15
2ej

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

DISEÑO, DESARROLLO E IMPLEMENTACION DEL SISTEMA DE SERVICIO DE ALERTA EN INTERNET PARA EL INSTITUTO DE INGENIERIA

TESIS PROFESIONAL

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A N :

SOFIA ROCIO GIL CASTILLO

CLAUDIA VIVAS HERNANDEZ

DIRECTOR: ING. GABRIEL CASTILLO HERNANDEZ

MEXICO, D. F. 1998

ÍNDICE

CAPÍTULO 3
EL SERVIDOR

CAPÍTULO 4
COMPONENTES DEL SISTEMA DEL SERVICIO DE ALERTA DEL INSTITUTO DE INGENIERÍA

Agradecimientos:

A Dios, por ser esa fuerza inexplicable que siempre me ayudó a seguir adelante y nunca darme por vencida.

A mis padres que me brindaron todo su amor y cariño alentándome a seguir siempre adelante, anhelando que siempre me preparara para enfrentarme a la vida, quiero que sientan que el objetivo logrado también es suyo y que la fuerza que me ayudó a conseguirlo fue su apoyo y comprensión. Este presente simboliza mi gratitud por toda la responsable e invaluable ayuda que siempre me brindaron .
Mi madre que a base de su sacrificio contribuyó grandemente a realizar mi más grande sueño, dándome su confianza transmitida día con día con tan solo haber creído en mi, por inculcarme los valores que ahora poseo que fueron piedra angular para poder terminar con éxito mi carrera profesional que ahora le dedico.
Mi padre por estar conmigo cuando más lo he necesitado, por darme su confianza, por creer en mi y por transmitirme ese orgullo que cuando a veces sentí caer me impulsó para no vencerme.

A mis hermanas Mari Lú y Fabis, que siempre estuvieron a mi lado compartiendo todos mis momentos de alegría, tristeza, éxitos y fracasos y a los detalles que me brindaron durante mi vida como estudiante.

A mi tío Luis que siempre estuvo al tanto de mi carrera y que siempre se preocupo porque nunca me desviara de mi camino.

A mi abuelo Jesús que sé , que hubiera estado feliz y muy orgulloso de ver culminado mi gran sueño que también era de él.

A todos mis amigos y seres queridos que siempre estuvieron a mi lado compartiendo gran parte de mi vida alentándome a cada momento a seguir adelante, regalándome un poco de su ser, transmitiéndome su sabiduría y brindándome su cariño (Estela, Fabiola, Diana, Ivette, Sandra, Vivis, Rocío, Rayo, Ricardo, Gus, Eduardo y Edgar).

A Alva por su ayuda incondicional y por todo su apoyo brindado en la realización de esta Tesis.

Al Ing. Gabriel Castillo Hernández por su paciencia y apoyo en la realización de esta Tesis.

Al Ing. Luis Palacios Hammeken y al Ing. Marco Ambriz Maguey por haberme permitido pertenecer al Instituto de Ingeniería y lograr que desarrollara mis conocimientos adquiridos en la carrera.

Al Instituto de Ingeniería, que me proporcionó todo lo necesario para la realización de esta Tesis.

Por eso, a Dios y a todos ustedes mil gracias.

Claudia Vivas Hernández

La vida es difícil, pero la mía es plena y feliz, puesto que tengo brillantes faros que me han guiado por la senda Que fácil es todo cuando hay apoyo, amor y comprensión, sin ustedes no sería lo que soy, ni estaría donde estoy. Gracias

A mi madre que es el ser más maravilloso del mundo. Gracias por el apoyo moral, su cariño y comprensión que desde pequeña me ha brindado, por guiar mi cariño y estar siempre junto a mí en los momentos difíciles.

A mi padre porque desde pequeña ha sido para mi un hombre grande y maravilloso, y que siempre he admirado. Gracias por guiar mi vida con energía, ésto es lo que ha hecho que sea lo que soy. Gracias por todo lo que me ha dado con amor, respeto y admiración.

A mi esposo Ernesto, por compartir tristezas y alegrías, éxitos y fracasos, por todos los detalles que me ha brindado durante mi vida como estudiante, por haberme ayudado día a día a lograr uno de mis más caros anhelos. Con amor y agradecimiento infinito.

A mi hija Cary por que no se cómo agradecer los momentos compartidos de su vida, ya que, me ha permitido robarle mucho del tiempo en el que merecía estar con ella. Quiero que sienta que el objetivo logrado también es suyo y que la fuerza que me ayudó a conseguirlo es el amor hacia ella.

A Gabriel por que no existen palabras que expresen el más profundo agradecimiento por el apoyo, la confianza recibido durante este tiempo, brindada aún en momentos difíciles y en especial por su paciencia y ayuda.

A Claudia, Alva, Yazmín y Dalila por la ayuda, comprensión y paciencia otorgadas en todo este tiempo. Gracias por haber compartido uno de los momentos más difíciles de mí vida

A mis hermanos, a mi suegra y a mí cuñado por su cariño, guía y apoyo. Este presente simboliza mi gratitud por toda la responsable e invaluable ayuda que siempre me proporcionaron.

Quiero expresar mil gracias al Instituto de Ingeniería y a las personas que se encuentran en él, por haberme apoyado de principio a fin en la realización de esta meta.

Sofía Rocío Gil Castillo

INTRODUCCIÓN

INTRODUCCIÓN

Día con día más personas están entrando al mundo de Internet, con el World Wide Web (Red Mundial de Cobertura Amplia, WWW), por lo que muchas de las empresas comienzan a utilizar dicha red para facilitar la comunicación y procesos que se llevan a cabo dentro de ella, automatizando los trabajos que se hacían manualmente, sustituyéndolos por las diferentes herramientas de Internet, las cuales facilitan y agilizan el manejo de éstos. Entre los posibles beneficios podemos mencionar: organizar y centralizar información, rapidez en el desarrollo de proyectos, distribución de aplicaciones, decremento de costos y aumento de eficiencia.

La informática, las telecomunicaciones y otras tecnologías de la información han revolucionado el mundo de las empresas y continuarán haciéndolo Las empresas tendrán que aprender a convivir y competir en este entorno y aprovechar al máximo el uso de la información. Esto no significa que los directivos de una empresa se conviertan en expertos informáticos o que sepan como utilizar una computadora, pero sí tienen que saber qué información necesita la empresa y cómo planificar, organizar y controlar este recurso.

El explosivo crecimiento de información ha hecho de esto un motivo de supervivencia para compañías que tienen a su disposición buenas herramientas de recuperación de información. El almacenamiento de información está siendo cada día más económico, y a causa de esto hay una gran cantidad de información.

Los sistemas de información serán fundamentales para las distintas organizaciones empresariales, ya que la información es uno de los recursos más valiosos de cualquier institución o empresa, de ahí la necesidad de obtenerla rápida y sin perder la objetividad de la misma. Estos sistemas de información no sólo tendrán que alcanzar a cubrir la gran demanda , sino que lo harán de manera eficiente y además realizados en tiempos razonables.

En los últimos años se ha producido una evolución en el entorno de la informática corporativa de sistemas centralizados residentes en "hosts" desde terminales remotas, se han ido asumiendo arquitecturas distribuidas que giran sobre todo alrededor del concepto Cliente/Servidor.

Pensando en aprovechar las ventajas que ofrece el WWW, se propuso un sistema en Internet llamado *Servicio de Alerta*, para el Instituto de Ingeniería. Este sistema permitirá difundir la gran cantidad de ejemplares (más de 100 títulos de revistas) que llegan a la Unidad de Servicios de Información (U.S.I.) Además, permitirá a personas ajenas al Instituto de Ingeniería enterarse de las revistas que se encuentran en éste, por medio del Web.

El sistema del *Servicio de Alerta* propuesto estará basado en una arquitectura cliente-servidor, en el cual, en la parte del cliente se tendrá una interfaz *Web* para consultar las revistas, los servicios de préstamo y suscripción de usuarios. Por otro lado se tendrá una interfaz en *Visual Basic* para la administración del *Servicio de Alerta* que se llevará a cabo por el personal de la U.S.I.. Para la parte del servidor se tendrá a Sybase como Sistema Manejador de Base de Datos Relacionales (RDBMS) y al servidor HTTPD.

La interfaz cliente del *Servicio de Alerta* consistirá de varias secciones en la parte del Web algunas de estas serán: *Sección de Búsqueda, Sección de Solicitud de Préstamo y Sección de Suscripción al Servicio de Alerta,* así mismo como las secciones de *agregar, consultar y eliminar revistas en la lista de suscripción;* estas secciones representan los servicios que ofrecerá el *Servicio de Alerta.*

También dentro de esta parte del cliente se tendrá para la administración del sistema una interfaz en Visual Basic que sea fácil de manejar y que tenga una presentación agradable para el personal de la U.S.I, esta administración consistirá en manipular información de las revistas, así como de los préstamos y suscripciones realizadas por los usuarios desde el web. Por otro lado, se podrán actualizar las tablas de contenido de los últimos ejemplares de cada revista.

El sistema de Servicio de Alerta manifestará algunos beneficios, como son:

- Que los usuarios consulten de manera rápida y eficaz las tablas de contenido, de los últimos ejemplares de cada revista, desde el Web del Instituto de Ingeniería.
- Difundir los ejemplares que se encuentran en la U.S.I. y
- Agilizar las tareas que se desarrollaban anteriormente por las personas encargadas de la U.S.I.

Esta tesis habla sobre los trabajos realizados para el desarrollo del sistema *de Servicio de Alerta* del Instituto de Ingeniería.

El presente documento consiste de cinco capítulos. En el capítulo uno se describe en qué consiste la arquitectura Cliente/Servidor y el entorno de red. En los capítulos dos y tres se explican las herramientas utilizadas en la parte cliente y la parte servidor del sistema, respectivamente. El capítulo cuatro define globalmente al sistema, este capítulo describe la problemática, el análisis, el diseño y el desarrollo del sistema del *Servicio de Alerta.* Además, se da a conocer la integración de mecanismos y recursos utilizados según las necesidades de los usuarios. Por último se presentan los comentarios finales originados por el desarrollo del sistema y se describen las tendencias en los sistemas de información, donde se tendrá un panorama general en los cambios tecnológicos.

Con este trabajo se pretende que el sistema de *Servicio de Alerta* proporcione servicios de manera factible, que refleje una estructura organizada al agrupar los procesos realizados, así como también optimizar los procedimientos administrativos que se hacían anteriormente, eliminando la necesidad de realizar manualmente los trámites de préstamo, y la posibilidad de difundir la información a los investigadores del Instituto de Ingeniería.

ARQUITECTURA CLIENTE/SERVIDOR

CAPÍTULO 1
ARQUITECTURA CLIENTE/SERVIDOR

1.1 Descripción y características:

Una arquitectura es un conjunto de definiciones, reglas y términos que se emplean para construir un producto, en consecuencia la arquitectura *Cliente/Servidor* se apega a estos principios.

La arquitectura *Cliente/Servidor* define una relación entre los usuarios que trabajan en computadoras denominadas sistemas frontales (front-end) y sistemas servidores denominados posteriores (back-end), que proporcionan servicios tales como el acceso a una base de datos, la gestión de red y el almacenamiento centralizado de archivos u otro tipo de sistema proveedor de servicios. Una red de computadoras ofrece la plataforma de comunicación en la que numerosos clientes pueden actuar con uno o más servidores. La interacción entre la aplicación que ejecutan los usuarios en el front-end y el programa (generalmente una base de datos o un sistema operativo de red) en el back-end se denomina relación *Cliente/Servidor*.

Las principales partes que integran un sistema *Cliente-Servidor* son: Cliente, Servidor y Red. A continuación se describirá en que consiste cada una de estas partes.

Cliente

El cliente es la combinación de software y hardware que invoca los servicios de uno o varios servidores, e incluso de otro cliente. Un cliente por lo general es una computadora personal y en algunos casos una estación de trabajo UNIX.

La forma más común que el cliente solicita un servicio a un servidor, es por medio de RPC's (Remote Procedure Call, llamada a un Procedimiento Remoto). Un RPC es un procedimiento que se ejecuta en otra máquina diferente a la que hizo la invocación del procedimiento, es decir, el cliente no ejecuta el procedimiento, sólo lo invoca en el servidor.

Los programas clientes son aquellos a los que los usuarios tienen acceso, un programa cliente provee una interfaz al usuario y funciones extras como ayuda y validación de datos, preprocesa los comandos antes de ser enviados al servidor, da formato a las peticiones de los servicios de la red y muestra la información o los mensajes enviados por el servidor.

Servidor

El servidor es la conjunción de software y hardware que responde a los requerimientos de los clientes. Los servidores en un entorno *Cliente/Servidor* son a menudo potentes sistemas superservidores, una estación de trabajo de alto nivel, minicomputadoras o computadoras centrales, capaces de gestionar adecuadamente las múltiples y simultáneas peticiones que reciben de los clientes, además de realizar tareas de seguridad y gestión de la red.

El programa servidor se encuentra en espera de alguna petición y actúa solo en respuesta a la solicitud de servicio de un programa cliente. Los programas servidores ejecutan los procesos correspondientes en respuesta a los comandos del cliente, transfieren la respuesta al cliente y esperan una nueva petición.

El servidor puede convertirse en un cliente de otro servidor al solicitar un servicio de otro servidor, gestiona el uso de un puerto (punto de comunicación lógica entre aplicaciones Cliente/Servidor) ante el sistema operativo.

Los servidores se clasifican en los siguientes tipos:

- *Servidores de archivos*
- *Servidores de datos*
- *Servidores de cómputo*
- *Servidores de aplicaciones*
- *Servidores de bases de datos*
- *Servidores de comunicaciones*

A continuación se describirá cada uno de éstos.

- ***Un servidor de archivo*** es aquel que permite compartir archivos entre un grupo de trabajo.

- ***Los servidores de datos y los servidores de cómputo*** se emplean conjuntamente, el servidor de datos se encarga de proporcionar los datos que el servidor de cómputo le solicita. El servidor de datos no efectúa ningún tipo de procesamiento incluido en la aplicación, sólo hace validación de datos basado en las reglas del negocio.

- ***El servidor de aplicaciones*** es aquel que efectúa tanto el manejo de datos, como el procesamiento de la aplicación, o sea, es un servidor de datos y un servidor de cómputo en la misma máquina.

- ***El servidor de base de datos*** es el caso más común en la arquitectura Cliente/Servidor. En él corre el manejador de base de datos, generalmente un RDBMS, que efectúa las funciones de actualización, consulta, verificación de integridad referencial, administración de recursos, etc.

La aplicación front-end en la PC cliente, por lo general es desarrollada en un ambiente totalmente gráfico como Windows y es la encargada de todos los procesos de interacción con el usuario, como entrada/salida de datos, pantallas, despliegue de gráficas y videos, etc.

El back-end en el servidor sostiene el procesamiento de datos y los accesos a disco, así como también otros servicios propios de un DBMS, y por norma se encuentra en un ambiente de multitareas y multiusuarios

La conectividad entre el front-end y el back-end, es la capacidad del front-end de poder accesar distintas bases de datos (de otros fabricantes), inclusive desde una misma aplicación, no importando el tipo de servidor ni el sistema operativo y por ende, el protocolo de comunicación en el que se encuentre el o los back-end, es decir, que se puede desarrollar una aplicación que acceda a los distintos servidores de bases de datos "al mismo tiempo", desde una misma PC y unir en una sola consulta los datos que se obtenga de todos ellos. Esto puede ser a través del estándar ODBC.

- Los ***servidores de comunicaciones*** adquieren dos formas, una computadora general con software específico para comunicaciones o servidores de comunicaciones como equipo de comunicaciones específico. En ambos los servidores de comunicaciones sirven principalmente para efectuar funciones de control de acceso, enrutamiento y puenteo de tráfico.

Red

Los servicios de la red son críticos en Cliente/Servidor, casi siempre el servicio es proporcionado en Ethernet. Usando TCP/IP como protocolo, aunque no es raro el empleo de otros.

La red de cómputo es el hardware y software de comunicaciones que enlaza a los clientes con los servidores.

Dependiendo de su cobertura, las redes se clasifican en redes de área local (LAN), redes de área metropolitana (MAN) y redes de área amplia (WAN). Difieren principalmente en la distancia que pueden cubrir, la tecnología de comunicaciones que emplean, el tipo de equipo, los canales de comunicación que pueden usar y la velocidad a la que pueden operar.

Una buena aplicación *Cliente/Servidor* deberá balancear adecuadamente el uso de los tres elementos, ya que cualquier falla de diseño o implementación podrá dar al traste rápidamente con el sistema.

Esta arquitectura de *Cliente/Servidor* abarca varias facetas de la informática. En la actualidad la mayoría de los sistemas *Cliente/Servidor* están orientados a transacciones de información con bases de datos casi siempre relacionales, su aplicación es en esencia, el conjunto de programas necesarios para que el usuario final maneje de manera transparente la información.

La arquitectura *Cliente/Servidor* aprovecha las características particulares de cada plataforma, aprovecha las características específicas de los grandes servidores para funciones que le son propias: arquitectura de entrada/salida mucho más amplias, CPU poderosas, subsistemas de discado que agilizan operaciones particulares, utilización de la CPU del lado cliente para tareas específicas que descarguen parcialmente equipos servidores ya saturados, descongestionamiento de equipos y servicios, disminuye los accesos a los programas del host, típicamente bases de datos, que consumen muchos recursos

La información y su proceso se puede distribuir entre múltiples equipos: procesos que implicaban por ejemplo, imprimir en el host, se pueden realizar localmente.

La figura 1.1 muestra un esquema de la arquitectura Cliente/Servidor, en la cual, varios clientes acceden a un único servidor. Esta es la configuración usual de una pequeña red de área local (LAN, Local Area Network) :

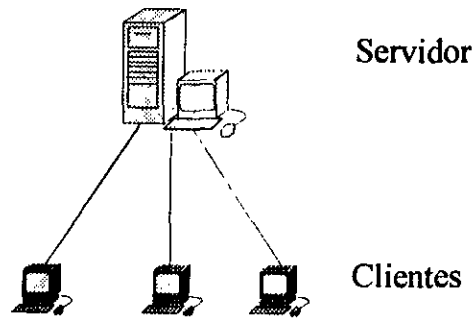


Fig. 1.1 Arquitectura Cliente/Servidor

Ventajas:

- Permite el acceso de un cliente a varios servidores en forma simultánea.
- La carga de trabajo asociada a las aplicaciones se divide entre las distintas computadoras. Los sistemas clientes realizan parte del procesamiento, que se distribuye sobre todos los sistemas de escritorio.
- Los sistemas servidores realizan la distribución de la información centralizada hacia unidades de almacenamiento conectadas directamente a ellos, reduciéndose así la información enviada a través de la red.
- Ayuda a las organizaciones a redimencionarse a partir de sus computadoras centrales y minicomputadoras hacia servidores y estaciones de trabajo sobre LAN's, que se constituyen así como plataformas de comunicaciones corporativa.
- Un porcentaje importante de información se ubica directamente en la memoria del servidor, no en la memoria de cada estación de trabajo que lo necesite.
- El tráfico en la red se reduce, ya que el servidor envía al cliente únicamente la información solicitada, no grandes bloques de información que deba procesar.
- Permite el uso de interfaces gráficas de usuario muy versátiles y amigables en los clientes
- Opera bajo sistemas abiertos.

Desventajas:

- Las aplicaciones *Cliente/Servidor* suelen ser más complejas que las tradicionales en la forma de anfitrión/terminales.

1.2 Entorno de Red

1.2.1 Internet: La red Mundial

En 1969 la Agencia de Investigación de Proyectos Avanzados de la Defensa de los Estados Unidos (DARPA Defense Advanced Research Projects Agency) fundó un proyecto de investigación y desarrollo para crear una red experimental de conmutación de paquetes (packet switching network). Esta red llamada ARPANET fue construida para estudiar técnicas de comunicación de datos que fueran robustas, confiables y no propietarias.

ARPANET tuvo mucho éxito y muchas de las empresas que estaban conectadas a ella empezaron a utilizarla para sus comunicaciones diarias. En 1975 ARPANET pasó a ser de una red experimental a una red operacional, y la responsabilidad de administrarla quedó en manos de la Agencia de Comunicaciones de la defensa (DCA-Defense Comunications Agency). La tecnología desarrollada por DARPA incluyó un conjunto de protocolos de comunicación y una serie de convenciones para interconectar redes y rutear tráfico, el cual fue llamado TCP/IP.

El conjunto de protocolos TCP/IP fue adoptado en 1983 como estándar militar (MIL STD), y todos los equipos conectados a la red fueron convertidos a los nuevos protocolos. Para alentar otras instituciones a adoptar TCP/IP, especialmente universidades y centros de investigación, DARPA fundó la compañía Bolt, Benarek, y Newman, Inc. (BBN) para implantar TCP/IP en Berkely UNIX (BSD). Así surgió la relación entre UNIX y TCP/IP.

A partir de que TCP/IP fue adoptado como un estándar, el término Internet fue usado comúnmente. En 1983, ARPANET fue dividida en MILNET, la parte no clasificada de la Red de la Defensa (DDN-Defense Data Network), y una nueva, pequeña ARPANET. Entonces el término Internet fue utilizado para referirse a la red entera: MILNET y ARPANET. En 1990, ARPANET desapareció formalmente, y el término *Internet* tomó un nuevo significado.

Una definición general de *Internet* es la que se da a continuación:

> ***Internet** es la colección mundial de redes interconectadas, que crecieron fuera de la original ARPANET, y que utilizan el conjunto de protocolos TCP/IP para ligar las distintas redes físicas en una sola red lógica.*

Características de Internet:

- ***Direccionamiento***: Para hacer de *Internet* un sistema de comunicación abierto, se aceptó un método general para identificar cada host conectado a ella. Para esto, a cada *host* se le asignó una dirección única de 32 bits, usualmente referida como dirección IP.

  La dirección de cada host contiene cuatro campos de 8 bits cada uno, cada uno de ellos separados por un punto. Cada dirección IP es dividida en dos partes : una parte de red y una parte de host como el siguiente formato:

  número1.número2.nímero3.número4

  Las direcciones IP se encuentran divididas en diferentes clases que dependen básicamente del espacio de direcciones que define la parte de red. Existen tres clases de direcciones: clase A, clase B, clase C.

En las *direcciones de clase A*, el primer número representa la parte de red, y los últimos tres son la parte local. En formato decimal, el primer número de una dirección de clase A debe de ser menor que 128, como 0 y 127 son números reservados, solo 126 redes de clase A existen, cada una de ellas puede contener hasta 2^{24} hosts.

Para las *direcciones de la clase B*, los primeros dos números de la dirección representan la parte de red y los últimos dos representan la parte local de la dirección. Esta parte es muy utilizada para redes muy grandes, como redes universitarias y redes de área amplia. El primer número debe ser mayor a 128 pero menor o igual a 191; el segundo número debe de estar entre 1 y 254, por lo que los números de esta clase de redes deben de estar en el rango de 128.1 y 191.254 (255 es un número reservado usado para enviar un mensaje a todos los sistemas que forman parte de la red), cada una de estas redes puede contener hasta 2^{16} *hosts*.

Para las *direcciones de la clase C*, los primeros tres números de la dirección representan la parte de red y el último representa la parte local de la dirección. Esta clase de direcciones son asignadas comúnmente a redes de área local (LAN), las direcciones de esta clase utilizan los primeros tres números en el rango de 192 1.1 hasta 223.254.254, esto permite que existan 254 host en cada red, pero puede haber muchas redes de este tipo.

Existen otras dos clases especiales de direcciones IP:
La clase D, cuyas direcciones son reservadas para enviar mensajes a todas las redes de *Internet* (Internet-wide multicast). La clase E, cuyas direcciones son reservadas para uso experimental.

El Centro de Información de Internet (InterINC) proporciona el servicio de registro de direcciones, quién es responsable de la asignación de direcciones IP únicas en *Internet*. InteNIC asigna la dirección correspondiente a la parte de red; el sitio local asigna la parte local a las máquinas de su subred.

- ***Subredes.*** Cuando el protocolo de *Internet* fue definido, el esquema de tener una parte de red y una parte local para cada dirección IP parecía ser adecuado. Nadie en ese tiempo tuvo la visión de que algunos sitios quisieran conectar redes locales muy grandes, algunas formadas por cientos o miles de hosts. Sin embargo, en poco tiempo muchos sitios de *Internet* fueron conectando redes de área local muy grandes, y algunos otros conectando redes pequeñas. Para este tipo de sitios cada red es conectada a otras redes por medio de ruteadores, por lo que cada red es una red independiente y necesita una dirección única. Sin embargo, como usualmente hay uno o quizás dos ruteadores entre un sitio e *Internet*, para propósitos de ruteo, éste se ve simplificado si cada sitio es identificado por un número de red, no importando cuantas redes se tengan internamente.

Este conflicto de necesidades fue resuelto con la implantación de subredes (subneting). Las subredes permiten que un sitio asigne a cada red un número único, mostrando solo una dirección de red en *Internet.* Las subredes agregan un nivel adicional entre las partes locales de una dirección IP. Para una dirección de clase B, los dos primeros serán la parte de red, el tercer número será la parte de subred y el último número la parte del host.

Parecería que siguiendo el esquema de subredes, cada host tiene que actuar como un ruteador con la finalidad de enviar adecuadamente paquetes a destinos a sus propias subredes, en el sitio más grande de la red, o en Internet. Si esto fuera cierto las subredes podrían no ser una respuesta eficiente para soportar el crecimiento del número de redes.

De hecho, las subredes de *Internet* están diseñadas de tal manera que un host en una subred necesite sólo determinar si un paquete particular es dirigido a un host en su propia subred o en cualquier otra red. Si es en otra red, necesita solo conocer la dirección de un ruteador de su propia red. No importa si el ruteador es el gateway entre una subred del host y la red a la cual el paquete es destinado. Si un host destina un ruteo inapropiado, el ruteador devolverá el mensaje para que este sea redireccionado al ruteador apropiado.

Un host determina si la destinación esta dentro de su propia subred usando un mecanismo conocido como enmascaramiento de dirección (Addres Mask o Net Mask). Un enmascaramiento consiste en la separación en una dirección IP, de la parte del host con la parte de la red y subred. En general, hay un solo enmascaramiento por cada red, al comparar esta máscara con su propia dirección y con la dirección destino, un host puede determinar cuando el destino se encuentra en su propia red o en otra red.

1.2.2 Protocolo TCP/IP

Debido a que *TCP/IP* es requerido para interconectarse a *Internet,* un gran número de organizaciones ha prestado interés a este conjunto de protocolos y con esto, nuevas aplicaciones y protocolos han sido desarrollados. En la actualidad es práctica de la comunidad de usuarios de UNIX utilizar *TCP/IP* en sus redes locales. Y muchas otras organizaciones utilizan cada vez más *TCP/IP* en sus redes, aún cuando estas no estén conectadas a *Internet.*

La popularidad de los protocolos de *TCP/IP* en *Internet* no creció rápidamente solo porque estuvieran allí, o porque agencias militares obligaron su uso. Ellos conocían una importante necesidad, la comunicación de datos a nivel mundial, y tenían características que satisfacían estas necesidades. Las más importantes son:

1. Protocolos estándares abiertos, disponibles gratuitamente y desarrollados independientemente de cualquier sistema operativo y cualquier hardware. Debido a que es ampliamente soportado, *TCP/IP* es ideal para unificar diferente hardware y software, aún no estando conectado a *Internet.*

2. Independencia de un específico hardware de red. Esto permite a *TCP/IP* integrar diferentes tipos de redes. *TCP/IP* puede ser utilizado en redes Ethernet, Token Ring, líneas seriales (dial-up lines), redes X.25 y virtualmente en cualquier otro medio físico de transmisión.

3. Cuenta con un esquema de direccionamiento que permite que cualquier dispositivo *TCP/IP* tenga un identificador único dentro de *Internet* o alguna otra red.

4. Amplia gama de aplicaciones y servicios soportados como son:
   - transferencia de archivos
   - comunicación electrónica y
   - acceso remoto.

1.2.3 Servicios de Internet a nivel de aplicación:

Parte de los servicios es brindar al usuario herramientas que le permitan acceder a equipos remotos y tener una vía de comunicación eficiente y confiable con el personal de su organización y de otras organizaciones, sin que haya diferencia en su localización física.

Entre los servicios básicos que se contemplan están:

- Sesiones remotas
- Comunicación electrónica
- Transferencia de archivos

A continuación se describirá cada uno de estos servicios:

1.2.3.1 Sesiones Remotas

Las ***sesiones remotas*** se efectúan mediante emuladores de terminal que proveen al usuario un mecanismo de enlace remoto con la mayoría de los equipos con que cuenta la organización que se encuentren conectados a la red, también con estos programas se logra el acceso a otros equipos dentro de la organización o aquellos localizados en otra red.

La mayoría de los accesos a las computadoras encargadas de llevar a cabo las tareas de procesamientos de datos en un ambiente distribuido, (manejo de correo electrónico, procesamiento distribuido, distribución de sistemas de archivos, accesos a sistemas y bancos de información, etc.), es efectuado principalmente desde *terminales remotas*, es decir se puede estar conectado a una de éstas computadoras a través de la red por medio de una computadora personal (PC) o una estación de trabajo (workstation).

Para acceder a la computadora remota deseada, se necesita encontrar un camino para pretender "ser" una terminal de esa computadora, esto se logra *emulando a la terminal.* Esta emulación es efectuada mediante *programas de comunicación o emuladores de terminal.*

Debido a que los protocolos de comunicación empleados en el modelo propuesto son *TCP/IP* e *IPX* en su mayoría, aplicaciones basadas en estos protocolos son utilizadas para llevar a cabo la emulación de terminales y así poder acceder a computadoras remotas.

La aplicación empleada bajo *TCP/IP* es *Telnet*, mientras que bajo *IPX* una de las aplicaciones utilizadas para efectuar sesiones remotas es *InfoView*, ambas herramientas proporcionan al usuario un medio eficiente y sencillo de enlace con computadoras remotas, permitiendo también que capacidades gráficas de las terminales (estaciones de trabajo y terminales gráficas) sean empleadas para ejecutar procesos complejos de simulación y visualización.

Dada la gran aceptación de *TCP/IP* como un estándar de comunicación a nivel mundial, *Telnet* se ha convertido en la herramienta más empleada para efectuar sesiones remotas en *Internet* y redes que operan con esta familia de protocolos.

Telnet permite al usuario acceder a una computadora remota desde su computadora local, una vez conectado el usuario al equipo remoto, éste puede introducir datos, ejecutar programas y llevar a cabo todas aquellas tareas que desee como si estuviera trabajando directamente con el equipo remoto.

Mientras *Telnet* está ejecutando, la computadora local se hace invisible durante la sesión con la máquina remota, ya que cada vez que se presiona una tecla esta información es transmitida directamente a la computadora remota. Cuando la sesión termina el control es devuelto nuevamente a la computadora local.

Telnet emplea el protocolo ***TELNET*** del conjunto *TCP/IP*, diseñado para soportar sesiones remotas. El protocolo *TELNET* define el proceso de comunicación entre computadoras locales y remotas que utiliza una sesión remota; *TELNET* asume que otros protocolos de *TCP/IP* se harán cargo de las actividades a nivel de red tales como, corrección de errores y aseguramiento de que los paquetes son transferidos correctamente entre las computadoras.

El funcionamiento de *TELNET* se basa en tres conceptos básicos, primeramente define una *terminal de red virtual (NVT, Network Virtual Terminal)* que provee una interfaz estándar para sistemas remotos, de esta manera, los programas cliente no tienen que conocer los detalles de todos los sistemas remotos existentes; por lo que son construidos para usar esta interfaz estándar.

En segundo término *TELNET* incluye un mecanismo de opciones negociables, que permite al cliente y al servidor negociar opciones, proporcionando un conjunto de opciones estándar (*por ejemplo una* de las opciones controla si los datos transmitidos usan el conjunto de caracteres ASCII de 7 bits o el de 8 bits).

Finalmente, *TELNET* opera ambos puntos de la conexión simétricamente, así permite que cualquier programa *sea un cliente. Estos conceptos permiten a TELNET* conectar dos programas de aplicación, dónde uno actúe como cliente. Estos conceptos permiten a *TELNET* conectar dos programas de aplicación, donde uno actúe como *cliente* y otro como *servidor*.

La figura 1.2 ilustra la forma en que los programas de aplicación implantan un cliente y un servidor de *TELNET*

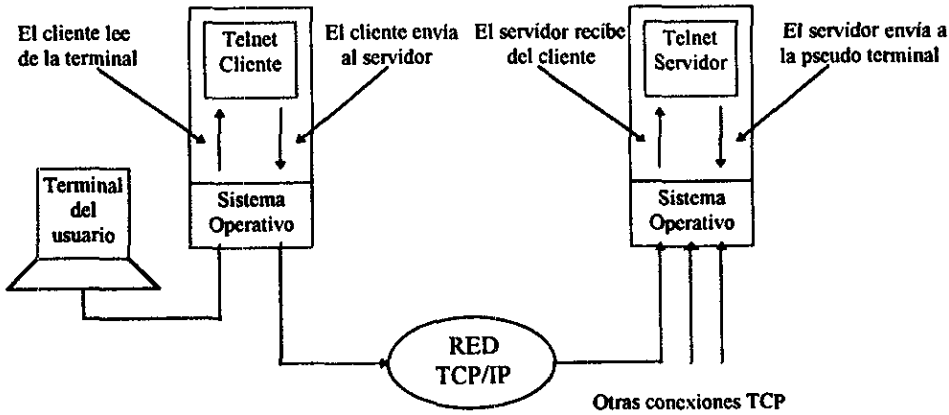


Fig. 1.2 Implementación de un cliente y un servidor de TELNET

La figura anterior muestra, cuando un usuario invoca *TELNET*, un programa de aplicación en la máquina del usuario se convierte en cliente. El cliente establece una conexión *TCP* con el servidor. Una vez que la conexión ha sido establecida, el cliente acepta secuencias de teclas de la terminal de usuario y las envía al servidor, mientras que concurrentemente acepta caracteres que el servidor regresa y los despliega en la terminal de usuario. El servidor debe aceptar una conexión *TCP* del cliente y transmite datos entre la conexión *TCP* y el sistema operativo local.

1.2.3.2 Comunicación electrónica

La ***comunicación electrónica*** aprovecha herramientas de comunicación en línea y de transferencia de mensajes, como el correo electrónico, con las cuáles el usuario puede entablar, en forma confiable y eficiente, comunicación con otros usuarios de una organización, así mismo tiene la posibilidad de comunicarse con personal de otras instituciones nacionales o extranjeras que estén integradas a una red mundial y que cuenten con los mismos servicios.

La aplicación más utilizada de comunicación electrónica entre usuarios de una red es el correo electrónico, considerado la herramienta de comunicación más utilizada en el mundo, en la cual diariamente millones de personas envían mensajes que viajan de una ciudad a otra, de país a país y de continente a continente.

La mayoría de estos mensajes son solo texto, pero también es posible enviar archivos conteniendo imágenes, tales como fotografías o planos arquitectónicos. Más recientemente, han surgido innovadores tipos de mensajes como lo son sonido digitalizado y animación.

A través del correo electrónico usuarios de una red (ya sea local, metropolitana o de área amplia) intercambian mensajes, el correo electrónico se utiliza para enviar mensajes a una persona en específico o a varias personas a la vez.

Esta forma de envío de mensajes presenta grandes ventajas sobre otros medios de envío como son el fax, mensajería y telefonía, ya que los costos son menores, pero sobre todo es una forma más rápida y confiable de entrega de mensajes.

El funcionamiento del correo electrónico difiere de otros servicios de red que emplean protocolos de comunicación para enviar paquetes directamente a sus destinos, utilizando intervalos de tiempo (timeout) y transmisión de paquetes individuales sin confirmación de entrega (acknowledgement returns).

En el caso de corre electrónico, el sistema debe proveer funciones extras cuando la máquina remota o la conexión de red fallen. El usuario no tiene que esperar a que la máquina remota este disponible para continuar su trabajo o que tenga que abortar la transmisión porque la comunicación con la máquina remota no está disponible, para ello se emplea un *mecanismo de entrega retardada.*

Para manipular el mecanismo de entrega retardada, los sistemas de correo electrónico emplean una técnica conocida como *spooling* (*áreas de almacenamiento temporal*). Cuando el usuario envía un mensaje, el sistema hace una copia del mensaje en esta área de almacenamiento conteniendo la identificación del remitente, la dirección del destinatario y el tiempo de depósito. Entonces el sistema inicia un proceso de *transferencia en segundo plano* (*background transfer process*) con la máquina remota, siendo totalmente transparente al usuario, permitiendo que éste lleve a cabo otras actividades. La figura 1.3 muestra esta idea.

Este proceso es encargado de transmitir el correo convirtiéndose en cliente, mapear el nombre de la máquina destino y la dirección IP, e intentar formar una conexión *TCP* con el servidor de correo en la máquina destino. Si es exitosa la conexión, el proceso de transferencia pasa una copia del mensaje al servidor remoto que la almacenará en su área de spool.

Una vez que el cliente y el servidor acuerdan que la copia ha sido aceptada y almacenada, el cliente remueve la copia local. Si el proceso de transferencia no puede establecer la conexión *TCP* o si la conexión falla, el proceso graba el tiempo en que será tratado de enviar nuevamente el mensaje y termina

Este proceso ve el área de spool periódicamente, checando el correo no enviado. Cuando encuentra un mensaje sin mandar trata de enviarlo nuevamente Si el proceso encuentra que un mensaje no ha podido ser enviado después del tiempo predeterminado, éste regresa el mensaje al remitente, como se muestra en la figura 1 3.

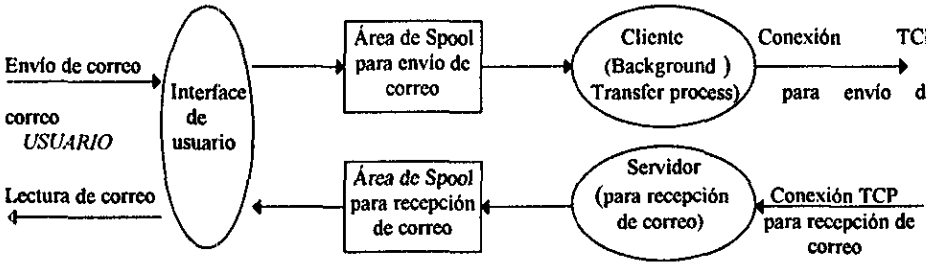


fig. 1.3 Elementos conceptuales de un sistema de correo electrónico. El usuario invoca a una interfaz de usuario para depositar o recoger correo , todas las transferencias ocurren ensegundo plano (background)

El conjunto de protocolos *TCP/IP* especifica un estándar para el intercambio de correo entre máquinas. El protocolo especificado para el manejo de correo electrónico es ***SMTP (Simple Mail Transfer Protocol)***, que describe las características bajo las cuales el correo electrónico debe ser implementado. En sistemas *UNIX*, *SMTP* es el protocolo utilizado por el programa ***Sendmail***. Este programa es capaz de comunicarse con otros servicios de correo basados en *SMTP*, así mismo, puede operar como un intérprete *(gateway)* entre diferentes sistemas de correo

Sendmail elimina problemas que otros sistemas de entrega de correo muestran, al utilizar un esquema de *direcciones electrónicas de correo (E-mail Addres)* para ruteo y entrega de mensajes. *Sendmail* acepta mensajes del usuario creados con algún cliente, posteriormente reescribe la dirección destino del mensaje en un formato adecuado para el programa de entrega y envía el mensaje al destino correcto, todo esto sin que el usuario tenga conocimiento de ello. Otra característica relevante de *sendmail* es su capacidad para direccionar mensajes entre los diferentes sistemas de correo que existe en *UNIX* y otros sistemas operativos.

Sendmail puede funcionar no solo como servidor, también como cliente, sin embargo, la mayoría de los usuarios nunca emplea *sendmail* directamente, en su lugar utilizan aplicaciones que permiten una fácil interacción con él (llamadas front-end's), como lo son: ***PINE, elm, Mail, mailx, eudora, popmail y mush.***

1.2.3.3 Transferencia de Archivos

A través de programas para ***transferencia de archivos***, la adquisición de datos desde equipos remotos o entre computadoras situadas en distintos puntos geográficos (ciudades, países y continentes), se convierte en una tarea que implicará al usuario invertir menores cantidades de tiempo y dinero para transferir información.

Uno de los grandes beneficios que se tiene al contar con una red de cómputo, es la facilidad de transferir archivos entre computadoras, convirtiéndose así en uno de los usos principales de la red. Los archivos a transferir pueden ser datos, imágenes, programas, texto y cualquier otro tipo de información.

La aplicación estándar utilizada para transferir archivos en redes que emplean *TCP/IP* es el programa ***ftp (file transfer program)***, que se basa en el protocolo ***FTP (File Transfer Protocol).***

FTP se basa también en el modelo *Cliente/Servidor* para efectuar transferencias, es decir, el usuario que inicia la conexión es el cliente mientras que la máquina remota asume la función de servidor. Como otras aplicaciones, la mayoría de las implementaciones de *FTP* permiten el acceso concurrente de clientes múltiples.

Los clientes utilizan una conexión *TCP* para conectarse al servidor, mientras que el servidor espera la conexión y crea un *proceso esclavo* para administrarla, este proceso acepta y administra el control de la *conexión del cliente* y crea un proceso adicional para administrar la *conexión de transferencia de datos.*

El control de la conexión del cliente le indica al servidor cual archivo transferir, mientras que la conexión de transferencia de datos, se encarga de transportar todos los datos entre el servidor y el cliente.

Usualmente ambos, cliente y servidor, crean procesos separados para administrar la transferencia de datos. Los detalles exactos de la arquitectura de los procesos dependen exclusivamente del sistema operativo, la figura 1 4 ilustra el ejemplo

Como se muestra en la figura, el control de la conexión del cliente se conecta al control de la conexión del servidor utilizando una conexión *TCP*, mientras que la conexión de transferencia de datos emplea su propia conexión *TCP*

En general, los procesos de control y el control de conexión permanecen activos tanto tiempo como el usuario mantenga la sesión de *ftp* activa. Sin embargo, *ftp* establece una nueva conexión de transferencia de datos por cada archivo transferido.

De hecho, muchas implantaciones de *FTP* crean un par de procesos de transferencia de datos, por cada conexión *TCP*, que se crea cuando el servidor necesita enviar información al cliente. Una vez que la conexión de control desaparece, la sesión es finalizada y ambos terminan los procesos de transferencia de datos, como se muestra en la figura 1.4 .

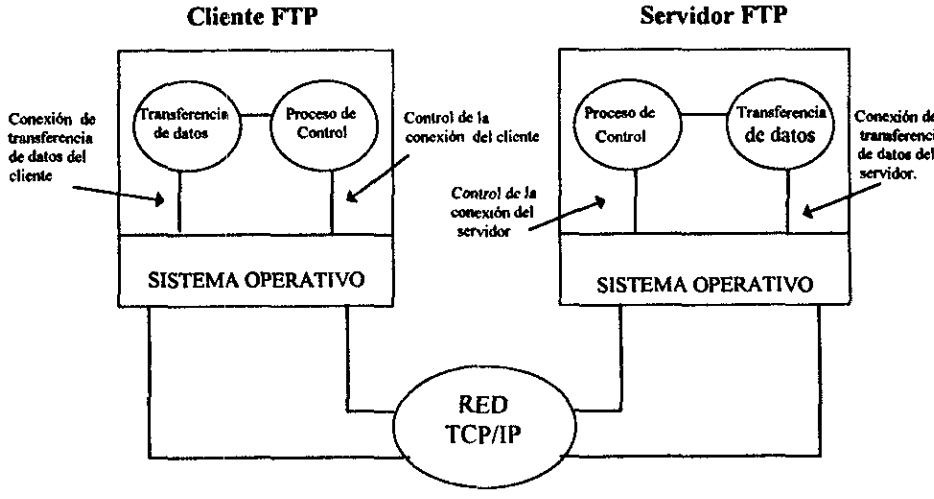


figura 1.4. Modelo de interacción de cliente y servidor de FTP

Entre las características más relevantes de *ftp* se encuentran:

- ***Acceso interactivo.*** La mayoría de las implementaciones de *ftp* proveen una interfaz interactiva que permita al usuario interactuar fácilmente con servidores remotos. Entre las operaciones que se pueden efectuar en una sesión de *ftp* son: listar, crear y eliminar directorios remotos, etc.
- ***Especificación de formato (representación).*** *ftp* permite al cliente especificar el tipo de formato de los archivos a ser transferidos, es decir, el usuario puede emplear diferentes modalidades de transferencia (*modo binario* para formatos especiales o *modo ascii* para archivos de texto), con esta característica la transferencia de datos puede hacerse incluso entre computadoras que empleen sistemas operativos diferentes o formatos de almacenamiento distintos.
- ***Control de autenticidad:*** *ftp* requiere que los clientes verifiquen su autenticidad enviando su nombre de usuario *(login name)* y su clave de acceso *(password)* al servidor antes de iniciar una sesión de transferencia de archivos. El servidor niega el acceso a aquellos clientes que no cuenten con un nombre de usuario y una clave de acceso válida.

1.3 World Wide Web (WWW)

El ***World Wide Web (WWW)*** fue desarrollado en Suiza, por el European Particle Physics Laboratory. El *WWW*, también referido como *W3* o *WEB*, combina obtención de información y uso de *hipertexto* para hacer un simple pero poderoso sistema de información. El *WWW* consiste en documentos virtuales que pueden contener otros documentos con ligas a otros servidores *WWW*. Los índices son considerados como documentos especiales que pueden ser examinados. Los servidores de *WWW* soportan documentos en una amplia variedad de formatos Post script, sonidos, imágenes y video, o ligas a otros servidores *WWW*.

El *WWW* utiliza el protocolo ***HTTP (Hyper Text Transport Protocol)*** como un programa de examinación que solicita documentos o búsquedas por medio de palabras claves de un servidor remoto. Estos programas son diseñados para acceder datos usando *HTTP* (conexión TCP utilizando el puerto 80) y protocolos existentes como *FTP* y el *Network News Transport Protocol* usado en Usenet. Originalmente el *WWW* incluía un cliente modo línea para utilizarse con *Telnet* y una interfaz gráfica limitada, actualmente la mayoría de los clientes *WWW* están basados en ***interfaces gráficas (GUI)*** que pueden desplegar imágenes y video, y agregar sonido al hipertexto de calidad PostScript, el cliente *GUI* más destacado para el *WWW* es Mosaic desarrollado por el National Center for Supercomputing Applications (NCSA) en Urbana Champaign, Illinois.

Para el uso efectivo de servidores *WWW* los documentos están escritos en lenguaje conocido como ***HyperText Markup Language (HTML).*** Muchas herramientas están siendo desarrolladas para convertir documentos en formato *HTML* a otros.

Cerca de 700 servidores *WWW* están siendo operados alrededor del mundo, 300 de los cuales se encuentran en E.U., los clientes específicos de *WWW* requieren de un gran ancho de banda en red para transferir imágenes, sonido y texto formateado del servidor, por lo que el uso de líneas telefónicas (dial-up) no es adecuado para una eficiente operación de *WWW.*

1.4 Plataformas

1.4.1 Windows 95

Windows 95 es un sistema operativo de Microsoft, que permite realizar tareas comunes, aprender aspectos más técnicos y encontrar sugerencias de ayuda en el manejo de éste. Además, en Windows 95 se mejoraron las características incluidas en *Windows.* Por lo que se agregó una ayuda acerca de procedimientos, la cual representa la principal fuente de información de Windows.

Windows 95 ofrece muchas características nuevas y de gran utilidad, además de mejorar muchas que ya aparecían en las versiones anteriores de *Windows* y con las que probablemente ya está familiarizado. Algunas de estas tareas se describen a continuación:

- ***Nueva interfaz mejorada:*** *Windows 95* incluye características como el botón de "Inicio" y la barra de tareas. El botón "Inicio" es utilizado para abrir programas, buscar documentos y utilizar herramientas del sistema rápidamente. La barra de tareas sirve para cambiar entre programas tan fácil como si se tratara de canales de televisión.
- ***Explorador de Windows***: El explorador de Windows es una vía útil para explorar y administrar archivos, unidades y conexiones de red.
- ***Nombres Largos de archivo:*** *Windows 95* acepta largos nombres de archivos para facilitar la organización y búsqueda de los archivos.
- ***Soporte mejorado para multimedia y juegos***: Se tiene una capacidad de vídeo mayor y más rápida para los juegos, da soporte mejorado para los juegos basados en MS-DOS y da un rendimiento más elevado para reproducir archivos de vídeo y de sonido.
- ***Compatibilidad del hardware Plug and Play***: Se puede insertar la tarjeta para hardware Plug and Play en la PC. Al encender la PC, Windows reconocerá e instalará su hardware de manera automática.
- ***Multitarea prioritaria de 32 bits***: Windows ahora permite utilizar varios programas a la vez: hacer más en menos tiempo.
- ***Microsoft Exchange***: Se utiliza Microsoft Exchange para ver y trabajar con todo tipo de comunicaciones electrónicas, incluidos el correo electrónico y el fax

- ***The Microsoft Network*:** Se podrá utilizar este nuevo, económico y sencillo servicio en línea para comunicarse con usuarios de todo el mundo a través del correo electrónico, de boletines electrónicos y de Internet.

Por otro lado, *Windows 95* contiene la seguridad para los archivos antiguos, a continuación se describirá en que consiste esta seguridad.

Antes de instalar Windows 95, será conveniente hacer copias de seguridad de determinados archivos del sistema. Aunque la mayoría de las instalaciones de Windows 95 son sencillas y no representan problema alguno, es posible que se produzca un error (p.e. un fallo del sistema debido a incompatibilidad de hardware, o un corte de corriente) que impida, temporal o permanentemente, el acceso a la información.
Los archivos de los cuales es conveniente hacer copias de seguridad son los siguientes:

- Todos los archivos de inicialización(.ini) del directorio Windows.
- Todos los archivos de registro (.dat) del directorio Windows.
- Todos los archivos de contraseñas (.pwl) del directorio Windows.
- Todos los archivos especificados en los archivos Config.sys y Autoexec.bat

Después deberá instalar Windows de la manera que comúnmente se conoce. Y dependiendo de la instalación de la PC, al iniciar Windows aparecerán determinados íconos en el escritorio; los más importantes se muestran en la tabla 1.1.

	Se podrá ver el contenido de la PC y administrar los archivos.
	Se podrán ver los recursos disponibles en la red si la PC tiene acceso a ésta, o puede conectarse.
	La papelera de reciclaje es un lugar de almacenamiento temporal de los archivos eliminados. Puede utilizarse para recuperar archivos eliminados por error.
	Se podrán iniciar programas, abrir documentos, cambiar la configuración del sistema, obtener ayuda, buscar elementos en la PC y mucho más.

Tabla 1.1 Íconos generados por default en la instalación de windows 95.

1.4.2 UNIX

Antecedentes:

UNIX nació en los laboratorios Bell AT&T, una de las instituciones investigadoras más amplias y mejor dotadas de los E.U. La intervención de esta historia es que los avances importantes no han venido principalmente de decisiones burocráticas, sino de las necesidades y creatividad de los usuarios.

Thompson y Ritchie después de haber creado un juego de "Viaje espacial" para el PDP-7, se pusieron a crear una nueva estructura de sistemas de ficheros y nuevo software muy similar al sistema de ficheros moderno. Le añadieron un entorno de procesos con planificación y completaron el resto de un sistema operativo rudimentario. El nombre UNIX fue pronto aplicado a los resultados, ya que su trabajo era una simplificación del sistema MULTICS. El sistema estuvo operando sobre el PDP-7 a principios de los 70's. Muchas de las ideas del UNIX actual estaban ya implantadas, incluyendo el sistema de ficheros, la implantación de procesos y la estructura de las líneas de órdenes aún utilizadas.

La implantación original del sistema UNIX fue codificada en lenguaje ensamblador, pero pronto se desarrolló el lenguaje de programación C. Inmediatamente, el lenguaje C fue utilizado en la continuación del desarrollo del sistema UNIX, y en 1973 se recodificó el núcleo en C.

El sistema UNIX captó la imaginación de los informáticos en los laboratorios Bell, y después de unos años UNIX se encontraba corriendo en varias máquinas diferentes. Se realizaron con frecuencia importantes mejoras de software, y AT&T comenzó a soportar el sistema como producto interno dentro de los laboratorios Bell.

El sistema operativo UNIX ha evolucionado durante los últimos veinte años desde su invención, hasta llegar a convertirse en uno de los entornos informativos más populares e influyentes en el mundo. En nuestros días el crecimiento de utilizar el sistema UNIX en maxi y supercomputadoras es potente y complejo, pero con en el transcurso del tiempo ha llegado a ser mucho más regular, controlable y amistoso, algunas características que han ayudado a UNIX a este crecimiento son:

Características:

- ***Herramientas Software:*** El sistema Unix introdujo una nueva idea en la computación: los problemas pueden ser resueltos y las aplicaciones creadas mediante interconexión de unos cuantos fragmentos simples. Estos fragmentos son generalmente componentes completos diseñados para realizar bien una tarea específica. Además, se pueden generar grandes aplicaciones a partir de secuencias de órdenes simples. Algunas subrutinas empaquetadas se combinan para formar nuevos programas ejecutables. El concepto de *reutilización de software* es una de las principales razones por las que el sistema UNIX resulta ser un entorno muy productivo en el trabajo

- ***Portabilidad.*** El sistema UNIX ha sido implantado en casi cualquier computadora construida de tamaño moderado o grande. El sistema UNIX es una evolución natural para las microcomputadoras, máquinas baratas pero potentes. Sólo unos cuantos cambios y adaptaciones mínimos han sido necesarios para hacer el sistema UNIX utilizable. El valor de esta portabilidad no puede ser sobrestimado, porque el desarrollo software es caro y tedioso. Aplicaciones importantes, tales como los procesadores de texto, las bases de datos y los sistemas gráficos, pueden llevar muchos años y esfuerzos de desarrollo. Generalmente se reconoce que el sistema UNIX proporciona el entorno adecuado para permitir el fácil traslado de aplicaciones desde microcomputadoras hasta maxicomputadoras.

- ***Flexibilidad.*** UNIX es un *banco de trabajo,* ha sido adaptado a aplicaciones tan divergentes como la automatización de fábricas, los sistemas de conmutación telefónica y los juegos personales. Es fácil extender el sistema UNIX básico que algunos paquetes de aplicaciones son dificilmente reconocibles en que se encuentran basados.

- ***Potencia.*** El sistema UNIX es uno de los sistemas operativos más potentes disponibles sobre cualquier computadora La sintaxis de UNIX permite al usuario realizar tareas rápidas y sencillas. Así también, los usuarios pueden aprovechar servicios internos de UNIX como herramientas adicionales.

- ***Multiusuario y Multitarea.*** UNIX es un sistema multitarea de tiempo compartido, porque se puede hacer fácilmente más de una cosa a la vez. El sistema UNIX está diseñado para manejar sin esfuerzo las necesidades múltiples y simultáneas de un usuario. También es un entorno multiusuario porque soporta las actividades de más de una persona a la vez. UNIX soporta cientos de usuarios a la vez, y todos éstos tienen un sistema privado sobre su microcomputadora.

- ***Elegancia.*** El sistema UNIX está considerado como un sistema elegante. En UNIX se pueden realizar muchas y grandes tareas de un modo sencillo y hermoso, siempre y cuando se tengan bien comprendidos los conceptos de éste.

- ***Orientación a Red.*** Las nuevas versiones de UNIX están organizadas para un uso de red fácil y funcional.
  Las herramientas de comunicación internas del sistema, la fácil aceptación de rutinas de dispositivos adicionales de bajo nivel y la organización flexible del sistema de ficheros son naturales para el entorno de red actual, en el cual los usuarios de estaciones de trabajo personales comparten algunos recursos centralizados tales como datos o dispositivos de comunicación Al ser cada vez más común el uso de computadoras en grupos de trabajo, las capacidades de conexión por red del sistema UNIX se vuelven más y más importantes.

Las redes de hoy en día se extienden desde pequeñas redes de área local (LAN, Local Area Network) hasta enlaces a nivel mundial que permiten transferencias de datos a alta velocidad de enormes bases de datos. El sistema UNIX, con su capacidad de multitarea y su enorme base de comunicaciones, hace que la computación por red sea simple y fácil.

- ***interoperabilidad.*** La gran flexbilidad del sistema UNIX se ha formalizado ahora en las interfaces binarias de aplicaciones (ABI) y la interfaz de programación de aplicaciones (API). UNIX se está desplazando hacia un mayor compromiso con el estándar internacional POSIX de sistemas operativos. El estándar POSIX proporciona un conjunto mínimo de funciones que serán soportadas por todos los sistemas de tipo UNIX. Por lo cual, POSIX, API y ABI ayudan a proporcionar una vía segura para usuarios y creadores de software.

Filosofía de UNIX

Un objetivo principal de su diseño fue hacer un sistema amplio en alcance y en diseño de órdenes, que no intentaran resolver todos los problemas posibles; por el contrario que fueran simples, silenciosas y muy buenas.

La facilidad de desarrollo y mantenimiento de software fue un resultado directo de la filosofía *"Lo pequeño es bello"* del sistema UNIX para la construcción de herramientas.

La filosofía original de *"hacer una cosa bien"* en el sistema UNIX primitivo se ha perdido hasta cierto punto en las nuevas versiones, ya que órdenes actuales contienen un gran número de opciones y controles. Por otro lado, con esta filosofía se ha ayudado a ampliar órdenes del sistema para satisfacer necesidades del usuario.

Este hecho tiene pros y contras, algunas de ellas se explicarán a continuación:

Ventajas y Desventajas

Comparando con MS-DOS, el sistema UNIX es mucho mayor y más complejo; por el contrario el sistema UNIX ofrece características, las cuales no se encuentran disponibles en MS-DOS.

El sistema UNIX está orientado principalmente a terminales *basados-en-carácter.* Se requiere un software especial denominado sistema de ventanas X para hacer que el sistema UNIX funcione con pantallas gráficas de mapa de bits Esta dependencia y soporte de las terminales ASCII ha ocasionado que la mayoría de las herramientas de ratón y menús que utilizan la pantalla gráfica tan sólo estén comenzando a aparecer masivamente Por otra parte, el sistema UNIX permite a un usuario invocar una máquina remota desde una terminal barata, sobre líneas telefónicas baratas. Esto hace que el sistema sea ideal para aplicaciones de *servidor*, en las cuales la interfaz primaria de usuario no está asociada con la máquina UNIX centralizada, sino que se ejecuta directamente sobre una terminal

"inteligente" remota. Esto proporciona una moderna arquitectura para entornos de redes de área local; de hecho, el sistema de ventanas X estándar está adaptado a red, significando que una aplicación puede ejecutarse en una máquina en red, mientras que la interfaz de usuario puede ejecutarse sobre máquinas baratas colocadas en la mesa de cada usuario. Este esquema puede reducir significativamente el costo de instalación de equipo de cómputo en un entorno de red.

El sistema UNIX está contra la tendencia actual de hacer los sistemas operativos "invisibles" al usuario. Muchos sistemas operativos han intentado crear interfaces de usuario que eliminen el sistema operativo de la vista del usuario. La interfaz de usuario orientada a ventanas y menús, ayuda a ocultar las órdenes, los sistemas de ficheros y las herramientas administrativas del usuario. En el sistema UNIX, por el contrario, mientras más consciente se está de las funciones internas del sistema, mejor se pueden controlar en beneficio propio y para mejorar su productividad. En el sistema UNIX se han preparado herramientas tales como los agentes de usuario para aislar muchas de las complejidades del sistema operativo.

EL CLIENTE

CAPÍTULO 2
EL CLIENTE

2.1 Herramientas

2.1.1 Browsers HTTP

Los *browsers Web (visualizadores web)* son usuarios finales (front-end), son la interfaz al *World Wide Web* . El *browser* se comunica con el *Servidor HTTP* para la transferencia de documentos de hipermedia. Esta transferencia puede tomar lugar en la misma máquina con una red de área local (LAN), a través de un red de área amplia (WAN) donde se quiera en todo *Internet.*

El *browser* es el responsable de recibir entradas para el usuario final, inicializando documentos, transfiriendo peticiones con el *servidor de Web*, recibiendo el código *HTML* para el servidor y convirtiendo este mismo en formato de salida para desplegarlo en la pantalla.

Los *browsers* que existen operan virtualmente en todas las plataformas y son capaces de comunicarse con todos los servidores Web.

La principal función de un *browser* es que acepta *lenguaje de Hipertexto de Marcas (Hyper Text Markup Language HTML)* de entrada a un servidor Web y despliega el correspondiente documento *HTML.*

El HTML contiene hyperlinks (superligas) que cuando se activan, causa que el *browser* se contacte con el apropiado servidor Web para traer el documento. Los *browsers* soportan predefinidas etiquetas de *HTML* ya especificadas. Estas especificaciones determinan como el *browser* administra los *tags HTML* contenidos en el documento, no todos los *browser* soportan las mismas etiquetas de *HTML.*

Tres ejemplos básicos de *browsers* son:

- (NCSA) Mosaic 2.1,
- Netscape Communications Navigator 2.0 y
- Microsoft Internet Explorer 2.0 .

El *browser* de Mosaic no despliega tablas como en los otros dos.

NCSA (Mosaic)

NCSA (Mosaic) es un *browser* de información en *Internet* y un cliente de *WWW*, desarrollado por la Universidad de Ilinois. *NCSA Mosaic* fue uno de los primeros *browsers* y el responsable del *overwhelming* soporta el protocolo *HTTP* en comunicación con el *WWW.*

Corre en las siguientes plataformas:

- X Windows System
- Macintosh
- Microsoft Windows 3.1, 95 y NT

Netscape Navigator

Netscape Navigator es un browser de información en Internet , es compatible con *NCSA Mosaic* y contiene características adicionales que no están disponibles en otros browsers, como:

Ejecución progresiva (Progresive Rendering).
Progresive Rendering permite bajar simultáneamente texto e imágenes. Los browsers manejan esto para usar múltiples conexiones al Servidor Web, cada conexión solicita un objeto diferente (imagen o texto). El beneficio real de Progresive Rendering, es que el usuario final carga rápido los documentos y accesa más rápido a algunas ligas.
Múltiples conexiones proveen acceso a los objetos de datos en paralelo en el orden en que aparecen en el documento, también incrementan la carga en el Servidor Web.

Continuous-Display Scrolling:
Con esta característica el usuario final puede ver el contenido del documento antes de que termine de cargar. El usuario final puede tomar la decisión de continuar o terminar cargando los procesos.

Secure Sockets Layer (SSL):
Usando el *SSL*, Netscape Navigator provee características de seguridad, este ofrece comunicación privada con servidores seguros certificados. El *SSL* provee encriptación de datos, servidor verificador, integridad de mensajes, y cliente verificador opcional para sesiones *TCP/IP*, Netscape Navigator soporta los recursos de método de acceso, el cual fue creado para servidores específicos que soportan *SSL.*

Microsoft Internet Explorer

Internet Explorer es un *browser* del *WWW* que integra todos los productos de Microsoft, es más reciente que Netscape Navigator y que NCSA Mosaic.

Internet Explorer Relase 2.0 soporta todas las etiquetas HTML Level 2 así como también Level 3.2 y extensiones de Netscape. *Internet Explorer* es soportado en todas las plataformas de Microsoft Windows.

Además no solo soporta tablas, sino extiende el soporte de tablas para incluir alineación de atributos que permiten fluir texto alrededor de la tabla.

Internet Explorer soporta seguridad SSL y STT (Secure Transaction Technology). STT es una tecnología usada para manejar transacciones financieras. Soporta PCT (Private Communication Technology) que es un canal seguro de protocolo.

2.2 El lenguaje como Cliente

2.2.1 Visual Basic

Existe en la actualidad mucha gente que se encuentra utilizando Visual Basic 3.0 para crear Base de Datos, por lo que si éstas cambiarán a Visual Basic 4.0, notarían varios cambios en los métodos utilizados para crear BD. Cualquiera de los programas existentes deben correr con Visual Basic 4.0, no importando la versión antigua, y algunos cambios mayores de los cuales deben estar pendientes, como los siguientes:

- Nuevos métodos CreateTableDef y CreateField son utilizados para definir tablas y campos.
- Se puede definir el nivel de validación de campo y tabla para Bases de Datos por medio de la colocación de las propiedades adecuadas.
- Se pueden crear relaciones entre tablas para establecer y reforzar la integridad referencial.
- La definición de datos en instrucciones SQL están disponibles para crear o modificar tablas.
- Se puede borrar un campo de una tabla bajo ciertas condiciones. En versiones anteriores, no se podían borrar campos de tablas de ninguna manera.

Visual Basic 4.0 también ofrece mejoras sobre la versión anterior en el área de Data Control y Bound Control. Estas mejoras dramáticamente extiende sus capacidades y superan la mayoría de las deficiencias de versión 3.0. Para el Data Control, las mejoras son las siguientes:

- La habilidad para crear cualquiera de los tres tipos de Recordset (Table, Dynaset o Snapshot).
- La habilidad para asignar un Recordset creado con los objetos de Acceso de datos hacia un control, o viceversa.
- Se agregaron propiedades de acción BOF y EOF, las cuales permiten al desarrollador que automáticamente solicite una acción específica cuando la línea de archivo sea alcanzada.

Cuando se crea un objeto Recordset, se debe de especificar el tipo de Recordset, el cual puede ser Dynaset, Snapshot o tabla, cuando se accesan datos reunidos por medio de la conexión ODBC.

Se pueden crear dos tipos: Dynaset o snapshot. Un objeto Recordset de tipo Dynaset o Snapshot regresa algún resultado de una consulta de datos.

Dynaset: Contiene una visión "viva" actualizable de datos en las tablas existentes. Cuando el Dynaset cambia, las tablas existentes se actualizan inmediatamente. De manera inversa, cuando la tablas cambian, el Dynaset está actualizado.

Snapshot: Es una visión estática o no actualizable de los datos en las tablas existentes.

Al crear un Snapshot, Visual Basic reforma todos los datos a las columnas seleccionadas de los renglones correspondientes, y coloca los datos reformados en el Recordset. Inversamente, cuando se crea un objeto Dynaset, VB reforma sólo la llave primaria (o Bookmark) de la consulta. Para ambos Recordset, VB almacena en la memoria los resultados de la consulta.

La habilidad para crear cualquiera de los tipos de Recordset es la más importante de estas tres mejoras. Esta habilidad permite poner cualquier índice para un Data Control que utiliza un Recordset tipo tabla, permitiendo el uso del método de búsqueda para encontrar rápidamente registros. Siendo capaz de poner significados de índices que pueden cambiar el orden de la presentación del Recordset más fácilmente. Otra ventaja de poder utilizar tablas es que cualquier actualización hecha por otros en un ambiente multiusuario están inmediatamente disponibles para el usuario actual. Éste no sucede cuando el Datacontrol esta limitado al utilizar solo Dynaset.

Además de las mejoras en el Data Control la versión 4.0 agrega cinco nuevos Data Bound Control. Estos controles se listarán a continuación:

- List Box
- Combo Box
- DBList Box
- DBCombo Box
- DbGrid

La figura 2.1 muestra estos cinco objetos en una forma en Visual Basic:

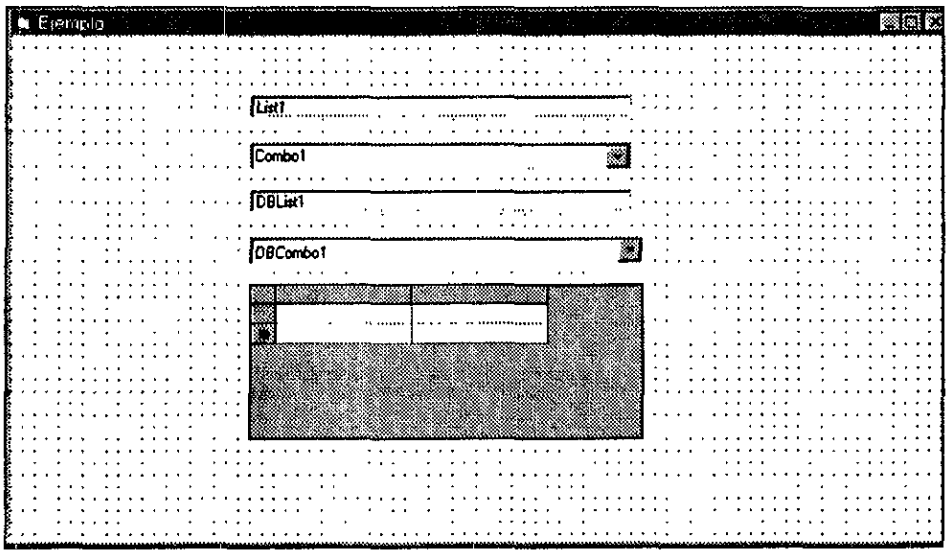


Fig. 2.1 Objetos de Visual Basic

Estos nuevos controles expanden la flexibilidad de programas que pueden ser desarrollados con los Bound control.

Visual Basic 4.0 mejora la conectividad de Base de Datos avanzadas y las características como manejadores de datos. Califica al lenguaje como un ambiente de desarrollo de aplicación de Base de Datos muy completa El acceso a Base de Datos ha puesto a Visual Basic como un producto competitivo en el mercado de Base de Datos.

Ventajas:
Las principales ventajas de VB4 ante sus competidores de Base de Datos son:

- Desempeño
- Simplicidad
- Flexibilidad
- Extensibilidad

Características de VB4

A continuación se listarán las caracteristicas que soporta Visual Basic 4.0:

- El Engine de Base de Datos Jet 3.0 de 32 bits ofrece un desempeño comparado con el de 16 bits del Jet 2 plus. Jet 3.0 es un multitareas con un mínimo de 3 tareas de ejecución.

- El Data control integrado en VB permite crear rápidamente una forma para proyectar la información de la Base de Datos con poco o nulo código en VB.

- VB4 es flexible porque éste no encerrará en una estructura de aplicación particular, como es el caso con la Interfaz de Documento Múltiple (MDI) de Access, no se requiere utilizar instrucciones DoCmd para manipular la actual Base de Datos y otros objetos de Access.

- Los controles OLE permiten adicionar nuevas características a las aplicaciones de BD de VB, con un esfuerzo de programación mínimo.

- La tercera parte de los desarrolladores están ocupados creando nuevos controles de visión de datos de VB4.0.

- La Interfaz de la Base de Datos de Visual Basic 4.0 requiere substancialmente menos recursos que otros como Access. Más aplicaciones BD de VB de 32 bits corren sobre Windows 95 con PC's de 8 MB de RAM y sobre Windows NT 3.51 plus en el rango de 16 MB. Microsoft dice que Access 95 requiere 12 MB para realizar un desempeño adecuado de todo, excepto aplicaciones triviales de Access 95.

Algunos fundamentos de la programación en Visual Basic:

Al utilizar el estilo de programación orientado a objetos, manejo de eventos, y métodos céntricos en Visual Basic para desarrollar plataformas de Base de Datos no se necesita crear un programa "Main", debido a que ya existe (MSWindows), este programa es un WinMain oculto en cada archivo ejecutable de Visual Basic. Todos los procedimientos de entrada de VB como Event-handlers, son métodos que la aplicación ejecuta en respuesta a eventos.

El código del Event-handlers está contenido en módulos que son componentes de cada forma. Estos módulos se pueden crear para declarar variables Public y para contener código que es común en varias formas, pero sólo código de manejadores de eventos pueden llamar procedimientos almacenados en los módulos Visual Basic.

La figura 2.2 muestra la ventana principal de VB4 y la forma que despliega por default.

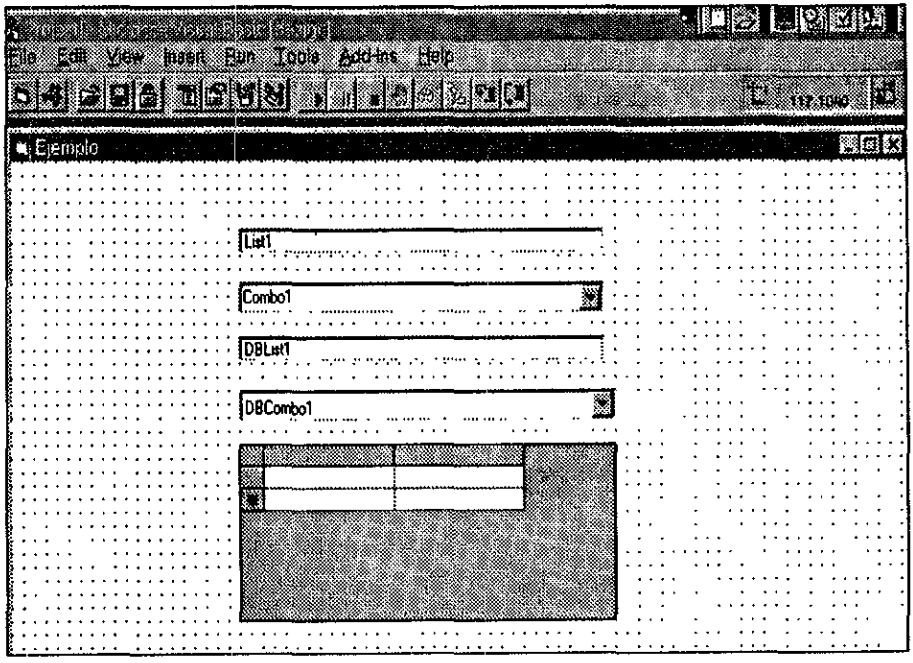


Fig. 2.2 Ventana principal de la aplicación de Visual Basic

De principio de cuenta al accesar a Visual Basic se genera un objeto de default llamado Forma, con su evento Load, por lo cual se llamará una rutina manejadora de evento "Form_Load( )". Visual Basic automáticamente asigna nombres, que consiste en el nombre de la forma, un guión bajo, y el nombre del evento. Visual Basic crea un sub (StartUp), sub StartUp_load...End sub (Sub declara una subrutina de la aplicación WinMain), en el cual se tecleará el código que se requiere para inicializar la aplicación.

La figura 2.3 muestra una subrutina en VB:

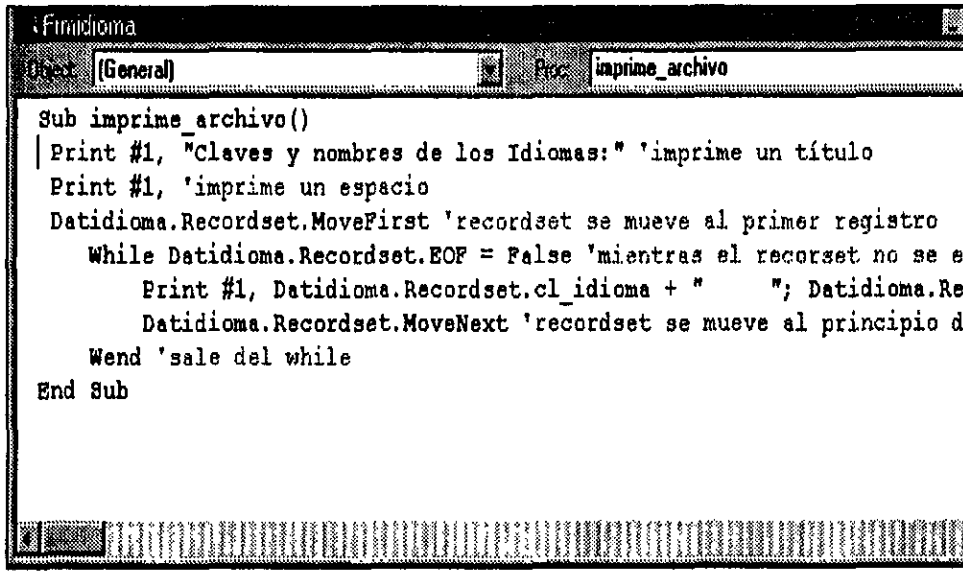


Fig. 2.3 Subrutina para imprimir datos de la BD en Visual Basic

Visual Basic clasifica formas, controles de las formas , base de datos y tablas como objetos. Un objeto posee características y comportamientos; las características de un objeto son las propiedades (properties) y el comportamiento del objeto se determina por los métodos (incorporados en código de manejo de evento). Un objeto contiene datos y código. Los objetos pueden contener otros objetos, pe. Las formas pueden contener a su vez botones, cajas de texto, imágenes, etc. Cada objeto de Visual Basic tiene su propio conjunto de propiedades predeterminado, para el cual se asignan valores por default.

La figura 2.4 muestra la lista de eventos para una forma y las propiedades de ésta:

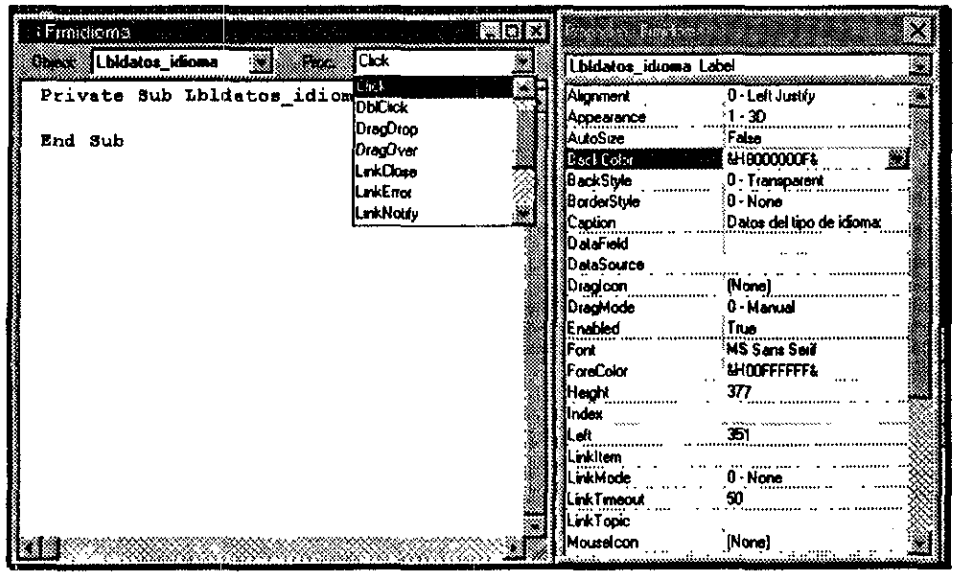


Fig. 2.4 Lista de eventos y propiedades en una forma de VB

Variables y tipos de datos

Visual Basic permite crear variables objeto por medio de la instrucción: "**Dim** *objvariable* **As** *Tipo de objeto* **Set** *variable objeto= nombre del objeto*". Algunos tipos de datos que maneja Visual Basic son los siguientes:

- Single 4 bytes
- Double 8 bytes
- Currency 8 bytes
- String 1 byte por caracter
- Byte 1 byte
- Boolean 2 bytes
- Object 4 bytes
- Date 8 bytes
- Variant 16 bytes más 1 byte por caracter

A continuación se proporcionará una lista de los objetos que maneja Visual Basic para crear formas de presentación final:

- Data Control
- DBList
- DBCombo
- DBGrid
- Label
- Text Box
- Combo Box
- Check Box
- Dialog Box
- Option Button
- Frame
- Command Button
- MaskEd Box
- Data Bound Drive
- Data Bound Directories
- Data Bound File
- Image
- List Box, etc

La figura 2.5 muestra los objetos del Toolbax de VB4:

Fig. 2 5 Barra de herramientas de VB

Utilizando VB4 como interfaz de una base de datos

El modelo componente de objetos (COM), es una infraestructura para crear módulos de código (llamados objetos) que son independientes de los lenguajes de programación y plataformas de computadoras. COM define un juego de interfaces que las aplicaciones deben soportar. OLE 2 plus es una implantación de alto nivel del COM diseñado para compartir objetos entre aplicaciones Windows y aplicaciones de programación.

La principal aplicación en Visual Basic es la conectividad a BD Cliente/Servidor por medio del ODBC. Por lo que VB4 incluye una aplicación improvisada Add-ins, un administrador de datos, los cuales permiten crear nuevas BD Jet, así como agregar, borrar y modificar en nuevos y existentes Jet's, DBase, FoxPro, Paradox y PDB Btrieve.

El administrador de datos (Data Manager) es una aplicación de BD de VB4 que utiliza formatos de interfaz de documento múltiple (MDI). El MDI permite crear aplicaciones de BD con varias ventanas que están contenidas dentro de una ventana convencional.

Espectro abierto de conectividad de BD de VB hace a VB un candidato ideal para desarrollar aplicaciones de presentación final de BD. El término BD de presentación final es utilizado para describir una aplicación de computadora que puede seleccionar un renglón de datos contenidos en una BD que se despliega el renglón escogido como una importante herramienta para el usuario.

Algunas aplicaciones de presentación final sólo despliegan datos, y otras también permiten actualizar la BD por medio de Editar, Agregar o Borrar registros. El sistema BD en sí mismo es llamado back-end, la BD back-end es como mínimo una colección de tablas relacionadas. Los administradores de BD tradicionales almacenan estas tablas relacionadas como archivos individuales en un directorio único, junto con los archivos de índice que son utilizados para acelerar el proceso de vaciado de datos. El Jet y el RDBMS guardan todas las tablas relacionadas e índices, además de las consultas almacenadas en un archivo de BDs único.

La figura 2.6 muestra una página de la interfaz del Servicio de Alerta realizada en VB4:

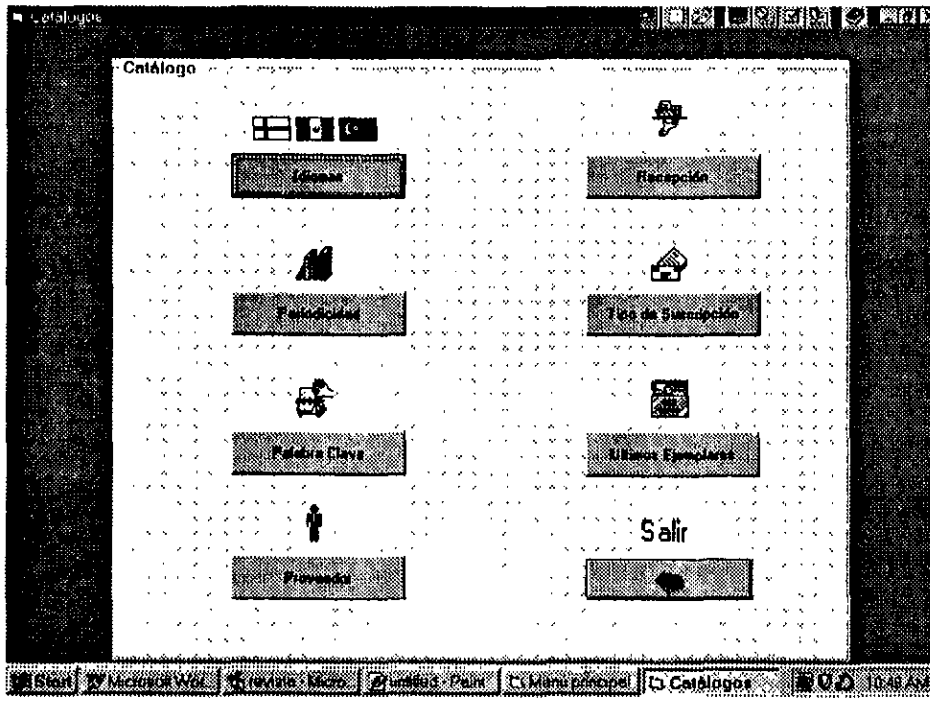


Fig 2.6 Forma para la interfaz en VB

En el siguiente tema se definirá más ampliamente el ODBC, el cual es una herramienta básica para realizar la conectividad a las Bases de Datos.

La herramienta de open database conectivity (ODBC)

Esta herramienta consta de 4 componentes que a continuación se mencionan:

- **Aplicaciones:** Es la responsabilidad para interactuar con el usuario por medio de la interfaz, además permite llamar funciones ODBC para someterse a declaraciones SQL y para recuperar resultados

- **El manejador driver**: Carga y maneja los driver según las aplicaciones

- **Driver:** Procesa las llamadas funciones ODBC, somete peticiones especificas con el Data source y regresa resultados hacia las aplicaciones Los drivers tienen responsabilidad en la interacción con algunas capas de software consideradas necesarias para accesar la información Tales capas incluyen software interfaces como redes fundamentales o archivos de sistemas.

- **Data source:** Consiste en un conjunto de datos y un ambiente asociado el cual incluye operaciones de el DBMS y redes. El término data source es utilizado para describir cualquier componente de software (que no sea uno de los tres descritos anteriormente).

A continuación se explicaran más detalladamente en que consiste cada uno de estos componentes, así como también se explicará el funcionamiento de los mismos.

Aplicaciones:

Las aplicaciones hacen todo el trabajo externo para la interfaz ODBC, así como también utilizan el ODBC para conectarse con el Data Source, enviar respuestas SQL a éste y de esta manera regresar resultados

El Driver Manager:

Administra interacciones entre programas de aplicación y drivers. Múltiples aplicaciones y múltiples drivers pueden ser administrados simultáneamente por el manejador de driver.

Una aplicación llama a una función ODBC por medio del manejador de driver, y éste enruta la llamada al driver correcto Cuando una aplicación utiliza ODBC para conectarse con alguna fuente de datos, el manejador de driver determina cual driver es requerido y carga este driver en la memoria. De ahí en adelante, el manejador de driver toma cualquier llamada de función de entrada de la aplicación y llama a la función con el mismo nombre en el driver. Cuando la aplicación llama a la función ODBC para desconectarse, el manejador de driver descarga el driver de memoria, o sea, si más de un aplicación utiliza el driver; éste no se descargará de la memoria hasta que finalice con la última aplicación. El manejador de driver también se encarga de verificar errores para asegurar que se realicen las funciones en el orden correcto, siempre y cuando contengan valores válidos.

Un driver tiene la función de recibir peticiones de su manejador para poder conectarse con el DBMS, y también puede usar algún software de comunicación para conectarse con el servidor. El ODBC puede conectarse directamente con drivers sin necesidad de utilizar el manejador.

Los Drivers:

Para obtener una funcionalidad adecuada en el uso del ODBC, se deben seleccionar los drivers requeridos, a continuación se describen los diferentes drivers existentes:

- **El Driver de un canal** accesa a un archivo de base de datos, a un ISAM o a un archivo plano. Se distingue de otros tipos de driver porque realiza todos los procesamientos SQL por sí mismo, debido a que la fuente de información en un sistema de este tipo no es una motor de base de datos SQL o un servidor Cliente/Servidor. El programa de interfaz para tal fuente de información consiste en un archivo E/S (Excel o Texto) o llamadas de función ISAM (Access, FoxPro, Paradox y Dbase) para que el driver se convierta en el motor de bases de datos SQL.

- **El Driver de dos canales** es el sistema clásico Cliente/Servidor; el driver (cliente) envía y recibe el protocolo de datos del DBMS (servidor), o mapea a la base de datos API, pero no accesa directamente la información. El DBMS recibe las peticiones SQL del cliente, ejecuta éstas y le envía el resultado.

  En un sistema de dos canales, un driver que utiliza directamente el protocolo de datos actúa como un conductor de información para el DBMS, justo como el *runtime library* de la base de datos API. Un driver que mapea entre el ODBC y la base de datos API actúa como traductor entre API's.

- **El Driver de tres o más canales** no tiene muchas diferencias con el doble, la diferencia es que el cliente es un sistema de tres canales; en vez de conectarse directamente con el DBMS, se conecta al servidor pero actúa como un *gateway* para el DBMS en cuestión (generalmente puede conectar a múltiples DBMS's).

  Una característica de la arquitectura de tres canales es que maneja gran parte de la complejidad del cliente hacia el servidor, el cual puede ser de bastante ayuda en la simplificación de la instalación, configuración, y administración de drivers. Todos los clientes utilizan un driver sencillo para conectarse al *gateway*, éste a su vez, envía peticiones al driver apropiado en el servidor.

Data sources:

Parte de la misión del ODBC es la de esconder mucha de la complejidad del software de comunicación que se utiliza que generalmente presenta en los sistemas Cliente/Servidor. El ODBC utiliza el *DSN (data source name)* para mapear un solo nombre para todos los componentes del software existentes requeridos para accesar la información.

El DSN es escogido por un usuario final o un administrador de sistema y debe de ser un nombre que represente esta información.

La utilización del ODBC para el acceso a bases de datos

Es un método de comunicación para Base de datos cliente/servidor en VB. El ODBC es parte de la arquitectura de los sistemas abiertos de microsoft windows (WOSA), el cual provee una serie de interfaces y programas de aplicación (API) para simplificar y proveer normas para varias actividades de programación La importancia es que todas las aplicaciones se comuniquen a través del mismo conjunto de API's.

El ODBC es sólo una pieza del WOSA. Antes de accesar al ODBC, se deben configurar los nombres de la fuente de información, sus driver ODBC y los valores de configuración que se utilizan en el ODBC.ini.

Cada driver ODBC se conforma en uno de los tres niveles de capacidad, los cuales pueden ser nivel Base, nivel 1 y nivel 2:

1. *El nivel base* es el conjunto de habilidades que un driver ODBC debe contener; todos los driver ODBC deben de cumplir con los requerimientos de este nivel.

   Las habilidades para este driver son las siguientes:

   - Proveer aplicaciones con conexiones de BD.
   - Preparar y ejecutar declaraciones SQL
   - Regresar resultados de las consultas.
   - Procesar transacciones, incluyendo regresos y peticiones.
   - Proveer información de errores.

2. *El nivel 1* incluye las capacidades del nivel central del conductor ODBC más la conectividad de la fuente de información a través del conductor de diálogos específicos. Todos los conductores ODBC que el Jet engine utiliza es necesario que se reúnan o se excedan en este nivel.

3. *El nivel 2* incluye las capacidades del nivel 1 del conductor ODBC más las siguientes capacidades:

- Conexiones accesibles de buscadores para la fuente de datos
- Regresa resultados de las consultas en formato arreglo
- Obteniendo información adicional tal como los procedimientos almacenados

Creando y nombrando una fuente de datos ODBC

Ha sido nombrado "fuente de datos ODBC" debido a la conexión que realiza a una Base de datos. El nombre y la localización de una base de datos, que se requieren para realizar esta conexión se almacenan como un "Data Source" en el archivo ODBC.INI.

Para desarrollar una Fuente de datos ODBC se deben tomar en cuenta dos partes importantes:

- El Administrador de Datos ODBC
- Register Database, por medio del DAO que utiliza el Enginer Database

Si se desea cargar la fuente de datos, a través del *Administrador de datos ODBC* , los pasos a seguir son los siguientes:

1. Accesar al panel de control de windows, como se muestra en la figura 2.7, hacer click en el administrador ODBC, por lo cual, aparecerá una lista de todas las fuentes de datos, como en el ODBC.INI. Otra manera más sencilla, es ejecutar simplemente el archivo ODBCADM.EXE.

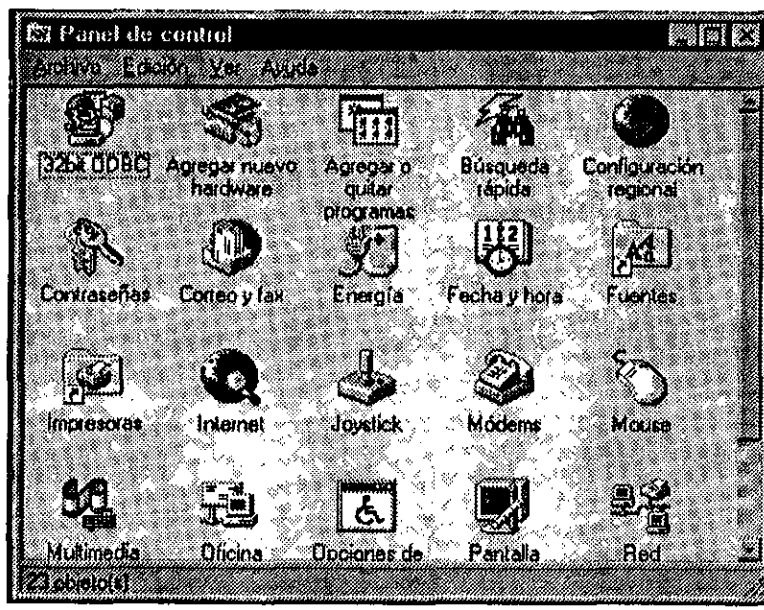


Fig. 2.7 Ventana del Panel de Control en win95

Para cargar una nueva fuente de datos se debe presionar el botón Add como se muestra en la figura 2.8, el cual desplegará otra ventana con una lista de driver's ODBC que se encuentran instalados en el sistema.

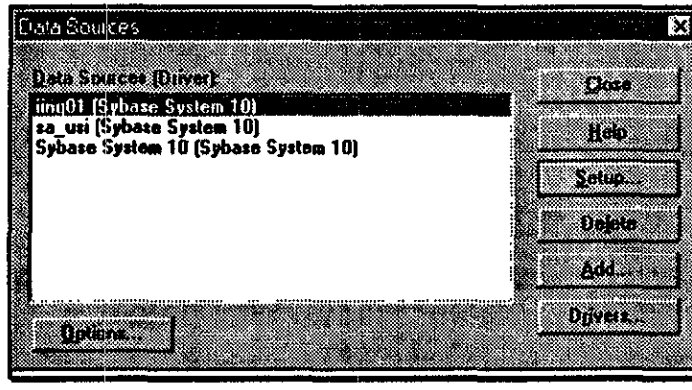


Fig 2.8 Ventana de selección de fuente de datos para el ODBC

2. Se seleccionará el driver requerido y por consiguiente aparecerá otra ventana con el ODBC setup, como se muestra en las figuras 2.9 y 2.10.

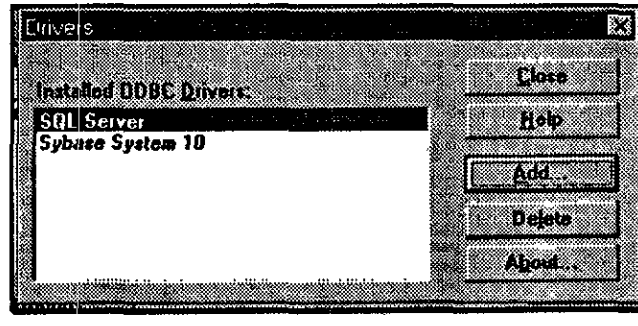


Fig. 2.9 Ventana de selección de los driver's para el ODBC

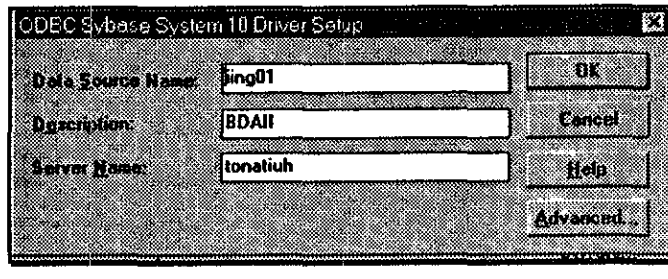


Fig. 2.10 Ventana de datos del driver seleccionado para el ODBC

3. Se insertarán los datos de nombre y descripción de la fuente de datos, así como el nombre del driver específico, los cuales son necesarios para la instalación; después se deberá oprimir el botón de OK, donde el administrador ODBC cargará la fuente de datos a la lista de DataSources.

Otra manera de cargar una fuente de datos es por medio del método *Register Database en el DBEngine*, en el cual se deben tomar en cuenta los parámetros que se muestran en la tabla 2.1:

PARÁMETROS	DEFINICIÓN
dbname	El usuario define un string que especifica el nombre de la fuente de datos.
Driver	Es una cadena que determina los nombres de los driver's instalados, en una lista en el archivo ODBC.INI, ej. ORACLE
silent	TRUE: toda la información de conexión. FALSE: se desplegará la ventana setup e ignorará el contenido de los atributos.
Atributos	Es toda información de conexión para utilizar el driver ODBC.

Tabla 2.1 Parámetros para cargar una fuente de datos en DBEngine

Tipos de datos ODBC

Cuando se leen aplicaciones con tipos de datos externos, una correspondencia uno a uno entre tipos, no siempre ocurre; tal es el caso de las lecturas externas de fuentes de datos ODBC. Para comprender los tipos de datos del ODBC ver la siguiente tabla:

TIPOS DE DATOS DEL ODBC	DESCRIPCIÓN	TIPOS DE DATOS DE VISUAL BASIC
SQL_BIT	Dato binario de un solo bit	YES/NO
SQL_TINYINT	Un número entero entre 0 y 255.	Integer
SQL_SMALLINT	Un número entero entre 32,767 y -32768	Integer
SQL_INTEGER	Un número entero entre 2,147,483,647 y -2,147,483,648	Long
SQL_REAL	Un número punto flotante con una precisión de 7 dígitos.	Single
SQL_FLOAT SQL_DOUBLE	Un número punto flotante con 15 dígitos de precisión	Double

SQL_TIMESTAMP SQL_DATE, SQL_TIME	Dato tipo Tiempo-Día	DateTime
SQL_CHAR	Cadena de caracteres	Si es 255 caracteres o más, para texto. Si es menor que 255 caracteres, para memorándum.
SQL_VARCHAR	Una longitud variable de cadena de caracteres con una longitud máxima de 255	Text
SQL_BINARY	Dato binario de longitud redondeada	255 caracteres o más en binario y si es menor de 255 caracteres en Registro OLE.
SQL_VARBINARY	Longitud variable de datos binarios con una longitud máxima de 255 caracteres	Binary
SQL_LONGVARBINARY	Longitud variable de datos binarios con una longitud máxima de fuente dependiente	Registro OLE
SQL_LONGVARCHAR	Una longitud variable de cadena de caracteres con una fuente dependiente de longitud máxima	Memo
SQL_DECIMAL SQL_NUMERIC	Valor con signo, exacto ó numérico con precisión y escala	Si la escala es 0, después ?, si la precisión es 4 ó mayor para entero; si es precisión de 9 para longitudes; si tiene precisión 15 o mayor para Double. Si la escala es menor que 0 y la precisión mayor que 15, es para Double.

Tabla 2.2 Tipos de datos del ODBC

A continuación, se describirán algunas diferencias entre el ODBC y el Jet Engine.

Comparación entre el ODBC API y el Objeto de Acceso de Datos Jet

Estos dos métodos sirven para accesar a una Base de Datos ODBC, y sus diferencias son:

- El ODBC API: Consiste de aproximadamente 30 funciones que se encuentran en el ODBC.DLL. Este soporta registro a registro regresando datos y utilizando sintaxis SQL para manipular y definir datos. Los diferentes driver's conforman uno de los tres niveles de capacidades.

- El Jet DAO: Puede utilizar driver's ODBC que proporcionan soporte en el nivel 1, por esa razón, el Jet no soporta todas las funciones ODBC. A continuación se mencionará cuales son las funciones ODBC que sí soporta el Jet

A continuación se determinan las ventajas y desventajas al utilizar el ODBC API.

Ventajas al utilizar ODBC API directamente:

- El API es bastante flexible, porque es un API de nivel muy bajo, o sea, el ODBC API es utilizado para comando de salida en el servidor de BD directamente.
- El desarrollador necesita construir todas las estructuras de datos para regresar información. Esto dará al programador un máximo poder en el diseño de la aplicación.
- También el API es rápido, va directamente al servidor de BD.
- Está diseñado para proporcionar la conectividad a todas las bases de datos, y de este modo, ser portátil a través de lenguajes y BD.

Desventajas:

- Se necesita un desarrollo externo, el ODBC SDK, para obtener los prototipos y documentación requerida para utilizar las funciones del API; porque el ODBC es un API de nivel muy bajo, por lo que se comenzará construyendo funciones de ayuda para proporcionar una interfaz de alto nivel para su aplicación.

- El API no proporciona objetos modelo para utilizarlo o construir

- Raramente en una aplicación es probable requerir el ODBC API directamente para accesar.

El Jet Dao.
Se puede accesar por medio del DAO, inicializando y utilizando el método DBEngine's Open Database y especificando una cadena ODBC de conexión, que consiste de los siguientes 2 renglones:

1. El tipo de Base de Datos, el cual puede ser ODBC.
2. Una cadena de parámetros para especificar el driver ODBC utilizado.

Esta cadena debe contener el nombre de la BD o path, el login del usuario, el password, el nombre de la fuente de datos y un login de tiempo de salida, el cual es opcional.

Cuando se liga una Base de Datos ODBC, el Jet Engine almacena una gran parte de la información sobre la tabla localizada, incluyendo tabla e información del registro como fuente del servidor.

Este método es el preferido para accesar a una Base de Datos, porque utiliza el ODBC API internamente, proporciona una interfaz de nivel mayor y está basado en un modelo objeto. El Jet DAO se encuentra en los siguientes conjuntos de funciones:

- DDL (Lenguaje de Definición de Datos)
- DML (Lenguaje de Manipulación de Datos)

En la tabla 2.3 se describirán las interfaces y propósitos con los que cuenta el DAO.

INTERFAZ OBJETO DE ACCESO A DATOS	PROPÓSITOS	TIPOS
DBEngine	Es un objeto con un último nivel que corresponde al Jet Engine	DML,DDL
Wokspace	Un contador para BD abiertas que soporta transacciones simultáneas	DML, DDL
Database	Representa el esquema BD, corresponde a un nativo Jet BD, una BD externa, ó una conexión ODBC	DML, DDL
TableDef	Representa la definición de una tabla física de BD	DDL
QueryDef	Define un almacenamiento de consultas ó instrucciones SQL precompiladas, almacenadas en la BD antes que el código	DDL
Recordset	Regresa el resultado de una consulta dentro de una BD	DDL
Field	Representa una columna de datos en una tabla	DDL
Index	Representa un índice almacenado por una tabla	DDL

Parameter	Representa un almacenamiento de consultas con parámetros asociados con una consulta parametrizada.	DDL
User	Define y refuerza la seguridad en la BD	DDL
Group	Una colección de usuarios con privilegios similares	DDL
Relation	Define la relación de registros entre dos ó más tablas	DDL
Property	Representa un almacenamiento de propiedades asociadas a un objeto	DDL
Container	Enumera todos los objetos almacenados en una BD	DDL
Document	Objetos de un tipo común que comparten un contador.	DDL

Tabla 2.3 Interfaz del DAO (Data Access Object)

2.2.2 Lenguaje Perl

¿Que es Perl?

Perl significa: ***Practical Extraction and Report Languaje.***
Es un lenguaje interprete optimizado para explorar arbitrariamente archivos de texto, extrayendo información de estos archivos, e imprimir reportes basados en esta información. Este es un buen lenguaje para varios sistemas manejadores de tareas .El lenguaje intenta ser práctico, rápido de usar, eficiente y completo.

PERL es un lenguaje de programación surgido a inicios de los noventas, que busca antes que nada el facilitar la elaboración de tareas comunes en sistemas tipo UNIX, donde tradicionalmente las tareas de administración y proceso de datos se realiza con herramientas muy rudimentarias y por demás hostiles al usuario o administrador. Pero que se aplican sobre grandes cantidades de información (por lo regular texto) por lo que se requiere que sean de alto rendimiento.

Su autor, Larry Wall realizó *PERL* casi como una obra altruista, de modo que hasta *PERL 5.X* su distribución es gratuita, sin que por eso tenga menos poder o consistencia.

¿Para que sirve PERL?

PERL surgió como una opción para una gran cantidad de herramientas de UNIX en las cuales basa su propia sintaxis, buscando el mínimo sacrificio de su desempeño por una máxima facilidad de programación e integración, sigue la filosofía de mantener un ambiente que sea capaz de detectar y corregir pequeñas omisiones del programador, y de proporcionar una forma abreviada de realizar múltiples tareas.

Anteriormente se hacían múltiples pruebas con diferentes sintaxis y se tenían que manipular múltiples lenguajes para compilar estas tareas, por ejemplo, para procesar un archivo, un administrador de sistemas tenía que escribir un *shell* usando *sh*, procesar un archivo usando *awk* o *grep* y editar el archivo usando *sed*, crear un programa en *C*, compilarlo y debuguearlo directamente. Ahora un administrador puede combinar varias de estas tareas en un simple lenguaje que es rápido de escribir, desarrollar, eficiente y completo como *Perl*.

En un solo lenguaje *PERL* combina algo de *C*, *sed*, *awk* y *sh*. La sintaxis de *PERL* es muy parecida a la del lenguaje *C*. *Perl* usa sofisticadas técnicas de *pattern-matching* (patrones de búsqueda) para buscar gran cantidad de datos muy rápido.
En una palabra, es una utilería que pretende facilitar el proceso de grandes volúmenes de información sin sacrificar el rendimiento.

¿Dónde Puede Usarse PERL?

Las plataformas donde *PERL* se ha desarrollado más son los servidores UNIX, por sus necesidades de administración y lo robusto de su manejo de memoria y de procesos (requisitos de PERL hacia el S.O.) además de la facilidad de *PERL* para realizar los así llamados CGI's, interfaces para comunicar recursos del servidor con un servicio de Internet particular (como podría ser *WWW* o *gopher*).

En otras plataformas, PC en particular, se han desarrollado versiones que mantienen un razonable grado de funcionalidad, pero en realidad, el sistema DOS no tiene un manejo lo bastante bueno de los procesos o de la memoria para permitir a *PERL* dar un buen desempeño, además de que no es común ver en PC necesidades de administración de la magnitud de un servidor institucional. Sin embargo, puede practicarse la programación en *PERL* de PC, o incluso elaborar programas de reporteo en el, sin embargo, es algo que no se ha popularizado hasta hoy.

Filosofía de PERL

Según Larry Wall, autor del lenguaje de programación *PERL*, dice que: *PERL* no establece ninguna filosofía de programación (de hecho, no se puede decir que sea orientado a objetos, modular o estructurado aún cuando soporta directamente todos estos paradigmas), los objetivos que se tuvieron en cuenta al diseñar la sintaxis de *PERL* fueron la facilidad de aprendizaje y de uso y la claridad de código, las cuales, se considera que son necesarias (aunque pueden escribirse programas en *PERL* complejos si así se desea).

Por si fuese poco, *PERL* no es ningún compilador ni un interprete, está en un punto intermedio, cuando se manda a ejecutar un programa en *PERL*, se compila el código fuente a un código intermedio en memoria, se le optimiza (como si fuese a elaborar un programa ejecutable) pero es ejecutado por un motor, como si se tratase de un interprete. El resultado final, es que se utiliza algo que se comporta como un intérprete pero que tiene un rendimiento comparativo al de programas compilados. Sin embargo, ya existen compiladores de Perl con la versión 5

Algunas de las principales facilidades del lenguaje PERL son:

- Permite construir filtros, para seleccionar información desde un archivo cualquiera, empleando expresiones regulares de búsqueda.
- Permite una comunicación interactiva, por medio de comandos ejecutados en el sistema operacional.
- Puede emplearse en cualquier línea de un programa estructurado en cualquier ambiente, sin problemas de ejecución.
- Al igual que cualquier script de los diferentes interpretadores de comandos (Shells), permite almacenar y ejecutar sin problema un script en particular desde el sistema operativo.
- Permite la manipulación de la estructura de cualquier archivo en el ambiente UNIX. Toda la traducción lógica de Oracle Libraries, se estructuró en scripts elaborados en *PERL*, sus resultados fueron excelentes y la traducción fue exitosa.

Como ejecutar Perl

Para ejecutar un programa en *PERL* basta crear un archivo en el sistema operativo con permisos de ejecución e ingresar como primer entrada (primera línea) la siguiente instrucción:

#!/usr/local/bin/perl

donde los caracteres #! indican que los caracteres siguientes representan un interpretador de comandos, la ruta (path) /usr//local/bin determina el directorio donde se encuentra el interpretador de comandos, para el caso del lenguaje *PERL* se denomina perl.

Diferencias entre PERL 4.3 y 5.X, Como elegir versión

Actualmente existen dos versiones altamente populares de Perl, la 4.3 y la 5.0X, de hecho hay diferencias importantes entre una versión y otra. Seguramente surge la duda entre usar una versión vieja y una nueva, por regla general las nuevas versiones son mejores que las anteriores de modo que las opacan en todo sentido, *PERL* no es la excepción a esta regla, el único factor que impide una transición inmediata es que no son 100% compatibles. La versión 5 de *PERL* es una reescritura total que ya incluye un manejo de estructuras abstractas de datos mucho más poderoso, incluso, soporta la orientación a objetos a su manera. De modo que las librerías, por ejemplo para creación de CGI's no funcionan de una función a otra por lo que la migración es poco práctica.

Así pues, la decisión sobre que versión utilizar depende del trabajo que haya sido realizado con anterioridad, si ya se tiene un sistema completo o grande en *PERL 4*, es recomendable mantenerlo en *PERL 4* por el resto de su vida útil, pero para desarrollos nuevos es por mucho, más recomendable iniciarlos con *PERL 5* o en su caso, la versión más reciente que esté disponible. Una tercera opción (que es por mucho la más recomendable si se tienen los recursos) consiste en instalar las dos versiones de modo que convivan en el sistema.

Algunas fuentes de información existentes:

Los libros que son ya clásicos sobre *PERL* son los libros del camello y la llama de la editorial Nutshell, además, existen magníficas introducciones y manuales de referencia que se pueden obtener vía Internet.
Para buscar información:
http://www.yahoo.com/Computers_and_Internet/Languajes/Perl/.
Fuente de información general y referencia de documentos:
http://pubweb.nexor.co.uk/public/perl/perl.html
http://www.khoros.unm.edu:80/staff/neilb/perl/perl5.html
Una buena introducción:
http://www.khoros.unm.edu:80/staff/neilb/perl/introduction/home.html
Documentación:
http://www-cgi.cs.cmu.edu/cgi-bin/perl-man
http://www.perl.com/perl/info/documentation.html

2.2.3 JavaScript

Con el acrecentamiento reciente del interés en Internet, no es de sorprenderse que nuevos conceptos relacionados con esta surjan. En esta nueva era de la conectividad encontramos a diseñadores y programadores que se apresuran a inventar metodologías nuevas para difundir grandes cantidades de información. El *WWW (World Wide Web)* ha sido uno de estos nuevos recursos, cada día nuevos autores del *Web* dejan su marca en Internet.

¿Qué es JavaScript?

JavaScript fue desarrollado por *Netscape*. Es como una derivación de *Java* y contiene un juego más reducido y más simple de comandos que varían ligeramente en su implementación

Netscape en su esfuerzo extiende la funcionalidad de su *browser*, desarrolló un lenguaje de programación que se puede colocar dentro de las páginas del *Web*.
Originalmente lo llamo *LiveScript*, después lo renombró a *JavaScript* con la idea de capitalizar la fama de *Java*, lenguaje desarrollado por *Sun Microsystems*.

La sintaxis y estructura de *JavaScript* son similares a los de *Java*, pero *JavaScript* sólo es funcional cuando se incluye como parte de una página HTML *JavaScript* no puede desarrollar applets o aplicaciones independientes, sólo puede residir dentro de un script HTML y funciona cuando se carga en un navegador compatible como *Netscape 2.0*.

Los scripts se realizan después de eventos específicos accionados por el usuario. Crear documentos Web con *JavaScript* requiere un editor de texto y un navegador compatible. *Netscape Gold* también incluye un editor dentro del navegador mismo, por lo que no es necesario contar con un editor externo.

Aunque no está directamente relacionado con *Java*, *JavaScript* puede interactuar con las propiedades y métodos expuestos de applets *Java* incrustados en una página HTML.
La diferencia se reduce a lo siguiente: los applets de *Java* existen fuera del navegador, mientras que *JavaScript* sólo existe dentro de un navegador.

JavaScript es un lenguaje de guiones

Las aplicaciones de computación no pueden satisfacer las necesidades de cada persona, para compensar esto es común dejar que el usuario personalice un programa por pequeños guiones (scripts), llamados *Snippets del programa*, como por ejemplo en Excel en las macros.

La mayor parte de estos miniprogramas no se compilan, pero se interpretan línea por línea, mientras la aplicación corre. *JavaScript* es un lenguaje de guiones, y su sintaxis se parece mucho a *C*, *C++*, *Pascal* o *Delphi*. *JavaScript* ordena y se introducen en el Web funciones del documento y las etiquetas HTML. Cuando el browser de un usuario carga una página, el browser "corre" el programa y ejecuta los procedimientos que estén indicados en el guión (script).

Debido a que la corrida depende del browser, es necesario recordar que no todos los browsers son capaces de interpretar el *JavaScript*, en general sólo la versión 2.0 o mayor (en versión estándar o gold) e *Internet Explorer 3.0*, soportan *JavaScript*.

JavaScript es basado en Objetos:

Un lenguaje orientado a objetos es el que intenta repartir con un programa una colección de partes individuales (objetos) que realizan cosas diferentes y no como una sucesión de declaraciones que ejecutan una tarea específica. Además de contar con métodos para cada objeto, *JavaScript* es un lenguaje basado en objetos, por ejemplo no se tiene que construir un objeto fecha (date) porque en *JavaScript* ya existe uno.

Manejando eventos:

Cuando se desea que algo pase en una página del Web, que un evento ocurra, por ejemplo apretar un botón cualquiera, arrastrar el ratón, cargar o descargar una página, enviar un formulario, por mencionar algunos, *JavaScript* está hecho para detectarlos y reaccionar de la forma en que el programador desee.

JavaScript es fiable:

JavaScript está diseñado para manipular la información presente en el browser, pero no para recuperar información de otro archivo o guardar información al servidor o a la computadora del cliente, esto significa que no es posible hacer un programa en *JavaScript* que pueda borrar o salvar archivos en la máquina de un usuario.

Independencia de la plataforma:

Es por demás conocido que un programa hecho para una computadora en plataforma *Windows* nunca va a funcionar si se intenta ejecutar en otra con plataforma *UNIX* o Macintosh.

JavaScript sólo se limita por la versión del browser en el que se ejecute, pues éste (el browser) es el que se tiene que atener a la plataforma.

Diferencias entre JavaScript y Java:

Es importante hacer notar que ***JavaScript*** es totalmente diferente de ***Java***.
Este último es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems y necesita una variedad de compiladores y archivos de soporte para funcionar, en cambio *JavaScript* tiene una estructura basada en objetos.

JavaScript toma prestada la mayor parte de la sintaxis de *Java* pero en esencia son diferentes y tienen propósitos distintos.

Java es útil para los desarrolladores que tienen experiencia en programación previa con lenguajes como *C++*.

Los programas desarrollados con el *Java Development Kit* pueden funcionar como aplicaciones independientes y completas o como applets incrustados en páginas HTML. Aunque los *applets* se encuentran incrustados en páginas HTML, llegan a la computadora del cliente como archivos separados.

Los *applets* son programas en binario independientes de la plataforma, significa que, a semejanza de *C* se compila en forma binaria Se puede correr este *applet* en cualquier plataforma con la única restricción de que se tenga un software para operar el applet.

La tabla 2.4 muestra las diferencias entre *Java* y *JavaScript*

JAVA	JAVASCRIPT
Se compila en un archivo binario que se da al cliente (browser) a ejecutar.	En cambio JavaScript pasa un archivo de texto y el browser lo interpreta .
Java (applets) es referenciado desde un documento HTML, pero pasado como archivo diferente.	JavaScript empotra la codificación dentro de un HTML.
Java requiere de la declaración de variables.	JavaScript no requiere de la declaración de variables.
La etiqueta para llamar a un applet es: <APPLET>	La etiqueta para declarar un código es : <SCRIPT>
Las referencias a los objetos se verifican cuando se compila.	Las referencias del objeto se verifican en tiempo de corrida.

Tabla 2.4 Diferencias entre Java y JavaScript

Uso de JavaScript:

- Es apropiado para manejar la actividad en el extremo del cliente.
- Esta diseñado para programas sencillos y pequeños.
- Hace posible programar respuestas a requerimientos de usuario.
- Es adecuado para implementar un sistema de ayuda

Ventajas:

- Desarrollo rápido.
- Fácil de aprender.
- Independencia de plataforma
- Gastos mínimos

Desventajas:

- Rango limitado de métodos integrados
- No cuenta con ocultación de código
- No permite impresión de su salida.

Términos generales en JavaScript

- **Manejadores de eventos**: Los manejadores de eventos son un elemento especial de *Javascript* y le dan gran parte de su poder Permiten que el programador busque un comportamiento específico en relación con la página HTML, como hacer clic en un botón de formulario o mover y colocar el puntero del ratón sobre un ancla.

  Los manejadores de evento están incrustados en etiquetas HTML que, por lo general, se usan como parte de los formularios, pero también se incluyen como parte de algunas anclas y vínculos. Virtualmente cualquier cosa que un usuario pueda hacer para interactuar con una página se cubre con los manejadores de evento, desde mover el ratón hasta abandonar la página actual.

- **Funciones**: Una función es un método definido por el usuario o integrado que realiza una tarea, puede dar como resultado un valor cuando se le usa con el enunciado return. Las funciones son universales y no tienen que estar asociadas con un objeto para ejecutarse, mientras que los métodos están integrados con los objetos.

  Como regla general, es mejor colocar las definiciones de función dentro de las etiquetas <HEAD> de un documento. Esta práctica asegura que todas las funciones estén cargadas y listas para usarse antes de que el usuario tenga la oportunidad de interactuar con el resto de la página.

- **Jerarquías**: En una Jerarquía, los objetos existen en una determinada relación entre unos y otros. Por ejemplo, los objetos *Navigator* tiene una estructura que refleja la construcción de una página HTML. A esto se le llama jerarquía de instancias porque sólo funciona con instancias específicas de objetos en lugar de funcionar con clases generales.

  El objeto *window* es el padre de todos los demás objetos *Navigator* Debajo del objeto *window*, los objetos *location*, *history* y *document* comparten todos la precedencia. Bajo *document* se encuentran otros objetos como *forms*, *links* y *anchors*.
  Cada objeto es un descendiente de un objeto más alto. Un formulario llamado *orderForm* es un objeto y también es una propiedad de *document*. Como tal, se le llama *document.orderForm*

  Otra forma de ver una jerarquía es pensando en la relación que los elementos del mundo real tienen entre sí. Frenos, manubrio y pedales son todos objetos que pertenecen a una bicicleta. Una bicicleta es un objeto que pertenece al transporte terrestre. El transporte terrestre es un objeto que pertenece a las formas de viajar.

  Si se representan como objetos JavaScript, estas relaciones se expresarían de esta manera:

  formaViajar transporteTerrestre bicicleta.manubrio

El objeto más alto y menos específico se encuentra a la izquierda y gana especificidad conforme se mueve a la derecha y sus descendientes comienzan a ramificarse.

- **Literales:** Una literal es un valor que puede asignarse a una variable. Las literales son lo que son y no cambian.
- En *JavaScript* varios tipos de literales corresponden a tipos de variables.

- **Enteros:** Los enteros son números como 1,16 y 25,896. Se pueden expresar en forma decimal, hexadecimal u octal.

- **Números de punto flotante:** Los números de punto flotante son porciones fraccionarias de enteros y deben incluir por lo menos un dígito y un punto decimal o símbolo de exponente ("e" o "E").

- **Literales booleanas:** Las literales booleanas sólo tienen valores, verdadero o falso. En algunas implementaciones de JavaScript, 0 (falso) y 1 (verdadero) no pueden usarse como sustitutos de valores booleanos. Las versiones actuales de Netscape Navigator y Gold dan soporte al 0 y al 1 como falso y verdadero respectivamente.

- **Cadenas:** Las cadenas están definidas por cualquier cantidad de caracteres dentro de comillas dobles o sencillas. El uso de la diagonal inversa "\" permite imprimir caracteres especiales.

- **Métodos:** Un método es una función asignada a un objeto. Por ejemplo, userName.toUpperCase() da como resultado una versión en mayúsculas de la cadena contenida en userName.

- **Objetos:** Un objeto es una construcción con propiedades que son variables *JavaScript* u otros objetos. Las funciones asociadas con un objeto se conocen como métodos de objeto.

- **Operadores:** Un operador realiza una función sobre uno o más operandos o variables. Los operadores se dividen en dos clases básicas: binarios y unarios. Los operadores binarios necesitan dos operandos y los operadores unarios necesitan un solo operando. Por ejemplo, la adición es un operando binario

```
sum=1+5
```

Los operandos unarios se usan con frecuencia para actualizar contadores
El siguiente ejemplo incrementa la variable counter en 1

```
counter++
```

- **Propiedades:** Las propiedades se emplean para describir un objeto o su valor actual. Una propiedad se define asignándole un valor. El valor puede ser asignado por el navegador, el programa o conforme el usuario interactúa con la página.

  Varias propiedades en *JavaScript* contienen constantes (valores que nunca cambian). Estos son elementos tales como el valor de Pi o las constantes de Euler (E)

  Otros elementos cambian de una página a otra, pero no pueden ser modificados como los elementos de formulario.

- **Scripts:** Uno o más comandos *JavaScript* pueden ser encerrados en una etiqueta <SCRIPT>. La aparición de varios lenguajes de creación de scripts ha hecho necesario identificar ante el navegador el lenguaje que se esta empleando. Para JavaScript la sintaxis es:

  ```
  <SCRIPT LANGUAGE="JavaScript">
  <! --
  ...Enunciados...
  // -- >
  </SCRIPT>
  ```

  El empleo del atributo LANGUAGE sigue siendo opcional en los navegadores Netscape. El empleo de las etiquetas de comentario HTML, <!-- Y -- >. Si la página que contiene el script se usa en un navegador que no es compatible con los lenguajes de creación de scripts, los enunciados de script se despliegan como si fueran cualquier texto en la página, agregando basura a la pantalla.

  Si se usan las etiquetas de comentario, un navegador no compatible ignora la porción de *scripts* del documento. Las diagonales dobles que preceden a la etiqueta final de comentario HTML aseguran que la etiqueta no se confundirá con un enunciado JavaScript.

2.2.4 HTML (Lenguaje de Hipertexto)

Es un protocolo con formato texto, que se utiliza para definir una página en el World Wide Web. HTML está basado en el lenguaje de marca generalizada en la SGML, el cual es una norma ISO. Como un conjunto de las normas ISO, cualquier aplicación que pueda *interpretar el formato* SGML puede también leer el formato HTML.

HTML está basado en el código formateado en un documento que puede servir como ajuste de fonts, crear listas bulleted, desplegado de imágenes o alguna otra página formateada.

Utilizando el web

El World Wide Web estándar está basado en dos básicas tecnologías: El HTTP (Protocolo de Transferencia de Hipertexto) y el HTML (Lenguaje Marcador de Hipertexto). HTTP proporciona los medios para mover documentos del servidor al cliente. HTML proporciona el formato de controles para documentos Web

El servidor de red puede postformatear páginas HTML sin tener un vendedor o sistema especial con formato de código específico.

Algunos browser que se pueden manejar para correr las páginas HTML, son los siguientes:

- NCSA Mosaic: Es el precursor de los browser.
- Microsoft Explorer: La Empresa más grande de Software para PC's.
- Netscape Comunicator Pro: Es una de las herramientas más valiosas para navegar por Internet, ya que se pueden manejar todos los comandos de HTML, debido a que es uno de los más actuales.

Estructura básica html:

Los códigos de formato de documento pueden dividirse dentro de 2 grupos básicos: *propiedades del documento*; las cuales contienen el título y otras cosas a cerca del documento como un todo, *y propiedades formato*, las cuales especifican la distribución del texto e imágenes dentro del documento.

A continuación se nombrarán los parámetros que aparecen en un encabezado de un documento para identificarlo y saber su utilidad dentro del sistema de un cliente.

<HEAD>
<TITLE> Partes de un Catálogo </TITLE>
</HEAD>

Las propiedades de documento son identificados por corchetes de ángulos (<>), el cual encierra el nombre de la propiedad. Además existe un identificador de cierre para la propiedad, el cual debe estar precedido por una diagonal (/), por ejemplo:

<TITLE> Identificador para abrir una propiedad
</TITLE> Identificador para cerrar una propiedad

Un documento HTML comienza con la etiqueta <HTML>, y termina con </HTML>. Dentro del documento de HTML (entre las etiquetas de principio y fin de html) existen dos partes importantes para el programa uno de estos es para el encabezamiento, delimitado por <HEAD> y </HEAD>, que sirve para definir diversos valores válidos en todo el documento; otro es para el cuerpo del documento, delimitado por <BODY> y </BODY>, donde reside la información del documento.

La utilidad de la directiva <TITLE>, en el encabezamiento, es que permite especificar el título del documento HTML. Este título no forma parte del documento en sí, ya que sirve como título de la ventana del programa mostrándolo en minúsculas.

Por ejemplo: <TITLE>Tema de HTML</TITLE>

Obsérvese que el título de este texto se ha escrito con algunas mayúsculas, para distinguirlo del título global del documento.

El cuerpo de un documento HTML contiene el texto que con la presentación y los efectos que se determinen, se presentará ante el hiper lector. Dentro del cuerpo son aplicables los efectos que se especifican exclusivamente a través de directivas. Esto quiere decir que los espacios, tabulaciones y retornos de carro que se introduzcan en el fichero fuente no tienen ningún efecto a la hora de la presentación final del documento.

En resumen, una estructura básica de un documento HTML quedará de la siguiente forma:

<HTML>
<HEAD>
<TITLE>Título</TITLE>
</HEAD>
<BODY>
Texto del documento, menciones a gráficos, enlaces, etc.
</BODY>
</HTML>

Ver en la figura 2.10, la representación de los comandos antes mencionados.

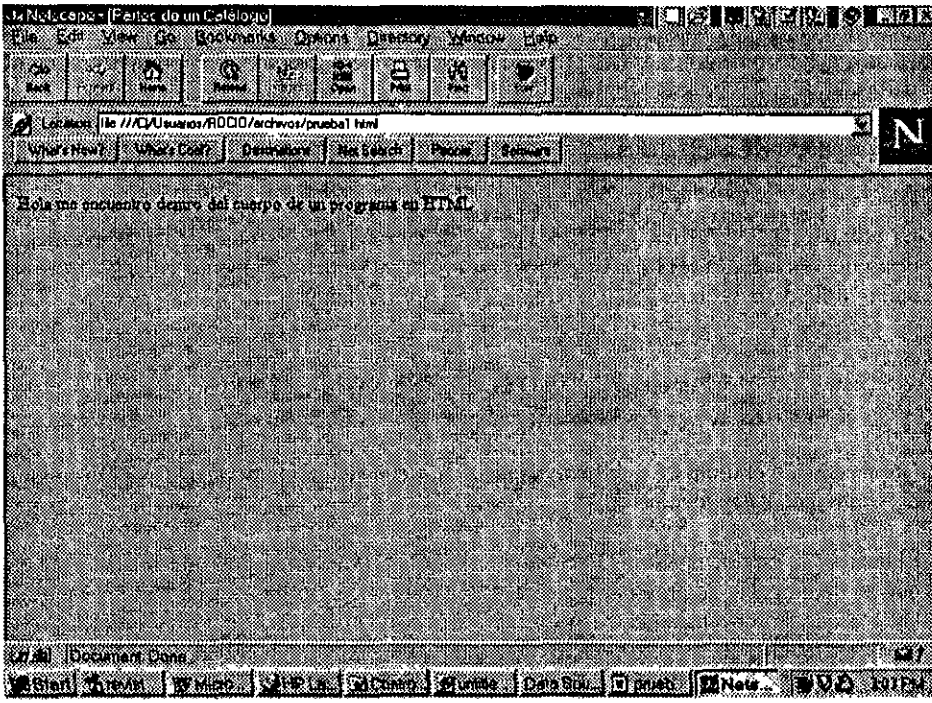


Figura 2.10 Página desarrollada en HTML

Utilizando editores de hipertexto

Para producir documentos de hipertexto, se utilizará cualquier editor para escribir el texto ASCII, algunos de estos editores especiales permiten revisar errores en los documentos, a continuación se explicarán algunos editores:

1. Hotmetal:
Este editor se encuentra entre 2 versiones: una versión de dominio público, y otra versión comercial. Es uno de los mejores editores de HTML disponible, HOTMETAL permite ver los documentos y provee una imagen cercana a la WYSIWYG al editar.

2. Asistente HTML:
El asistente HTML es un editor que permite utilizar métodos de punto y click cuando se está creando un documento HTML. Este soporta el pasar un documento HTML en proceso a un browser local para ver y cargar información URL de Mosaic para utilizarse en el documento local.

El editor asistente HTML tiene una barra de herramientas que permite crear documentos HTML más fácilmente.

3. Plantillas de word para Windows:
Sólo unas pocas plantillas de documentos de MSWord para Windows (versiones 2 y 6) existen para editar documentos HTML. Estas plantillas permiten muchas de las características que proveen otros editores de HTML, así como la capacidad de revisar la ortografía del documento. A continuación se muestran algunas plantillas con sus características.

Plantillas	Descripción
CU_HTML.DOT	Permite imágenes en línea para ser proyectadas durante la edición.
GT_HTML.DOT	Proporciona el editor WYSIWYG de HTML
HTML for Word	Plantillas con macros que proporcionan un ambiente para estructurar correctos documentos HTML y SGML

4. Asistente de internet:
Microsoft ha liberado la aplicación de asistente para internet, también basado en Word para Windows Este asistente utiliza las capacidades OLE de Word para Windows versión 6.0a

Estilos y efectos básicos

Primero se presenta el texto original HTML es decir, lo que se edita, y después se presenta el efecto que dicho texto fuente produce al interpretarse y representarse por el programa adecuado.

Títulos:
Mediante los títulos, en sus diferentes niveles de importancia, se puede definir el esqueleto del documento, por ejemplo·

```
<h1>Grande</h1>
<h2>Menos Grande</h2>
<h3>Pequeño</h3>
```

La estructura básica de los tamaños de letra se ven reflejados en la figura 2.11.

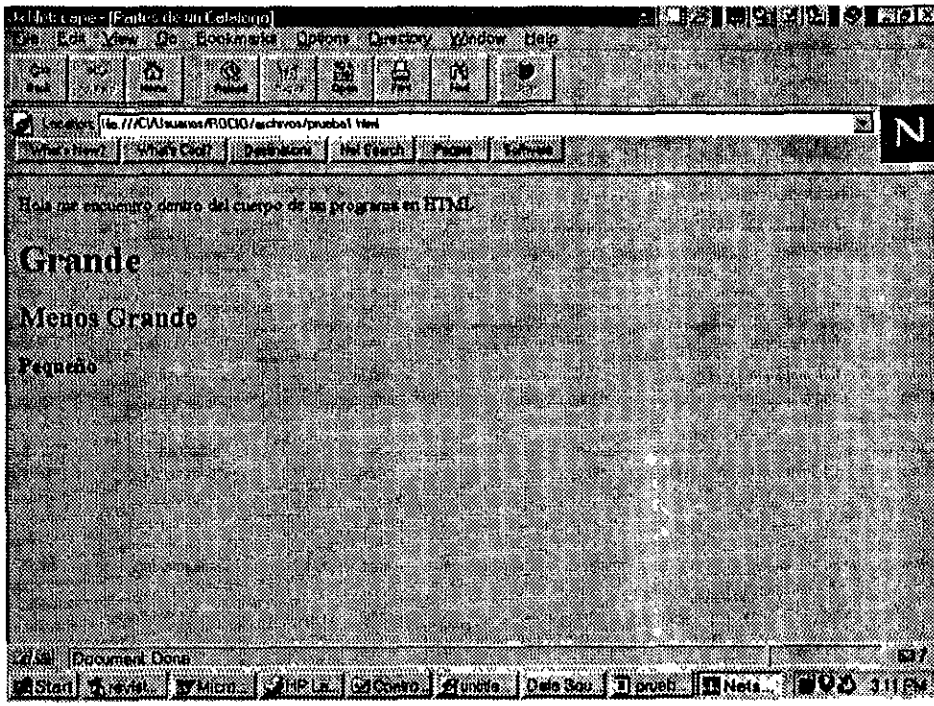


Fig. 2.11 Página en HTML que muestra los tamaños de letra

Atributos del texto:

Los atributos determinan el estilo y el tipo de letra que tendrá la presentación del documento final. Algunos de estos atributos se listan a continuación:

- Para definir un párrafo de forma normal no es necesario poner ninguna etiqueta. Al presentar el documento se hace caso omiso de los espacios, tabulaciones y retornos de carro que se encuentren en el texto fuente. Por ello cuando se quiera forzar un final de línea es necesario utilizar dos comandos especiales: <P> para marcar un fin de párrafo, y
 para un retorno de carro. La diferencia entre ambas es que la separación de líneas que provoca <P>, es algo mayor que la de
 para que los párrafos se distingan bien entre sí. Las dos etiquetas mencionadas se sitúan en el lugar donde se quiere poner la separación.

- El texto preformateado (o etiqueta <PRE>) permite respetar en la presentación final del documento los espacios y retornos de carro que se hayan puesto en el texto.

- Para hacer una cita textual dentro del documento, se puede utilizar la etiqueta <BLOCKQUOTE>·

  <BLOCKQUOTE>Muchos años después, frente al pelotón de fusilamiento,
  el coronel Aureliano Buendía había de recordar aquella tarde remota
  en que su padre lo llevó a conocer el hielo.
  (Gabriel García Márquez, Cien años de soledad)</BLOCKQUOTE>

- Las direcciones de correo electrónico se suelen marcar con la siguiente etiqueta:

  <ADDRESS>Dirección: webmaster@. </ADDRESS>

  Que se presentará de la siguiente manera:
  Dirección: webmaster@..

- Se pueden establecer los atributos negrita y cursiva de la siguiente manera

  <B>**Esto en negrita**</B> y <I>*esto en cursiva*</I>

- Para centrar texto o gráfico(o cualquier cosa) se usa la etiqueta <CENTER>

  <CENTER>Verde que te quiero verde</CENTER>

En la figura 2.12 se muestra el resultado al utilizar los comandos anteriores:

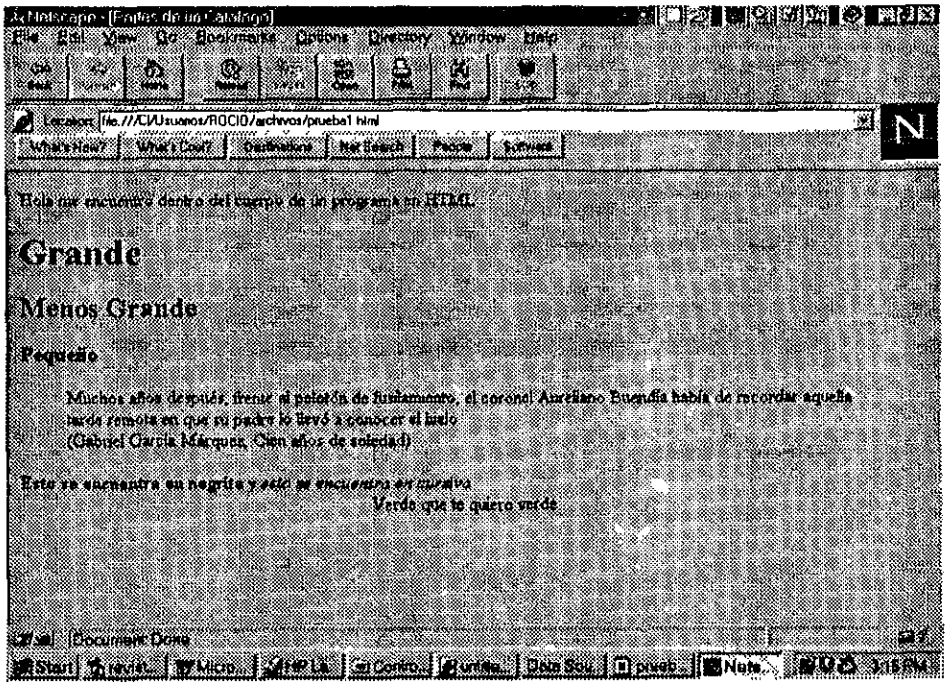


Fig. 2.12 Página que muestra algunos atributos de texto en HTML

Listas:

Existen tres tipos distintos de listas, numeradas, no numeradas y listas de definiciones (glosarios), las cuales contienen una presentación diferente·

Las listas se pueden anidar, es decir, se podrá poner una lista dentro de otra lista que no sea del mismo tipo. A continuación se muestra las sintaxis para cada tipo de lista:

- Esto es una lista no numerada:
  <UL>
  <LI>Tomates
  <LI>Zanahorias
  <LI>Perros
  </UL>

- Esto es una lista numerada:

```
<OL>
<LI>Miguel Induráin
<LI>Tony Rominger
<LI>Eugeni Berzin
</OL>
```

Este tipo de lista se desplegará de la siguiente manera:

1.Miguel Induráin 2.Tony Rominger 3 Eugeni Berzin

- Un glosario está formado por una serie de parejas de término (marcado con <DT> al principio de línea) y definición (con <DD>).

```
<DL>
<DT>Perro (<I>n masc.</I>)
<DD>Animal de cuatro patas que ladra.
<DT>Gato (<I>n masc </I>)
<DD>Animal de cuatro patas que maúlla y se lleva muy mal con el
perro.
<DT>Pescadilla (<I>n. fem.</I>)
<DD>Animal que vive en el mar y está recubierto de escamas
</DL>
```

- La etiqueta <HR> permite colocar una línea horizontal en un documento para especificar una separación.

- Para especificar un comentario en un documento HTML, es decir, una etiqueta que no aparecerá en la presentación final del documento, se encierra el texto que formará el comentario entre los símbolos "<!--" y "-->". Por ejemplo:

<!-- Esto es un comentario -->

Ver la figura 2.13, la cual mostrará los atributos anteriormente mencionados.

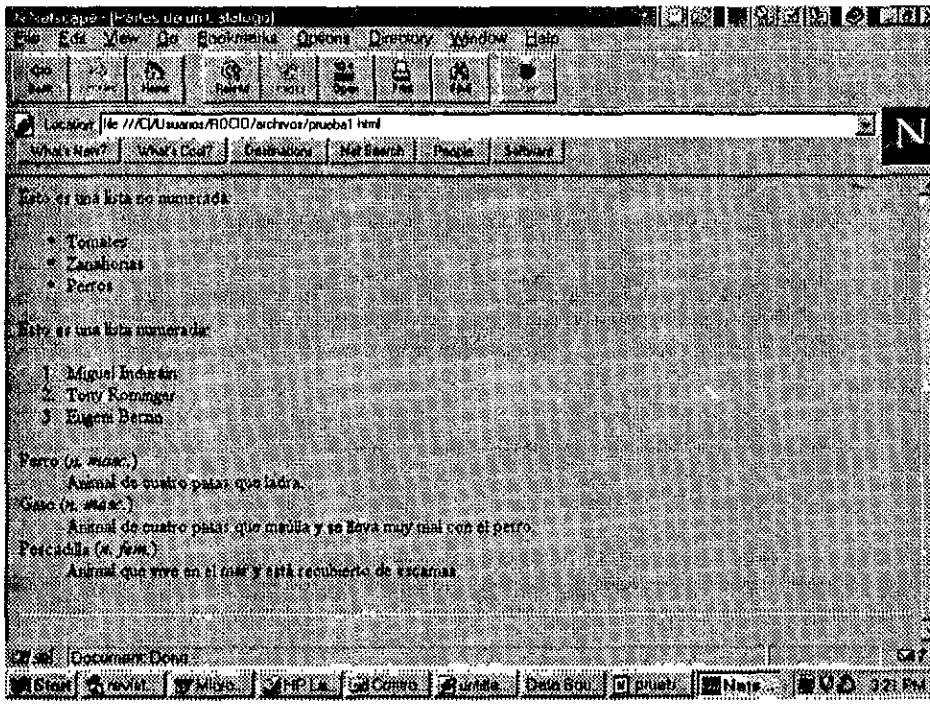


Fig. 2.13 Página que muestra los tipos de listas en HTML

Enlaces y gráficos

Además de los muchos estilos y capacidades de presentación que ofrece HTML para estructurar el documento en sí, además proporciona varias etiquetas que permitirán definir relaciones entre diferentes documentos y estructurar todo un conjunto de documentos para crear una unidad lógica. La facilidad para definir este tipo de enlaces es una de las razones de la potencia y versatilidad de HTML. Por la similitud que tienen los enlaces y los gráficos, a continuación se dará una explicación de cómo pueden incluirse estos últimos en un documento.

Enlaces:

¿Cómo se introducen enlaces en un documento HTML?. Para definir un enlace es necesario marcar con la etiqueta <A> el objeto del cual va a partir dicho enlace. Dicha etiqueta debe incluir el parámetro href="URL" para especificar el destino del enlace. Es decir, que antes del objeto elegido debemos abrir con <A href="URL">, y después cerrar con </A>. Por ejemplo, si se desea que un fragmento de texto se enlace para visitar al Instituto permitirá accesar a la home page del Instituto, se debe escribir en el texto HTML:

<A href="http://pumas.iingen.unam.mx/">Pulse aquí para visitar al Instituto</A>

El resultado que se desplegará en la presentación final será:

Pulse aquí para visitar al Instituto

Por lo general, no se va muy lejos sino sencillamente se enlaza con otro documento que se encuentra en el mismo servidor, e incluso en el mismo subdirectorio. En este caso no es necesario escribir el camino completo al destino del enlace, sino que basta con dar la mínima información imprescindible Basta con poner el nombre:

<A href="el_otro_camino">pulse aquí</A>.

Si se encuentra en otro subdirectorio del mismo servidor, es suficiente con poner:

<A href="/la/ruta/que/sea/fichero.html">pulse aquí</A>.

También pueden utilizarse rutas relativas:

<A href="ruta/relativa/cosa html">cosa</A>

Los enlaces en HTML se expresan con la etiqueta Ancla <A>, el objeto que permitirá el enlace podrá ser un gráfico o un fragmento de texto. Al ponerse una ancla <A> en una imagen, se podrá pulsar con el ratón sobre dicha imagen en el documento final, se brincará al objeto referenciado en el enlace, que puede ser otro documento, un vídeo musical, o un servidor de información meteorológica

Gráficos:

Para incluir un gráfico en un documento HTML se utiliza la etiqueta <IMG>. En dicha etiqueta debe incluirse el parámetro SRC="URL", el cual indica dónde está el camino con el gráfico concreto que se desea para algún documento.

Existe alguna limitación respecto a los formatos gráficos que los programas lectores de HTML puede interpretar sin problemas El formato fundamental es el GIF, que cualquier programa con capacidades gráficas debería poder mostrar directamente en el texto (Mosaic

y Netscape pueden hacerlo) Si se utiliza otro formato diferente, lo más probable es que cuando un lector esté accediendo al documento, el programa no comprenda ese formato y se tenga que solicitar la ayuda de otro programa, con lo cual al final el gráfico no se insertará en el lugar estratégico del documento, sino que aparecerá en otra ventana diferente.

Por lo general no es necesario escribir el URL completo, sino que basta con dar la mínima información como ocurre con los enlaces.

Se puede incluir también un dibujo que esté en otro lugar especificando un URL completo, por ejemplo:

<IMG src="http://no_se_nada_de_ti/images/nada-nueva.gif"><P>
<Picture>

Además se puede hacer que un gráfico sea un enlace, utilizando la etiqueta <A>. En este caso no se debe olvidar utilizar la opción alt="texto" para que los usuarios puedan seguir el enlace:

<A href="http://www.nada.deti/"><img src="http://no_se_nada_de_ti/images/nada-nueva.gif" ALT="NADA"></A><P>
<Picture: NUEVA>

El URL:

Para especificar de manera uniforme el objeto al que apunta el enlace, se utiliza una forma estandarizada que se denomina URL (Uniform Resource Locator, es decir, Localizador Uniforme de Recursos). Un URL está formado de la siguiente manera:

protocolo://maquina/ruta.

La función del URL es que identificará el tipo de servicio que va a proporcionar en el destino del enlace. Los esquemas más frecuentes son ftp, gopher o telnet. La razón de los esquemas es que el WWW unifique el acceso a servicios de información que previamente eran incompatibles entre sí. El esquema más utilizado es http, correspondiente al propio WWW (es decir, que cualquier referencia a un documento HTML debería comenzar con http://).

La máquina y la ruta sirven para localizar el objeto al que apunta el enlace. La máquina es la identificación del servidor en el cual está situado el objeto al que apunta el enlace. Puede ser simplemente el nombre de la máquina o también el nombre y el puerto.

La ruta es el nombre del fichero que contiene el documento en concreto, incluyendo el nombre del subdirectorio donde se encuentra La ruta completa del archivo se deberá separar con la barra inclinada hacia la derecha (/), como se hace en UNIX. La razón de este

convenio es precisamente que la mayor parte de los servidores de WWW que hay en Internet son computadoras basadas en UNIX, debido a la gran superioridad tecnológica de este sistema sobre MS-DOS. Esto se nota también en que por lo general los nombres de las rutas no tienen muchas limitaciones: pueden ser casi tan largos como se quiera, contener varios puntos, etc.. Se debe tener en cuenta que en UNIX las mayúsculas y las minúsculas en los nombres de las rutas son diferentes· no es igual RUTA que ruta..

La información que se puede encontrar en un servidor de WWW se distribuye a partir del directorio raíz en distintos subdirectorios y archivos. Una norma relativa al nombre de las rutas es hacer que los archivos que contengan documentos HTML terminen en **.html.**

Caracteres especiales

Existen también ciertas limitaciones relativas al uso de ciertos símbolos en HTML, como el de menor que (<) o el signo inglés de and (llamado ampersand: &).

Existe una razón evidente que impide que se puedan escribir ciertos símbolos directamente en un texto HTML, dichos símbolos tienen un significado en HTML, y es necesario diferenciar claramente cuándo poseen ese significado y cuándo se quiere que aparezcan literalmente en el documento final. Por ejemplo <, que indica el comienzo de una etiqueta, y, por ello, si se desea que aparezca en el texto como tal, se debe escribir algo que no de lugar a confusión, en este caso *&lt.* Los símbolos afectados por esta limitación, y la forma de escribirlos, se detallan a continuación:

•< (Menor que): <
•> (Mayor que): >
•& (símbolo de and, o ampersand): &amp,
•" (comillas dobles): &quot.

Es decir, para escribir <"> en el texto HTML original se debe poner:

<"&gt

El otro caso especial se da cuando en un texto HTML se quiere escribir una "ñ", por ejemplo. Las entidades comienzan siempre con el símbolo &, y terminan con un punto y coma (;). Entre estas entidades va un identificador del caracter que se desea escribir. Las entidades necesarias en el idioma español son:

•á: á
•é: é
•í· &iacute,
•ó: ó

•ú: ú
•Á: Á
•É: É
•Í: Í
•Ó: Ó
•Ú: Ú
•ü: ü
•Ü: Ü
•ñ: ñ
•Ñ: Ñ
•¿: ¿
•¡: ¡

Como se puede ver, las vocales acentuadas se identifican añadiendo el sufijo "acute" a la vocal sin acentuar (puesto que se trata de un acento agudo). Para la *u* con diéresis y la eñe se usa *u* tras una *u* y tilde detrás una *n*, respectivamente. La equivalencia de los signos de abrir interrogación y exclamación es algo más oscura: a falta de una denominación más evidente, tenemos que usar el valor numérico de dichos caracteres en el código estándar latín1 (ISO-8859-1). Esto se puede hacer con cualquier otro carácter del código latín1, que es el código de caracteres básico en HTML, escribiendo &#número;.

EL SERVIDOR

CAPITULO 3
EL SERVIDOR

3.1 Introducción a las Bases de Datos.

Una base de datos es una colección integrada de datos, los cuales se pueden almacenar, pero sin redundancia[1] perjudicial para servir a una o más aplicaciones. Las bases de datos se pueden clasificar u ordenar para la búsqueda de información, ya que facilitan al usuario o usuarios el manejo de ésta.

Las bases de datos dentro de las organizaciones son necesarias para:

- Organizar la información
- Hacer consistentes[1] los datos
- Optimizar los accesos a la información
- Generar estándares
- Integrar[2] los datos
- Controlar la redundancia[3]
- Facilitar la programación
- Asegurar los datos
- Reducir el mantenimiento al sistema
- Independizar los programas de la base de datos

Arquitectura de *una base de datos*

Para el desarrollo de una BD el programador debe entender claramente los diferentes niveles de abstracción que constituyen el Sistema Administrador de Base de Datos (SMBD) con el fin de facilitar su tarea. A continuación se presentan dichos niveles:

- **Nivel físico**: El nivel más bajo de abstracción describe cómo se almacenan realmente los datos . En el nivel físico, se describe en detalle las estructuras de datos complejas del nivel bajo.

[1] Ver definición en el glosario de términos

[2] ibídem
[3] ibídem

- **Nivel conceptual**: Es el siguiente nivel más alto de abstracción describe qué datos son realmente almacenados en la base de datos y las relaciones que existen entre los datos Aquí se describe la base de datos completa en términos de un número pequeño de estructuras relativamente sencillas. Aunque la implantación de las estructuras sencillas en el nivel conceptual puede implicar estructuras complejas del nivel físico, el usuario del nivel conceptual no necesita darse cuenta de esto. El nivel conceptual de abstracción lo usan los administradores de bases de datos, quienes deben decidir qué información se va a guardar en la base de datos.

- **Nivel de visión**: Es el nivel más alto de abstracción describe sólo parte de la base de datos completa. A pesar del uso de estructuras más sencillas en el nivel conceptual, permanece algo de complejidad debido al gran tamaño de la base de datos no se interesarán por toda esta información. En cambio, dichos usuarios sólo necesitan una parte de la base de datos. Para simplificar su interacción con el sistema, se define el nivel de abstracción de visión. El sistema puede proporcionar muchas visiones para la misma base de datos.

En la figura 3.1 se muestran los niveles de abstracción:

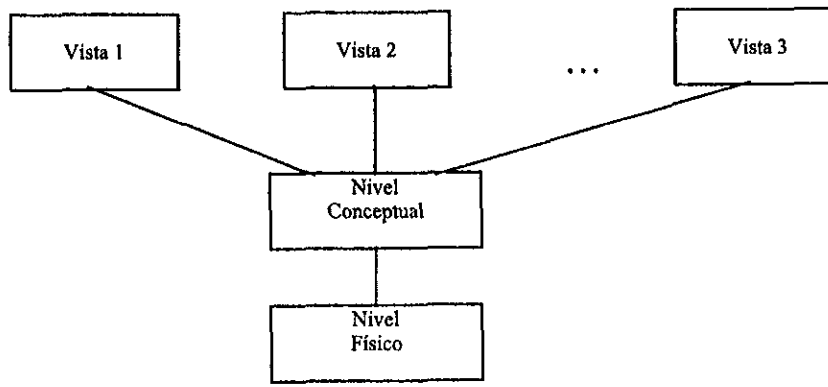


Fig. 3.1 Niveles de Abstracción

¿Qué es un Sistema de Bases de Datos?

Un sistema de Bases de datos consiste en una colección de datos inter-relacionados y un conjunto de programas que permiten al usuario acceder y modificar esos datos. El objetivo primordial es proporcionar a los usuarios un entorno que sea a la vez conveniente y eficiente[4] para ser utilizado al extraer y almacenar información de la base de datos.

Modelo de datos

Un modelo de datos es una colección de herramientas conceptuales para describir datos, relaciones entre ellos, semántica asociada a los datos y restricciones de consistencia . Los diversos modelos de datos que se han propuesto se dividen en tres grupos: modelos lógicos basados en objetos, modelos lógicos basados en registros y modelos físicos de datos. A continuación se explicarán cada uno de estos modelos:

- **Modelos lógicos basados en objetos**: Se usan para describir datos en los niveles conceptual y de visión. Se caracterizan por proporcionar capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente. Hay muchos modelos, pero los más conocidos son: El modelo entidad-relación, el modelo orientado a objetos, el modelo binario, el modelo semántico de datos, el modelo infológico, y el modelo funcional de datos.

A continuación se dará una breve explicación del modelo entidad-relación y el modelo orientado a objetos. Tales temas se definirán con más detalle posteriormente:

⇒ **El modelo Entidad-Relación:** Consiste en una colección de objetos básicos llamados entidades y relaciones entre estos objetos. Una entidad es un objeto que es distinguible de otros objetos por medio de un conjunto específico de atributos. Una relación es una asociación entre varias entidades. Este modelo ha ganado aceptación en el diseño de bases de datos y se utiliza de manera mayoritaria.

El modelo *Entidad-Relación representa ciertas restricciones* a las que se deben ajustar los contenidos de una base de datos. Una restricción importante es la de cardinalidad de asignación, que expresa el número de entidades a las que se puede asociar otra entidad mediante un conjunto de relaciones.

⇒ **El modelo orientado a objetos:** Se basa en una colección de objetos, donde un objeto contiene valores almacenados en variables de instancia. A diferencia de los modelos orientados a registros, estos valores son objetos por sí mismos. Así, los objetos contienen objetos a un nivel de anidamiento de profundidad *arbitraria*. Un objeto también contiene partes de código que operan sobre el objeto. Estas partes se llaman métodos.

[4] Ver definición en el glosario de términos

Los objetos que contienen los mismos tipos de valores y los mismos métodos se agrupan en clases. Una clase es parecida al concepto de tipos de datos abstractos en lenguajes de programación.

La única forma en la que un objeto puede acceder a los datos de otro objeto es invocado a un método de ese otro objeto. Esto se llama envío de un mensaje al objeto. Así, la interfaz de llamada de los métodos de un objeto define su parte visible externamente. La parte interna del objeto, las variables de instancia y el código de método, no son visibles externamente.
A diferencia de las entidades en el modelo E-R, cada objeto tiene su propia identidad única independiente de los valores que contiene. Así, dos objetos que contienen los mismos valores son, sin embargo, distintos. La distinción entre objetos individuales se mantiene en el nivel físico por medio de identificadores de objeto.

- **Modelos lógicos basados en registros**: Se llaman así porque la base de datos está estructurada en registros de formato fijo de varios tipos. Cada tipo de registros define un número fijo de campos, o atributos, y cada campo normalmente es de longitud fija; se usan para especificar la estructura lógica global de la base de datos y para proporcionar una descripción a nivel más alto de la implantación. Los tres modelos de datos más aceptados dentro de este modelo lógico son: el modelo relacional, el modelo de red y el modelo jerárquico.

A continuación se explicarán brevemente cada uno de éstos:

⇒ **Modelo relacional**: Representa los datos y las relaciones entre los datos mediante una colección de tablas, cada una de las cuales tiene un número de columnas con nombres únicos.

⇒ **Modelo de red**: Los datos en este modelo se representan mediante colecciones de registros y las relaciones entre los datos se representan mediante enlaces, los cuales pueden verse como punteros. Los registros en la base de datos se organizan como colecciones de grafos arbitrarios.

⇒ **Modelo Jerárquico**: Se asemeja con el modelo de red en el sentido de que los datos y las relaciones entre los datos se representan mediante registros y enlaces, respectivamente. Se diferencia del modelo de red en que los registros están organizados como colecciones de árboles en vez de gráfos arbitrarios.

Nota: Los modelos relacionales se diferencian de los modelos de red y jerárquico en que no usan punteros o enlaces. En cambio, el modelo relacional conecta registros mediante los valores que estos contienen. Esta libertad del uso de punteros permite que se defina una base matemática formal.

- **Modelos físicos de datos**: Se usan para describir datos en el nivel más bajo. A diferencia de los modelos lógicos de datos, hay muy pocos modelos físicos de datos en uso, dos de éstos son: modelo unificador y modelo de elementos

Estos sistemas están diseñados para gestionar grandes bloques de información. La gestión de datos implica tanto la definición de estructuras para el almacenamiento de información como la provisión de mecanismos para la gestión de la información. Además, estos sistemas deben mantener la seguridad[5] de la información almacenada, pese a caídas del sistema o intentos de accesos no autorizados. Si los datos van a ser compartidos por varios usuarios, el sistema debe evitar posibles resultados anómalos.

Modelo Entidad-Relación

En este apartado se describirá más a detalle lo que es un modelo Entidad-Relación.
Un modelo **Entidad-Relación** permite diseñar un fenómeno real para un sistema de base de datos, a través de diagramas con entidades y sus relaciones. Se explicarán algunos conceptos básicos dentro de este modelo:

Una **entidad** es un elemento u objeto que contiene ciertos atributos que lo caracterizan, como elemento básico que se desea administrar, guardándose información del mismo. Además, una entidad puede ser abstracta o concreta, puede diferenciarse de otros elementos.

Algunas de sus características que deben cumplir las entidades son:
- debe tener existencia
- debe ser un concepto genérico
- debe contar con propiedades de interés

Una **relación** es la interacción o enlace entre dos o mas entidades y corresponde al número de ocurrencias que pueden enlazarse de una entidad a otra.

Los ***tipos de relaciones*** nos determinan la relación que existe entre las entidades:
- Una a Una
- Una a Muchas
- Muchas a Una
- Muchas a Muchas

Un **atributo** es un valor descriptivo o propiedad asociada a una entidad, el cual posee un rango de validación y una estructura de datos asociada.

[5] Ver definición en el glosario de términos

La estructura lógica global de una base de datos puede expresarse gráficamente por medio de un diagrama E-R:

Diagrama Entidad-Relación:

Para un **diagrama de Entidad-Relación** como se muestra en la figura 3.2 es necesario tomar en cuenta los siguientes componentes:

- Rectángulo: determinan los conjuntos de entidades,
- Elipses: determinan los atributos,
- Rombos: determinan las relaciones entre conjuntos de entidades y,
- Líneas: determinan las conexiones de atributos a conjuntos de entidades y conjuntos de entidades a relaciones.

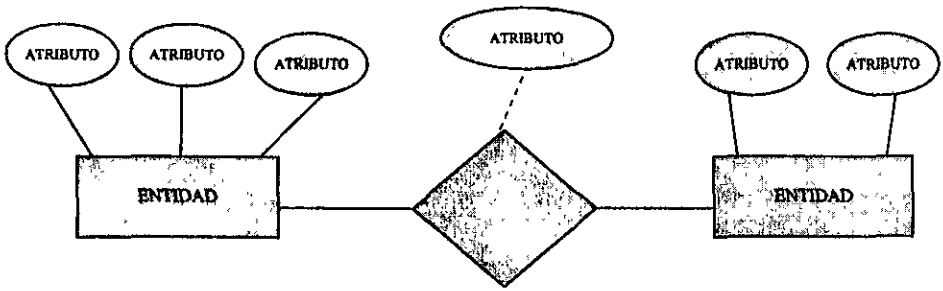


Fig. 3.2 Diagrama E-R

Una base de datos que se ajusta a un diagrama E-R puede representarse por medio de una colección de tablas.

Tablas:
Una tabla es un arreglo bidireccional o colección de registros. Formada por renglones (registros) y columnas (atributos), se deben tomar en cuenta todas las entidades primordiales (son conjuntos de entidades); ya que, por cada entidad primordial se tendrá una tabla. Cada tabla tiene un número de columnas que, a su vez, tienen nombres únicos. A continuación se darán algunas ventajas y desventajas al respecto:

Ventajas:
Determinan los beneficios dentro de las tablas.

- La tabla es fácil de manejar.
- Una tabla puede relacionarse con otra tabla, mediante un atributo común.
- Las columnas pueden ir en cualquier orden.
- Al modificar, borrar o insertar, no es necesario reorganizar la Base de datos física, sólo los índices.

Desventajas:
Determinan algunas complicaciones que existen al desarrollar tablas.
- Cierto nivel de redundancia.
- Difícil al diseñar vectores dentro de registros.
- Acceso lento basado en el tipo de índice y su tamaño.

Tipos de tablas:
Existen varios tipos de tablas, algunos de ellos:
1. Primordiales.
2. De referencia.
3. De trabajo.
4. De operación.
5. Histórica.
6. De descarga.

Normalización

Es un proceso de reestructuración de las tablas para disminuir la redundancia y optimizar el diseño de la Base de datos. Dentro de la normalización es necesario conocer los siguientes conceptos que permiten entender el uso de las formas normales:

Dependencia funcional:

Dentro de la normalización se va ha encontrar este término, por lo que es conveniente retomarlo:
Dependencia funcional, es cuando la existencia de un atributo depende directamente de otro que no es llave, para eliminar esta dependencia es necesario descomponer las tablas, que llevará a generar nuevas tablas.

Dependencia Transitiva:

Dado un atributo **A** que depende de **B** y un atributo **B** que depende de **C** se dice que hay dependencia transitiva si el atributo **A** depende de **C**.

***Multidependencia* ó Dependencia Multivaluada:**

La figura 3.3 muestra el caso cuando existen dos relaciones dependientes de muchas a muchas en una tabla.

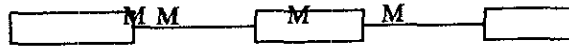


Fig. 3.3 Relación muchos a muchos

Reglas de Integridad:

Son características de la Base de datos donde el acceso y los registros de datos por diferentes aplicaciones debe ser coordinado y especificado en las tablas de asociación.

1. Integridad[6] de Entidades: ningún componente de la llave primaria (PK) puede tener valores nulos.

2. Integridad Referencial: Si A es un atributo primario entonces cada valor del atributo A deberá ser **NULO** y corresponder a un valor de la llave primaria relacionada.

Las formas normales

Para la normalización se deben tomar en cuenta algunas formas normales que permitirán normalizar tablas, ya que, debido a estas normas se evitan muchas inconsistencias, dichas formas se muestran en la figura 3.4.

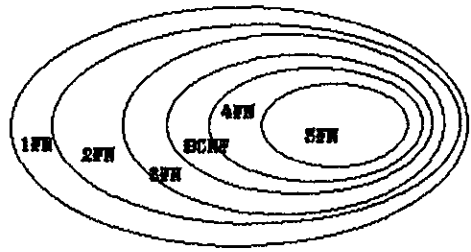


Fig 3.4 Formas Normales

1ª Forma Normal: Cuando todos los valores de las columnas son atómicos; es decir, cada celda de la tabla debe contener sólo un valor simple (no debe contener vectores). Para la eliminación de vectores es necesario crear más renglones.

2ª Forma Normal: La estructura debe estar en la primera forma normal, antes de pasar a esta forma y además todas las columnas "no llave" dependen directamente de la columna llave y no de manera funcional de otra columna.

3ª Forma Normal: Siempre y cuando hayan pasado las dos primeras formas y que no exista dependencia transitiva en ninguno de sus atributos no llaves.

[6] Ver definición en el glosario de términos

Forma Normal De Boyce / Codd: Una columna es determinante cuando alguna otra depende de ésta. Por lo que, cada determinante es una llave candidata. Llave candidata es cualquier columna (o grupo de columnas) las cuales son únicas y mínimas. Esta interpretación se muestra en la figura 3.5.

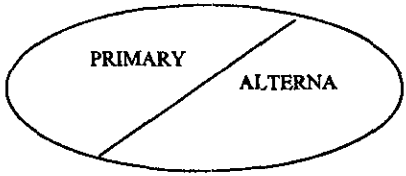


Fig. 3.5 Llaves Candidatas

4ª Forma Normal: Es cuando existe la multidependencia en tablas.
5ª Forma Normal o Forma Normal De Proyección / Producto: Es cuando hay redundancia de relaciones dependientes muchas a muchas, un ejemplo es el de la figura 3.6:

Relaciones dependientes = Relaciones con *n* caminos

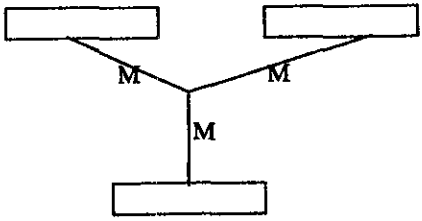


Fig. 3.6 Relaciones dependientes

Las formas normales decrementan la redundancia en las bases de datos; así como también, facilitan el uso y acceso a las bases de datos. Todas y cada una de estas, se explicaran claramente con el siguiente ejemplo:

Como principio de cuenta se tiene la tabla 3.1, la cual se va a normalizar:

NUM_EMP	CODIGO_EMP	NUM_PROJ	CARGO	PRECIO
E1	Programador	p1	Supervisor Inspector	60.00 38.80
E1	Programador	p2	Científico	25.50
E1	Programador	p3	Personal	9.95
E3	Ventas	p3	Supervisor Supervisor	60.00 3.35
E5	Escritor	p3	Supervisor	60.00

Tabla 3.1 Tabla a normalizar

1FN:Cada celda de la tabla debe contener solo un valor simple. Por lo tanto la tabla 3.1 normalizada se muestra en la tabla 3.2:

NUM_EMP	CODIGO_EMP	NUM_PROJ	CARGO	PRECIO
E1	Programador	p1	Supervisor	60.00
E1	Programador	p2	Científico	25.50
E1	Programador	p3	Personal	9.95
E3	Ventas	p3	Supervisor	60.00
E5	Escritor	p3	Supervisor	60.00

Tabla 3.2 Tabla resultante al aplicar 1FN

2FN: La tabla 3.2 está en la 1FN, pero puede ser redundante.

- Una causa de redundancia es que la columna "CODIGO_EMP" depende solo de la llave primaria de la columna "NUM_EMP".
- Para eliminar la redundancia se debe descomponer la tabla 3.2 , por lo que toda columna no llave depende de toda llave primaria.

Para comprender en que consiste esta forma normal se muestran las tablas 3.3 y 3.4 :

NUM_EMP	NUM_PROJ	CARGO	PRECIO
E1	p1	Supervisor	60.00
E1	p2	Científico	25.50
E1	p3	Personal	9.95
E3	p3	Supervisor	60.00
E5	p3	Supervisor	60.00

Tabla 3.3

NUM_EMP	CODIGO_EMP
E1	Programador
E1	Programador
E1	Programador
E3	Ventas
E5	Escritor

Tabla 3.4

Las tablas 3.3. y 3.4 son el resultado al aplicar la 2FN

3FN : Las tablas anteriores se encuentran normalizadas por la 2FN, pero puede ser redundante.

- La causa de la redundancia es que la columna PRECIO depende indirectamente de la llave primaria de la columna CARGO, como se muestra en la figura 3.7 .

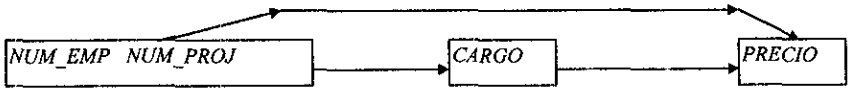


Fig. 3.7 Dependencia entre campos de una tabla

- Para eliminar la dependencia indirecta se lleva a cabo por medio de las tablas de descomposición, como se muestran en las tablas 3.5 y 3.6.

NUM_EMP	NUM_PROJ	CARGO
E1	p1	Supervisor
E1	p2	Científico
E1	p3	Personal
E3	p3	Supervisor
E5	p3	Supervisor

Tabla 3.5

CARGO	PRECIO
Supervisor	60.00
Científico	25.50
Personal	9.95
Supervisor	60.00
Supervisor	60.00

Tabla 3.6

Las tablas 3.5. y 3.6 son el resultado al aplicar la 3FN

- Cada columna "No llave" depende directamente de la llave Primaria, como se muestra en la figura 3.8 .

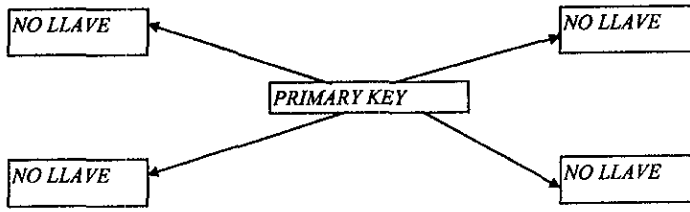


Fig. 3.8 Dependencia de los campos a la llave primaria

BCNF: Las tablas anteriores se encuentran normalizadas por la 3FN, pero pueden ser redundantes. Un ejemplo para esta forma normal es la tabla 3.7:

NUM_EMP	NUM_PROJ	CUENTA	CARGO
E1	p2	RMJ245	Científico
E1	p3	ZOO	Personal
E3	p3	ZOO	Supervisor
E4	p1	1211	Científico
E5	p3	ABC45	Supervisor
E6	p3	ABC45	Coordinador

Tabla 3.7 Tabla a la cual se le aplicará la BCNF

- La causa de la redundancia es que la llave primaria de la columna NUM_PROJ depende de la columna CUENTA.
- Para eliminar la redundancia se debe descomponer la tabla 3.7 en las tablas, 3.8 y 3.9:

NUM_EMP	CUENTA	CARGO
E1	RMJ245	Científico
E1	ZOO	Personal
E3	ZOO	Supervisor
E4	1211	Científico
E5	ABC45	Supervisor
E6	ABC45	Coordinador

Tabla 3.8

NUM_PROJ	CUENTA
p2	RMJ245
p3	ZOO
p3	ZOO
p1	1211
p3	ABC45
p3	ABC45

Tabla 3.9

Las tablas 3.8 y 3.9 son el resultado al aplicar la BCNF

BCNF remueve toda redundancia de las relaciones singulares.

Para ver como actúa la 4FN se tomarán las tablas 3.10, 3.11, 3.12 Y 3.13:

NUM_EMP	LENGUAJE	EXTENSION
E1	FORTRAN	NULL
E1	COBOL	NULL
E1	BASIC	NULL
E1	NULL	222
E1	NULL	501

Tabla 3.10

NUM_EMP	LENGUAJE	EXTENSION
E1	FORTRAN	222
E1	COBOL	222
E1	BASIC	222
E1	FORTRAN	501
E1	COBOL	501
E1	BASIC	501

Tabla 3.11

NUM_EMP	LENGUAJE	EXTENSION
E1	FORTRAN	222
E1	COBOL	501
E1	BASIC	NULL

Tabla 3.12

NUM_EMP	LENGUAJE	EXTENSION
E1	FORTRAN	NULL
E1	NULL	501
E1	BASIC	501
E1	COBOL	222

Tabla 3.13

Las tablas 3.10, 3.11, 3.12 y 3.13 se utilizarán para aplicar la forma normal 4FN.

4FN: Establece que no debe haber relaciones muchas a muchas independientes.
Al aplicar ésta norma las tablas 3.10, 3.11, 3.12 y 3.13 quedan descompuestas finalmente en las tablas siguientes:

NUM_EMP	LENGUAJE
E1	FORTRAN
E1	COBOL
E1	BASIC

Tabla 3.14

NUM_EMP	EXTENSION
E1	222
E1	501

Tabla 3.15

Las tablas 3.14 y 3.15 son el resultado al aplicar la 4NF

Modelo relacional

El modelo relacional es el principal modelo de datos para aplicaciones comerciales de procesamiento de datos. En este apartado se definirá más ampliamente este Modelo.

Una **base de datos relacional** consiste en una colección de tablas, a cada una de las cuales se asigna un nombre único. Cada tabla tiene una estructura similar a las bases de datos E-R mediante tablas. Una fila de una tabla representa una relación entre un conjunto de valores. Puesto que una tabla es una colección de dichas relaciones es por eso que se toma el nombre de modelo de datos relacional.

Álgebra relacional

El álgebra relacional es un lenguaje de consulta. Consta de un conjunto de operaciones que toman una o dos relaciones como entrada y producen una nueva relación como resultado.

Las operaciones fundamentales en el álgebra relacional son seleccionar, proyectar, producto cartesiano, renombrar, unión y diferencia de conjuntos. Aunque existen otras operaciones como intersección de conjuntos, producto natural, división y asignación.

Operaciones fundamentales:

Las operaciones fundamentales como seleccionar, proyectar y renombrar se llaman operaciones unitarias, ya que operan sobre una relación y las otras operan sobre pares de relaciones y, por tanto, se llaman operaciones binarias.

1. **Seleccionar** (σ): Se seleccionan tuplas que satisfacen un predicado dado. El predicado aparecerá como subíndice de σ. Además, se podrán usar las comparaciones de =, <, , >, en el predicado, así como también se pueden combinar varios predicados utilizando los conectores y ($\wedge$) y o ($\vee$).

2. **Proyectar (∏):** Es una operación unitaria que devuelve su relación argumento con ciertas columnas omitidas. Se listan los atributos que se quieren proyectar en el resultado como subíndices de ∏. La relación argumento se escribe a continuación de ∏ entre paréntesis.

3. **Producto cartesiano (x):** Esta es una operación binaria. Si r1 y r2 son relaciones, entonces r1 x r2 es el producto cartesiano de las relaciones.

4. **Renombrar (ρ):** Esta operación se usa para eliminar posibles ambigüedades. Al ser renombrada una referencia a una relación, es posible que se pueda hacer referencia a una sola relación pero sin ambigüedades:

Características:
Para realizar un alias es necesario tomar en cuenta los siguientes puntos:

- Se usan cuando tenemos homónimos.
- Renombra temporalmente una tabla.
- Sólo existe mientras se ejecuta la operación.
- Permite usar una misma tabla más de una vez.
- Para crear un alias o renombre se pone después de la tabla.

5. **Unión (*U*):** Se pueden unir dos conjuntos por medio de esta operación, nada más que se debe de asegurar de que las uniones se tomen entre relaciones compatibles, por lo tanto, para que sea valida esta unión debe cumplir dos condiciones:

   - Las relaciones r1 y r2 deben tener el mismo número de atributos.
   - Los *dominios* del atributo iésimo de r1 y del atributo iésimo de r2 deben ser los mismos.

6. **Diferencia de conjuntos (-):** Permite encontrar tuplas que estén en una relación pero no en otra. La expresión r1 -- r2 da como resultado una relación que contiene aquellas tuplas que están en r1 pero no en r2.

Operaciones adicionales:

7. **Intersección de conjuntos ( ):** Esta operación se puede leer o sustituir por la siguiente expresión, aunque es más sencillo escribirla de la primera forma:

r1 - r2 = r1 - (r1 - r2)

8. **Producto natural (IxI):** Es una operación binaria que permite combinar ciertas selecciones y un producto cartesiano en una operación. Esta operación forma un producto cartesiano con dos argumentos, realiza una selección forzando la igualdad en aquellos atributos que aparezcan en ambas planificaciones de relaciones y, finalmente, quita las columnas duplicadas.

9. **División (÷):** Se establece para aquellas consultas que incluyen la frase "para todos".

10. **Asignación (←):** La asignación debe hacerse siempre a una variable de relación temporal. Las asignaciones a relaciones permanentes constituyen una modificación de base de datos.

Relaciones temporales:

Este tipo de relaciones se deben tomar en cuenta, en algunos casos, al realizar alguna operación.

- Divide operaciones muy extensa en suboperaciones.
- El resultado de una suboperación puede originarse a una nueva relación llamada relación temporal.

Los atributos de la relación temporal son los mismos de la relación que la genera y los renombres se pasan como referencia.

Cálculo relacional

Al escribir una expresión en álgebra relacional, se da una secuencia de procedimientos que generan la respuesta a una consulta. El cálculo relacional es equivalente al álgebra relacional, es decir, para cada expresión en el álgebra relacional existe una expresión equivalente en el cálculo relacional de tuplas e inversamente, para cada expresión en el cálculo relacional de tuplas existe una expresión equivalente en el álgebra relacional.

Tipos de cálculos relacionales:

El cálculo relacional se divide en dos tipos, el cálculo relacional de tuplas y el cálculo relacional de dominios, a continuación se definirán en que consiste cada uno, además, se darán dos sencillas definiciones de lo que es una tupla y un dominio:

Una **tupla** (t) está representada por medio de (V_1, V_2), donde V_1 y V_2 son valores, pero de diferente dominio.

Un **dominio** es un conjunto de valores de los atributos.

El **cálculo relacional de tuplas**, es un lenguaje de consultas no procedimental. Describe la información deseada sin dar un procedimiento específico para obtener la información. Una consulta en el cálculo relacional de tuplas se expresa como:

$$\{t \mid P(t)\}$$

es decir, el conjunto de todas las tuplas t, tal que el predicado P, es verdadero para t.

El **cálculo relacional de dominios** utiliza variables de dominio que toman valores del dominio de un atributo más que valores de una tupla completa. Aunque, éste se encuentra relacionado íntimamente con el cálculo relacional de tuplas.

Una expresión en el cálculo relacional de dominios es de la forma $\{X_1, X_2, \ldots\ldots, X_n \mid P(X_1, X_2, \ldots\ldots, X_n)\}$, donde $X_1, X_2, \ldots\ldots, X_n$ representan variables de dominio. P representa una fórmula compuesta por átomos, como en el caso del cálculo relacional de tuplas. Un átomo en el cálculo relacional de dominios tiene una de las formas siguientes:

- $< X_1, X_2, \ldots\ldots, X_n > \in r_1$, donde r_1 es una relación en n atributos y $X_1, X_2, \ldots\ldots, X_n$ son variables de dominio o constantes de dominio.
- x Θ y, donde x e y son variables de dominio y Θ es un operador de comparación (<, , =, >,). Es requisito que los atributos x e y tengan dominios que puedan compararse por medio de Θ.
- x Θ c, donde x e y son variables de dominio y Θ es un operador de comparación

3.2 Herramientas

3.2.1 SYBASE (Sistema Manejador de Bases de Datos Relacionales - RDBMS)

En 1984 se fundó Sybase, en un mercado donde Oracle, Ingress, Informix y DB2 eran productos RDBMS. Sybase es un Sistema Manejador de Bases de Datos Relacional (RDBMS), Sybase fue el primero en incorporar la arquitectura Cliente/Servidor dentro de una base de datos relacional y el primero en promover el concepto de un nuevo camino a construir sistemas.

Características:

- Soporta 2 millones de tablas por cada base de datos
- 32767 bases de datos en Sybase
- Soporta 16 bases de datos abiertas en una misma consulta
- Soporta 16 tablas o más en una consulta

Requerimientos de Hardware:

- Para SQL Server 8MB en RAM
- *Espacio en disco para soportar librerías en el sistema 6MB*
- Para instalarlo completo son de 50 a 60 MB

Sybase y su uso en arquitectura Cliente/Servidor

- Diagrama Cliente/Servidor básico
- Consultas en línea
- Procesador de transacciones
- Arquitectura Cliente/Servidor
- Lenguaje: Structured Query Language
- Abierto

Funcionamiento de Sybase (Tareas del Servidor):

- Controlar integridad
- Manejar la concurrencia
- Ejecutar las peticiones.

Funcionamiento del Cliente:

- Proporcionar la Interfaz.

Plataformas para el Servidor de Sybase:

La tabla 3.16 muestra estas plataformas:

Servidor	Cliente
OS/2 IBM HP VMS Netware UNIX NT Sun	PC's, Macs y UNIX

Tabla 3.16 Plataformas para el servidor de Sybase

Productos:

Los Productos son los siguientes:
- Servidor de bases de datos SQL Server
- Herramientas para el uso de Sybase en terminales : DWB, APT e ISQL
- Herramientas de interoperabilidad: Open Server, Omni SQL Gateway
- Herramientas de replicación: Replication server
- Herramientas para desarrollo de aplicaciones: DB library
- Otras herramientas: BCP

Roles y responsabilidades

- System Administrator (Administrador del sistema): Resuelve problemas críticos.
- Operator System: Realiza respaldos, da de alta a usuarios.
- DBO: Dueño de la base de datos, coordinación con los usuarios para que se hagan respaldos, responsable de su propia información.

Componentes de SQL Server

El servidor debe poder alojar varias bases de datos para esto las organiza en dispositivos lógicos.
El procedimiento para hacerlo es:

- Crear un dispositivo.
- Crear las bases de datos y la bitácora (logs).

La figura 3.9 muestra esta organización:

Organización de Sybase

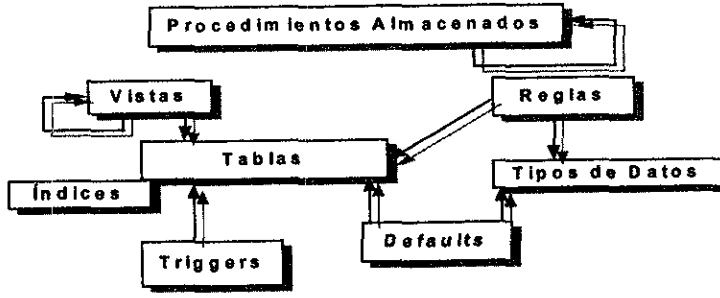


Fig. 3.9 Organización de Sybase

A continuación se describe cada uno de estos componentes:

Procedimientos Almacenados:

Un procedimiento almacenado es un conjunto de sentencias SQL que han sido almacenadas en una base de datos que pueden ser ejecutadas por su nombre.
Los procedimientos almacenados pueden aceptar y regresar parámetros, pueden correr más rápidamente que un archivo batch, reducen el tráfico de la red, refuerzan consistencia e integridad referencial en la base, mejoran significativamente el desempeño de los queries de SQL.

Vistas:

Una vista es una ventana con datos de una o más ventanas, a través de la cual se puede ver o cambiar información de una tabla muy grande . Es una tabla virtual, se ve y se usa como una tabla pero no existe, sus datos se derivan de tablas, sin embargo no son copiados. Las Vistas dan un mecanismo de seguridad a las base de datos, ya que dichas vistas no contienen datos, son como un espejo de los datos.

Triggers:

Los triggers son un tipo especial de procedimientos almacenados y son de gran utilidad en las tablas ya que se activan automáticamente por el SQL Server al insertar, borrar o actualizar (insert, delete, update) en una tabla.
Los triggers no aceptan parámetros y tampoco pueden ser llamados por su nombre.

Reglas:

Las reglas pueden especificar una máscara para una columna o un tipo definido por un usuario.
Una regla puede ser una lista de valores, un rango o un patrón; una regla puede ser aplicada para varias columnas de una o distintas tablas.

Defaults:

Los defaults especifican los datos que deben de estar dentro de una columna si el usuario no especifica una entrada. Una columna puede tener un solo default.

Tipos de Datos:

Los tipos de datos proveen consistencia entre tablas. Se pueden aplicar reglas y defaults a un tipo de dato definido por el usuario, y este usuario en varias columnas.
Una columna puede tener un solo tipo de dato definido.

Índices:

Un índice es una estructura de almacenamiento física y ocupa espacio. Los índices pueden reforzar la unicidad de las llaves.

Hay dos tipos de índices:

- non-clustered index: No afecta los datos y sólo provee apuntadores a renglones, puede haber hasta 249 índices por tabla.
- clustered index: Ordena la página es orden del índice. Sólo puede haber uno por tabla.

Análisis Comparativo de Bases de Datos:

La evolución de las bases de datos ha permitido que los usuarios tengan una mayor ventaja en la portabilidad, compatibilidad, integración, arquitectura Cliente/Servidor, soporte SQL, orientación a objetos, etc.

Ante este panorama se muestra el cuadro general 3.1, de análisis comparativo de bases de datos:

LO MEJOR	LO PEOR
Adabas	
• Facilidad de uso • Confianza en el servicio • Capacidad de transacciones • Integridad de datos	• Alto costo de mantenimiento • Poca flexibilidad
DB2	
• Ambiente multiplataformas • Herramientas CASE • Aplicaciones de soporte a la decisión	• Difícil migración • Herramientas de usuario final
Informix	
• Buen desempeño • Flexibilidad y portabilidad • Buena herramienta en Unix	• Administración compleja • Caídas regulares
Ingres	
• Buen precio • Fácil manejo • Integridad de datos	• Pocos programadores con experiencia • Tiempos de respuesta no cumplidos
Oracle	
• Seguridad • Fácil desarrollo • Integridad de datos	• Manejo complejo de la administración • Consume muchos recursos
Progress	
• Rápida implementación • Lenguaje robusto • Integridad de la información	• Problemas con el soporte técnico • Empieza a ser obsoleta
Sybase	
• Fácil de administrar • Tecnología Cliente/Servidor • Integridad de datos	• Herramientas de desarrollo • Migración de aplicaciones

Cuadro 3.1 Análisis comparativo de bases de datos

Los resultados anteriormente mostrados fueron realizados por Computerworld/México en una encuesta nacional de bases de datos con 224 usuarios de los productos más importantes en el área, a saber: ADABAS, de Software AG; DB2, de IBM; de informix Software; INGRES, de Ingres; PROGRESS, de Progress Software y SYBASE de Sybase.
La tabla 3.17 muestra los resultados obtenidos de las diferentes herramientas que se basan en puntuaciones en escala de 1 a 6, donde 6 es la mejor.

Herramientas	**Adabas**	**DB2**	**Informix**	**Ingress**	**Oracle**	**Progress**	**Sybase**
Eficiencia	4.5	4.0	4.9	3.9	4.9	5.0	5.1
Recuperación de caídas	4.7	3.9	4.9	4.2	4.8	5.3	4.3
Capacidad de transacciones on-line	4.8	4.0	5.0	4.0	5.0	5.3	4.1
Integridad de datos	4.5	3.9	4.9	4.1	5.2	5.4	5.6
Herramientas de programación	4.0	3.9	4.5	4.0	4.5	4.8	4.1
Administración del Sistema	4.0	4.2	4.6	3.9	4.2	4.7	4.5
Herramientas del usuario final	4.0	3.9	4.1	3.8	4.4	4.6	4.0
Integración del SGBD y seguridad de S.O.	4.0	4.0	4.6	4.2	4.5	4.6	4.8
Seguridad Multinivel	4.3	3.9	4.6	4.0	4.5	5.0	4.6
Soporte SQL Estándar	4.3	4.0	5.0	4.0	5.3	4.7	5.3
Aplicaciones de soporte a la decisión	4.2	4.2	4.4	3.9	4.6	4.6	4.3
Funciones de auditoría	4.0	4.1	4.1	3.9	4.3	4.3	4.3
Facilidades de migración desde SGBD	4.0	3.7	4.5	3.8	4.3	4.6	4.4
Extensiones SQL	3.9	3.9	5.0	3.9	5.0	4.3	5.0
Actualización, recuperación y admón. remota	4.2	4.0	4.9	3.5	4.4	4.8	4.3
Integración de herramientas CASE	4.0	3.9	3.8	3.9	4.9	4.2	4.3
Facilidad de herramientas de desarrollo	4.0	4.0	4.6	4.0	4.6	5.0	4.4
Soporte técnico recibido del proveedor	4.2	3.8	3.8	3.8	4.5	3.8	4.3

Tabla 3.17 Tabla de resultados de diferentes herramientas

En este caso Sybase ocupó el segundo lugar

3.2.2 Servidores HTTPD

En un principio el WWW fué utilizado para fines científicos y educacionales.
Se puede accesar a otras máquinas por medio de Internet por varios caminos o comandos de programas como Telnet, FTP, Gopher, WAIS, Verónica y Mail.

HTTPd (Hiper Text Transfer Protocol Servers)

Provee acceso para documentos distribuidos de hypermedia, aplicaciones y bases de datos.
Uno de los mayores defectos o limitaciones de HTTP es su incapacidad de mantener aplicaciones de información (a veces llamadas estados) a través de múltiples sesiones.
Para superar esta limitación se tiene que ser creativo con el HTML pasando información del estado de los documentos.

HTTP Server se basa en el modelo cliente/servidor por medio del cual un cliente establece una conexión con un servidor y somete una petición. El servidor responde con la línea de status, incluyendo el protocolo de mensaje y el código de éxito o de error seguido por el mensaje MIME-like.
A continuación se mencionan los principales servidores Web disponibles en Internet.

National Center for Supercomputing Aplications (NCSA) HTTPd

NCSA es un derivado del Servidor Web HTTP/1.0 y es creditado como uno de los primeros Servidores de HTTP disponibles. NCSA HTTPd soporta múltiples esquemas de autentificación incluyendo las autentificaciones de HTTP, MD5, Kerberos v.4 y v.5. El software de NCSA esta disponible tanto en binario como en fuente. Aunque el software de NCSA HTTPd es propiedad de The Board of Trustees de la Universidad de Ilinois, se otorgó una licencia y no causa honorarios al usar HTTPd por academias, búsquedas y propósitos de negocios internos solamente.

En seguida se muestra una lista de las diferentes plataformas donde corre el precompilador ejecutable de NCSA para HTTPd 1.5
IRIX 4.0.5, 5.3
SunOS 4.1.3/Solaris 1.x
SunOS 5.4/Solaris 2.4.x86
SunOS 5.4/Solaris 2.4 SPARC
SunOS 5.3/Solaris 2.3 SPARC
AIX 3.2.5
HP-UX 9.05
OSF/1 3.0
Ultrix 4.0
Linux 1.2.13, libc 5.0.9 ELF

Apache HTTP Server

Apache HTTP Server es mantenido por El Grupo Apache y es derivado del original NCSA HTTPD 1.3. El software Apache está disponible y es libre de cargos y puede ser libremente distribuido. La versión actual (1.3.0) incluye soporte para DBM validación de bases de datos, cliente HTML/CGI respuesta a servidor de errores, múltiples directivas Directory Index, ilimitado número de Alias y directivas Redirect, contiene negociación y servidores multi-homed.

Adicionalmente El Grupo Apache mantiene y guarda una lista Servidor Majordomo de usuarios informados del nuevo bug fixes, security fixes y noticias en general.

La lista de servidores provee un mecanismo automático de correo electrónico de subscripción de usuarios que los mantiene informados al día de información pertinente del contenido de la lista.

Por ejemplo, el e-mail lista apache-nnounce contiene información acerca de los nuevos usuarios relacionados al Servidor Web Apache.

Las plataformas donde corre son las siguientes:
A/UX 3.1
BSDI 1.1
FreeBSD 2.1
HP/UX 9.07
IRIX 5.3
LINUX
NETBSD 1.1
SunOS 5.4
SunOS 4.1.3
Unixware 1.2

Netscape Communications Server y Commerce Server

El Servidor Netscape Communications y el Servidor Commerce son descendientes del Servidor HTTP/1.0. El Servidor Communication y el Servidor Commerce son compatibles con cualquier HTTP cliente o servidor. Estos soportan al máximo uso de Multipurpose Internet Mail Extension (MIME), tipos y formatos estándar de imágenes como GIF y JPEG. Adicionalmente soporta la interface CGI, pero el Servidor Netscape incluye un conjunto de API's que se pueden usar e integrar estas aplicaciones dentro del ambiente del Servidor Web.

El Servidor Netscape Commerce difiere del Servidor Netscape Communication en que éste provee encriptación de datos, validación de servidores y la verificación integral de mensajes usando el protocolo abierto SSL.

Plataformas donde corre:
OSF/1 2.0
HP-UX 9.0.3, 9.0.4
AIX 3.2.5
IRIX 5.2
Solaris 2.3 y 2.4, SunOS 4.1.3
Windows NT
BSDI 1.1, y 2.0 (BSD/OS)

Microsoft Internet Information Server
Microsoft Internet Information Server (IIS) es una base de Windows NT WWW Server, éstos incorporaron servicios World Wide Web, servicios de Gopher y FTP, un manejador de Internet, conexión de Bases de Datos en Internet, y el Secure Sockets Layer (SSL). Este contiene visualizadores cliente para Windows 3.11, Windows para trabajo en grupo, Windows NT y Windows 95.

Características:
- Flexible construcción en login
- Segura información en Internet en el servidor de administración
- Alto desempeño en acceso a Bases de Datos
- Conexión a Bases de Datos en Internet

Usando el Internet Database Connector se puede ligar directamente a Microsoft SQL Server, Microsoft Acces, Oracle, Informix, Sybase, y otros (ODBC) Open Database Connectivity.

Otros Web Server
La tabla 3.18 muestra algunos servidores de Web:

Servidor Web	Plataforma
Alibaba	NT,95
Basis Web Server	95
Cheetah	NT, 95, Solaris,VMS
CL-HTTP	UNIX, 3.1, Mac
CommerceBuilder	NT, 95
ConnectionServer	3.1, 95
HTTPd	Perl
Plexus HTTPd	Perl
Purveyor	NT, 95, VMS, Netware

Tabla 3.18 Plataformas de algunos servidores de red

3.2.3 Interfaz CGI

Fue el primer paso para lograr convertir el HTML en un lenguaje más poderoso gracias a la integración con otras herramientas, su nombre son las iniciales de Common Gateway Interface (CGI) es el estándar para servir de interfaz para comunicar aplicaciones externas con servidores de información, se ejecuta en tiempo real y es un contenido dinámico de información.

El Common Gateway Interface (CGI) es simplemente una especificación que le dice al servidor cómo enviar la información hacia el script, y qué es lo que el servidor debería hacer después de recibir la información previamente del script.

Un programa CGI puede ser tan simple o tan complejo, y puede desempeñar cualquier tarea de un programa, CGI Script puede proveer salidas tan simples como una lista de usuarios generalmente logged para el servidor Web, o tan complejo como un sistema. El programa se comunica con el mundo real usando el lenguaje CGI.

No hay limite para lo que puede hacerse con CGI, lo único que debe tenerse en cuenta es que sea lo que sea que deseemos hacer el proceso no debe ser muy largo por que de otra manera el usuario no sabe a que atenerse al estar pidiendo la realización del proceso y nada pasa.

Un buen uso de CGI Script es que dinámicamente cambia la presentación de los documentos Web cada vez que son accesados. Las aplicaciones CGI son usadas para producir páginas HTML (los contenidos pueden ser cambiados en cada petición).

Los CGI's son frecuentemente usados para procesar la información introducida en las formas HTML.

CGI tiene varios beneficios. Uno de éstos es que habilita aplicaciones existentes para generar salidas (salida estándar, archivos de salida y otros) esto puede ser usado como entrada en HTML-generating CGI Script.

No se necesita re-escribir el código existente para generar salidas HTML, simplemente se tiene que ejecutar el CGI Script en la aplicación existente, recuperar la salida de la aplicación, analizar la salida y generar el código HTML.

La especificación CGI es implementada en servidores Web, como también sobre programas construidos sobre el Web. Éste no es parte del HyperText Transfer Protocol (HTTP), pero actualmente más servidores Web cambian para implementar esto en el futuro.

Las aplicaciones CGI se usan en varios servidores Web conocidos incluyendo a NCSA httpd, CERN httpd, Apache httpd, y otros servidores comerciales.

Estos servidores Web son usualmente distribuidos con un conjunto de programas CGI de propósito general, estos residen en un directorio llamado cgi-bin en el directorio root de el servidor Web. Este es el directorio comúnmente usado para almacenar programas CGI, pero el Webmaster puede definir otra ruta.

En CGI Scripts, el usuario y el visualizador pueden interactuar con el servidor Web. CGI Script son archivos ejecutables (como archivos batch de DOS, un Shell Script de UNIX, lenguaje Perl Script, programación en C, y otros), éstos permiten una comunicación dinámica entre dos aplicaciones. Por ejemplo, considérese una aplicación de bases de datos en el servidor Web.

Para hacer esto más claro digamos que deseamos que la base de datos implementada bajo Unix que se encuentra en nuestra empresa sea consultada por todos los visitantes a nuestra página web. Lo que necesitamos crear es un programa en CGI que se ejecute cada vez que sea requerida una información desde el cliente hasta el motor de búsqueda de la base de datos y que reciba un resultado y lo transmita al cliente que deseaba esa información. Éste es un ejemplo sencillo de cómo funciona este "puente".

Como un usuario final de la aplicación, se quiere una consulta y recuperar información tan pronto como este disponible. Teniendo CGI Script, el Servidor Web puede consultar la base de datos para la información que se solicita y construir el código en HTML para que se despliegue en el visualizador Web, como se muestra en la figura 3.10 :

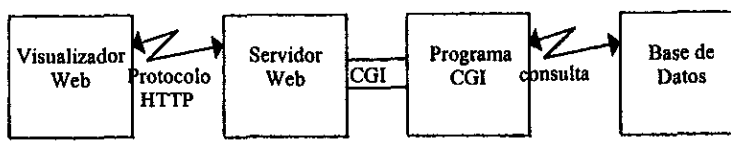


Fig. 3.10. Accesando a una Base de datos usando un visualizador Web

No es necesario generar el código HTML, siempre que nuevos documentos o datos son introducidos dentro de la base de datos. CGI provee a los usuarios finales acceso al instante para modificar la base de datos.

Por ejemplo, un programa CGI llama fácilmente a generar un número aleatorio usado como índice en la tabla de la B.D. Cada vez que el CGI es ejecutado, una nueva consulta es desplegada al usuario final.

Lenguajes:

Las aplicaciones CGI pueden ser escritas en cualquier lenguaje que pueda ser ejecutado en una computadora, en particular una plataforma Web. En realidad se puede escoger cualquiera de los lenguajes comunes para una aplicación CGI. La selección depende de lo que se quiera hacer porque los diferentes lenguajes pueden ser especializados para diferentes propósitos. Perl por ejemplo es muy bueno para cadenas y manipulación de archivos, mientras que C es mejor por lo robusto para programas complejos. Perl y C son probablemente los lenguajes más usados para programar CGI's.

Los lenguajes para programar CGI's son :

- C
- C++
- Perl
- Tcl
- Python
- Shell scripts (UNIX)
- Visual Basic
- Applescript

Estos lenguajes como también muchos otros proveen la programación necesaria para compilar con la especificación CGI y usar todo su potencial.

Métodos CGI:

Un método es un camino para invocar un programa CGI. En realidad para ejecutar el programa, se hace una petición al servidor usando un método, el cual, define como el programa recibe los datos. Hay tres métodos :

Método GET

Normalmente se usa este método para hacer una petición por una página Web. GET *simplemente le dice al servidor* que envíe la información deseada al cliente.
Cuando se usa este método, el programa recibe el dato en la variable de ambiente QUERY_STRING. El programa analiza (procesa) la cadena en orden para interpretar el dato y ejecutar la acción necesaria.

El método GET es usado cuando se quiere obtener datos desde el servidor y no serán alterados los datos en el servidor. Excepciones pueden aparecer cuando el dato transmitido es muy largo entonces estos problemas eventuales en el tamaño de las variables son prevenidos. En este caso, el método POST es preferible.

Método POST

Este método le dice al servidor que se enviará la información del URI[7] como un subproceso, sin hacer interpretaciones.
Cuando se usa el método POST el servidor Web transmite el dato al programa CGI hacia la salida estándar (stdin standard input).
El servidor no marca el final del dato con el carácter EOF, entonces el programa debe usar el valor CONTENT_LENGTH en orden para leer correctamente la salida estándar. El método POST se usa cuando el dato enviado altera cualquier dato en el servidor Web o cuando se desea enviar cantidades largas de datos al programa CGI (usualmente más de 1024 bytes, el límite de un URL[8]).

Método HEAD

Este método es muy útil ya que regresa al cliente la información del header. El método HEAD es similar al método GET, excepto que en el método GET sólo los headers HTTP (y no el dato mismo) son enviados por el servidor Web al visualizador.

Especificación de Interfaz

A continuación se presentan los cuatro mejores métodos de comunicación entre un servidor Web y un programa CGI :

- Variables de ambiente
- Línea de comando
- Entrada estándar
- Salida estándar

Estos están basados en la versión (1.1) de la especificación del Common Gateway Interface.

Variables de ambiente

Las variables de ambiente son un conjunto de variables de un sistema específico por el servidor web cuando este ejecuta una aplicación CGI. Las variables de ambiente suministran información acerca del servidor, del cliente, del CGI, y algunas veces de los datos enviados del servidor. A continuación se listan algunas variables de ambiente disponibles.
Nota : algunos servidores pueden incluir algunas variables extras propiamente como se muestran en la tabla 3.19.

[7] Un URI (Uniform Resource Identificator) es un identificador de recursos
[8] Un URL (Uniform Resource Identificator) es un localizador de recursos

Variables generales:

VARIABLES	DESCRIPCIÓN
GATEWAY_INTERFASE	Describe la versión del protocolo CGI usado. La versión actual del protocolo es la 1.1
SERVER_PROTOCOL	Contienen la versión del protocolo HTTP usado. Lo más común es que contenga la versión 1.0
REQUEST_METHODS	Puede ser GET o POST, dependiendo del método usado para enviar los datos al programa CGI.

Tabla 3.19 Tabla de variables utilizadas en programas CGI's

Variables pasadas del servidor al programa CGI

VARIABLES	DESCRIPCIÓN
PATH_INFO	El usuario puede especificar un path (relativo a la raíz de los documentos HTML del servidor) donde accesa al programa CGI añadiendo una diagonal (/) seguida por la información del path deseado. Por ejemplo, si se accesa el siguiente URL, donde el programa CGI es mail.cgi, PATH :INFO deberá ser igual a /images : *http ://mi.servidor.mx/cgi-bin/mail.cgi/images*
PATH _TRANSLATED	Es el valor equivalente de PATH_INFO relativo a un sistema de archivos. Si la raíz de documentos es */netscape /server/docs,* y accesa el siguiente URL, PATH_TRANSLATED es igual a */netscape/server/docs/images :*

Tabla 3.20 Tabla de algunas variables enviadas del servidor al programa CGI

Línea de Comando

La línea de comando CGI es usado sólo con una consulta ISINDEX. Una consulta ISINDEX es una consulta especial obtenida con la etiqueta <ISINDEX> y la etiqueta <BASE HREF="..."> (referenciando el script). El dato entrado por el usuario es enviado al programa CGI vía línea de comando, a menos que contenga el signo igual (=), en este caso el QUERY_STRING es usado en su lugar. Más de un parámetro puede ser pasado a la línea de comando de un programa CGI, porque el servidor Web remplaza cualquier signo(+) recibido desde el cliente con espacios.

Entrada estándar

La entrada estándar (stdin) es usada por el servidor web para pasar información al programa CGI cuando el método POST es usado. El servidor Web es el responsable para enviar el valor CONTENT_TYPE y CONTENT_LENGTH, entonces el programa CGI sabe o conoce que tipo de dato recibieron y que tan largo es. El CONTENT_LENGTH es una cantidad de bytes del código de datos URL(los espacios son remplazados por los signos tilde, caracteres como %7E, y otros).

Salida estándar

El programa CGI manda resultados a la salida estándar. Éstos tal vez son enviados directamente a los usuarios de un visualizador o pueden ser interpretados por el servidor Web en orden para una acción que puede ser ejecutada (por ejemplo, una dirección URL existente).

Los programas CGI pueden sobrepasar al servidor y llamar directamente al browser. Para distinguir estos programas, primero sus nombres deben iniciar con nph (esto es No Parse Header, estos resultados ignorando cualquier información, encabezados HTTP o MIME). Esto es arriba del programa CGI para regresar encabezados HTTP válidos al browser.

Pero si un programa nph no es usado el servidor observa por tres especiales encabezados que el programa CGI puede regresar:

Content-type: Éste es el tipo de encabezado MIME. Usualmente, como programas CGI a la salida texto HTML para desplegar en el visualizador, esto es común para usar Content-type: text/ html\n\n. Los dos caracteres de salto de línea para el final de la línea, esto es obligatorio para una línea en blanco antes de un encabezado HTTP.
Location: Llama el servidor referenciando otros documento. El servidor puede decidir una re-dirección para el cliente o enviar el contenido del documento referenciado, dependiendo si es un URL completo o una ruta virtual (ruta relativa).
El programa CGI puede enviar un archivo existente a un sistema de archivos o de otro servidor Web por medio del *header location*. Para usarlo, se debe especificar ya sea la locación del archivo relativo al directorio raíz del directorio Web, en cuyo caso se especificaría como:
Location: absolute URI
bien un URL válido, como por ejemplo:
Location http://www.msc.com/

Status: Esta es la línea de status, el servidor envía al cliente un formato: nnn xxxxx, cuando nnn es un código de tres dígitos y xxxxx corresponden a descripción de texto.

La línea de status le dice al cliente el estado de la petición. La línea consiste en la versión del protocolo HTTP, el código de status del documento, y una breve explicación del código de status, ejemplo:
HTTP-versión CODE CODE-description
HTTP/1.0 200OK

Como trabajan los CGI's

Dependiendo del método que el cliente utilice para hacer la petición al servidor, este recolecta la información suministrada por el cliente y envía la información al script CGI. El script CGI procesa entonces la información y regresa los resultados al servidor, el cual los envía al cliente, como se muestra en la figura 3.11:

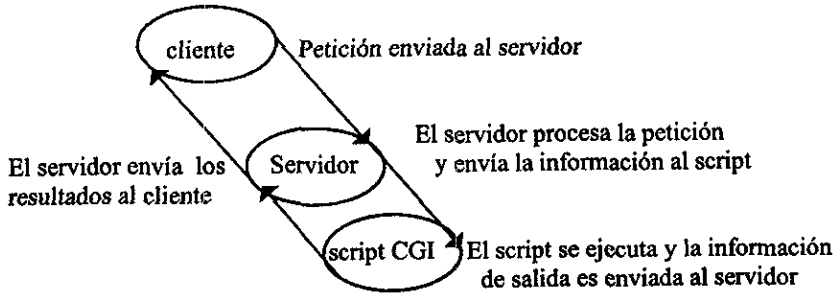


Fig. 3.11 Funcionamiento de un programa CGI

3.2.4 Active X

De la pantalla surgen animaciones y sonido, ejecutados en tiempo real, al tiempo que se pueden invocar pequeñas herramientas para ejecutar operaciones sencillas, como una calculadora. Además de incluir herramientas como los marcos (frames) que permiten visualizar varias páginas Web en una sola pantalla. Con la aparición del Explorer 3.0 Microsoft presentó una nueva tecnología denominada ActiveX, que pretende convivir con la fiebre de Java, que pareciera marcar la dirección en el entorno del software de los próximos años.

¿ Qué es ActiveX ?

ActiveX es una serie de herramientas tecnológicas de Microsoft que permite el contenido interactivo dentro del World Wide Web.
ActiveX permite a desarrolladores de webs generar páginas con contenido dinámico, de la misma forma que lo permite el Java. Se pueden escribir pequeñas aplicaciones como controladores ActiveX que a la vez pueden manejar audio, video y animación. Lo principal de todo esto es que los controladores pueden ser escritos en lenguajes tradicionales como el C++ o Visual Basic y adaptados casi automáticamente al nuevo estándar.

Aunque en un principio se pensó en que sería el reemplazo de Java, Microsoft se encargó de posicionar el único elemento que permitirá la conexión de diferentes componentes y la interacción de cada uno de ellos, como el Java, el JavaScript, Visual Basic Script, etc.

Otra de las incorporaciones es JavaScript, denominado JScript por parte de Microsoft dado que es un lenguaje desarrollado íntegramente por la empresa debido a problemas legales con Netscape. El Navigator ya lo tiene incorporado desde la versión 2.0.

Active X es una tecnología diseñada para trabajar con componentes u objetos. Un "objeto" es entidad de software que puede contener tanto código (la parte del programa en donde se especifica las cosas que debe hacer) como datos y que es independiente por sí mismo, lo cual implica que puede ejecutarse sin necesidad de otras herramientas.

Sin embargo, un grupo de objetos entrelazados de alguna manera pueden formar parte de una utilidad de software más grande que la suma de sus partes.

Microsoft y otras compañías ofrecen herramientas para la construcción de esos componentes (Visual Basic, VC++, Borland Delphi, etc.) así como implementaciones de los mismos en productos como Internet Explorer y Microsoft Internet Information Server.

Este marco de trabajo incluye a OLE, Visual Basic y Java. Además está basado en estándares abiertos. ActiveX no es un concepto nuevo, el nombre es lo único que es nuevo. La idea nació en 1993 cuando Microsoft liberó OLE 2.0 (Object Linking and Embedding, Objetos Ligados y Pegados).
El objetivo era, y es hoy en día, habilitar la integración de aplicaciones al nivel de documentos compuestos, como puede ser editar una hoja de Excel dentro de Word.

OLE 2.0 fue la primera tecnología basada en otra llamada COM o Component Object Model (Modelo de Componentes de Objetos). En ese entonces (es decir, 1993) Microsoft decidió llamar a todas las tecnologías construidas entorno a COM con el nombre OLE, aún cuando muchas de ellas no tuvieran mucho que ver con el concepto (Ligar y Pegar o Link and Embed).

Así nacieron términos como: OLE Automation y OLE Controls que están más relacionados a Objetos y Componentes que a Ligar y Pegar. Con la introducción de diferentes tecnologías basadas en COM se decidió arreglar este problema de nomenclatura y así nació el nombre ActiveX.
Las tecnologías relacionadas a Ligar y Pegar seguirán siendo llamadas OLE.

Entre los beneficios primarios de esta tecnología encontramos que es totalmente multiplataforma, permite integrar al HTML herramientas de desarrollo como Visual Basic, Visual C++, Delphi y Java. Los desarrolladores pueden usar la herramienta que conozcan mejor.

Entre los elementos más importantes de esta tecnología está el permitir ver, a través de los browsers, archivos que no estén en HTML como documentos de Word o de Excel por poner un ejemplo. Aparte permite integrar Java applets con controles ActiveX en browsers que así lo soporten, por ahora sólo lo soporta el Internet Explorer 3.0.

El ActiveX incluye también arquitectura cliente servidor y se encuentra en el Internet Explorer 3.0 y para los usuarios de Netscape como un plug-in.

¿Cómo se complementan ActiveX y Java?

ActiveX provee un mecanismo estándar para extender cualquier lenguaje de programación, incluyendo Java. ActiveX extiende las capacidades de Java permitiéndole integrar los applets con la riqueza de ActiveX.
Este lenguaje empata las applets con objetos creados en otros lenguaje esto permite vincular controles de ActiveX directamente desde los programas de Java. De la misma manera objetos escritos en otros lenguajes de programación de distintas compañías pueden ser vinculados con aplicaciones Java.
Recuerde que estas herramientas están diseñadas pensando en aplicaciones a la medida del cliente y que son un tanto complejas pero que de veras es sencillo el programar o formatear documentos en HTML.
Active X, es una tecnología ya probada, abierta y que no implica los costos de asumir el lenguaje Java.

Algunas fuentes de información:

ActiveX PROGRAMMING , Snaders Kaufman, Jeff Perkins, Dina Fleet ; Editorial Sams Net.
http://www.colgado.com.ar/aa-main/06feat/1009_tre/tre_tec.htm
http://www.unired.net.pe/~overnet/online/opina2.htm
http://www.venweb.com/venweb/cglobal/52/Html.html

3.3 Lenguajes

3.3.1 SQL (Structure Query Languaje)

El SQL es un lenguaje de bases de datos relacional estándar. Combina instrucciones del álgebra relacional y del cálculo relacional, para estructurar una base de datos, para los arreglos previos a ésta, y también para implementarle una medida de seguridad a la BD.

Partes que integran al sql

Algunos han dividido a SQL en varias partes, enseguida se darán pequeñas definiciones para cada una de estas:

- **Lenguaje de definición de datos (DDL):** El SQL DDL proporciona órdenes para definir esquemas de relación, eliminar relaciones, crear índices y modificar esquemas de relación.

- **Lenguaje de manipulación de datos interactivo:** El SQL DML incluye un lenguaje de consultas basado en el álgebra relacional y en el cálculo relacional de tuplas. También incluye órdenes para insertar, suprimir y modificar tuplas de la base de datos.

- **Lenguaje de manipulación de datos inmerso (DML):** La forma inmersa de SQL está diseñada para usar dentro de los lenguajes de programación de propósito general, tales como PL/I, Cobol, Pascal, Fortran y C.

- **Definición de vista:** El SQL DDL incluye órdenes para definir vistas.

- **Autorización:** El SQL DDL incluye órdenes para especificar derechos de acceso a relaciones y vistas.

- **Integridad:** El lenguaje Sequel del Sistema R original incluye órdenes para especificar restricciones de integridad complejas. Versiones recientes de SQL, incluyendo el ANSI estándar, proporcionan únicamente una forma limitada de comprobación de integridad. Los productos futuros y estándares futuros es probable que incluyan características intensificadas para la comprobación de integridad.

- **Control de transacciones:** SQL incluye órdenes para especificar el comienzo y final de las transacciones. Varias implementaciones, incluyendo IBM SAA-SQL, permiten el bloqueo explícito de los datos para control de concurrencia.

Estructura básica

La estructura básica de una expresión en SQL consta de tres cláusulas: **select, from** y **where,** que a continuación se explicarán cada una de ellas.

SELECT: Corresponde a la operación de proyección del álgebra relacional. Se usa para listar los atributos que se desean en el resultado de una consulta.

FROM: Corresponde a la operación de producto cartesiano del álgebra relacional. Lista las relaciones que se van a examinar en la evaluación de la expresión.

WHERE: Corresponde al predicado de selección del álgebra relacional. Consta de un predicado que implica atributos de las relaciones que aparecen en la cláusula FROM. En caso que se omitiera la *cláusula* **Where** , *el predicado* **P** es **verdadero.**

SQL usa los conectores lógicos **and, or** y **not** en vez de los símbolos matemáticos, $\wedge$, $\vee$, $\neg$. Permite el uso de expresiones aritméticas como operandos de los operadores de comparación. Una expresión aritmética puede implicar cualquiera de los operadores, +, −, * y /, operando sobre constantes o valores de tuplas. Muchas implementaciones de SQL incluyen funciones aritméticas especiales para tipos de datos particulares.

Operaciones lógicas

Algunas de las operaciones más utilizadas en SQL son las siguientes:

- = igual
- ≠ ó <> no igual
- > mayor que
- < menor que
- >= mayor ó igual que
- <= menor ó igual que

Operadores aritméticos

Dentro de las cláusulas Select, Where, Order by, se pueden utilizar las siguientes expresiones aritméticas con datos tipo DATE, que nos van a permitir realizar las operaciones de suma, resta, multiplicación y división.

- *, /: Estos permiten realizar primero la operación de multiplicación, y a continuación la de división. Ya que, se tiene un orden de precedencia.
- +, -: Estos permiten realizar sumas y restas de fechas; y al igual, que las expresiones anteriores estas tienen un orden de precedencia de izquierda a derecha.

Otros operadores

Dentro de SQL, también se tienen operadores como los que a continuación se mencionan, que ayudarán en la realización de consultas:

- **Between**: SQL incluye un operador de comparación **between** para simplificar cláusulas **where** que especifican que un valor que sea menor o igual que un valor dado y mayor o igual que otro valor dado. Y análogamente, podemos usar el operador de comparación **not between**.

- SQL también incluye un operador de selección para comparaciones de cadenas de caracteres. Los modelos se describen usando dos caracteres especiales:
  - Tanto por ciento (%). El carácter % es igual a cualquier subcadena de uno o más carácteres.
  - Subrayado (_). El carácter _ es igual a cualquier cadena de un carácter.

Los modelos son sensibles al tipo de letra; es decir, los caracteres en mayúsculas no son iguales a los caracteres en minúsculas, o viceversa.

- **Like** (Busca un patrón): Los modelos se expresan en SQL usando el operador de comparación **like**. Para que los modelos incluyan caracteres de modelos especiales (es decir, %, _), SQL permite la especificación de un carácter de **escape**. El carácter de escape se usa inmediatamente antes de un carácter de modelo especial para indicar que el carácter de modelo especial se va a tratar como un carácter normal. Se define el carácter de escape para una comparación **like** usando la palabra clave escape. Para ilustrarlo considérense los siguientes modelos que usan una barra inclinada (\) como carácter de **escape**.

Además, SQL permite buscar desigualdades en vez de igualdades usando el operador de comparación **not like**.

- **In** (lista): SQL se basa en el cálculo relacional para operaciones que permiten probar la pertenencia de tuplas a una relación. El conector **in** prueba si se es igual a cualquier miembro de una lista ó conjunto, donde el conjunto es una colección de valores producidos por una cláusula **select**. Al igual que otros operadores este tiene su inverso. El conector **not in** prueba la no pertenencia al conjunto.

- **Is NULL** e **Is NOT NULL**: permitirá darle a un atributo, excepto el PK (Primary Key), un valor nulo.

  - Un valor nulo no es lo mismo que un cero, ni espacios en blanco.
  - El cero es un número, NULL no.
  - NULL significa que el valor es desconocido, es faltante o no se aplica y "no" debe ser tratado como un cero.
  - No ocupa espacios en la base de datos.

- **Not**: Permitirá realizar la negación de un operador.

Comandos de sql estándar:

Estos comandos son usados para crear, almacenar, cambiar, recuperar y dar mantenimiento a la información en una base de datos relacional. Estos comandos no necesitan ";"al ejecutarse. Por lo tanto, los comando SQL se almacenan en el buffer de SQL, hasta que otro comando es tecleado.

A continuación, se darán los nombres y conceptos de cada uno de los comandos:

- **Alter:** Altera la estructura de una tabla.
- **Audit:** Activa el proceso de auditoría interna.
- **Commit:** Realiza físicamente el cambio a la base de datos, pero para evitar problemas se debe salvar la información.
- **Comment:** Para escribir documentación en los programas.
- **Create:** Crea los objetos de la base de datos. Estos objetos son vistas, tablas e índices.
- **Delete:** Elimina registros de una base de datos. Para eliminar registros parcialmente se usa **update** con **null** y **where**.
- **Drop:** Elimina objetos de una base de datos.
- **Grant:** Asigna privilegios de un usuario sobre un objeto.
- **Insert:** Inserta datos dentro de una tabla.
- **Lock:** Coloca un candado en un objeto.
- **No audit:** Desactiva el sistema de auditoría de la base.
- **Rename:** Renombra a un objeto de la base de datos.
- **Revoke:** Elimina privilegios a un usuario sobre un objeto.
- **Rollback:** Permite regresar al estado anterior de la base de datos, es decir, hasta el **commit** anterior.
- **Select:** Ejecuta una consulta a la base de datos.
- **Update:** Permite hacer modificaciones a los registros o datos de una tabla.
- **Validate:** Válida la información de la base de datos con respecto a un dominio especificado.

Tipos de datos utilizados en sql

Los tipos de datos que se utilizan en SQL para la creación de una tabla son los siguientes:

- **CHAR (n):** Cadena de hasta n [1,240] caracteres.
- **NUMBER (n,d):** Numérico entero (d = 0) o decimal (d< >0) con n dígitos.
- **DATE:** Dato tipo fecha.
- **LONG:** De hasta 64 kbytes de dato tipo texto.
- **RAW:** Dato binario.

3.3.2 Web.sql

Panorama de web.sql

La llave para tener éxito en cualquier tipo de comercio es entendiendo a los clientes y efectivamente determinando sus necesidades.

Con el último desarrollo en software para manejadores de bases de datos, Sybase ha hecho todo lo posible para lograr este objetivo. Sybase web.sql representa poderosamente una nueva tecnología capaz de alterar radicalmente en línea el mercado y el servicio a clientes.

¿Qué es Web.sql?

Web.sql es una interfaz para bases de datos relacionales como Sybase y se encuentra transparentemente a un lado del servidor HTTP, y se liga directamente con este, web.sql representa una mejor innovación en el desempeño de base de datos relacionales, ya que se tiene mejor integración entre el servidor de HTTP y el servidor de base de datos.

Con web.sql se disminuye el número de documentos web que el servidor HTTP debe manejar, el resultado es dramáticamente mejorado en el acceso a bases de datos relacionales y en tiempos de respuesta desde el World Wide Web. Web.sql permite generar páginas estáticas y dinámicas en lenguaje HTML (Hypertext Markup Language; Lenguaje Hipertexto de Marcas), también permite la creación de templates dinámicos que pueden ser continuamente alterados con nuevo contenido.

El web.sql soporta en línea llamadas a scripts de SQL y de Perl antes de que se separe el script que es requerido por el CGI, es decir, con web.sql se pueden insertar instrucciones de bases de datos y de Perl dentro del texto de una página HTML. Cuando el browser solicita o hace una petición a estas páginas, web.sql corre los scripts y el resultado final es interpolado dentro del archivo

El web.sql reconoce dos diferentes extensiones de archivo: .hts y .pl. Cuando los archivos tienen extensión .hts, el browser recibe únicamente a la salida código HTML, las extensiones soportadas por web.sql son procesadas en una parte del servidor. Sin embargo, si se desea que el browser reciba otro tipo de documentos, tales como archivos .gif, entonces se tendrá que especificar el tipo de contenido del documento en el archivo .pl antes de enviar los datos al browser.

La función ws_content_type() de web.sql permite especificar el tipo de contenido.

El web.sql incluye soporte para Native Netscape Server API (NSAPI), ofrece escalabilidad y soporta el incremento del tráfico en el Web, a través de mantener la conexión de bases de datos abiertas mejorando el tiempo de respuesta y la sobre carga del sistema.

El web.sql integra perfectamente dentro de la arquitectura empresarial de Sybase una construcción capaz para accesar a Sybase IQ: libera sofisticados análisis de datos, Replication Server: tiempo real de replicación, SQL Server: soporta voluminosos sistemas en paralelo, y Enterprise CONNECT: el cuál provee acceso múltiple a los datos.

Componentes en web.sql

Para analizar más detalladamente los componentes de web.sql se dará una breve explicación de como esta integrada la arquitectura de un servidor de Web, la figura 3.12 muestra la típica arquitectura de un servidor de Web. El browser realiza una petición al servidor Web especificando un URL (Uniform Resource Location).
El servidor HTTP interpreta este URL dentro de la ruta de un archivo en el servidor. Si el archivo tiene un formato .html, .gif, .jpeg, u otro tipo de formato de archivo que el servidor Web reconozca, el servidor HTTP envía el archivo directamente al browser.
Si el archivo es un programa residiendo en un directorio autorizado, el servidor HTTP ejecuta el programa acorde con el CGI (Common Gateway Interface) y envía la salida del programa al browser.

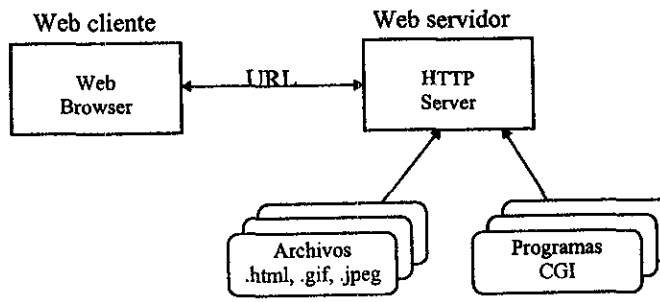


Fig. 3.12. Arquitectura típica del servidor Web

La figura 3.13 muestra la arquitectura de un servidor Web con web.sql. El servidor Web manipula las peticiones por medio de un archivo HTML justo como en la figura 1. Sin

embargo, cuando un browser solicita un URL que es interpretado por un archivo .hts (HyperText Sybase) o por un archivo .pl de Perl (solo Perl, no HTML), el servidor HTTP pasa la petición al programa web.sql.

El programa web.sql lee el archivo HTS (HyperText Sybase), procesa las peticiones de bases de datos y de Perl contenidas en el archivo, y compone la salida en el formato estándar HTML para el servidor HTTP, y éste, lo pasa al browser. El programa web.sql usa un mapa de acceso a bases de datos para determinar cual base de datos, login y password se usará por cada archivo HTS (.hts o .pl).

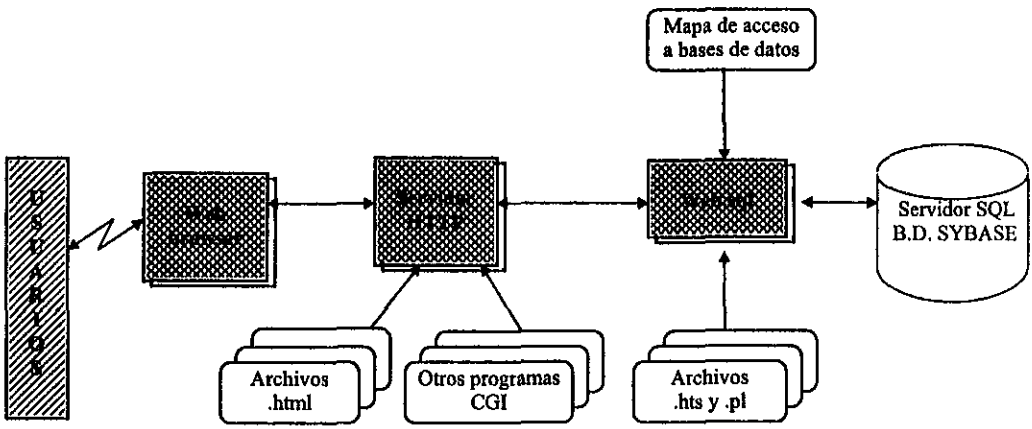


Fig.3.13 Arquitectura del servidor Web con web.sql

El browser accesa archivos HTS (.hts o .pl) como cualquier otra página Web y recibe los datos en el estándar HTML. El browser no requiere de extensiones especiales o aplicaciones de ayuda.

El programa web.sql puede ser usado para regresar páginas dinámicas HTML al browser con la ayuda de un archivo HTS. Web.sql puede correr un script de Perl para regresar: texto, imágenes, sonido u otros datos que el browser pueda desplegar o pasar a una aplicación de ayuda. El script de Perl puede interactuar con una base de datos como un archivo HTS. Por ejemplo se puede usar un script de Perl para recuperar una imagen de una base de datos.

Hay dos versiones de web.sql disponibles: CGI y NSAPI (Netscape Server Application Programming Interface). En la versión de CGI, web.sql corre como un programa CGI. El servidor HTTP corre el programa cada vez que recibe una petición de un archivo HTS.

Sobre una plataforma UNIX, la versión NSAPI de web.sql se liga directamente dentro del servidor de Netscape HTTP el cuál elimina la sobrecarga de los resultados desde la

inicialización del proceso web.sql para cada petición HTS o Perl. Esta versión tiene alto desempeño ya que permite mejorar la conexión a las bases de datos, reduciendo la sobrecarga.

Archivos de formato HTS (HyperText Sybase)

El formato de archivo HTS es una extensión del formato estándar HTML. Se puede incluir cualquier cosa que se quiera de un archivo HTML en un archivo HTS, además se pueden incluir otras extensiones de etiquetas tales como Java o JavaScript.

Los archivos HTS soportan todas las etiquetas de HTML 3.0 y una etiqueta adicional, <SYB> para la funcionalidad de web.sql.
Web.sql interpreta todas las líneas entre la etiqueta <SYB> y su correspondiente etiqueta </SYB> para cualquiera de los dos códigos: transact-SQL o Perl. Inserta la salida de estas declaraciones dentro del código HTML y son enviadas al servidor HTTP.

La etiqueta <SYB> tiene un atributo opcional TYPE. Incluyendo "TYPE=SQL" el cual indica que el bloque de <SYB> contendrá únicamente sentencias de Transact-SQL, mientras que con "TYPE=PERL" indica que el bloque contendrá sentencias en Perl, (Las sentencias de Perl pueden contener funciones que llamen a ejecutar sentencias en Transact-SQL). Si no se incluye el atributo TYPE, el procesador de web.sql asume que el bloque <SYB> contiene código en Perl.

Nota: Como otras etiquetas de HTML, la etiqueta de <SYB> y sus atributos no son sensitivos a mayúsculas o minúsculas, para este caso se puede usar upper, lower o mixed case.

El archivo de formato HTS regresa únicamente contenido HTML a la salida. Sin embargo si se desea tener otro tipo de contenido a la salida que el tipo de contenido HTML se necesita usar un archivo Perl (.pl) y especificar el tipo de contenido, usando la función: ws_content_type(). Por ejemplo en un archivo que se llama *contenido.pl,* se puede especificar el tipo de contenido como "image/gif" y entonces imprimir el archivo, como por ejemplo:

```
ws_content_type("image/gif")
print "cat contenido.gif";
```

En seguida se muestra un ejemplo de un archivo HTS, que muestra como desplegar la fecha.

Cuando el browser realiza una petición a un archivo .HTS, el procesador de web.sql ejecuta las líneas que aparecen dentro de las etiquetas de <SYB> y </SYB>, y busca el día y la hora en el servidor e imprime los resultados a la salida estándar con etiquetas de HTML.

El programa de web.sql ejecuta el código e inserta la salida dentro de HTML y los envía al servidor de HTTP, como se muestra a continuación:

```
<HTML>
<HEAD>
<TITLE> Ejemplo de Perl </TITLE>
</HEAD>
<BODY>
<H2><CENTER> Ejemplo de Sybase SQL con Perl </CENTER><H2>

<SYB TYPE=PERL>
        require "ctime.pl";
        print "<P>";
        print "Fecha: ", "<STRONG>", &ctime(time), "</STRONG>";
        print "<P>";
</SYB>

</BODY>
</HTML>
```

El resultado en HTML enviado al servidor es:

```
<HTML>
<HEAD>
<TITLE> Ejemplo de Perl </TITLE>
</HEAD>
<BODY>
<H2><CENTER> Ejemplo de Sybase SQL con Perl </CENTER></H2>


<P><HR>Fecha: <STRONG>Wed Jan 14 15:21:42 México/General 1998 </STRONG>
<P><HR>
</BODY>
</HTML>
```

API's de web.sql: Convenience y Client-Library

Web.sql incluye dos APIs (Application Programming Interfaces) en Perl para acceso a bases de datos:

- El API de web.sql "convenience", que se usa para obtener los resultados generados por el servidor de SQL en tablas de HTML que se generan rápido y automáticamente.
- El API web.sql Client-Library, para cuando se quieren manipular los datos regresados por el servidor renglón por renglón. Provee una interfaz similar al API de Sybase Open Client-Library (CT lib). Esta interfaz es más compleja que el API de Convenience, pero se tiene más control en la interacción del Servidor SQL, el web.sql Convenience requiere de menos programación.

Usando el API de web.sql Convenience

El API de web.sql Convenience provee un conjunto de rutinas que cumplen las tareas más comunes en un archivo HTS. Se pueden usar estas rutinas individualmente para un mejor desempeño en la interacción con la base de datos o se pueden combinar estas rutinas con las Client-Library.

En la tabla 3.21 se listan las rutinas del API Convenience y su descripción.

Función	Descripción
ws_connect	Conecta al servidor SQL y regresa una conexión manejable
ws_content_type	Conjunto de tipos de contenidos para los datos regresado por un archivo .pl.
ws_error	Imprime un mensaje de error, una cadena opcional, y termina el procesamiento de la página actual.
ws_fetch_rows	Trae e imprime renglones regresados por una llamada a **ct_sql.**
ws_print	Imprime una cadena, expandiendo una referencia a una variable de perl.
ws_sql	Ejecuta uno o más comandos de SQL e imprime el resultado.
ws_rpc	Ejecuta un procedimiento almacenado, actualiza, e imprime los resultados.

Tabla 3.21 Rutinas API de web.sql Convenience

Usando el API de web.sql Client-Library

En web.sql Client-Library se pueden manipular los datos regresados por el servidor renglón por renglón. Éstas proveen una interfaz similar a las de Sybase Open Client Client-Library (CT Lib). Esta interfaz es más compleja que las API Convenience, pero se tiene más control en la interacción con el servidor de SQL.

Las Client-Library difieren de las Open Client en diversos caminos, programando, facilitando y reflejando las diferencias entre programar en C y programar en Perl:

- Las API's de web.sql no incluye todas las rutinas de Open Client Client-Library.
- El formato de las rutinas de web.sql difiere de las Open Client.
- Algunas rutinas de web.sql no soportan el valor de parámetros soportados por las Open Client.

Se pueden usar las rutinas de web.sql Client-Library en conjunción con las API Convenience de web.sql. Por ejemplo se puede usar **ct_connect** para conectarse a la base de datos, **ws_error** para mandar un error y **ct_fetch** para buscar un renglón del resultado de la consulta.

La tabla 3.22 lista las rutinas de web.sql Client-Library

Función	**Descripción**
ct_callback	Instala o restaura una rutina callback para manipular errores.
ct_cancel	Cancela un comando o el resultado de un comando.
ct_col_types	Recupera un arreglo de tipo de columnas para el resultado de una consulta.
ct_connect	Establece la conexión de una base de datos al servidor.
ct_fetch	Obtiene un renglón del resultado de los datos.
ct_fetch_parameters	Modifica las variables de los parámetros de salida suministrados por ct_rpc.
ct_options	Establece, recupera o limpia los valores de la consulta del servidor.
ct_res_info	Recupera información acerca de un conjunto de resultados o comando.
ct_results	Determina el estado de un comando SQL y el tipo de resultado que regresa.
ct_rpc	Llama un procedimiento almacenado que reside en un servidor remoto.
ct_sql	Envía uno o más comandos de SQL al servidor de la base de datos.

Tabla 3.22 Rutinas API de Web.sql Client-Library

Accesando Archivo HTS

Se puede accesar a un archivo HTS por medio de un browser usando URL´s justo como se accesa a cualquier otro archivo Web.

- Si se está usando la versión NSAPI de web.sql sobre la plataforma UNIX:

http://<servername>/<websql.dir>/welcome.hts

<servername> es la dirección IP (hostname) y el número de puerto del servidor web, *<websql.dir>* es la ruta del subdirectorio de web.sql en los documentos del directorio root , y *welcome.hts* es el nombre del archivo HTS.

- Si se esta usando la versión (high-performance) CGI de web.sql en una plataforma UNIX:

http://<servername>/<cgi-name>/ws.exe/<websql.dir>/welcome.hts

<servername> es la dirección IP (hostname) y el número de puerto del servidor web, *<cgi-name>* es el nombre del directorio del script CGI en el servidor Web, **ws.exe** es el nombre del programa de CGI web.sql, *<websql.dir>* es la ruta del subdirectorio de web.sql en los documentos del directorio root , y *welcome.hts* es el nombre del archivo HTS.

- Si se esta usando la versión de CGI shared-library de web.sql en una plataforma UNIX:

http://<servername>/<cgi-name>/websql/<websql.dir>/welcome.hts

<servername> es la dirección IP (hostname) y el número de puerto del servidor web, *<cgi-name>* es el nombre del directorio del script CGI en el servidor Web, **websql** es el nombre del programa de CGI web.sql, *<websql.dir>* es la ruta del subdirectorio de web.sql en los documentos del directorio root , y *welcome.hts* es el nombre del archivo HTS.

- Si se esta usando la versión NSAPIA de web.sql sobre una plataforma NT:

http://<servername>/<websql>/welcome.hts

<servername> es la dirección IP (hostname) y el número de puerto del servidor web, *<websql>* es la ruta del subdirectorio de web.sql en los documentos del directorio root, y *welcome.hts* es el nombre del archivo HTS.

- Si se usa la versión de CGI de web.sql en una plataforma NT:

http://<servername>/<cgi-name>/ws.exe/<websql>/welcome.hts

<servername> es la dirección IP (hostname) y el número de puerto del servidor web, *<cgi-name>* es el nombre del directorio del script CGI en el servidor Web, **ws.exe** es el nombre del programa de CGI web.sql, *<websql>* es la ruta del subdirectorio de web.sql en los documentos del directorio root , y *welcome.hts* es el nombre del archivo HTS.

Nota: Antes de especificar el nombre del programa CGI web.sql en web.sql CGI URL´s la información del documento root y el nombre del archivo HTS son pasados al programa de CGI web.sql.

Requerimientos para Sybase web.sql:

Sybase web.sql corre en las siguientes plataformas: Sun Solaris de Sun Microsystems, IRIX de Silicon Graphics, Inc., HP-UX de Hewlett Packard y Microsoft Windows NT de Intel.
Las tabla 3.23, 3.24, 3.25, 3.26 y 3.27 muestran los requerimientos disponibles de web.sql en versión Alpha y Beta para las siguientes plataformas:

Requerimientos disponibles para:
- Sun SPARC Solaris 1.1 Beta 1
- Windows NT 1.1 Beta1
- Sun SPARC Solaris 1.0 GA
- Silicon Graphics IRIX 1.0 GA
- Hewlett Packard HP-UX 1.0 Alpha

Requerimientos disponibles para web.sql 1.1 Beta para Sun Solaris	
Hardware:	Sun SPARCstation
Sistema Operativo:	Solaris 2.4, 2.5 ó 2.5.1
Memoria:	16 MB mínimo
Espacio libre en disco:	14 MB
Base de Datos	Sybase SQL Server System 10(10.0.2 ó 10.0.3), 11 ó 11.0.1 corriendo en cualquier plataforma en la red.
Servidor Web (Web Server)	• Para la versión de web.sql CGI: Cualquier Web server que soporte CGI ver.1.1 (Netscape NCSA, Open Market, etc.). • Para la versión de web.sql NSAPI: Netscape Communications Server ver. 1.10, Commerce Server ver.1.12, FastTrack Server ver.2.0, Enterprise Server ver.2.0.
Cliente Web (Web Browser)	Un Web Borwser que soporte tablas HTML 3.0, web.sql requiere HTML 2.0 como mínimo, pero HTML 3.0 es recomendado desde que soporta tablas web.sql no usa cualquier web browser. Éste provee soporte para algunos tales como el método post query y tablas que pueden no estar disponibles en todos los browser.

Tabla 3.23 Requerimientos para web.sql 1.1 Beta para Sun Solaris

Requerimientos disponibles para web.sql 1.1 Beta para Windows NT	
Hardware:	Una PC con 486 o procesador Intel, monitor VGA color
Sistema Operativo:	Windows NT 3.5.1
Memoria:	16 MB mínimo
Espacio libre en disco:	35 MB
Base de Datos	Sybase SQL Server System 10(10.0.2 ó 10.0.3), 11 ó 11.0.1 corriendo en cualquier plataforma en la red.
Servidor Web (Web Server)	• Para la versión de web.sql CGI: Cualquier servidor Web que soporte CGI ver.1.1 en Win NT tales como (Netscape i.x Netscape 2.0, WebSite 1.1, Microsoft Internet Information Server 1.0). • Para la versión de web.sql NSAPI: Netscape Communications Server ver. 1.12, FastTrack Server ver.2.0, Enterprise Server ver2.0.
Cliente Web (Web Browser)	Un Web Borwser que soporte tablas HTML 3.0, web.sql requiere HTML 2.0 como mínimo, pero HTML 3.0 es recomendado desde que soporta tablas web.sql no usa cualquier web browser. Éste provee soporte para algunos tales como el método post query y tablas que pueden no estar disponibles en todos los browser soporta para algunos tales como el método post query y tablas que pueden no estar disponibles en todos los browser.

Tabla 3.24 Requerimientos para web.sql 1.1 Beta para Windows NT

Requerimientos disponibles para web.sql 1.0 GA para Sun Solaris esta disponible para 90 días libres	
Hardware:	Sun SPARCstation
Sistema Operativo:	Solaris 2.4 ó 2.5
Memoria:	16 MB mínimo
Espacio libre en disco:	12 MB
Base de Datos	Sybase SQL Server System 10(10.0.2 ó 10.0.3), 11 corriendo en cualquier plataforma en la red.
Servidor Web (Web Server)	• Para la versión de web.sql CGI: Cualquier Web server que soporte CGI ver.1.1 (Netscape NCSA, Open Market, etc.). • Para la versión de web.sql NSAPI: Netscape Communications Server ver. 1.10, Commerce Server ver.1.1 ó 1.12. Nota: web.sql 1.0 no trabaja con el Netscape 2.0

	servers).
Cliente Web (Web Browser)	Un Web Borwser que soporte tablas HTML 3.0, web.sql requiere HTML 2.0 como mínimo, pero HTML 3.0 es recomendado desde que soporta tablas web.sql no usa cualquier web browser. Éste provee soporte para algunos tales como el método post query y tablas que pueden no estar disponibles en todos los browser.

Tabla 3.25 Requerimientos para web.sql 1.0 GA para Sun Solaris

Requerimientos disponibles para web.sql 1.0 GA para SGI IRIX esta disponible para 90 dias libres	
Hardware:	Silicon Graphics worksation
Sistema Operativo:	IRIX 5.3/6.2
Memoria:	16 MB mínimo
Espacio libre en disco:	12 MB
Base de Datos	Sybase SQL Server System 10(10.0.2 ó 10.0.3), 11 corriendo en cualquier plataforma en la red.
Servidor Web (Web Server)	• Para la versión de web.sql CGI: Cualquier Web server que soporte CGI ver.1.1 (Netscape NCSA, Open Market, etc.). • Para la versión de web.sql NSAPI: Netscape Communications Server ver. 1.10, Commerce Server ver.1.1 ó 1.12. Nota: web.sql 1.0 no trabaja con el Netscape 2.0 servers).
	Un Web Borwser que soporte tablas HTML 3.0, web.sql requiere HTML 2.0 como mínimo, pero HTML 3.0 es recomendado desde que soporta tablas web.sql no usa cualquier web browser. Éste provee soporte para algunos tales como el método post query y tablas que pueden no estar disponibles en todos los browser.

Tabla 3.26 Requerimientos para web.sql 1.0 GA para SGI IRIX

Requerimientos disponibles para web.sql 1.0 "Alpha" para HP-UX y es gratis:	
Hardware:	*HP-UX workstation*
Sistema Operativo:	HP-UX 9.x
Memoria:	16 MB mínimo
Espacio libre en disco:	12 MB
Base de Datos	Sybase SQL Server System 10(10.0.2 ó 10.0.3), 11 *corriendo en cualquier plataforma* en la red.
Servidor Web (Web Server)	• Para la versión de web.sql CGI: Cualquier Web server que soporte CGI ver.1.1 (Netscape NCSA, Open Market, etc.).
Cliente Web (Web Browser)	Un Web Borwser que soporte tablas HTML 3.0, *web.sql requiere* HTML 2.0 como mínimo, pero HTML 3.0 es recomendado desde que soporta tablas web.sql no usa cualquier web browser. Éste provee soporte para algunos tales como el método post query y tablas que pueden no estar disponibles en todos los browser.

Tabla 3.27 Requerimientos para web.sql 1.0 "Alpha" para HP-UX

COMPONENTES DEL SISTEMA DEL SERVICIO DE ALERTA DEL INSTITUTO DE INGENIERÍA

CAPÍTULO 4
COMPONENTES DEL SISTEMA DEL SERVICIO DE ALERTA DEL INSTITUTO DE INGENIERÍA

4.1 Necesidades (Descripción del problema).

Día con día más personas están entrando al mundo de Internet, con el WWW (World Wide Web - Red Mundial de Cobertura Amplia), muchas de las empresas comienzan a utilizar dicha red para facilitar su comunicación y procesos, que se llevan a cabo dentro de ella. Estos procesos se logran automatizar cuando los trabajos que se hacían manualmente son sustituidos por las diferentes herramientas de Internet, las cuales facilitan y agilizan el manejo de éstos. Entre los posibles beneficios podemos mencionar: organizar y centralizar información, rapidez en el desarrollo de proyectos, distribución de aplicaciones, decremento de costos y aumentar la eficiencia.

Es por esto que el Instituto automatizará sus procesos emigrando a esta nueva tecnología en el Web del Instituto de Ingeniería.

A continuación se especificarán las necesidades y problemas que actualmente existen en la *Unidad de Servicios de Información* (U.S.I):

> *La Unidad de Servicios de Información es la encargada de adquirir las revistas que necesitan los investigadores para el desarrollo de sus labores diarias. En el caso de las revistas, la U.S.I. se encarga de solicitar las revistas, libros, documentos, etc., de tal manera que se puede tener "a vistas" para su futura compra si el ejemplar es requerido. La U.S.I. adquiere el ejemplar a través de un proyecto que el investigador le proporciona; ésta lo da de alta en su sistema de revistas; y se encarga de fotocopiar las tablas de contenido de dicha revista para colocarla en un mostrador y posteriormente asignarla a la coordinación correspondiente.*
>
> El proceso que se lleva para consultar y solicitar una revista en particular es el siguiente:
>
> *El investigador tiene que ir a consultar las fotocopias de los últimos ejemplares de las tablas de contenido, en el mostrador que se encuentran en la U.S.I., si el investigador se interesa por algún tema y desea solicitar el préstamo de la revista, la U.S.I. verifica en qué coordinación del Instituto de Ingeniería se encuentra físicamente la revista para pedirla, una vez que se tiene la revista el investigador llena una solicitud de préstamo y finalmente se le entrega.*

Todo este proceso es engorroso tanto para las personas de la U.S.I. como para los investigadores, lo cual origina la problemática que a continuación se presenta:

- *Uno de los problemas existentes es que muy pocas personas se enteran de la existencia de los ejemplares con los que cuenta el Instituto de Ingeniería, por lo que las personas de la U.S.I. no pueden saber qué temas de revistas son los más solicitados.*

- *Debido a un control deficiente de registros de revistas se originan compras repetidas, y pérdidas de ejemplares, puesto que a veces se pierden o se traspapelan las formas de solicitud de préstamo y no se sabe quién posee la revista.*

- *Otro problema es el costo que implica el fotocopiar las tablas de contenido de cada revista para que los investigadores puedan consultarlas.*

- *Finalmente, el tiempo perdido de los investigadores para ir a la U.S.I. y consultar las tablas de contenido, para posteriormente solicitar el préstamo.*

Por estas circunstancias se procedió a automatizar este proceso para corregir la problemática anteriormente citada.

En la siguiente sección se describirá el análisis que se siguió para llevar a cabo el sistema.

4.2 Análisis.

Uno de los medios de mayor difusión es el WWW. Dicho medio permite a personas de todo el mundo mantenerse actualizados con diversos tipos de información, así como agilizar algunos procesos desde una computadora personal o una estación de trabajo.
Pensando en aprovechar las ventajas que ofrece el WWW, se propuso un sistema de *Servicio de Alerta* en Internet, el cual permitirá:

- Que los usuarios consulten de manera rápida y eficaz las tablas de contenido de las revistas desde el Web del Instituto de Ingeniería.

- Difundir la gran cantidad de ejemplares (más de 100 títulos de revistas) que llegan a la U.S.I., y

- Agilizar las tareas que se desarrollaban anteriormente por las personas encargadas de la U.S.I.

Basados en el objetivo anteriormente presentado, el sistema de *Servicio de Alerta* propuesto consistirá de lo siguiente :

- *Sección de Búsqueda.* En esta sección el usuario podrá buscar una revista para consultar las tablas de contenido por medio del título de la revista, por coordinación dónde se encuentra físicamente la revista y por el tema asociado a la revista.

- *Sección de Solicitud de Préstamo.* El sistema dará un servicio de solicitud de préstamo en el cual el usuario consultará las tablas de contenido a través del Web y podrá llenar una solicitud de préstamo que contendrá los datos del usuario y de la revista dentro de una página Web.

- *Sección de Suscripción al Servicio de Alerta.* El sistema contará con un servicio de suscripción, donde el usuario después de localizar una revista podrá suscribirse a ésta desde una página web, suministrando información como: datos personales del usuario, correo electrónico, así como un login y un password que se usará posteriormente para el mantenimiento de la lista de revistas a las cuales el usuario estará suscrito para el servicio de alerta.

- *Sección de agregar revista al Servicio de Alerta.* El usuario a través del login y password, podrá agregar la revista que desee a su lista de suscripción del servicio de alerta.

- *Sección de consultas.* El usuario podrá ver la lista de revistas que tenga en el servicio de alerta, a través del login y password, y seleccionará alguna revista para consultar las tablas de contenido.

- *Sección de eliminar revista de la lista de suscripción al Servicio de Alerta.* El usuario a través del login y password, podrá ver la lista de revistas que tiene en el servicio de alerta y seleccionará ,la revista que desee eliminar.

Para que las personas encargadas de la U.S.I. puedan administrar las revistas, los préstamos y suscripciones de los usuarios, se propuso desarrollar una interfaz que fuese fácil de manejar y que tuviese una presentación agradable al usuario.
Por esta razón se optó por utilizar la herramienta de Visual Basic, debido a que se tenía disponible en el Instituto y además cumplía con las especificaciones anteriormente citadas.

Esta interfaz consistirá de lo siguiente:

- *Sección de Revistas:* En esta sección las personas encargadas de la U.S.I. podrán agregar una nueva revista, buscar una revista en particular, eliminar cualquier revista, imprimir todas las revistas e imprimir las revistas con sus temas asociados, y asociar temas a una revista.

- *Sección de Usuario:*. En esta sección las personas encargadas de la U.S.I. podrán agregar, buscar, eliminar e imprimir a los usuarios que hayan solicitado un préstamo o una suscripción al servicio de alerta.

- *Sección de solicitud de préstamo:* En esta sección las personas encargadas de la U.S.I. podrán agregar, buscar, eliminar e imprimir los préstamos que realicen los usuarios

- *Sección de suscripción al Servicio de Alerta:* En esta sección las personas encargadas de la U.S.I. podrán agregar, buscar, eliminar e imprimir las suscripciones que realicen los usuarios

- *Sección de últimos ejemplares*: En esta sección las personas encargadas de la U.S.I. actualizarán las tablas de contenido de recién ingreso a la U.S.I. y se reflejaran en la interfaz del web.

Con este sistema se pretende mejorar la problemática existente, ya que los usuarios no tendrán que ir hasta la U.S.I. para consultar las tablas de contenido, pues las podrán consultar desde su propia computadora personal o estación de trabajo a través del Web, lo cual implica un menor tiempo de respuesta para el usuario.

Con este sistema la U.S.I. dará un mejor servicio y organizará sus tareas realizadas. Uno de sus servicios es notificar al usuario la llegada de revistas, por medio de un correo electrónico.

4.3 Diseño:

El sistema propuesto está basado en una arquitectura cliente-servidor, en el cual, en la interfaz del cliente se tiene por un lado al **Web** para consulta de revistas, servicios de préstamo y suscripción y, por otro lado, a **Visual Basic** para la administración del sistema.

Con el análisis realizado en la sección anterior, se diseñó cada una de las secciones que conforman el sistema, proporcionando una visión general de su funcionamiento.

4.3.1 Web

En las figuras 4.1, 4.2, 4.3, 4.4 y 4.5 muestran los esquemas elaborados para la interfaz del web y la simbología utilizada, para cada una de las secciones que conforman el sistema del *Servicio de Alerta*:

Página principal del sistema para el *Servicio de Alerta*

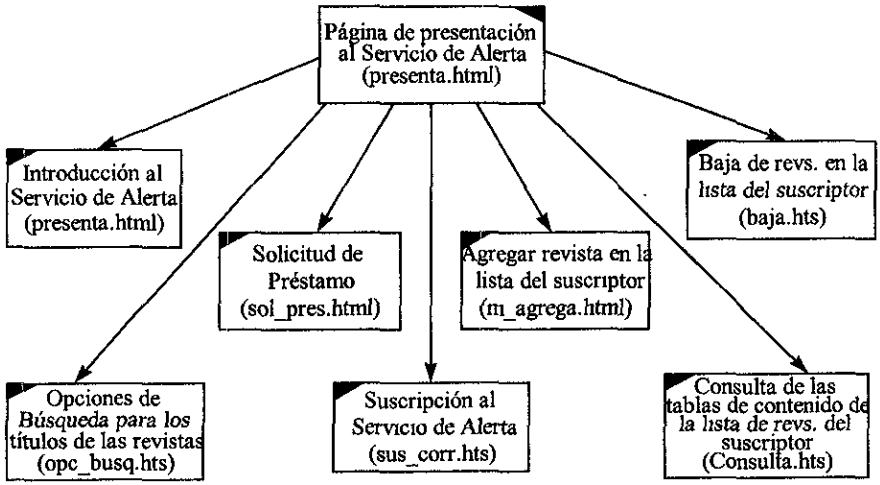
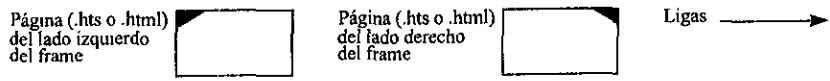


Fig. 4.1 Esquema que muestra como está constituida la página principal del sistema del Servicio de Alerta



Sección de opciones de búsqueda para los títulos de las revistas

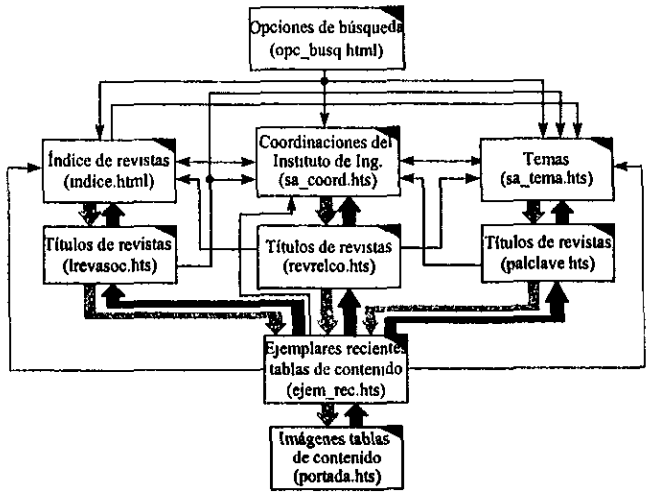


Fig. 4.2 Esquema para la sección de búsqueda

Secciones de solicitud de préstamo, suscripción al *Servicio de Alerta* y agregar revista a la lista del suscriptor

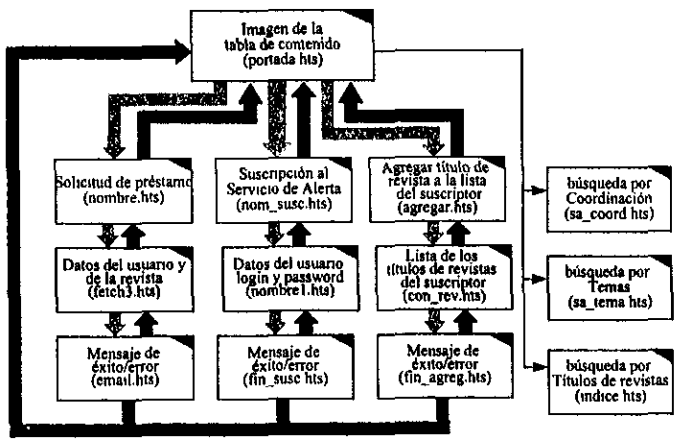
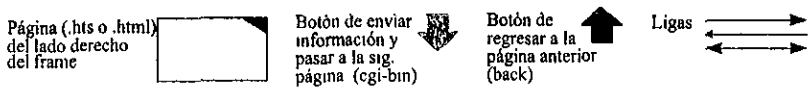


Fig. 4.3 Esquema que representa la página que incluye las secciones de préstamo, suscripción y agregar revista a la lista de suscripción al Servicio de Alerta.



Sección de Consulta a las tablas de contenido de la lista de revistas del suscriptor

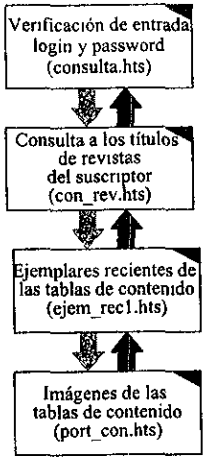


Fig.4.4 Esquema que representa la página de la sección de Consultas

Sección de Bajas para los títulos de revistas de la lista del suscriptor

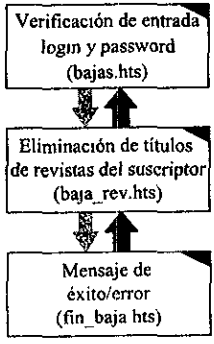
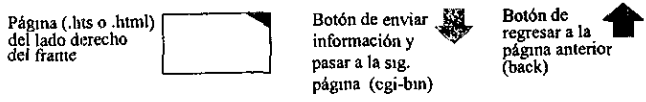


Fig. 4.5 Esquema que representa la página de sección de bajas de revistas de la lista de suscripción.



Con la ayuda de estos esquemas, se procedió a realizar una descripción detallada[9] de cómo funcionaría el sistema, se subrayaron todos los sustantivos y verbos[10] contenidos dentro de la descripción con el fin de identificar los objetos y los métodos, a continuación se listaron, para seleccionar sólo aquellos que fuesen útiles al sistema asociándole a cada objeto sus atributos y métodos correspondientes, y se representaron esquemáticamente.[11]

Para conocer mejor el funcionamiento de la interfaz del web se representaron las responsabilidades de cada objeto por medio de escenarios, (un escenario es el estado o la relación que guardan los objetos en un momento dado). La figura 4.6 muestra un ejemplo del escenario para búsqueda por coordinación:

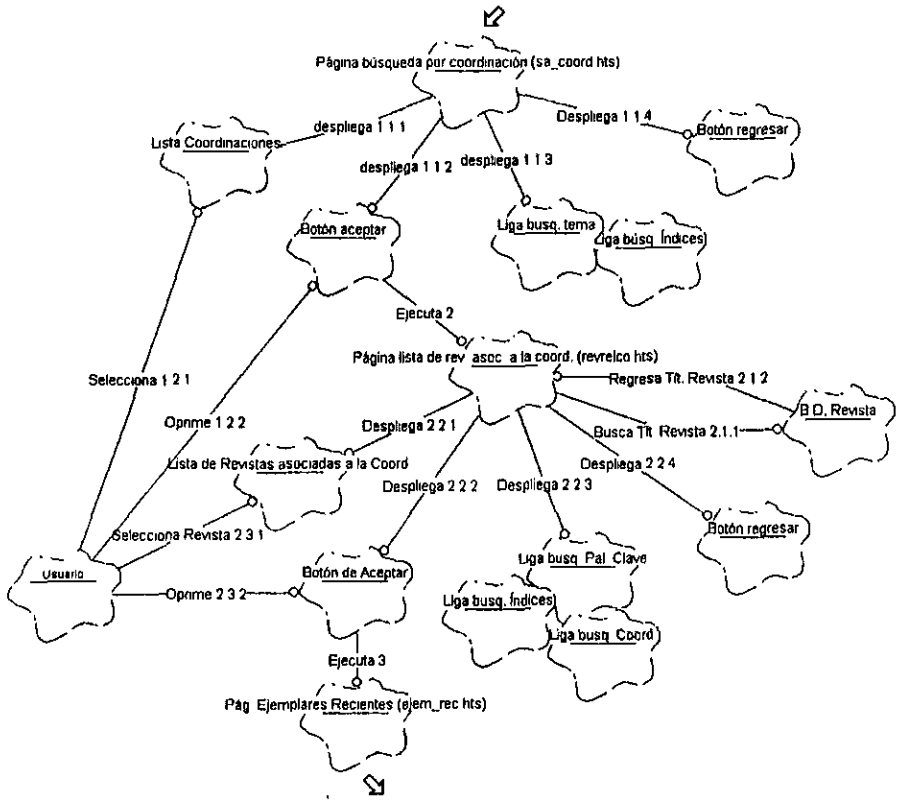


Fig. 4.6 Escenario que representa el método de búsqueda por coordinación

[9] Para mayor información ver la descripción detallada del sistema en el Manual técnico del sistema de Servicio de Alerta: "Desarrollo de sistemas informáticos para la intranet del Instituto de Ingeniería", proyecto 7705, autores: Ing. Gabriel Castillo Hernández, Rocío Gil Castillo y Claudia Vivas Hernández.

[10] Los sustantivos indican los posibles objetos del sistema , y los verbos indican los métodos o acciones que realizarán los objetos del sistema.

[11] Para más detalles, ver representación de objetos con atributos y métodos del informe mencionado en la referencia (9)

Estos escenarios[12] ayudaron a identificar la relación entre los objetos de las diferentes páginas html que conformarían la interfaz del web.

Las páginas del Web se diseñaron de tal manera que al usuario se le hiciera amigable el funcionamiento del sistema en el Web, y ver si se cumplía con los objetivos y especificaciones preestablecidos.

4.3.2 Visual Basic

En las figuras 4.7, 4.8, 4.9, 4.10. 4.11, 4.12 y 4.13 muestran los esquemas para el diseño de la interfaz cliente de Visual Basic:

Esquema de la interfaz de Visual Basic

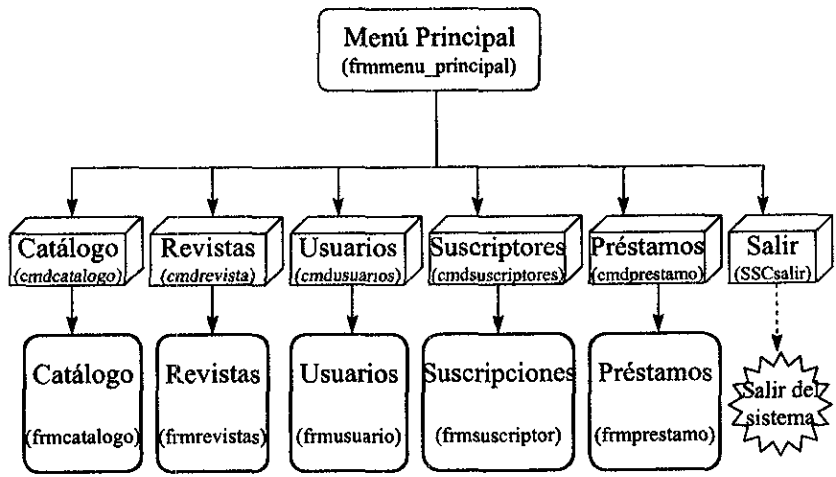
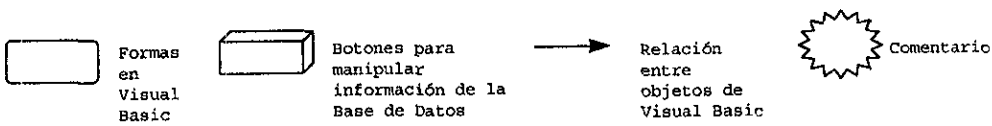


Fig. 4.7 Forma principal de la interfaz de Visual Basic



[12] Por cuestiones de espacio no se incluyen todos los escenarios, para más detalles, ver escenarios del sistema en el informe mencionado en la referencia (9)

Esquema de la forma de Catálogo

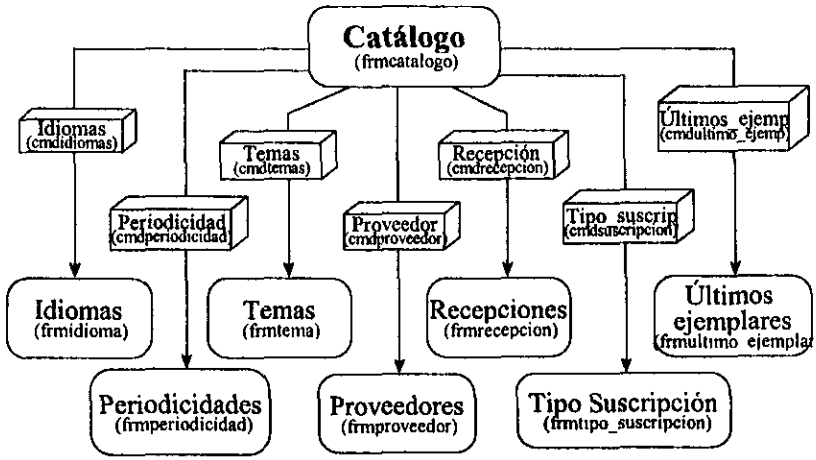


Fig. 4.8 Esquema que representa la forma de catálogo en VB.

Esquema de la Forma de Revistas

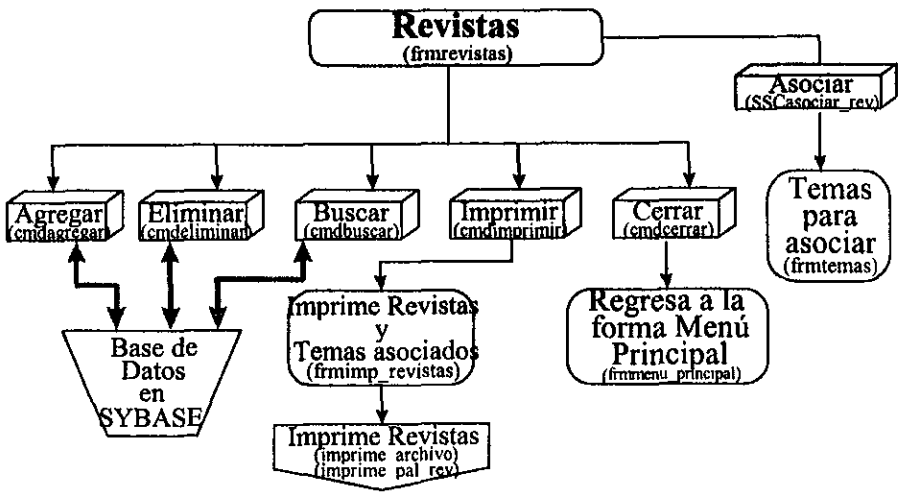
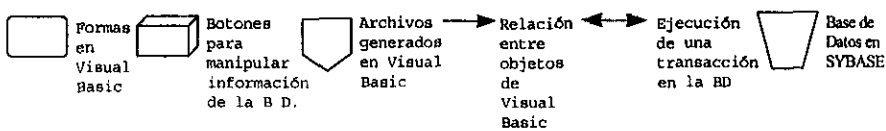


Fig. 4.9 Esquema que representa la forma de revistas en VB.



Esquema de la Forma de Usuarios

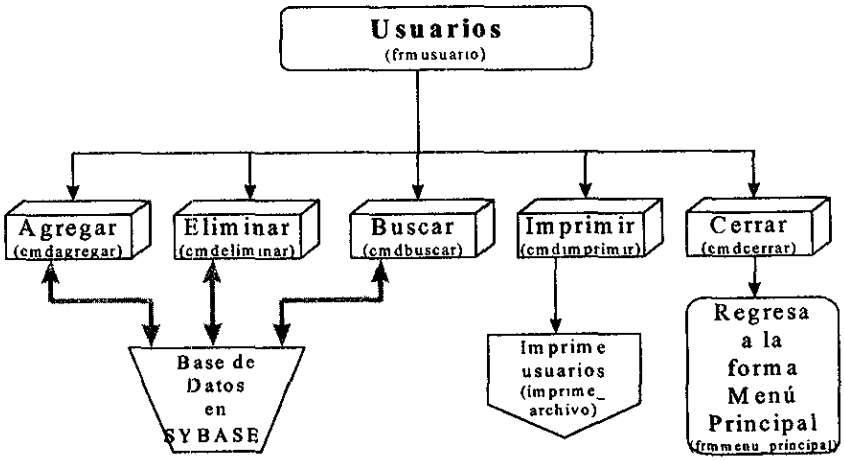


Fig. 4.10 Esquema que representa la forma de Usuarios en VB

Esquema de la Forma de Suscripciones

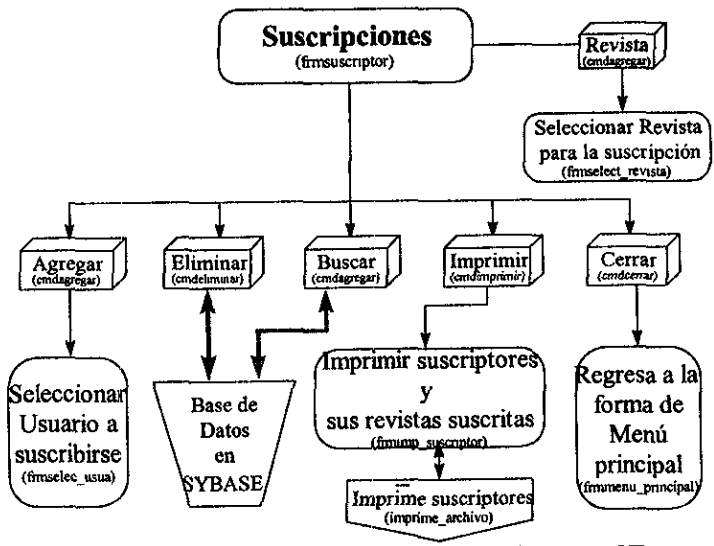
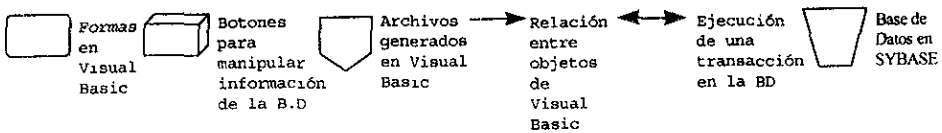


Fig. 4.11 Esquema que representa la forma de suscripciones en VB



Esquema de la Forma de Préstamo

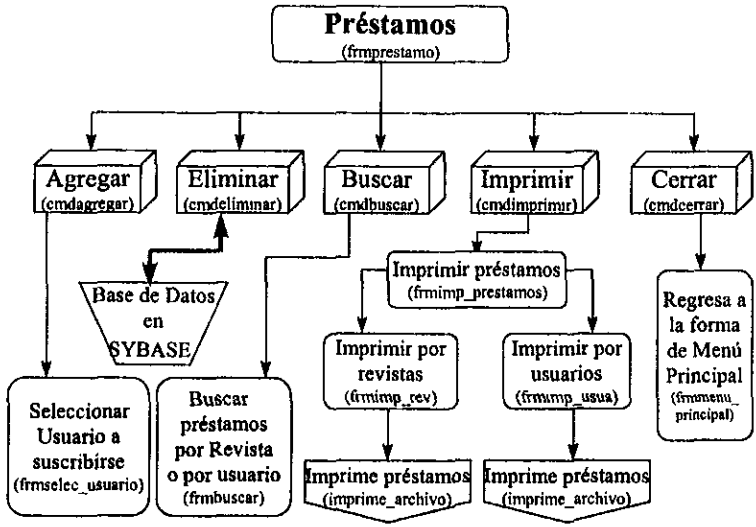


Fig. 4.12 Esquema que representa la forma de préstamos en VB

Esquema de la Forma de Últimos Ejemplares

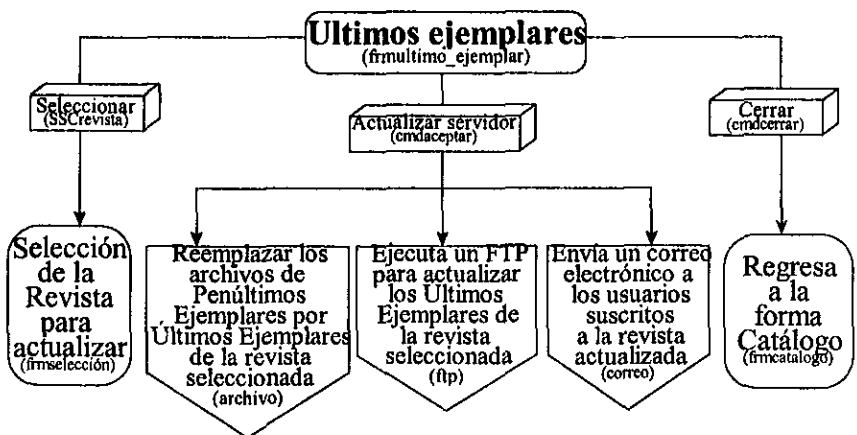
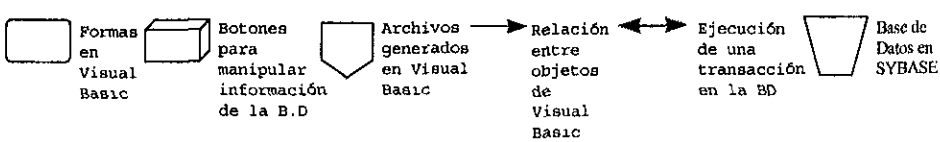


Fig. 4.13 Esquema que representa la forma de Últimos ejemplares en VB



Con la ayuda de estos esquemas y la descripción detallada[13] del sistema se diseñaron las formas que conformarían la interfaz de Visual Basic para la administración de la U.S.I., una vez terminadas las formas, se les presentaron a las personas encargadas de la U.S.I. para acordar si cumplían con los objetivos preestablecidos.

4.3.3 Diagrama Entidad-relación de la base de datos:

El diseño de la base de datos, se facilitó a través de la representación de objetos citados anteriormente, los cuales ayudaron a identificar las posibles tablas con sus respectivos atributos que conformarían la base de datos del sistema. En la figura 4.14 se muestra dicho diagrama.

Diagrama Entidad-relación para la base de datos del sistema del Servicio de Alerta.

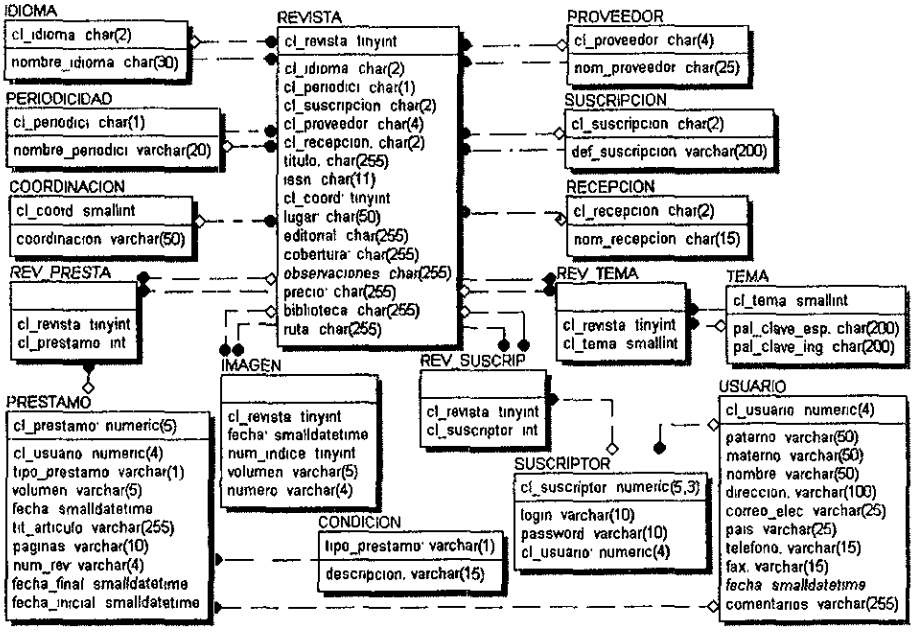


Fig. 4.14 Diagrama entidad-relación para el sistema del Servicio de Alerta.

Para evitar la redundancia en la base de datos del sistema, es decir la repetición de datos y generación de inconsistencia, se utilizaron las formas normales de bases de datos para mantener la integridad referencial, la eficiencia y la seguridad en el sistema.
A continuación se describirá el desarrollo del sistema del servicio de alerta.

[13] Para mayor información ver la descripción detallada del sistema en el informe mencionado en la referencia (9)

4.4 Desarrollo

Para llevar a cabo el desarrollo de las interfaces del sistema y de la base de datos anteriormente citadas, se necesitó la ayuda de herramientas que facilitaran el trabajo, para esto, se aprovecharon las herramientas con las que se contaba en el Instituto de Ingeniería.

Para trabajar con las páginas del web, se cuenta con el *servidor de web NCSA*, el cual permite atender las solicitudes de las páginas que se encuentran en el home page, y con Netscape Comunicator como browser institucional, el cual proporciona un editor de texto para crear páginas HTML.

Para la interfaz de la administración de la U.S.I. se utilizó *Visual Basic*, puesto que se tenía disponible y además cumplía con los requerimientos para el desarrollo de la administración.

Para la creación de la base de datos se utilizó *Sybase* que permite generar bases de datos relacionales y también se tiene disponible en el Instituto.

Además se utilizó web.sql, el cual es una buena opción en el desempeño de bases de datos ya que se liga directamente con el servidor, mejorando la integración entre el servidor Web y la base de datos, proporcionando así rápidos accesos a bases de datos relacionales y en consecuencia un mejor tiempo de respuesta desde el World Wide Web.

Para mostrar como se encuentra integrado el sistema, en la figura 4.15 se presenta la arquitectura general del sistema del Servicio de Alerta.

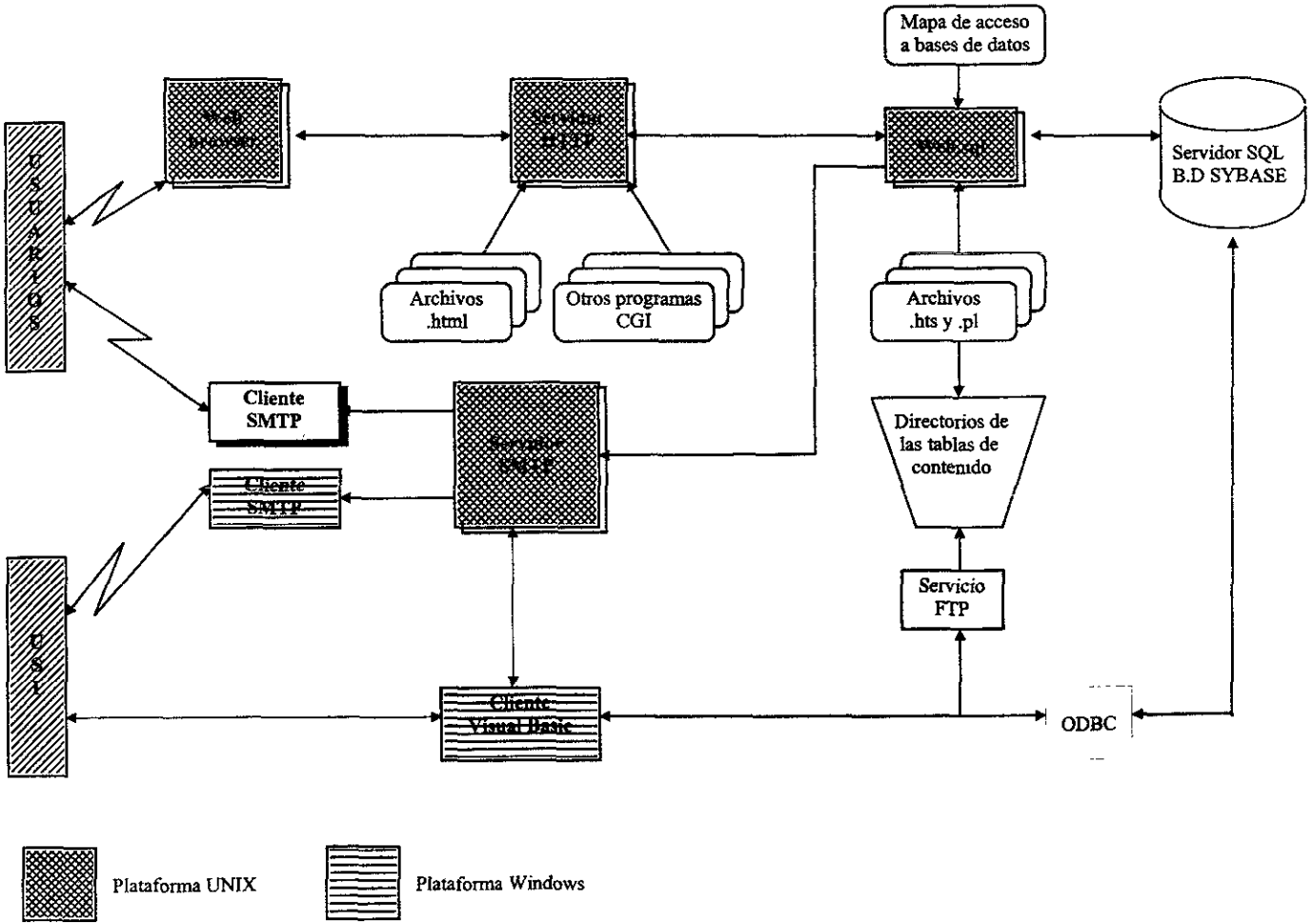


Fig. 4.15 Arquitectura general del sistema para el Servicio de Alerta

A continuación se describirá el funcionamiento de la arquitectura mostrada en la figura 4.15:

Cuando un usuario consulta el web del Instituto, por medio de un browser (Netscape, Internet Explorer u otro), este browser realiza una petición al servidor Web a través de un Uniform Resource Location (URL). Este URL es interpretado por el servidor HTTP dentro de la ruta de un archivo en el servidor. Si el archivo tiene un formato .html, .gif, .jpeg, u otro tipo de formato que el servidor web reconozca, el servidor HTTP envía el archivo directamente al browser.

Sin embargo, cuando un browser solicita un URL que es interpretado por un archivo .hts (HyperText Sybase) o por un archivo .pl de Perl (solo Perl, no HTML), el servidor HTTP envía la petición al programa de web.sql.

El programa de web.sql lee los archivos HTS o .pl, procesa las peticiones de la base de datos, así como el código de Perl contenidos en el archivo utilizando un mapa de acceso para determinar que base de datos se va a utilizar. El programa web.sql compone la salida en el formato estándar HTML para el servidor HTTP y éste finalmente lo envía al browser para desplegarlo al usuario.

Cuando un usuario realiza una solicitud de préstamo, el web.sql envía un mensaje al servidor Simple Mail Transfer Protocol (SMTP), el cual envía un correo electrónico a las personas encargadas de la U.S.I. por medio de un cliente SMTP (Eudora, etc).

La interfaz de Visual Basic se podrá conectar a la Base de Datos por medio del Open Database Connectivity (ODBC).

Cuando los usuarios de la U.S.I. realicen la actualización del último ejemplar de una revista en particular, a través de una forma en Visual Basic, dicho cliente se encargará de reemplazar los archivos .gif (imágenes de las tablas de contenido) de los directorios de penúltimos ejemplares por los archivos .gif contenidos en el directorio de últimos ejemplares, a través de un RPC (Remote Procedure Call). Posteriormente enviará un FTP que transferirá los nuevos archivos .gif para las imágenes de las tablas de contenido al directorio de últimos ejemplares. Por último el cliente Visual Basic enviará un mensaje al servidor SMTP, el cual enviará un correo electrónico a los usuarios suscritos al servicio de alerta, a través del cliente SMTP (Eudora u otro).

A continuación se describirá el desarrollo de las interfaces del Web y de Visual Basic:

El siguiente paso del desarrollo fue la programación de las formas en Visual Basic y las páginas del Web. Este proceso de programación consiste en traducir el diseño lógico de un sistema, a código fuente; para ello, se utilizaron herramientas de software como Visual Basic, web.sql, Perl, html y Java Script.

4.4.1 Web

A continuación se mostrará el desarrollo de tres de las páginas del web más representativas del funcionamiento del sistema. Para este desarrollo se procedió de la siguiente manera:

La figura 4.16 muestra la presentación final de una página web en la sección de búsqueda por coordinación.

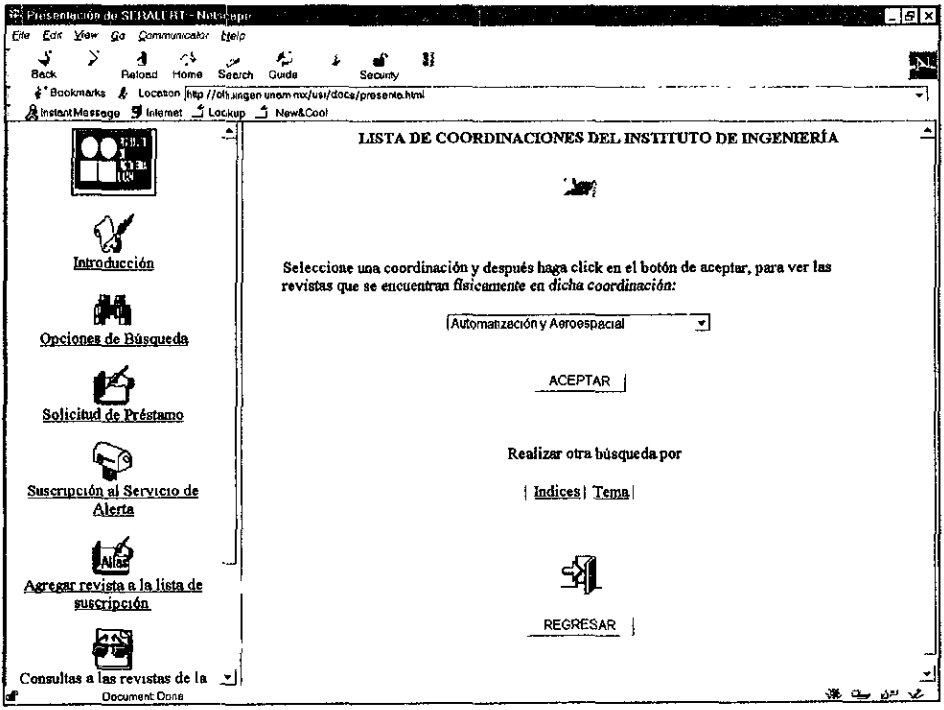


Fig. 4.16 Página que representa la sección de búsqueda por coordinación.

El código que se utilizó para el desarrollo de esta página fue HTML, Perl y Sybase.
Con HTML dentro de una forma se crearon los botones que realizan la acción de submit y back (enviar y regresar), las ligas que llevan a las diferentes opciones de búsqueda, y algunos detalles más como lo son: título de la página, color de fondo y una breve descripción de la función que realiza dicha página como se muestra a continuación:

```
<HTML><HEAD><TITLE>RELACION DE REVISTAS POR COORDINACIÓN</TITLE></HEAD>
<BODY BGCOLOR = "WHITE" TEXT = "#000000" LINK = "#rrggbb" VLINK = "#rrggbb" >
<CENTER><B><blockquote><FONT COLOR="#FF0000">T&Iacute;TULOS DE REVISTAS
ASOCIADAS A LA COORDINACI&Oacute;N </FONT>
</blockquote></B>
<IMG SRC="/usi/imag/books01.gif" ALT="coordinaciones" WIDTH=50 HEIGHT=50> <HR><P>

</CENTER><blockquote>Seleccione el t&iacute;tulo de la revista asociado a la coordinaci
&oacute;n que desee consultar y despu&eacute;s haga click en el bot&oacute;n de aceptar
:</blockquote><P>
```

Dentro de la forma de HTML se insertó un bloque de <SYB TYPE=PERL> que provee *web.sql*, en el cual se puede programar en Perl y utilizar algunas funciones para hacer consultas a bases de datos en Sybase; este bloque se utilizó para construir una lista con las diferentes coordinaciones del Instituto de Ingeniería; para esto, se recurrió a la función de *ws_sql* que permite realizar consultas a bases de datos e imprimir los resultados en el formato HTML, en este caso fue una lista (*<select>*), como la que se muestra en la figura 4.16, a continuación se muestra lo descrito anteriormente.

```
<CENTER><form action=ejem_rec.hts method=post>

<select name="revistas">

<SYB TYPE=PERL>

ws_rpc($ws_db,"b_usi.dbo.coord_r",[{'value'=>"$coordina",'type'=>CS_SMALLINT_TYPE}],
"<option>%s");
</syb>
</select>
<BR><BR><input type="submit" value="ACEPTAR"><BR>
<P><HR><P><blockquote>Realizar otra búsqueda por:</blockquote>
 <TABLE><TR>
<TD> | </TD>
<TD><A HREF="http://olli.iingen.unam.mx/cgi-bin/ws.exe/usi/docs/sa_coord.hts">Coordinacion
</A></TD>
<TD> | </TD>
<TD><A HREF="http://olli.iingen.unam.mx/cgi-bin/ws.exe/usi/docs/sa_tema.hts">Tema </A> </TD>
<TD> | </TD>
<TD><A HREF="http://olli.iingen.unam.mx/cgi-bin/ws.exe/usi/docs/indice.hts"> Indice </A></TD>
<TD> | </TD>
</TR></TABLE><BR><HR><BR>
<IMG SRC="/usi/imag/puerta.gif" ALT="regresa" WIDTH=50 HEIGHT=50 ><P>
<input type="button" name="BACK" value="REGRESAR" onClick="history.back()"> <BR><BR><HR>
</FORM>
</BODY>
</HTML>
```

Las demás páginas de la sección de búsquedas, se desarrollaron de una forma muy similar.

La figura 4.17 muestra un ejemplo de la página realizada para la sección de solicitud de préstamo:

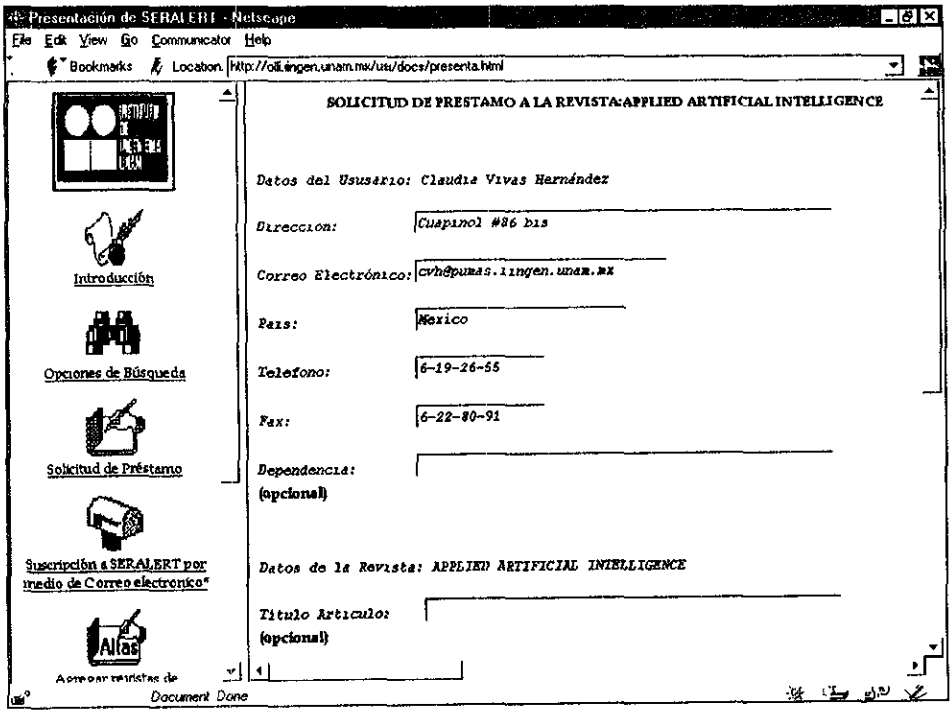


Fig. 4.17 Página que representa la sección de solicitud de préstamo.

Para el desarrollo de la página de solicitud de préstamo, se utilizó HTML, JavaScript, Perl, y Sybase, para esta página se tomaron en cuenta dos casos, uno cuando el usuario ya se encuentra en la base de datos y el otro cuando el usuario es uno nuevo. La figura 4.17 muestra el caso en el que el usuario es uno ya existente en la base de datos.

En este caso se procedió a construir las cajas de texto correspondientes a los datos del usuario (dirección, correo electrónico, país, teléfono y fax), dentro de un bloque de <SYB TYPE=PERL> usando la función de *ws_sql* , para traer de la base de datos cada uno de los datos del usuario, e imprimir el resultado en la caja de texto correspondiente. A continuación se muestra un ejemplo (para los datos de dirección y correo electrónico), de como se construyeron dichas cajas de texto:

```
<SYB TYPE=PERL>
ws_sql($ws_db, "select direccion from USUARIO where nombre = '$nombre' and paterno = '$paterno' and
materno='$materno'","<I><PRE>Dirección:          <input type=\"text\" name=\"direccion\" value=\"%s\"
size=\"50\" maxlength=50></PRE></I>")

ws_sql($ws_db, "select correo_elec from USUARIO where nombre = '$nombre' and paterno = '$paterno' and
materno='$materno'","<I><PRE>Correo electrónico:<input type=\"text\" name=\"correo_elec\" value=\"%s\"
size=\"30\" maxlength=25></PRE></I>"),
.
.
.
etc.
<SYB>
```

Las demás cajas de texto correspondientes a los datos de la revista las cuales deben estar vacías para que el usuario llene los datos, se construyeron con HTML dentro de una etiqueta *print* de Perl. A continuación se muestra un ejemplo:

```
print <<ETI;

<I><PRE>Dependencia:      <input type="text " name="dependencia" value="" size=50 maxlength=150>
<h3>(opcional)</h3></PRE></I><P>
<HR><P><P><P><P><I><PRE>Datos de la Revista: <font color="FF0000"> $revista </FONT>
</I></PRE><P>
<I><PRE>Título Articulo:  <input type="text "name="tit_articulo" size=50 maxlength=255>
<h3>(opcional)</h3></PRE></I><P>
.
.
.
etc.
ETI
```

Para el caso en el que el usuario es uno nuevo, se procedió a construir todas las cajas de texto vacías, correspondientes a los datos del usuario y de la revista en HTML, dentro de una etiqueta *print* de Perl como la que se mostró en el ejemplo anterior.

Las cajas de texto anteriormente citadas para ambos casos, se construyeron dentro de una forma de HTML, así como los botones de enviar y regresar.
Para validar los datos, se programó en JavaScript, para evitar que el usuario deje algún campo vacío que no debe estar nulo.

Por último la figura 4.18 muestra el desarrollo de la página para la sección de consultas:

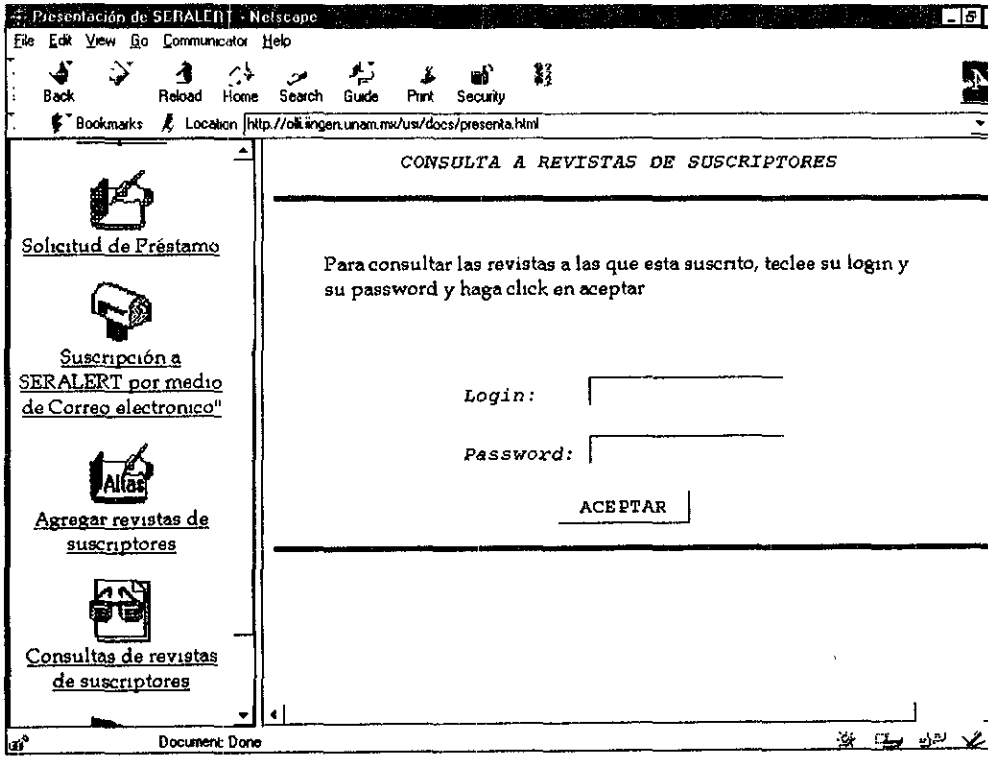


Fig. 4.18 Página que representa la sección de consultas.

Para el desarrollo de esta página se procedió a crear dentro de una forma de HTML, las cajas de texto correspondientes a login y password, un botón de aceptar y algunos detalles más. Para validar los campos de login y password, se programó en JavaScript para evitar que el usuario deje algún dato vacío que afecte a la base de datos, además dentro de un bloque de <SYB TYPE=PERL>, se usó la función de *ws_sql* para realizar una consulta a la base de datos y verificar que el login y password sean los correctos, y de esta manera permitirle o no al usuario entrar a ver las revistas a las que esta suscrito.

Las páginas de la sección de bajas y agregar revista a la lista de suscriptores, se desarrollaron de una forma muy similar.

4.4.2 Visual Basic.

Las siguientes formas fueron desarrolladas en Visual Basic para la administración del Servicio de Alerta. Esta administración la llevarán a cabo las personas encargadas de la USI. A continuación se muestra el desarrollo de las formas más representativas para la administración del sistema:

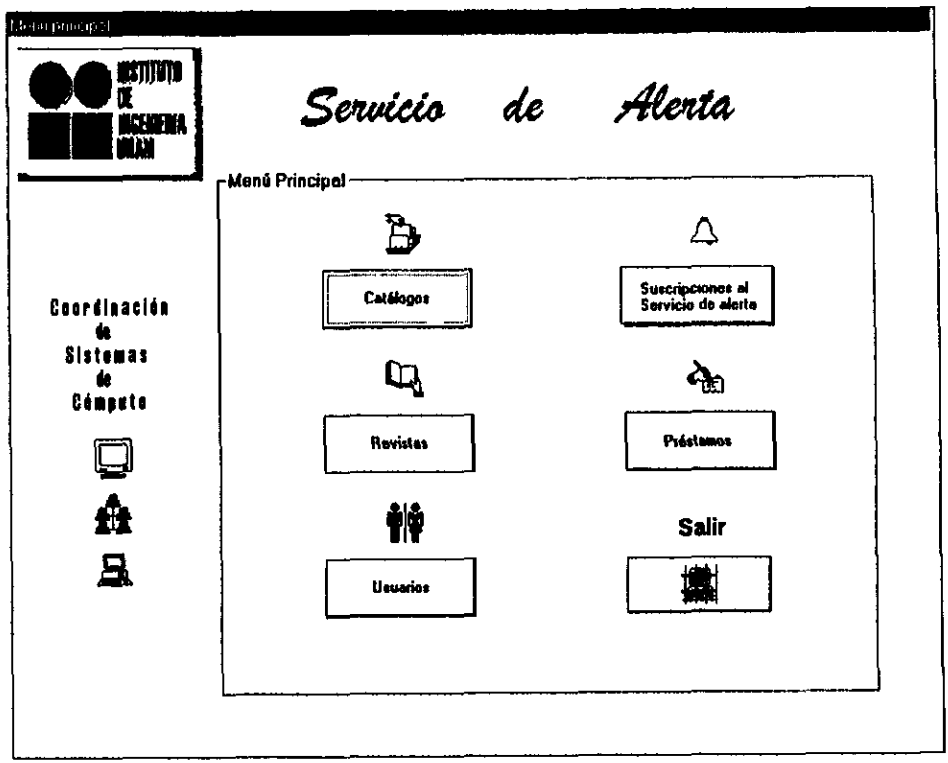


Fig. 4.19 Forma de Menú principal de la interfaz VB.

La figura 4.19 muestra el menú principal para la administración del Servicio de Alerta y se generó a través de objetos preestablecidos en Visual Basic, es decir como primer paso se insertó una nueva ***Forma***.

El segundo paso fue establecer las propiedades a cada objeto. Dentro de este objeto se utilizó la propiedad de *Back Color* que determina el color de la forma, otra propiedad del objeto es la de *Caption* que determina que título desea ponerle a la forma, por ejemplo en esta forma fue "Menú Principal", otras propiedades utilizadas fueron la de *Fore Color y Font* que determinan el color y el tipo de letra deseada, la última propiedad utilizada fue la de *name* que permite nombrar a nuestro objeto, conteniendo en el nombre el prefijo "Frm",

por ejemplo en este caso la forma se llamo "Frmmenuprincipal", por supuesto se pueden utilizar otras propiedades de la tabla, pero eso lo determina el diseño deseado.

El tercer paso fue insertar código en algunos eventos, en este caso el evento utilizado fue *"Load"*:

```
Private Sub Form_Load()
Set bsusi = OpenDatabase("", False, False,
"ODBC;database=mybase;uid=myid;pwd=xxxxx;dsn=sabase") 'selecciona los registros de la tabla de
la BD

Cmdprestamo.Enabled = True
Cmdsuscriptores.Enabled = True
cmdusuarios.Enabled = True

End Sub
```

El código anterior permite conectarse a la base de datos por medio de la ruta de acceso mostrada en el ejemplo.

Otros objetos insertados en la forma fueron: ***frame, etiquetas, botones e Imágenes;*** a los cuales se les determinaron sus respectivas propiedades de manera similar al objeto anterior.

El evento utilizado en el objeto botón fue *"Click"*, y el código insertado en éste fue:

```
Private Sub Cmdcatalogo_Click()
Screen.MousePointer = 11
Load Frmcatalogo
Frmcatalogo.Show 1
Screen.MousePointer = 0
End Sub
```

El código anterior permite llamar y visualizar a otra forma. Para los demás botones se utilizó un código específico dependiendo las tareas a realizar.

En la figura 4.20 se muestra la forma donde se podrá manejar la información de las Revistas almacenadas en la base de datos:

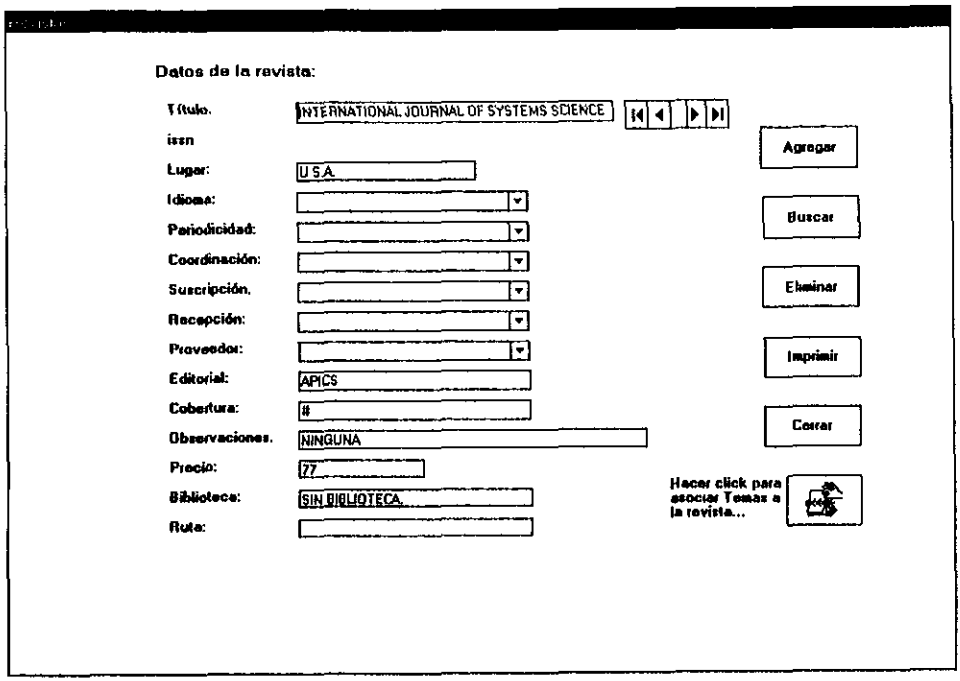


Fig. 4.20 Forma para administrar las revistas.

Para desarrollar esta forma llamada "Frmrevistas" se utilizaron objetos como cajas de texto, combos, etiquetas, botones, imágenes y un Data control que permite accesar a la Base de Datos.

Los eventos utilizados en la caja de texto fueron *"Keypress" y "Lost Focus"*. A continuación se muestra el código ejemplo para cada uno.

Código del Evento Keypress.

```
Private Sub Txttitulo_KeyPress(KeyAscii As Integer)
KeyAscii = Asc(UCase(Chr(KeyAscii)))
Select Case KeyAscii
Case 0 'permite el backspace
  Exit Sub
Case 13
  KeyAscii = 0
  SendKeys "{tab}"
Case Else
If proceso_agregar = False Then
```

```
    Txttitulo.SetFocus
    If bandera = True Then
        iresponse = MsgBox("Se modificará la información de la Base de datos ¿Deseas hacerlo?",
vbYesNo)
        If iresponse = vbYes Then
            Txttitulo.Locked = False
            Txttitulo.SetFocus
        Else
            Txttitulo.Locked = True
            Exit Sub
        End If
    End If
    bandera = False
    Exit Sub
End If
End Select
If IsNumeric(Chr(KeyAscii)) Then
  KeyAscii = 0
  Beep
End If
End Sub
```

El código anterior valida los datos insertados en las cajas de texto, ya que pueden ser letras ó números.

Código del Evento LostFocus:

```
Private Sub Txttitulo_LostFocus()
If proceso_cancelar = False Then
    If proceso_agregar = True Then
        If Not (PermiteAcceso) Then 'Aceptar acceso
            If Txttitulo.Text = "" Then
                PermiteAcceso = True
                Txttitulo.SetFocus
                MsgBox "Proporcione el título de revista"
                DoEvents
                PermiteAcceso = False
            End If
        End If
    End If
End If
bandera = True
End Sub
```

El código anterior verifica que las cajas de texto se les inserte información, sino contiene información mandará un mensaje. De manera similar se realizó para todas las cajas de texto.

En los DBCombos el evento utilizado fue el "*Click*" un ejemplo del código es:

```
Private Sub DBCidioma_Click(Area As Integer)
Lblcl_idioma.Caption = DBCidioma.BoundText
If proceso_agregar = False Then
   If bandera = True Then
      iresponse = MsgBox("Se modificará la Base de datos ¿Deseas hacerlo?", vbYesNo)
      If iresponse = vbYes Then
         DBCidioma.Locked = False
         DBCidioma.SetFocus
      Else
         DBCidioma.Locked = True
         Exit Sub
      End If
   End If
   bandera = False
   Exit Sub
End If
End Sub
```

El código anterior verifica antes de borrar la información mandando un mensaje de alerta.

El objeto **Data Control** permite conectarse a la Base de Datos por medio de sus propiedades:

Algunas propiedades utilizadas dentro de este objeto fueron: *Connect* que permite conectarse a la Base de Datos, en este caso fue por medio del ODBC; *Record Source* permite seleccionar una tabla de la BD, por ejemplo "dbo.IDIOMA", *Recordset Type* se le especifica el tipo de recordset que se va a utilizar, por ejemplo "Dynaset", otra propiedad importante es la de *Name* que permite asignar un nombre al Data Control con su prefijo "Dat", por ejemplo "Datidioma".

Dentro de la forma de Suscriptor se utilizaron los objetos y eventos similares a los de la forma de Revistas, la única diferencia es que en está forma no se utilizó el DBCcombo y se va a manipular información de las tablas Suscriptor y Usuario.

En la figura 4.21 se muestra la forma que permitirá manejar la información de la tabla de Préstamos:

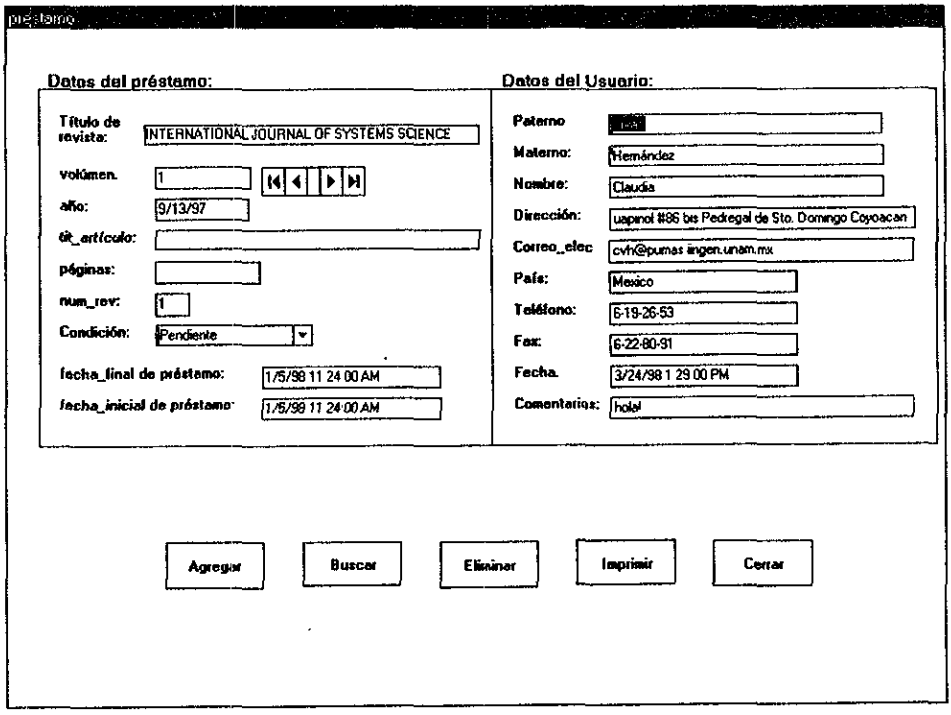


Fig. 4.21 Forma para administrar los préstamos.

En esta forma se podrán consultar los préstamo realizados desde la interfaz del Web. Estos préstamo se encuentran almacenados en la Base de Datos de Préstamo, por lo cual desde esta interfaz de Visual se podrá manipular la información.

Los objetos y eventos utilizados son similares a los de las formas Revistas; la única diferencia es que esta forma fue realizada para manipular los préstamos de los usuarios.

La figura 4.22 muestra la forma que permitirá actualizar los últimos ejemplares de cada revista:

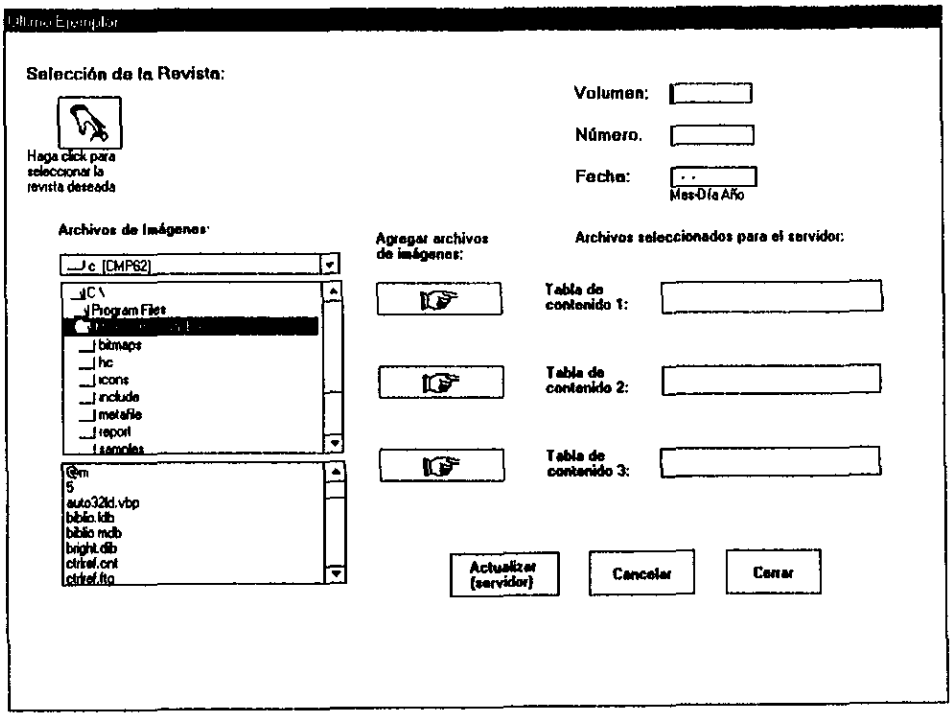


Fig. 4.22 Forma para actualizar los últimos ejemplares.

Para esta forma se utilizaron algunos otros objetos como el **Drive List Box,** que permite desplegar los drives que se encuentran en la máquina de trabajo.

El evento utilizado en este objeto fue "*Change*", para permitir cambiar la lista de directorios dependiendo el Drive seleccionado:

```
Private Sub Drvrevista_Change()
On Error GoTo DriveHandler
Dirrevista.Path = Drvrevista.Drive
Exit Sub
DriveHandler:
   Drvrevista.Drive = Dirrevista.Path
   Exit Sub
End Sub
```

Otro objeto utilizado fue **Dir List Box** que permite desplegar los directorios del drive seleccionado.

El evento utilizado en este objeto es el "*Change*", que va a permitir cambiar la lista de archivos a partir del directorio seleccionado.

```
Private Sub Dirrevista_Change()
Filrevista Path = Dirrevista.Path
End Sub
```

El objeto **File List Box** permite desplegar los archivos dependiendo del directorio seleccionado.
El objeto **MaskEdBox** permite poner un formato para fechas.

Para los botones el evento utilizado fue "*Click*".

El código del evento "*Click*" en el botón para selección de la revista, permitirá ir a otra forma, en la cual se va a seleccionar la revista para la actualización.
El código del evento "*Click*" en los botones "➡", permitirá insertar el archivo seleccionado, de la lista de archivos, en la caja de texto correspondiente.
El código del evento "*Click*" en el botón de actualizar servidor, permitirá realizar la actualización de los últimos y penúltimos ejemplares de cada revista, así como enviar correos a los usuarios suscritos al Servicio de Alerta.

4.5 Implantación y Administración

Una vez terminado el Sistema, la parte de la administración se implantó en una máquina de la U.S.I., instalando la herramienta de ODBC para permitir la conexión a la base de datos a través de SYBASE ; el Winrsh, que es una herramienta que permite crear archivos batch para ejecutar comandos de Ftp y correo electrónico, también se instaló el Aplication Setup Wizard de Visual Basic 4.0 para poder implantar el programa ejecutable del sistema de administración.

Ya instaladas las herramientas, se les proporcionó una breve explicación del funcionamiento del sistema a las personas encargadas de la administración, para qué trabajaran en éste y nos comunicarán sus comentarios o dudas, y así realizar las mejoras necesarias al sistema.

En la administración del sistema se dará el mantenimiento necesario, el cual consiste en realizar mejoras o correcciones al sistema, así como también en caso de generarse un error y no saberlo corregir, poderle ayudar al usuario.

Para futuras mejoras se realizó un informe para el usuario, el cual contiene las siguientes etapas del sistema: una explicación detallada, el diseño, el desarrollo y la instalación del mismo.

COMENTARIOS FINALES

COMENTARIOS FINALES.

Los resultados esperados para el sistema del Servicio de Alerta han sido satisfactorios, ya que se lograron alcanzar los objetivos propuestos, puesto que ahora los usuarios internos y externos del Instituto de Ingeniería, se mantendrán informados de una manera más rápida y cómoda al consultar las tablas de contenido de las revistas que llegan al Instituto a través del Web.

Por otra parte se facilitó el proceso realizado anteriormente por las personas encargadas de la U.S.I., ya que, debido a la publicación electrónica a través del Web se difunden los ejemplares que llegan a ésta, se reduce el consumo de papel y los costos derivados de la impresión, además se ofrecen los servicios de préstamo y suscripción al servicio de alerta por medio del web, eliminando los tiempos de presentación y entrega de información, tanto para las personas encargadas de la U.S.I. como para los usuarios.

Con la implantación del sistema de Servicio de Alerta se llevó a cabo un paso más en el desarrollo de nuevas aplicaciones, teniendo en cuenta que los sistemas de información tienden día con día a ser una Intranet, ya que se tiene la posibilidad de utilizar el protocolo de Internet como un estándar de todas las computadoras.

El sistema de Servicio de Alerta marca el inicio de una Intranet, puesto que la Intranet proporciona herramientas competitivas, suficientemente poderosas como para conseguir ahorrar costos y tiempo en el trabajo diario, permite compartir y publicar información, y reducir la desventaja de trabajar a distancia por parte de los empleados con conocimientos sobre las operaciones frecuentes y productos habituales de su Institución.

Como parte de la documentación del sistema de Servicio de Alerta, esta tesis proporcionará la información necesaria para futuras modificaciones y adiciones.

GLOSARIO DE TÉRMINOS

ARPANET Advanced Research Projects Agency: Departamento de defensa encargado de la administración de redes de organizaciones militares y de investigación.

Awk: Utilería de programación basado en la búsqueda de patrones. Forma parte de las utilerías de UNIX.

Base de Datos: Colección de información, organizada y presentada para servir a un propósito específico, como la facilitación de búsquedas, ordenamientos o procesamiento de los datos.

Campo: Elemento de información contenido dentro de un renglón o registro. Equivalente lógico de una columna.

Bourne Shell: Uno de los tres principales shells que Unix provee. Fue desarrollado por Steve Bourne y es el shell original de Unix.

C Shell: Uno de los tres principales shells que Unix provee. Fue desarrollado por Bill Joy en la Universidad de Berkeley, cuenta con una sintaxis similar a la del lenguaje C

Cliente: Sistema que recibe recursos de un sistema remoto llamado servidor a través de una red.

Columna: Conjunto de todos los renglones de una tabla que tienen un atributo común. Contiene un dato individual dentro de cada renglón o registro.

Concurrencia: Son múltiples accesos por diferentes usuarios a la misma información.

Consistencia: La información es consistente, si al consultar datos en cualquier parte del sistema, estos son los mismos.

Dato: Representación codificada de información para usarla en una computadora. Los datos tienen atributos como tipo y longitud.

Dependencia Funcional: El atributo A de una relación es funcionalmente dependiente del atributo B, si el valor de A esta determinado por el valor de B.

Dirección electrónica: En el ámbito de las comunicaciones electrónicas como el correo electrónico, ésta especifica el nombre de la máquina destino y el usuario que recibirá un mensaje. En Internet esta dirección tiene el formato *usuario@máquina_destino*

Dirección IP: Identificador universal de 32 bits asignado a una máquina que forma parte de Internet, consiste de una parte que identifica a la red a la cual pertenece la máquina y una parte que identifica a la máquina misma.

DNS: Acrónimo de **Domain Name System**, base distribuida de nombres de **hosts** y direcciones IP de redes, esta base de datos es controlada por servidores jerárquicos dentro de Internet.

Domain name: Nombre del host compuesto por componentes que permiten a hosts localizados en diferentes sitios tener el mismo nombre por ejemplo: *myhost.Institute.mx* y *myhost.company.com.*

Eficiencia*: Es el tiempo de respuesta de una aplicación en el momento de accesar a una base de datos.*

Elemento: Sinónimo de campo en una tabla. Intersección de un renglón y una columna.

Front end: En términos de software un front end es el programa con el cual el usuario interactúa con un servicio muy especializado.

FTP: Una parte del protocolo de transferencia de archivos (File Transfer Protocol) que permite al usuario conectarse como individuo a un host remoto utilizando un login, éste es uno de los servicios de mayor éxito en Internet.

Hardware: Parte física de un sistema de cómputo.

Hipertexto: El concepto de Hipertexto se refiere a datos contenidos en un documento que están organizados para proveer ligas entre documentos, de modo que relacionan conceptos y temas que pueden ser ligados entre sí.

Home directory: Parte de un sistema de archivos que es asignado para el uso exclusivo de un usuario.

Host: Este término se aplica comúnmente a toda computadora que se encuentra conectada a una red.

HTML HyperText Markup Language: Lenguaje de documentos de hipertexto.

HTTP HyperText Transport Protocol: Protocolo de transportación de hipertexto.

Inconsistencia*:* Obtener diferentes salidas para repeticiones similares en un mismo momento.

Índice: Conjunto de apuntadores ordenados lógicamente por los valores de una llave. Un índice es un elemento de la base de datos que proporciona acceso a los registros de una tabla, mediante el valor de una llave.

Integración: Es la unión de muchos archivos separados.

Integridad: Reglas que los datos deben cumplir especificadas por el mundo real.

internet: Colección de redes conectadas con gateways y ruteadores.

Internet: Colección de hosts conectados de varias redes regionales alrededor del mundo que utilizan los protocolos TCP/IP

IP: Acrónimo de **Internet Protocol**, IP es uno de los principales protocolos que forman a la familia TCP/IP

LAN Local Area Network: Red de área local para que una red pueda ser considerada dentro de esta clasificación debe tener una área menor a 10 Km a la redonda.

Login name: Nombre que define al usuario de un sistema, con éste y el password el usuario puede tener acceso al sistema.

Llave: Uno o más campos usados para identificar un registro, frecuentemente se utiliza como índice de una tabla.

Llave foránea: Columna o combinación de columnas cuyos valores se relacionan con la llave primaria de alguna otra tabla. Una llave foránea no tiene que ser única. No deben existir valores de las llaves foráneas, excepto "nulo", a menos que el mismo valor exista en una llave primaria.

Llave primaria: Columna o combinación de columnas que identifican de manera única una tabla. Siempre deben ser diferentes de "nulo" y tener un índice único. Una llave primaria se usa para relacionarse con llaves foráneas de otras tablas.

Primera forma normal: Una relación está en la primera forma normal si todos los campos en cada registro contienen un solo valor tomado de sus dominios respectivos.

Procesamiento distribuido: Las tareas de procesamiento son efectuadas por uno o más sistemas interconectados mediante una red de cómputo.

Protocolo: Descripción formal de formatos de mensaje y las reglas de como dos computadoras deben intercambiar mensajes. Los protocolos describen detalles de interfaces de computadora a computadora de bajo nivel o intercambios de mensaje de alto nivel entre programas.

Puerto: Abstracción que los protocolos de transporte utilizan para distinguir entre muchos destinos en un host remoto. Los puertos son asociados a una aplicación específica y será a través de este puerto que la aplicación recibirá peticiones de servicio, un claro servicio de esto es el servicio de TELNET al cual está asociado el puerto 21.

Recuperación: Capacidad de restaurar la integridad y la consistencia de una B.D, después de una falla en el sistema.

Redundancia: Es la repetición de datos, datos derivados, por lo que genera inconsistencia.

Registro: Grupo de campos (columnas) cuya información se trata como una unidad. Equivalente lógico de un renglón.

Relación: Dada una serie de conjuntos D1, D2, ..., Dn (no necesariamente distintos) se dice que R es una relación sobre estos n conjuntos si es un conjunto de n tuplas ordenadas <d1, d2,...dn>, tales que d1 pertenece a D1, d2 pertenece a D2, ..., dn pertenece a Dn. Los conjuntos D1, D2, ..., Dn son los dominios de R. El valor n es el grado de R.

RPC: Conjunto de protocolos de red que permiten a un nodo llamar procedimientos que son ejecutados en máquinas remotas.

Rlogin (Remote log-in): Comando de BSD Unix que permite accesar un sistema remoto sin tener que identificarse ante él previamente.

Root: El login name convencional para el superusuario de Unix , el root es el único usuario de un sistema que posee ilimitado poder sobre el sistema.
Sistemas Abiertos (Open Systems): Basados en ambientes independientes a fabricantes en el cual se puede mezclar hardware y software de varios fabricantes, los sistemas abiertos son sistemas de cómputo y comunicaciones cuyas especificaciones son ampliamente aceptadas, disponibles y estandarizadas.

Rsh: Comando TCP/IP que accesa un nodo remoto, su operación es muy similar a la de rlogin.

Sed (Stream Editor): Editor que permite efectuar operaciones sobre archivos de texto, su principal ventaja es que no modifica el contenido del archivo sobre el cual trabaja.

Segunda forma normal: Una relación está en segunda forma normal si está en primera forma normal y cada atributo que no forma parte de la llave principal está en forma total y funcionalmente dependiente de ella.

Seguridad: Protección de los datos contra accesos, modificaciones o pérdidas.

Sesiones remotas: Conexión con sistemas remotos encargados del procesamiento desde cualquier computadora conectada a la red.

SMTP: Acrónimo de Simple Mail Transfer Protocol, protocolo usado para intercambiar correo electrónico entre hosts de Internet.

Shell: Término que se refiere a la Interfaz del usuario y el Kernel del sistema operativo, en sistemas Unix, el C o Bourne shell son las interfaces principales.

Shell scripts: Archivo que contiene una serie de comandos ejecutables que son utilizados como entrada del shell.

SNMP: Acrónimo de **Simple Network Management Protocol**, protocolo usado para obtener información estadística de gateways, ruteadores, y otros elementos de red.

Software: Parte de un sistema computacional que se refiere a los programas que se utilizan para que funcione.

Tabla: Colección de renglones (o registros) que tienen columnas (o campos) asociados.

TCP: Acrónimo de **Transmission Control Protocol**, uno de los protocolos primarios usados en Internet. TCP permite a un proceso en una máquina enviar un flujo de datos a otro proceso en otra máquina. Para realizar una conexión TCP los participantes deben establecer una conexión antes de enviar información.

***TCP/IP* Transmission Control Protocol/Internet Protocol:** Conjunto de protocolos cuyo nombre está referido a sus dos principales protocolos, Transport Control Protocol (TCP) e Internet Protocol (IP), TCP/IP ofrece una transmisión confiable, sobre el cual varias aplicaciones se ejecutan. TCP/IP es el protocolo que se utiliza en Internet.

Tercera forma normal: *Una relación está en tercera forma normal si está en segunda forma* normal y ningún atributo involucrado en la relación es funcionalmente dependiente de algún otro atributo que no es parte de la llave.

Tipo de dato: Identificador que especifica la clase de información (números, valores lógicos, caracteres , texto, etc.) que contiene una columna y cómo será almacenada.

Trade-Offs:

A mayor integridad y consistencia mayor eficiencia.
A mayor seguridad menor eficiencia.
A mayor seguridad menor concurrencia.

WAN Wide Area Network: Redes de área amplia.

Workstations: Estación de trabajo, equipos de cómputo de gran desempeño con capacidad de multiprocesamiento y acceso multiusuario, estos sistemas fueron inicialmente desarrollados para efectuar tareas propias de ingeniería, en la actualidad estos sistemas han venido a sustituir a equipos como minicomputadoras y mainframes

BIBLIOGRAFÍA

BIBLIOGRAFÍA:

Fundamentos de Bases de Datos.
Henry F. Korth, Abraham Silberschatz.
Editorial Mc Graw-Hill, 2ª edición.
México D.F. 1994.
Pags: 25-45, 57-90, 103-105, 183-216.

Apuntes de Bases de Datos (1996).
- Conceptos Básicos de Bases de datos.
- Modelo Entidad-Relación.
  - Algebra Relacional.
  - Calculo relacional.
- Modelo Relacional .
- Formas Normales.
- SQL.

Sybase Developer's Guide.
Daniel J. Worden.
Editorial SAMS Publishing, 1ª edición.
E.U.A. 1994. Pags: 3-69.

World Wide Web. Database Developer's Guide.
Swank Mark and Drew Kittel.
Editorial SAMS-Net, 1ª edición.
E.U.A. 1996. Pags: 3-27, 49-123, 133-221, 255-285

UNIX (Manual de Referencia).
Stephen Coffin.
Editorial Mc Graw Hill, 5ª edición.
México. Pags: 5-18.

Internet Aplications with Visual Basic.
Michael Marchuck.
Editorial QUE, Edición para Visual Basic 3.0
E.U.A. 1995. Pags: 371-384.

Using Visual Basic 4.
Jeff Web, Mike Mckelvy, Ronald Martinsen, Taylor Maxwell, and Michael Regelski.
Editorial QUE, Edición para Visual Basic 4.0
E.U.A. 1995. Pags: 60-62, 121, 239-259

Developer's Guide with Visual Basic 4.
Roger Jennings.
Editorial SAMS Publishing, 2ª edición.
E.U.A. 1996. Pags: 3- 83, 929.

Manual de Referencia de HTML.
Todos los temas.
México D.F 1997.

Manual de Referencia de Windows 95
Temas: Introducción y Fundamentos.
Pags. 5-15.

Web Site Administrator's (Surival Guide).
Jerry Alban y Scott Yanoff.
Editorial Sams net.
Indianapolis, Indiana 46290
Pags 3-25

SYBASE SQLServer 11.
Ray Rankins, Jeffrey R. Garbus, David Solomon, Bennett Wm. Mc Ewan.
Sams Publishing.
E.U.A. 1996, 1ª edición.
Indianapolis , IN 46290
Pags. 3-22.

Perl 5.
Kamran Husain.
Sams Publishing.
E.U.A. 1996.
Indianapolis, Indiana 46290
Pags. 4-46, 96-117, 136-160, 406-431

CGI Programming.
Dan Berlin, et al
Sams Net.
E.U.A. 1996.
Indianapolis, Indiana 46290.
Capítulos 1,2,3,4,9 y 10.

Sybase web.sql Programer's Guide.
Document ID: 35725-01-0101-02.
Sybase, Inc. All rigths reserved.
Todos los temas.

Direcciones electrónicas:
http://www.sybase.com/
http://language.perl com/info/documentation.html
http://macaria.pucp.edu.pe/taller-html/URL-General.html