ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería Eléctrica y Computación

"WEB SERVER ORIENTADO A BASE DE DATOS MEDIANTE ODBC"

"Análisis y Diseño de Aplicaciones Orientado a Objetos"

Trabajo de Tópico de Graduación
Previo a la Obtención del Título de:

Ingeniero en Computación

Presentado por:

Katty Sucre Martínez
Víctor Zambrano Avilés
Erick Ricaurte Zambrano
Darío Andrade Ramírez
Christian Molina Gómez
Oscar Jaramillo Burgos
Medardo Jirón Novillo

Guayaquil - Ecuador

AÑO 1997

ESCUELA SUPERIOR POLITÉCNICA

DEL LITORAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y COMPUTACIÓN

"WEB SERVER ORIENTADO A BASE DE DATOS MEDIANTE ODBC"

"Análisis y Diseño de Aplicaciones Orientado a Objetos"

Trabajo de Tópico de Graduación

Previo a la Obtención del Título de:

INGENIERO EN COMPUTACIÓN

Presentado por:

KATTY SUCRE MARTINEZ

VICTOR ZAMBRANO AVILÉS

ERICK RICAURTE ZAMBRANO

DARÍO ANDRADE RAMÍREZ

CHRISTIAN MOLINA GÓMEZ

OSCAR JARAMILLO BURGOS

MEDARDO JIRÓN NOVILLO

Guayaquil - Ecuador

Año 1997

"La responsabilidad por los hechos, ideas y doctrinas expuestos en esta tesis, corresponden exclusivamente a los autores; y el patrimonio intelectual de la misma, a la ESCUELA SUPERIOR POLITECNICA DEL LITORAL". (Reglamento de exámenes y Títulos profesionales de la ESPOL)

Katty Suere Martínez

Dario Andrade Ramírez

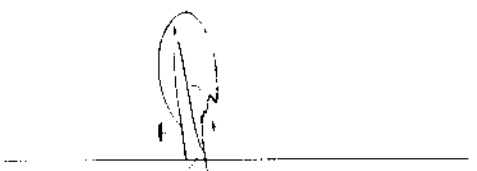
Christian Molina Gómez

Oscar Jaramillo Burgos

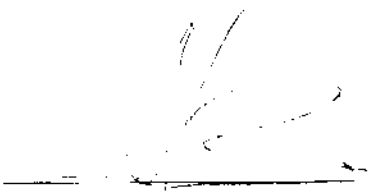
Erick Ricaurte Zambrano

Víctor Zambrano Avilés

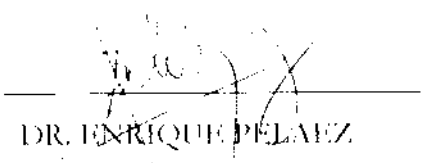
Medardo Jirón Novillo



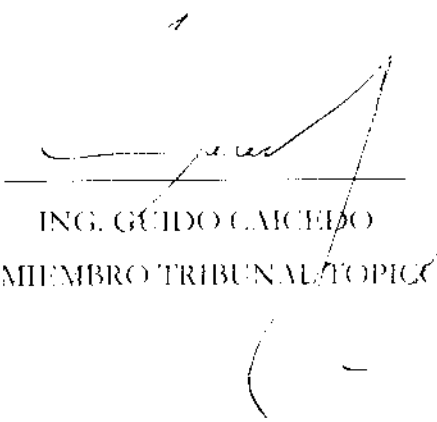
ING. ARMANDO ALTAMIRANO
SUBDECANO FIEC - ESPOL



ING. JHONNY MACIAS
DIRECTOR DE TOPICO



DR. ENRIQUE PELAEZ
MIEMBRO TRIBUNAL TOPICO



ING. GUIDO CAICEDO
MIEMBRO TRIBUNAL TOPICO

Indice General

Resumen

El presente documento recopila la información acerca del análisis, diseño e implementación de nuestro proyecto de graduación utilizando la Técnica Orientada a Objetos. Este proyecto fue seleccionado entre los siguientes temas propuestos:

- WebServer orientado a Base de Datos utilizando ODBC
- Wizard HTML
- Correo de Voz

- Aplicación Multimedia con Acceso a Base de Datos Vía ODBC
- Cada uno de los temas presentados cubría tópicos importantes e interesantes, sin embargo el tema *WebServer orientado a Base de Datos utilizando ODBC* nos interesó sobre manera por que abarcaba un tópico novedoso, diferente y que tiene una amplia gama de usos y aplicaciones. Además nos permitiría extender nuestros conocimientos en lo referente a Internet y la forma de accesar a múltiples Bases de Datos.

Una vez seleccionado el tema, se realizó un análisis exhaustivo sobre las áreas hacia dónde encaminar nuestro esfuerzo para la solución de sus problemas. Finalmente se decidió orientarlo hacia el área de prestación de servicios, fundamentalmente a la consulta de información en las empresas de servicio público, quienes se convertirían en nuestros clientes naturales.

- La aplicación nos permite accesar a la base de datos de nuestros clientes para recuperar la información requerida por un usuario de nuestro servicio, esta información es puesta en un formato que permita ser mostrada en la Internet, lugar desde el que se generó el requerimiento.

De lo expuesto en el párrafo anterior podemos entrever la generación de páginas HTML dinámicas, lo que constituye un punto sobresaliente en nuestra aplicación permitiendonos mostrar información en línea de cada empresa que se consulte. Además hemos aprovechado las bondades que permite la Internet para mostrar información de una manera agradable para el usuario.

Esperamos que la lectura de este documento y la utilización de la aplicación asociada sea del máximo provecho para Ud., sábiendo que hemos puesto todo el esfuerzo posible para que así sea.

Introducción

El presente documento describe el estudio e implementación de un Web Server, al que de ahora en adelante lo referenciaremos con el nombre de ICARO, además del desarrollo de una aplicación utilizando ICARO Web Server, la que será llamada ICARO Centro de Servicios.

Inicialmente hacemos un planteamiento del problema para luego describir algunos conceptos importantes de diversos tópicos relacionados con el problema planteado, estos conceptos nos proveen de las bases para un mejor entendimiento acerca del funcionamiento de la aplicación.

Como siguiente paso hacemos una exposición de la Técnica Orientada a Objetos y su aplicación a ICARO Web Server, así como de los algoritmos de las funciones principales resultante de este análisis.

Por ser ICARO Web Server una aplicación de Internet esta se sirve del conjunto de protocolos TCP/IP y de la interface de comunicaciones conocida como

La API Windows Sockets[1]. Los protocolos TCP/IP son los encargados de la transferencia de información entre dos computadores anfitriones, mientras que Winsock define las estructuras o sockets a través de las que se realiza la transferencia de información.

ICARO es un Web Server orientado a Base de Datos, utiliza ODBC como herramienta para realizar la conexión y consulta de datos. El uso de ODBC nos permite el acceso a una amplia gama de Bases de Datos y de esta manera incrementar sustancialmente los clientes potenciales de nuestro servicio.

ICARO Centro de Servicios es una aplicación que demuestra una de las áreas de utilización de ICARO Web Server, el escojimiento de esta área fué producto de un análisis del mercado y sus tendencias, además del estudio técnico respectivo.

Finalmente indicamos un conjunto de conclusiones y recomendaciones resultantes del desarrollo de este proyecto.

[1] Winsock

Capítulo 1

Generalidades

1.1 Descripción del problema

El Internet dio un giro total al cambiar radicalmente su interface con el ingreso de WWW, es decir al pasar de su interface aburrida y poco amigable a una interface gráfica; la tendencia actual es realizar todo tipo de transacción y de investigación vía esta autopista de la información. Originalmente nació el WWW con páginas en formato HTML previamente creadas y que reposaban en el servidor Web, el cliente solicita información y se busca el archivo respectivo y se lo envía al cliente para ser presentado por el visualizador al usuario. Fue tan grande la aceptación de esta tecnología que cada vez mas empresarios buscan este medio para publicitarse y hacer negocios a nivel internacional, con todo esto surge la necesidad de tener información en línea.

El propósito de este proyecto es crear un prototipo de un Servidor Web (WebServer) que sea una alternativa de solución para dar información en línea por medio de acceso a Bases de Datos, lo que da como consecuencia que las páginas

HTMLs deben ser generadas dinámicamente, esto es, generadas en el momento que el usuario las solicite y con la información respectiva. Además se desea que la aplicación sea independiente del DBMS (DataBase Management System 'Sistema Administrador de Base de Datos') utilizando para esto ODBC (Open DataBase Connectivity).

1.2 Objetivos.

- Implementación de un WebServer orientado a Base de Datos.
- Aplicar nuestros conocimientos de Análisis y Diseño Orientado a Objetos para la implementación de un WebServer
- Implementar el programa servidor utilizando TCP/IP a través de la API Windows Socket.
- Implementar las aplicaciones a usarse, en una herramienta de cuarta generación como es MS Visual C ++, en la versión 4.0
- Accesar a Base de Datos a través de ODBC
- Generar dinámicamente las páginas HTML como resultado de una consulta a la Base de Datos
- Dar un ejemplo práctico de las capacidades de Web Server.

1.3 Alternativas de Implementación

ICARO WebServer 1.0 fue implementado en su totalidad en Visual C++ versión 4.0 utilizando programación orientada a objetos, sin embargo ICARO WebServer pudo haber sido implementado en otra herramienta.

No necesariamente la herramienta utilizada para la implementación de ICARO WebServer es la mejor, por tal razón en este capítulo ofrecemos otras alternativas de herramientas que pudieron ser utilizadas.

1.3.1 Análisis de Herramientas

1.3.1.1 Visual Basic

Una de estas alternativas lo constituye Visual Basic versión 5.0 Edición Profesional, en esta versión se incluyen tres controles diseñados específicamente para encapsular tecnología relacionada con Internet, los cuales incluyen lo siguiente:

- Control de Transferencias Internet: empaqueta tres protocolos comunes de Internet, el protocolo de transferencia de hiepertexto (HTTP), el protocolo de transferencia de archivos (FTP) y Gopher.
- Control Winsock: le permite conectarse a un equipo remoto e intercambiar datos mediante el protocolo de datagramas de usuario (UDP) o el protocolo de control de transmisión (TCP). (Figura 1.1.a)

- Control WebBrowser: incorpora gran funcionalidad disponible mediante Internet Explorer. (Figura 1.1.b)

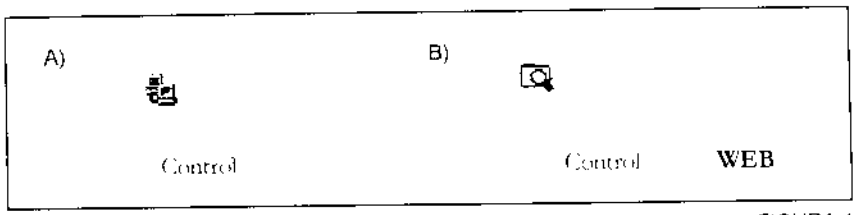


FIGURA 1.1

Estos controles diseñados que pueden ser utilizados en la programación lo cual ahorra muchas líneas de código y complejidad en el momento de la implementación.

Por ejemplo para nuestros propósitos el control Winsock hubiera sido de mucha importancia. Este control permite conectarse a un equipo remoto e intercambiar datos con el protocolo UDP o TCP. Ambos protocolos se pueden utilizar para crear aplicaciones cliente y servidor[1].

Entre las propiedades que tiene este control se encuentra la dirección del equipo a la cual nos vamos a conectar, esta dirección puede ser cambiada en tiempo de ejecución. Cada control Winsock constituye una conexión, por lo tanto si

[1] Para nuestro caso usaríamos TCP

deseamos una determinada cantidad de conexiones debemos definir una matriz de controles Winsock.

Además de los controles ya descritos anteriormente Visual Basic nos da mucha facilidad para el uso de controles Active X[2] en una aplicación para la Internet lo cual permitiría principalmente ampliar la funcionalidad de la aplicación para Internet mediante protocolos comunes, generar páginas HTML con mayor funcionalidad en el cliente.

En cuanto al acceso a Base de Datos en Visual Basic lo podemos hacer utilizando DAO[4] y estableciendo la propiedad Connect a un origen de datos ODBC, los resultados de estas consultas pueden ser almacenadas en cualquier dispositivo o control de Visual Basic.

1.3.1.2 Ventajas sobre Visual C++

Entre las ventajas que tiene Visual Basic sobre Visual C++ podemos citar las siguientes:

---

[2] Antes llamados controles OLE

[4] Data Acess Object

- Controles para Internet ya diseñados y que pueden ser incluidos en un proyecto, estos controles incluyen las propiedades básicas de todo control de Visual Basic además de sus propiedades específicas.
- Gran cantidad de componentes que pueden ser añadidos a la aplicación.
- Mayor facilidad en el diseño de las interfaces y en el manejo de las propiedades de los controles para cambiar el estado del mismo.
- Si Ud. no ha trabajado antes con Visual C++ y con POO[3] será mucho más sencillo aprender a manejar Visual Basic utilizando la metodología tradicional o utilizando POO.

1.3.1.3 Desventajas de Visual Basic

Entre las desventajas que podemos citar de Visual Basic son:

- El poder de la aplicación esta limitado a los controles de Visual Basic, si se desea dar una característica diferente a la aplicación y esta no existe entre los componentes se debe programar el mismo para luego añadirlo. Esta programación del componente puede constituirse en algo muy complicado.

[3] Programación Orientada a Objetos

- El nivel de programación es en un nivel más alto del que ofrece Visual C++, lo cual en muchas ocasiones no es muy deseable.
- El equipo en el que va a trabajar la aplicación necesita de mayor perfil de hardware y software para su correcto funcionamiento.

1.3.2 Conclusiones

ICARO WebServer ha sido implementado en Visual C++ utilizando POO, y como indicamos al incio de este documento es posible que no haya sido la mejor forma de implemetarlo, pero uno de los requerimientos para este proyecto era desarrollarlo utilizando el lenguaje y la técnica antes mencionada.

Sin emabargo fue una experiencia interesante y creemos que la herramienta se presta para darle toda la funcionalidad necesaria a ICARO WebServer.

1.4 ¿Qué es Orientado a Objeto?

Superficialmente el término Orientado a Objeto, significa que organizamos software como una colección de objetos discretos que incorpora estructura de datos y comportamiento. Esto está en contraste a la programación convencional en el cual la estructura de datos y el comportamiento están libremente conectadas. Hay algunas

disputas de acerca de que características exactamente son requeridas para un enfoque orientado a Objeto, pero por lo general incluyen cuatro aspectos:

- Identidad
- Clasificación
- Polimorfismo
- Herencia

1.4.1 Características de los Objetos

Identidad.- Significa que el dato es cuantificado en dentro de lo discreto, entidades distinguibles llamadas objetos

Clasificación: Significa que objetos con la misma estructura de datos (atributos) y comportamiento (operaciones) son agrupados dentro de clases. Una clase es una abstracción que describe propiedades importantes a una aplicación e ignora el resto, cada clase describe un conjunto infinito posible de objetos individuales.

Polimorfismo.- Significa que la misma operación podría comportarse diferentemente en clases distintas.

Herencia.- Es la compartición de atributos y operaciones de clases basadas en una relación jerárquica.

1.5 Programas servidores web (Web Servers)

Para implementar el nuevo paradigma de WWW intervienen partes bien diferenciadas:

- aplicaciones cliente,
- aplicación servidor y
- un protocolo de transferencia de hipertexto (HTTP).

Los programas servidores web, generalmente corren en plataformas multitarea, como Unix, NT, OS/2, AS/400, entre otros. Su funcion es ser el receptor de todos los requerimientos de los browser.

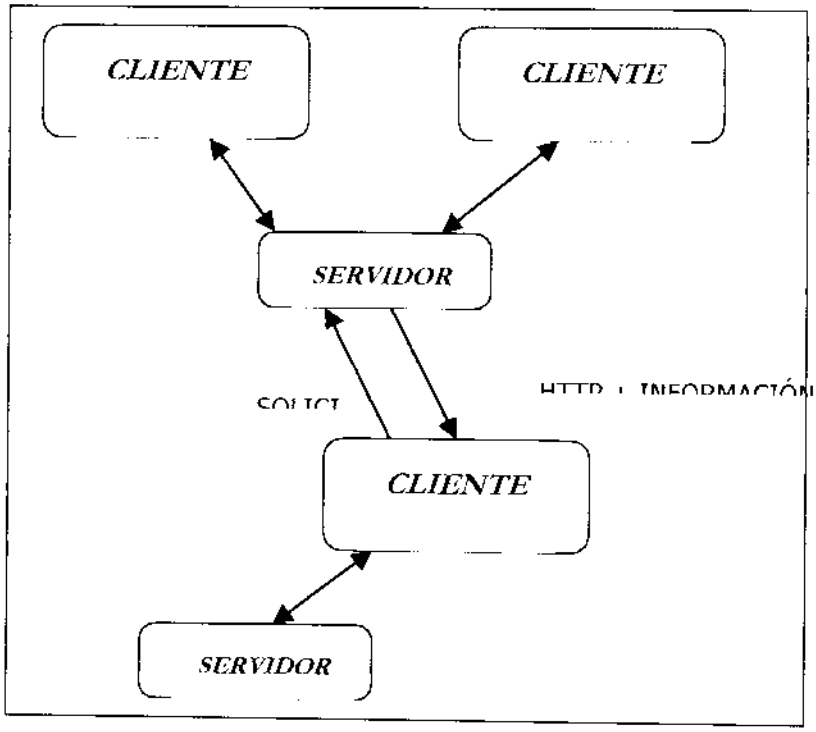


FIGURA 1.2

En general, un programa cliente HTTP establece una conexión TCP en anfitrión remoto. Enseguida, el cliente envía una solicitud al servidor HTTP. Después de que este envía una respuesta, él o el servidor cierra la conexión. (Figura 1.2)

Un poco más detallado, el proceso es: un cliente HTTP solicita que el servidor HTTP envíe un archivo (digamos, un archivo HTML) o un archivo hipermedia (un archivo de imagen, video, audio o animación). En la mayoría de los

casos la respuesta del servidor tan solo consiste en una transferencia de flujo de bytes de un archivo al puerto de protocolo local del cliente.

Pero el servidor web no solo es un servidor de archivos. Realmente las solicitudes hechas por los clientes especifican un nombre de archivo, una ubicación en Internet (dirección del anfitrión) y un método (habitualmente un protocolo como HTTP o FTP).

1.6 Open DataBase Connectivity (ODBC)

Open DataBase Connectivity (Conectividad abierta de bases de datos). Se trata de un Administrador de controladores, junto con diversos controladores ODBC, que permiten a las aplicaciones accesar a los datos utilizando SQL como lenguaje estándar.

Los drivers son el enlace crítico entre aplicaciones cliente/servidor bajo ODBC y bases de datos. Algunos colegas cuestionan el uso del ODBC SQL para la implementación de soluciones Cliente/Servidor porque es una solución genérica, y se preguntan: ¿no sería mucho más eficiente utilizar una solución específica para cada Sistema de Base de Datos?.

La intención de Microsoft cuando introdujo la especificación del ODBC, que luego se tornó un estándar de hecho, fue que los programas resultaran totalmente independientes del Servidor de Base de Datos y que, luego, en tiempo de ejecución, una DLL genérica resolviera automáticamente todas las conversiones necesarias.

Sin embargo, los Sistemas de Gerencia de Base de Datos soportados por los módulos ODBC actualmente en el mercado son muy diferentes unos de otros: sistemas relacionales (sin embargo todos diferentes), archivos *.DBF, que provienen de los diferentes dialectos del Dbase, archivos propietarios del Btrieve, etc.

Si se tratara de utilizar estos módulos de una forma transparente y generar el mismo código para Sistemas de Gerencia de Base de Datos diferentes, luego de resolver algunos pequeños problemas de compatibilidad que probablemente se presentarían de todas maneras, tendríamos soluciones ineficientes ya que cada Sistema de Gerencia de Base de Datos tiene sus particularidades que deben ser tenidas en cuenta e, incluso, explotadas para obtener las soluciones realmente eficientes.

1.6.1 Arquitectura ODBC

La arquitectura ODBC tiene cuatro componentes: (Figura 1.3)

- Aplicación - (Hoja Electrónica, Procesador de Palabra, Acceso a Datos & Herramientas de Desarrollo, Lenguaje de Desarrollo, etc.) realiza procesamiento pasando sentencias SQL y recibiendo resultados del

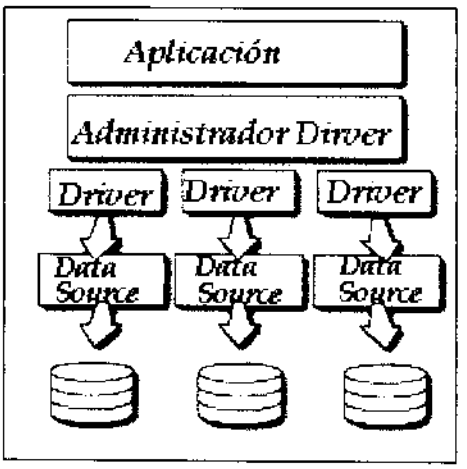


FIGURA 1.3

Administrador de Driver ODBC.

- Administrador de Driver - una Librería de Enlace Dinámico que carga los drivers en la corrida(behalf) de una aplicación.
- Driver – una Librería de Enlace Dinámico que procesa llamadas a procesos ODBC recibidas del Administrador de Drivers, enviando los pedidos resultantes SQL a un data source específico, y retornando los resultados a la

aplicación. Si es necesario, el driver modifica el pedido de la aplicación de tal forma que vaya de acuerdo con la sintaxis soportada por DBMS asociado.

- Data Source consiste de DBMS, el sistema operativo en el que el DBMS corre, y la red usada para accesar al DBMS (si existe alguna).

El Administrador del Driver y el Driver aparecen a la aplicación como una sola unidad que procesa las llamadas a las funciones ODBC.

Tipos de Drivers

ODBC define dos tipos de drivers:

- single-tier. El driver procesa llamadas ODBC y sentencias SQL.
- Multiple-tier. El driver procesa llamadas ODBC y pasa sentencias SQL al data source.

Un sistema puede contener ambos tipos de configuraciones. El siguiente párrafo describe las configuraciones single tier y multiple tier de forma más detallada.

Configuración Single-Tier

En una implementación single-tier, la base de datos es procesada directamente por el driver. El driver procesa sentencias SQL y recibe información de la base de datos. Un ejemplo de una implementación single-tier es un driver que manipula bases de datos de escritorio tales como DBASE, Paradox, FoxPro, etc.

El siguiente diagrama muestra dos tipos de configuraciones single-tier – el uno es stand alone y el otro usa una red. En un single-tier el driver de ambiente de red el software de acceso de datos reside en el PC, esto implica que la inteligencia de resolución del query reside en el cliente.13

Configuración Multiple-Tier

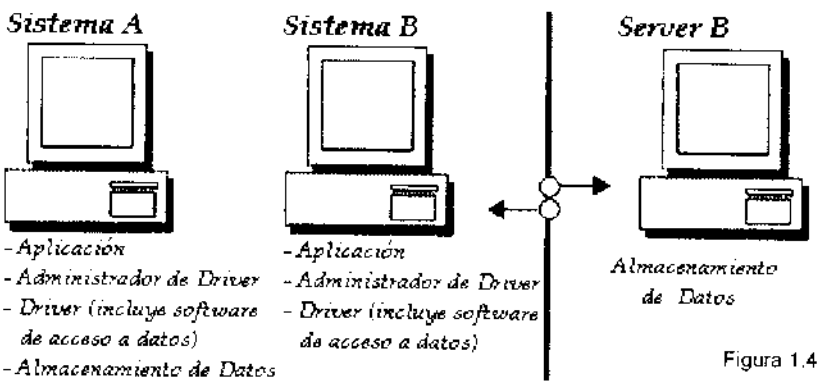


Figura 1.4

En una configuración multiple-tier, el driver envía los pedidos SQL a un servidor que procesa pedidos SQL.

La aplicación, driver, y Administrador de Driver residen en un solo sistema, típicamente llamado cliente. La base da datos y el software que controla el acceso a la base de datos reside comúnmente en otro sistema, típicamente llamado servidor. Esto implica que la resolución inteligente del query reside en el servidor.

Una variante de la configuración multiple-tier es una arquitectura gateway, donde el driver pasa pedidos SQL a un proceso gateway.

El proceso gateway envía los pedidos a la data source. El gateway en este escenario puede ser una pieza de hardware o software de acceso a datos en la forma de interface a bajo nivel a bases de datos foráneas por vendedores de RDBMS.

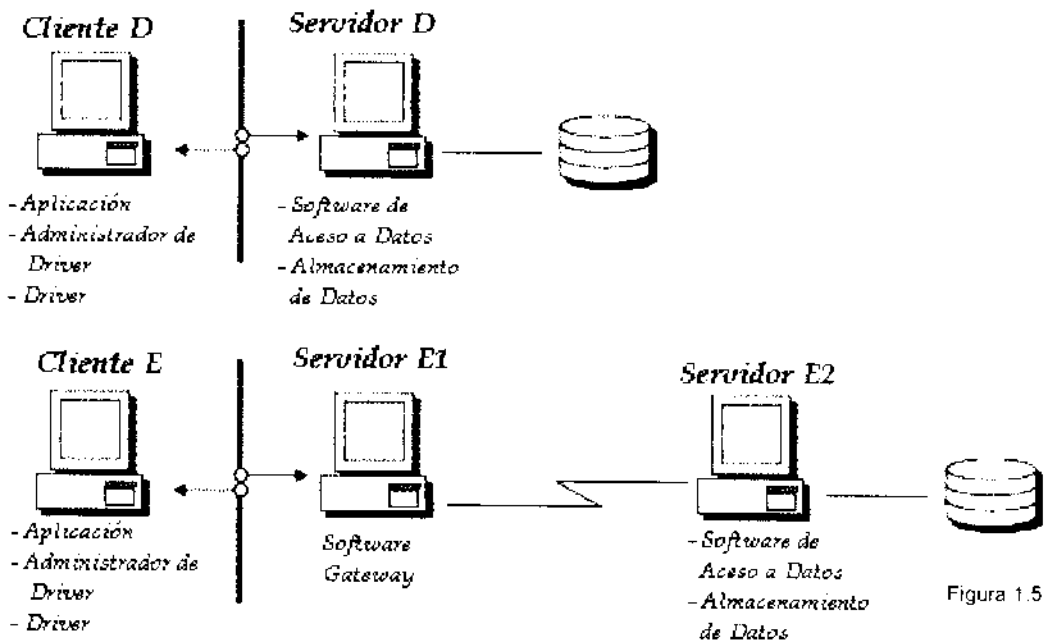


Figura 1.5

Los siguientes diagramas muestran dos tipos de configuraciones multiple-tier. Desde la perspectiva de una aplicación, ambas configuraciones con idénticas.

1.7 Cliente/Servidor.

Clientes y servidores son entidades lógicas separadas que trabajan juntas sobre una red para ejecutar una tarea.

Todos los sistemas cliente /servidor tienen las siguientes características:

Servicio: Cliente/Servidor es una relación entre procesos corriendo sobre máquinas separadas. El proceso Server es un proveedor de servicios. El cliente es un consumidor de servicios.

Recursos compartidos: Un servidor puede atender a muchos clientes al mismo tiempo y regular sus accesos a los recursos compartidos.

Protocolos Asimétricos: Hay una relación de Muchos a Uno entre clientes y servidor. Los clientes siempre inician el dialogo porque requieren un servicio. El Servidor esta pasivamente esperando requerimientos del cliente.

Transparencia de localización: El servidor es un proceso el cual puede residir sobre la misma máquina como el cliente o sobre una máquina diferente a través de una red. Software Cliente/Servidor usualmente disfraza la localización del

servidor de los clientes redireccionando las llamadas del servicio cuando se necesitan. Un programa puede ser un cliente, un servidor o ambos.

Mezcla e igualación: El software ideal cliente/servidor es independiente del hardware o plataformas de sistemas operativos.

Intercambio basado en mensajes: El mensaje es el mecanismo de liberar para la petición y respuesta de servicio.

Encapsulación de servicios: Un mensaje dice al Server que servicio es requerido; este es entonces subido al servidor para determinar como obtener el trabajo hecho.

Escalable: Sistemas cliente/servidor pueden ser escalables horizontalmente (agregando o removiendo estaciones cliente) o verticalmente (migrando a un servidor más grande y rápido o multiservers).

Integridad: El código y datos del servidor es centralmente mantenido. Los clientes permanecen independientes.

1.7.1 Infraestructura Cliente/Servidor

Bloque de construcción de cliente: Corre en el lado cliente de la aplicación sobre un sistema operativo que provee GUI o OOUI y que puede accesar a servicios distribuidos. También corre un componente de DSM.

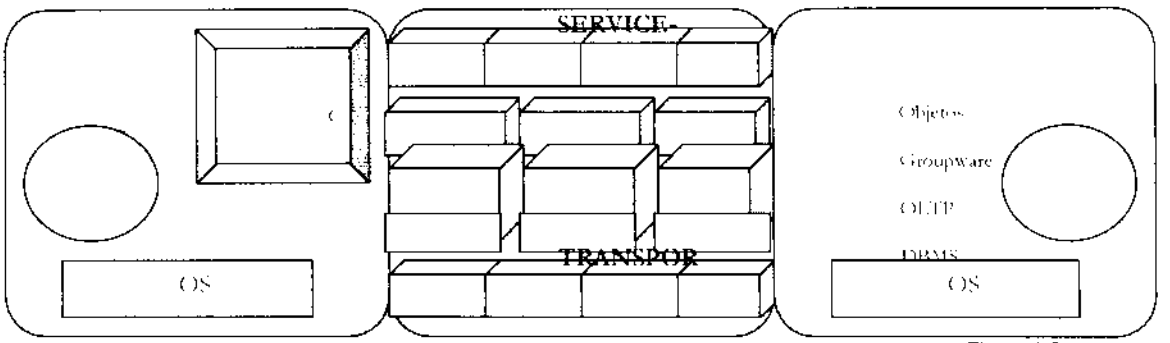


Figura 1.6

Bloque de construcción del servidor. Corre en el lado servidor de la aplicación. Para crear aplicaciones tiene: SQL Database Server, TP Monitors, Groupware Servers y Object Servers.

Bloque de construcción middleware: Corre sobre ambos lados el cliente y el servidor de una aplicación. Esta dividido en 3 categorías: Pila de transporte, sistema operativo de red (NOSs) y servicio-específico middleware. Middleware es el sistema nervioso de la infraestructura cliente/servidor y tiene componentes de software DSM.

plicación de Administración de Sistemas Distribuidos (DSM) sobre cada nodo en una red cliente/servidor. Una estación de trabajo de administración colecta información de todos sus agentes sobre la red y despliega esto gráficamente, también puede instruir a sus agentes ejecutar acciones sobre su interés.

Capítulo 2

Análisis y Diseño

2.1 Planteamiento

El problema que debemos analizar incluye en forma general la Implementación de un WebServer orientados a Base de Datos utilizando para esto ODBC. El Web Server debe accesar a su base de datos de donde recuperará información de las empresas clientes y de las consultas que ofrecen. En estas consultas están incluidas las sentencias SQL que generan las consultas de cada usuario. Esta información que retorna de las bases de datos será mostrada en archivos HTML, generados dinámicamente.

El objetivo es que los clientes (empresas públicas o privadas) pueden realizar transacciones comerciales con sus usuarios. Para demostración de las bondades del WebServer orientado a Base de Datos, solo se permitirán consultas de deudas de los servicios básicos como luz, teléfono, agua potable, predios, es decir, nuestros clientes serán empresas estatales y los usuarios inexpertos (esto es importante para el diseño de la Interacción Hombre Máquina).

2.2 ¿Por qué Orientado a Objetos?

Cuando nos enfrentamos a un problema y debemos hallar una solución de tipo computacional a éste, encontramos que básicamente lo podemos hacer de dos formas:

- Usando la Técnica Convencional[6]
- Usando la Técnica OO

La Técnica que utilicemos depende principalmente de la complejidad del problema que se plantee. A continuación exponemos algunas razones por las cuales nos inclinamos por la técnica OO antes que por la convencional:

- El mantenimiento de los sistemas OO es mucho más sencillo que el mantenimiento de sistemas convencionales.
- Un sistema OO se construye sin tener que pensar en ciclos, ramificaciones y estructuras para el control de programa. Para construirlo pensamos en términos de objetos y su comportamiento.

[6] Programación Estructurada

- La codificación es menor cuando la hacemos por objeto antes que por procedimientos de esta manera tendremos aplicaciones menos prospensas a errores (o más fáciles de detectar).
- La POO[2] nos permite ir generando una librería de objetos que puede ser utilizada por otros sistemas.
- La POO aumenta la confiabilidad. El hecho de usar objetos, de librerias que ya han sido probadas reduce el grado de error comparado con un sistema que es desarrollado con la técnica convencional desde cero.
- La POO es más sencilla. Los programas se forman a partir de piezas pequeñas, cada una de las cuales se puede crear y unificar fácilmente así como darles propiedades y características propias.
- La técnica OO utiliza el mismo paradigma para el análisis, diseño e implementación, y este paradigma es más realista o natural que en la técnica convencional.
- Los sistemas OO están diseñados para ser independientes de la plataforma, hardware y software. Éstos utilizan solicitudes y respuestas estándares, lo que

[1] Orientada a Objetos

[2] Programación Orientada a Objetos

permite utilizarlas en múltiples sistemas operativos y controladores de bases de datos.

- En los sistemas clientes-servidor la clase servidor puede ser usada por diferentes clientes. Los clientes sólo pueden tener acceso a los datos del servidor a través de los métodos de la clase, lo cual permite mayor seguridad de los datos.
- El desarrollo de un Sistema utilizando la técnica y la programación OO fué un requerimiento del tópico. Además esto nos permite evaluar nuestros conocimientos, la metodología y la herramienta de desarrollo[3].

2.3 Metodología

2.3.1 Metodologia de Análisis y Diseño

En análisis Orientado a Objetos, construimos 2 tipos de modelos:

- Modelo de tipos de datos y su estructura
- Módelo de lo que ocurre a los objetos

Los modelos se representan mediante diagramas llamados esquemas.

[3] Visual C++ 4.0

- Esquema de Objetos y Esquema de Eventos

Utilizamos como metodología

- AEO Análisis de la estructura de Objetos
- ACO Análisis del comportamiento de Objetos
  - Esquema de eventos
  - Diagrama de dependencias entre procesos
  - Diagrama de flujo de objetos
- DEO Diseño de la estructura de Objetos
- DCO Diseño del comportamiento de Objetos

2.4 Análisis

Análisis de Estructura de Objetos

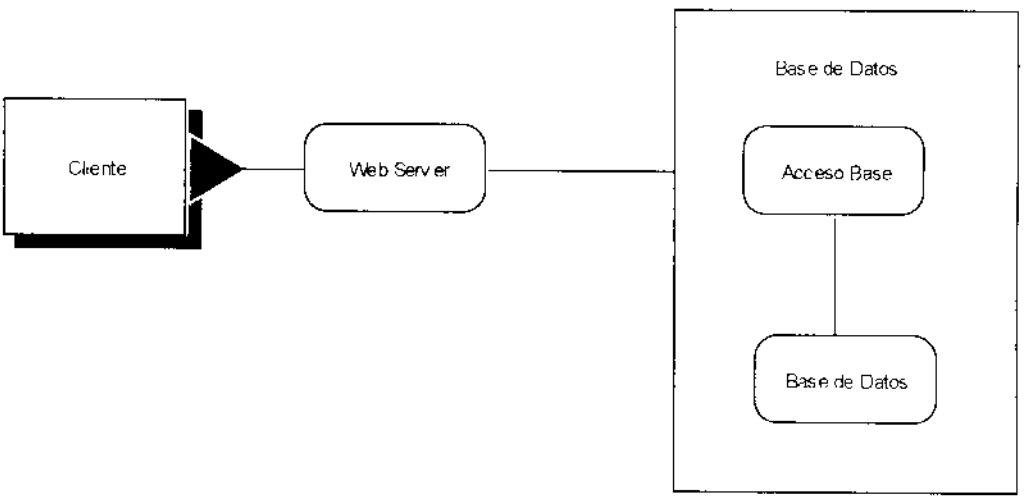


Diagrama de dependencia entre procesos

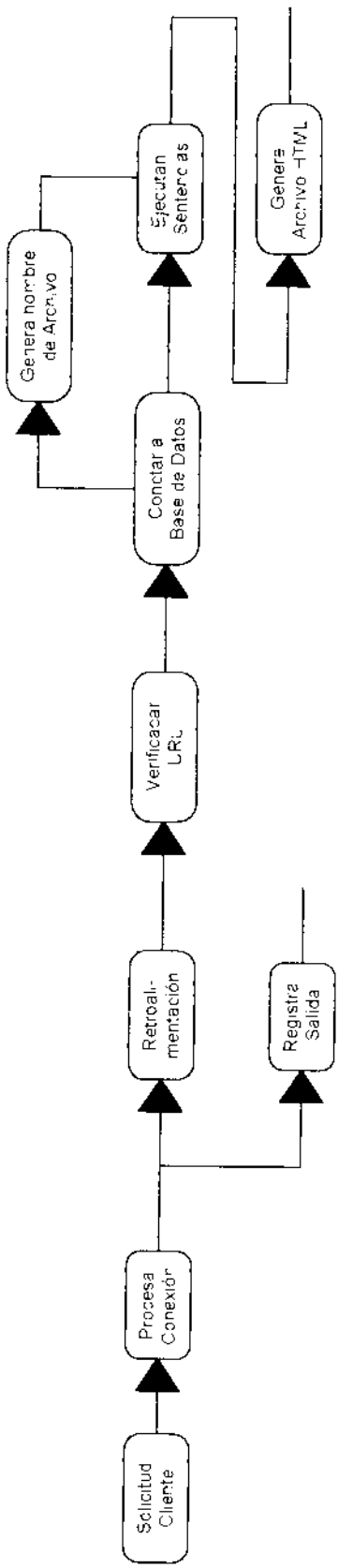
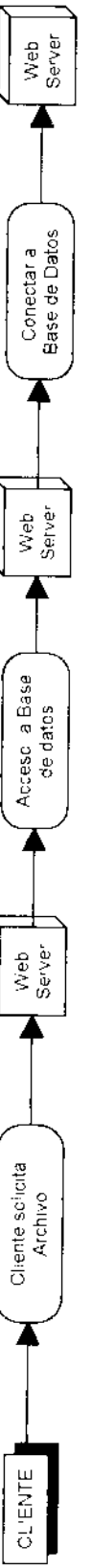
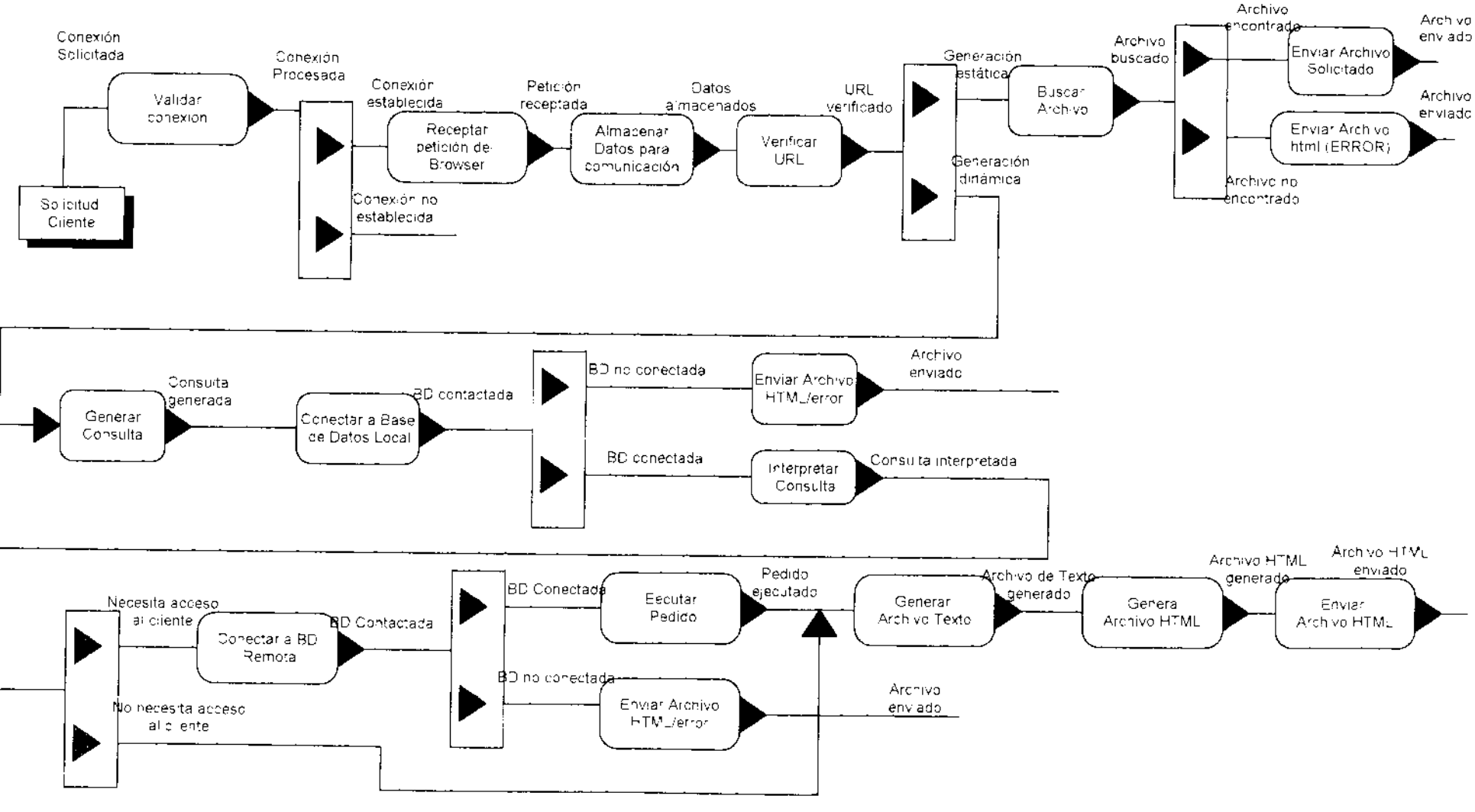


Diagrama de Flujo de Objetos



Esquema de Eventos



2.5 Diseño

Clase CIcaroView

Datos

Tipodedato WWWSockDef {

short Estado

SOCKET CliSock

CADENA BufferEntrada

CFile ArchivoEnvio

CADENA BufferEntrada[MAX_ENV_BUF]

int CantidadBufferEnvio

int RestoParaEnvio

CTime UltimaVez

CADENA CliIp

};

WWWSockDef WWWConv[MAX_CLIENTES+1];

SOCKET WWWServer;

SOCKET cli_sock;

Tipodedato SOCK_DIRECCION srv_addr, cli_addr;

LPSERVENT srv_info;

LPHOSTENT host_info;

SOCK_DIRECCION m_sockClientAddr;

int WWWPort;

int CantidadClientes;

CADENA Directorio;

CADENA NoExisteArchivo;

CADENA ServerCerrado;

Funciones

ConstrutorView()

DestructorView()

FUNCION_INICIALIZAR()

MatarConexion(ENTERO Numero_socket)

EnviandoBloque(ENTERO Numero_socket)

ProcesoRequerido(ENTERO Numero_socket)

ProcesoLectura(ENTERO Numero_socket)

LRESULT ClienteMensaje(WPARAM wParam, LPARAM lParam)

Clase CAccesoBase

Datos

CONSTANTE COLUMNAS_MAX 90

CONSTANTE MAX_CLIENTES 20

CONSTANTE SOCK_BUFER 500

CONSTANTE SOCK_LIBRE 4

CONSTANTE ESPERANDO 5

CONSTANTE ENVIANDO 6

CONSTANTE SOCK_LLENO 7

CONSTANTE CLIENTES_MSG WM_USER+1

CONSTANTE LLEGADA_MSG WM_USER+2

CONSTANTE MAX_REC_BUF 100

CONSTANTE MAX_ENV_BUF 500

EstructuraSocket PeticionSocket[MAX_CLIENTES+1];

SOCKET WWWServer;

SOCKET cli_sock;

LRESULT ServerMensaje(WPARAM wParam, LPARAM lParam)

Escribe_log(CADENA texto)

EstructuraSocketDireccion srv_addr, cli_addr;

SOCKADDR_IN m_sockClientAddr;

CADENA buf[500];

ENTERO Puerto;

ENTERO CantidadClientes;

CADENA Directorio;

CADENA ServerCerrado;

struct EstructuraSocket

{

ENTERO Estado;

SOCKET SocketCliente;

CADENA BufferInterno;

ARCHIVO EnviarArchivo;

CADENA BufferEnviado[MAX_ENV_BUF];

ENTERO CantidadBufEnv;

ENTERO Resto;

TIEMPO UltimaVez;

CADENA DireccionIP ;

};

Funciones

void Inicial();

void ImprimirCadena(CADENA ParaImprimir);

void ImprimirCadena(CADENA* ACadena);

void ImprimirCaracter(CADENA NroCaracter, CADENA FueraEsto = 'n');

void ProcesoRequerido(ENTERO NroSocket);

void ProcesoLectura(ENTERO NroSocket);

void EnviandoBloque(ENTERO NroSocket);

CADENA ImprimirPrimerCadena(CADENA LaCadena, ENTERO PalabraAdquir);

void MatarConexion(ENTERO NroSocket);

void Escribe_log(CADENA texto);

CAccesoBase()

~CAccesoBase()

RegistrarEntrada(String URL)

Retroalimentacion(String cliente, String URL, String IP)

RegistrarSalida(String IP)

VerificarURL(String URL)

InterpretaArchivo(String NombreArchivo)

GeneraNombreArchivo()

AnadirCabecera(String NombreArchivo)

Convertidor(String URL)

CrearSubmitHTML(String NombreArchivo)

Clase CBaseDatos

Datos

int longitud

Arreglo m_ArregloEmpresas

int ultimo_numero

puntero hstmt

puntero hdbc

puntero henv

int conectado

Funciones

CBaseDatos()

~ CBaseDatos()

Conectar(String dsn, String usuario, String clave)

Desconectar()

EjecutarSQL(String consulta)

ResultadoSQL(String URL)

2.5 Algoritmos de las funciones principales

2.5.1 Pseudocodigo Web Server

```
struct EstructuraSocket
{
ENTERO          Estado;
SOCKET          SocketCliente;
CADENA          BufferInterno;
ARCHIVO         EnviarArchivo;
CADENA     BufferEnviado[MAX_ENV_BUF];
ENTERO          CantidadBufEnv;
ENTERO          Resto;
TIEMPO          UltimaVez;
CADENA          DireccionIP ;
};

ConstructorServidor()
{
        WSADATA wsaData;
ENTERO error;
        CantidadClientes = GetProfileInt("ICARO","CantidadClientes", 10);
```

```
        error = WSAStartup(WS_VERSION_REQD, &wsaData);
        SI (error != 0)
        {
ImprimirCadena("WinSocket no pudo ser inicializado ");
WSACleanup();
                return;
        }
        VerificaVersionAdecuada();
SI (wsaData.iMaxSockets < (CantidadClientes + 1))
        {
ImprimirCadena("Esta aplicación requiere un mínimo de sockets soportados ");
        WSACleanup();
                return;
        }
        return;
}

DestructorServidor()
{
ENTERO Conteo;
ENTERO error;
CierraSocket(WWWServer);
CierraSocket(PeticionSocket.);
error = WSACleanup();
```

```
        if (error != 0)
        {
                ImprimirCadena("Error los sockets de Windows");
}
}

Inicial()
{
ENTERO err, Conteo;
DeclaraInterfaseGrafica();
ActualizarOpcionesMenu();
Directorio=LeeArchivoINI ("ICARO, "Directorio","C:\\ICARO");
NoArchivo=LeeArchivoINI ("ICARO, "NoArchivo","C:\\ICARO\\NoArchivo.htm");
ServerCerrado=LeeArchivoINI ("ICARO", "ServerCerrado","C:\\ICARO\\Cerrado.htm");

ConfigurarInterfaseGrafica();
WWWServer = SOCKET_INVALIDO;  // Si el server no esta conectado todavía
// Fija todas las estructuras de sockets no usadas
for (Conteo = 0; Conteo < CantidadClientes; Conteo++)
     PeticionSocket[Conteo].Estado = SOCK_LIBRE;
WWWServer = socket(AF_INET, SOCK_STREAM, 0);
VerificaSocket(WWWServer);
srv_addr.sin_family = PF_INET; //Familia de Protocolos Internet
srv_addr.sin_addr.s_addr = INADDR_ANY;
```

```
srv_addr.Puerto = Puerto;
SI (bind(WWWServer, &srv_addr, TamanoDe(srv_addr)) == SOCKET-ERROR)
        {
ImprimirCadena("No se puede ligar el socket del servidor");
        retum;
        }
        err = AsincronicaAdquisicion(WWWServer, Manejador, LLEGADA_MSG,
FD_ACCEPT);
SI (err == SOCKET-ERROR)
{
ImprimirCadena("el código de AsincronicaAdquisicion no trabaja en el server.");
return;
}
SI (listen(WWWServer,TiempoEspera) == SOCKET-ERROR)
{
ImprimirCadena("No se puede instalar el server para escuchar los clientes.");
return;
        }
        ImprimirCadena("INICIALIZACION <OK!>");
return;
}

MatarConexion(ENTERO NroSocket)
{
```

```
PeticionSocket[NroSocket].Estado = SOCK_LIBRE;
CerrarSocket( PeticionSocket[NroSocket].SocketCliente );
SI (PeticionSocket[NroSocket].EnviarArchivo.m_Archivo != Null)
PeticionSocket[NroCocket].EnviarArchivo.Close();
ImprimirCadena("Petición Cerrada ");
}

EnviandoBloque(ENTERO NroSocket)
{
ENTERO err;
while (1==1)
{
SI (PeticionSocket[NroSocket].CantidadBufEnv == 0)

PeticionSocket[NroSocket].CantidadBufEnv=
PeticionSocket[NroSocket].EnviarArchivo.Leer(PeticionSocket[NroCocket].BufferEnviado,
MAX_ENV_BUF);
PeticionSocket[NroSocket].Resto = 0;
                }
        PeticionSocket[NroSocket].UltimaVez = TIEMPO::AdquirirTiempo();
SI (PeticionSocket[NroSocket].CantidadBufEnv == 0)
{
                ImprimirCadena("ARCHIVO ENVIADO <OK!> - Terminando la
conexion");
```

```
        MatarConexion( NroCocket);
                        return;
        }
        ELSE
                {
                        err = send( PeticionSocket[NroSocket].SocketCliente,
        PeticionSocket[NroCocket].BufferEnviado+PeticionSocket[NroSocket].Resto,PeticionSock
et[NroCocket].CantidadBufEnv,0);
        SI (err != PeticionSocket[NroSocket].CantidadBufEnv)
                 {
        PeticionSocket[NroSocket].Resto = PeticionSocket[NroCocket].Resto+err;
        PeticionSocket[NroSocket].CantidadBufEnv = PeticionSocket[NroCocket].CantidadBufEnv
- err;
                 }
                ELSE
                {
        PeticionSocket[NroSocket].CantidadBufEnv = 0; PeticionSocket[NroSocket].Resto = 0;
                 }
        }
        }
        }

        ProcesoLectura(ENTERO NroCocket)
        {
```

```
ENTERO err;
ENTERO LecturaBytes=0;
CADENA Buf[500];

        SI (PeticionSocket[NroSocket].Estado != ESPERANDO)
{
return;
}
SI(PeticionSocket[NroSocket].BufferInterno=Obtener(PeticionSocket[NroSocket].SocketC
li
ente, Buf, LecturaBytes))
                ProcesoRequerido(NroSocket);
}

ProcesoRequerido(ENTERO NroSocket)
{
CADENA Texto, URL, tipo, mensaje, texto, log, URLreal;
ENTERO Conteo, err;
CADENA caracter;

Texto = ImprimirPrimerCadena( PeticionSocket[NroSocket].BufferInterno, 1);
SI (Texto.Comparar("GET") == 0)  // Browser con los que trabajamos
{
Texto = ImprimirPrimerCadena( PeticionSocket[NroSocket].BufferInterno, 2);
```

```
                SI (Texto.Comparar("/") == 1)
                {
        log = ObtenerArchivoLog();
                ImprimirCadena("log");
                        Escribe_log(log);          // Escribe en un archivo
        SI(Tipo=VerificarTexto(Text, '?'))
        Texto=ObtenerURLReal(Texto);
                SI (Texto.Mid(1,1) == 'D' && tipo.Comparar("htm")==0)

        CADENA URL_dinamico;
                CADENA ArchivoTxt;
                        ENTERO posicion_htm;
                                CTraductor TxtHtm;
                        posicion_htm=Texto.Find(".htm");
                                URL_dinamico=Texto.Mid(1,posicion_htm-1);
        CAccesoBase llama-base;
        ArchivoTxt=llama_base.Retroalimentacion(PeticionSocket[NroSocket].DireccionIP,texto,U
RL_dinamico);
        Texto=TxtHtm.Convertidor(ArchivoTxt);
        }
                }
        ELSE  //El Cliente ha solicitado el HOME

        ImprimirCadena("Un nuevo cliente esta siendo atendido:");
```

```
            Texto=("\icaro.htm");
        }
        Texto=ObtenerDireccionReal(Texto);
                    ImprimirCadena(": Tratando de enviar el archivo:");
                    ImprimirCadena(Texto);
                    SI(!PeticionSocket[NroSocket].EnviarArchivo.Open(Texto,
ModoLectura|typeBinary))
                    {
        ImprimirCadena("Archivo no existe");
                        MatarConexion( SockNumber );
        }
                    ELCE
                    {
        PeticionSocket[NroSocket].CantidadBufEnv = 0;
            EnviandoBloque(NroSocket);
                    }
        }
        ELSE
        {
                    ImprimirCadena("Información del Browse desconocida para el
sistema:");
            err = send( PeticionSocket[NroSocket].SocketCliente, "Método no
conocido",41,0);
                    MatarConexion( NroSocket );
```

```
}
}

RESULTADO ClienteMensaje(WPAMAMETRO IdenNroSocket, LPARAMETRO
IdenNroEvento)
{
ENTERO Conteo;
ENTERO LecturaBytes = 0L;

SI (AsincronicaAdquisicionError(IdenNroEvento) == 00)
        {
                ObtenerNumeroSocket() ;
                SI (Conteo == CantidadClientes)
                {
                        ImprimirCadena("Por favor espero unos minutos");
return 0L;
        }
        PeticionSocket[ Conteo ].UltimaVez = TIEMPO::AdquirirTiempo();
                SI (AsincronicaAdquisicionEvento(IdenNroEvento) == FD_LEER)
                {
                        ProcesoLectura( Conteo );
                        return 0L;
                }
                SI (AsincronicaAdquisicionEvento(IdenNroEvento) == FD_WRITE)
```

```
                {
        ImprimirCadena(": Ahora estoy habilitado para escribir en el cliente.");
        SI (PeticionSocket[Conteo].Estado == ENVIANDO)
                        EnviandoBloque(Conteo);
        return OL;
                }
        SI (AsincronicaAdquisicionEvento(IdenNroEvento) == FD_CLOSE)
                {
                        ImprimirCadena("Conexion terminada por el cliente.");
                        MatarConexion( Conteo );
                        return OL;
                }
            }
            return OL;
        }

        RESULTADO ServerMensaje(WPAMAMETRO IdenNroSocket, LPARAMETRO
IdenNroEvento)
        {
        PALABRA err;
            ENTERO Conteo;
            SOCKADDR_IN sad; // Accepta sockets variables
            ENTERO Ancho = TamanoDe(SOCKADDR);
            SOCKET TempSock;
```

```
ARCHIVO QFile;
CADENA mensaje;
SI (AsincronicaAdquisicionError(IdenNroEvento) == 0)
{
    SI (AsincronicaAdquisicionEvento(IdenNroEvento) == FD_ACCEPT)
    {
        ObtenerNumeroSocket();
SI (Conteo == CantidadClientes)
        {
            ImprimirCadena("Maximo número de conexiones entrantes");
return 0L;
        }
PeticionSocket[Conteo].BufferInterno = "";
        PeticionSocket[Conteo].Estado = ESPERANDO;
        PeticionSocket[Conteo].SocketCliente = accept(WWWServer, &sad, &Ancho);
        mensaje = CADENA("Cliente->");
        mensaje += sad.sin_addr;
        PeticionSocket[Conteo].DireccionIP = sad.sin_addr;
ValidarSocket(PeticionSocket[Conteo].SocketCliente);
err=AsincronicaAdquisicion(PeticionSocket[Conteo].SocketCliente, Manejador, CLIENTES_MSG, FD_LEER | FD_WRITE | FD_CLOSE);
    SI (err ==SOCKET_ERROR)
```

```
                    {
                    ImprimirCadena("No se trabaja mientras se acepta conexion");
                    MatarConexion( Conteo );
return OL;
}
}
        }
        retum OL;
}
```

2.5.2 Pseudocodigo Base De Datos

```
CONSTANTES:
        MAXBUFFER 3000
        LONGERROR 73

CADENA Retroalimentacion(CADENA IP,CADENA cliente,CADENA URL)
{
        CADENA resultado;
        CADENA archivo-resultado;
        CADENA cadena_ENTEROerpretar;
```

```
CADENA sentencia_2;
CADENA usuario;
CADENA DNS;
CADENA clave;
CBaseDatos base–remota;
CADENA error;
ENTERO remota = 0;
LIMPIAR archivo–resultado;
LIMPIAR sentencia_2;
SI (m_conectado != 1)
{
        estatus = LLAMAR m_base_local.Conectar("empresas","katty","katty");
        SI (estatus == 1) {
                VISUALIZAR("No se pudo conectar a la BD");
                RETORNAR(archivo_resultado);
        }
        CASO CONTRARIO{
                LLAMAR Registrar_entrada(IP);
                m_conectado = 1;
        }
}
resultado = VerSIicarURL(URL);
SI (resultado ESTA VACIO)
        CONCATENAR(URL, m_anadeURL);
```

```
        sentencia–2 = "Select m_empresa, em_descripcion, em_logo, em_mensaje, em_dns,
em_user, em_password from empresa ";
                        CONCATENAR (sentencia–2, " where em_codigo = '" + m_compania +
"'");
                cadena_ENTEROerpretar = LLAMAR
m_base_local.EjecutarSQLcampo(sentencia_2);
                        LLAMAR InterpretaArchivo(cadena_ENTEROerpretar);
                        UMPIAR sentencia–2;
                sentencia–2 = "Select plan–archivo from plantillas,pantalla_plantilla ";
        CONCATENAR (sentencia–2 ," where pp_codigo='" + URL + "' and pp_cod_plantilla =
plan_codigo");
                m_NombrePlantilla = LLAMAR m_base_local.EjecutarSQLcampo(sentencia_2);
                        SI (m_consulta NO ESTA VACIA){
                                LIMPIAR sentencia–2;
        sentencia–2 = "Select con_nombre from consulta where con_codigo='" + m_consulta +
'11111.
                                NombreConsulta = LLAMAR
m_base_local.EjecutarSQLcampo(sentencia_2);
                }
                SI(m_varios_registros == 0)
                {
                        SI (cliente NO ESTA VACIA){
                                CADENA queryretornado= LLAMAR
        m_base_local.EjecutarSQLcampo(resultado);
```

```
                    ENTERO estatus;
        estatus = LLAMAR base_remota.Conectar(m_Cabecera[4], m_Cabecera[5],
m_Cabecera[6]);
                    SI (estatus == 1) {
                        VISUALIZAR("No se pudo conectar a la BD");
                        RETORNAR(archivo_resultado);
                    }
                    CASO CONTRARIO{
                        remota = 1;
        archivo_resultado = LLAMAR base_remota.EjecutarSQLURL(queryretornado, URL);
                        LLAMAR base_remota.Desconectarse();
                    }
                }CASO CONTRARIO
                    {
                        remota = 1;
                        archivo_resultado = LLAMAR
base_remota.GenerarNombreArchivo();
                    }
            }
            CASO CONTRARIO{
                archivo_resultado = LLAMAR
m_base_local.EjecutarSQLURL(resultado,URL);
            }
            SELECCION (remota)
```

```
            {
            CASO 0:
                        LLAMAR
AnadirCabecera(archivo_resultado,m_base_local.longitud);
                        TERMINAR;
            CASO 1:
                        LLAMAR
AnadirCabecera(archivo_resultado,base_remota.longitud);
                        TERMINAR;
            CASO 2:
                        LLAMAR AnadirCabecera(archivo_resultado,error.GetLength());
                        TERMINAR;
            }
            RETORNAR(archivo_resultado);
      }

      CADENA VerificarURL(CADENA URL)
      {
      CADENA fragmento;
      CADENA sentencia;
      CADENA cod_contrasena;
      fragmento = EXTRAER(URL,0,1);
      SI (fragmento == "D"){
       m_compania = EXTRAER(URL,1,2);
```

```
m–consulta = EXTRAER(URL,3,2);
cod–contrasena = EXTRAER(URL5,1);
SI (m–compania ESTA VACIO && m_consultaESTA VACIO){
sentencia = "select em_codigo, em_status, ernernpresa from empresa where em_activo
='S' order by em_empresa";
m_varios_registros = 1;
}
CASO CONTRARIO
SI (m–compania NO ESTA VACIO && m–consulta NO ESTA VACIO){
sentencia="select con_codigo,con_estatus, con–nombre from empresa_consulta,
consulta";
sentencia +=" where ec_empresa='" + m–compania + "' and con_codigo =
ec_consulta" ;
m_varios_registros = 1;
}
CASO CONTRARIO{
SI (m–consulta ESTA VACIO && cod–contrasena ESTA VACIO){
sentencia = "Select ec_contrasena from empresa–consulta where ec_empresa='" +
m–compania + "'";
sentencia += " and ec_consulta = '" + m–consulta +
"'";
cod–contrasena = LLAMAR
m_base_local.EjecutarSQLcampo(sentencia);
SI (cod_contrasena=="S"){
```

```
                                        sentencia = "Select ec_contrasena, con–submit
";
                                        sentencia += " from empresa_consulta,consulta
where ";
        sentencia += " ec_empresa = '" + m_compania + "' and con–codigo = '" + m–consulta
+ "' and con–codigo = ec–consulta";
                                        m_varios_registros = 1;
                                    }
                                    CASO CONTRARIO(
                                        SI (m–consulta == "EM")
                                        {
        sentencia = "Select ec_contrasena, em_status, em_email from
empresa_consulta,empresa";
        sentencia += " where em_codigo = '"+ m_compania +"' and ec–consulta = '";
                                            sentencia+= m–consulta + "' and
em_codigo = c_empresa";
                                            m_varios_registros = 1;
                                        }
                                        CASO CONTRARIO(
                                            m_anadeURL = "N";
        sentencia="Select q_descripcion from query, empresa_consulta ";
        sentencia+= "where ec_empresa = '" + m_compania + "' and ec–consulta = '" +
m–consulta + "'";
```

```
                                                    sentencia+= " and
ec_codquery=q_codigo";
                                                    m_varios_registros = 0;
                                                }
                                            }
                                }
                                    CASO CONTRARIO{
                                        sentencia="Select q_descripcion from
query,empresa–consulta ";
        sentencia+= "where ec_empresa = '" + m_compania + "' and ec_consulta = '" +
m_consulta + "'";
                                        sentencia+= " and ec_codquery=q_codigo";
                                        m_varios_registros = 0;
                                }
                            }
                    }
        RETORNAR(sentencia);
        }
```

```
        CONSTANTES:
        TIMEOUT         5
        MAXFILAS    20
        MAXCOLUMNAS 50
```

```
MAXLONGITUD MAXFILAS*(MAXCOLUMNAS+1)
NULLDATASTRING "SQL_NULL_DATA"
LONGCADENA 62


CADENA EjecutarSQLURL(CADENA SentenciaSQL,CADENA URL)
{
 char* sentencia;
 CADENA NombreArchivo;
 RETCODE retcode;
 retcode = 0;
 ENTERO error = 0;
 SI (conectado == 1){
        sentencia= (char *)SentenciaSQL;
        retcode = LLAMAR SQLExecDirect(hstmt,sentencia,SQL_NTS);
        SI (retcode == EXITOSO)
        {
                retcode = LLAMAR SQLNumResultCols(hstmt, &nCols);
                SI ( nCols > 0) {
                        NombreArchivo = LLAMAR ResultadoSQLURL(URL);
                }CASO CONTRARIO
                        error = 1;
        } CASO CONTRARIO
                error = 1;
```

```
}CASO CONTRARIO
        error = 1;
SI (error ==1)
{
        NombreArchivo = '\0';
        longitud = 0;
}
RETORNAR (NombreArchivo);
}

CADENA ResultadoSQLURL(CADENA URL)
{
  SWORD cont;
  char rgbData[MAXFILAS][MAXCOLUMNAS] ;
  SDWORD dwDataLen[MAXFILAS];
  char szDispBuffer[MAXLONGITUD+1];
  ENTERO nRows;
  RETCODE nReturn;
  SWORD nCount;
  DWORD dwText;
  char contador[4];
  time_t tiempo;
  ARCHIVO f;
  CADENA NombreArchivo;
```

```
char url[MAXLONGITUD+15];
longitud = 0;
time(&tiempo);
itoa(tiempo,contador,10);
NombreArchivo= "c:\\poli"+ (CADENA) contador + ".txt";
ABRIR ARCHIVO f(NombreArchivo,CFile: :modeCreate | CFile: :modeWrite );
REPETIR (cont = 0;cont < nCols;cont++)
        SQLBindCol(hstmt,(cont+1),SQL_C_CHAR,rgbData[cont],MAXCOLUMNAS,&dwDa
taLen[cont]);
REPETIR(nRows = 0; (nReturn = SQLFetch(hstmt))==EXITOSO || nReturn == EXITOSO
CON     INFORMACION;)
{
        url[0]='\0';
        CONCATENAR(url,URL);
        REPETIR(nCount=0, szDispBuffer[0]='\0'; nCount<nCols; nCount++)
        {
                SI (nCount == 0){
                   CONCATENAR(url,rgbData[nCount]);
                   dwText = LONGITUD(url);
                   CONCATENAR(szDispBuffer,url);
                   dwText = LONGITUD(szDispBuffer);
                    szDispBuffer[dwText++] = '\t';
              szDispBuffer[dwText] = '\0';
                }
```

```
                CASO CONTRARIO{
                   SI (dwDataLen[nCount]  NO EXISTE DATOS)
                                CONCATENAR(szDispBuffer, NULLDATASTRING);
                        CASO CONTRARIO
                                CONCATENAR(szDispBuffer,rgbData[nCount]);
                        dwText = LONGITUD(szDispBuffer);
                        szDispBuffer[dwText++] = '\t';
                        szDispBuffer[dwText] = '\0';
                }
        }
        SI (*szDispBuffer)
        {
                        dwText = LONGITUD(szDispBuffer);
                        szDispBuffer[dwText++] = '\n';
                        szDispBuffer[dwText] = '\0';
        }
        CASO CONTRARIO
                RETORNAR;
        CONCATENAR(longitud,LONGITUD(szDispBuffer));
        ESCRIBIR EN ARCHIVO f(szDispBuffer,LONGITUD(szDispBuffer));
  }
  CERRAR ARCHIVO  f;
  RETORNAR (NombreArchivo);
}
```

Capítulo 3
TCP/IP y La API Windows Sockets

3.1 TCP/IP

3.1.1 Importancia de Protocolos TCP/IP

Si una red esta conectada a Internet esta depende totalmente de una serie de protocolos llamados en conjunto TCP/IP, los cuales mueven información a través de esta red, cada uno de estos protocolos transfiere datos usando formatos diferentes y con distintas opciones (CRC, sumas de comprobación, mensajes de confirmación, etc.).

3.1.2 Conjunto de Protocolos TCP/IP

Son una colección de protocolos cooperativos y complementarios que trabajan juntos para comunicar información a través de Internet.

- TCP

Es un protocolo basado en conexión y es análogo a una conexión telefónica, el usuario debe establecer una conexión antes de continuar con el proceso de transmisión.

- UDP

Es un protocolo sin conexión y la transacción entre los dos equipos es como pasar una nota: se envía un mensaje de un equipo a otro, pero no existe una conexión explícita entre ambos. Además, el tamaño máximo de los datos en envíos individuales esta determinado por la red.

- IP

El protocolo Internet actúa en la capa de red que mueve la información entre computadores anfitriones.

- ICMP

Es un protocolo de control de mensajes de Internet, lleva mensajes de error de a red y notifica otras condiciones que requieren atención del software de red.

- IGMP

Protocolo de manejo de grupos.

- ARP

Protocolo de resolución de direcciones.

- RARP

Protocolo de resolución de direcciones inverso.

3.1.2.1 Características

Existen unas características que definen a los protocolos TCP/IP. Estas características definen las diferencias entre los protocolos de transporte UDP y TCP, éstas se refieren a los tipos de conexiones de red, confiabilidad en transporte de datos y los servicios de información que prestan

3.1.2.1.1 Conexiones de Red

Estas pueden ser orientadas a *conexión* o *sin conexión.* Un protocolo orientado a conexión no puede comunicarse o transportar datos hasta que establezca una conexión (TCP). Un protocolo sin conexión es aquel que no establece un enlace antes de transmitir un mensaje, este es pasado a la siguiente capa en la pila de protocolos y confía en que la red lo entregue. Como ejemplo de protocolos sin conexión tenemos a IP y UDP.

3.1.2.1.2 Confiabilidad

Los protocolos pueden ser *confiables* y *no confiables.* Un protocolo confiable garantiza la entrega de datos, este utiliza sumas de comprobación, mensajes de

confirmación y otras técnicas para asegurar la entrega de datos confiable, el protocolo TCP es un ejemplo de un protocolo conhable. Un protocolo no confiable no asegura la entrega de datos, tratan de entregarlos pero no garantizan el éxito, además no informa a la aplicación transmisora cuando falla el esfuerzo para la entrega.

3.2.2.3 Servicio de Datos

Existen dos tipos básicos de servicios de información, el de *flujo de bytes* y el de *datagrama*. Si un protocolo utiliza el servicio de flujo de bytes, transmite la información como un flujo serial de bytes sin importar la longitud o el número de transmisiones requeridas para enviar o recibir toda la información, además garantiza que el otro lado de la conexión recibirá los datos en el mismo orden de la secuencia de transmisión. En contraste un protocolo que utiliza el servicio de datagrama entrega datos como unidades individuales e independientes de información, pudiendo estos tomar diferentes caminos para llegar a su destino y no necesariamente tienen que llegar en la misma secuencia de transmisión.

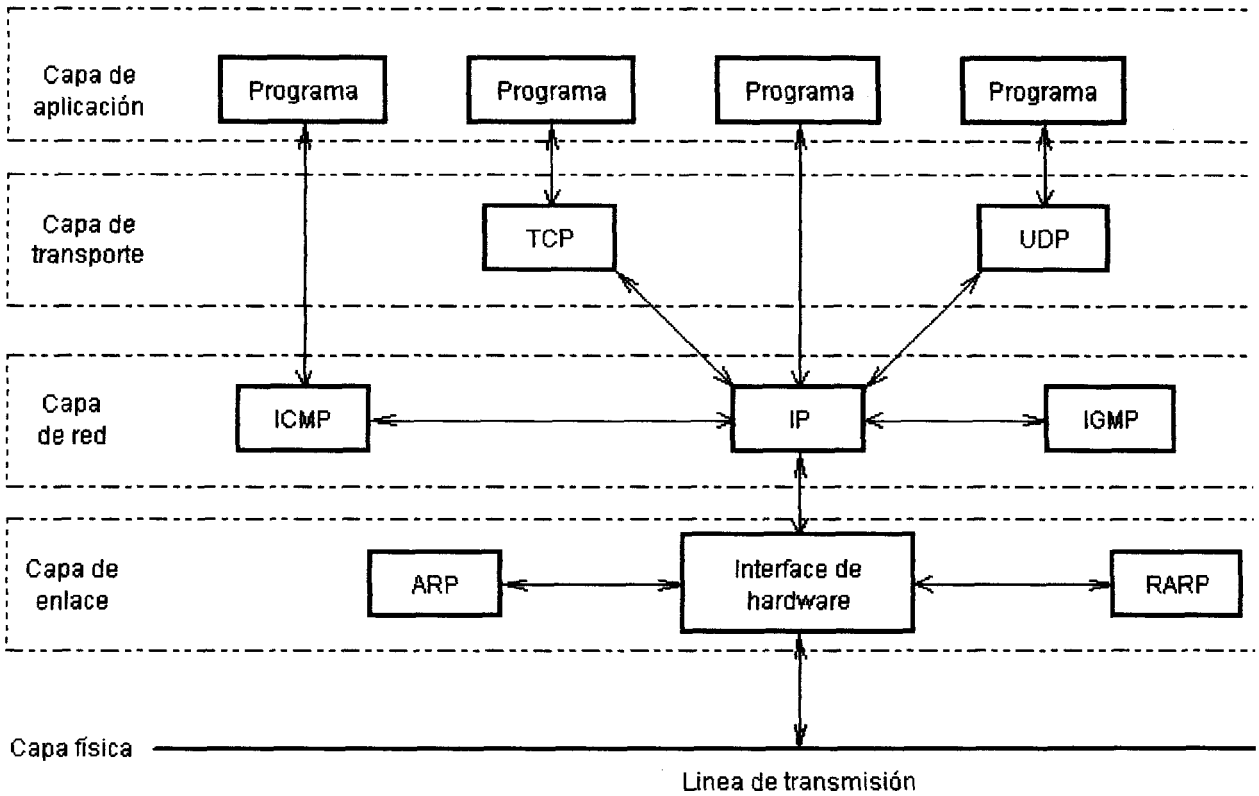


Figura 3.1 Modelo de red TCP/IP con protocolos asociados.

Por lo general un protocolo sin conexión es no confiable y tiene servicio de información de datagrama. Un protocolo orientado a conexión es confiable y tiene servicio de información de flujo de bytes.

3.2 Protocolo Internet

La capa de red decide como empaquetar los mensajes para transportarlos a través de la red TCP/IP. La capa de red emplea el Protocolo Internet (IP) para tal

propósito, IP es el sistema de entrega para el conjunto de protocolos TCP/IP. La capa de red es el corazón de cualquier red basada en TCP/IP, esta capa incluye IP, Protocolo de Control de Mensajes de Intemet (ICMP) y el Protocolo de Manejo de Grupos de Intemet (IGMP).

3.2.1 Direcciones de Internet

Una dirección de Intemet es una dirección IP. La mayoría de los usuarios e incluso la literatura sobre Intemet asocian las direcciones IP a las computadoras anfitrión, pero la computadora en realidad no tiene una dirección IP.

Así como Ethemet asocia sus direcciones con una tarjeta de interface de red, una red TCP/IP asocia las direcciones IP con una tarjeta de interface, no con la computadora anfitrión. Si consideramos que una computadora puede tener varias tarjetas de interface de red, significa que una computadora anfitrión puede tener varias direcciones IP válidas.

Una dirección IP válida es de *32* bits o **4** bytes de longitud. Aunque existen varios tipos de notación, pero la más utilizada es la "notación decimal", la cual

representa una serie de números decimales separados por puntos (Ej: 100.100.100.100).

3.2.2 Decodificación de Direcciones IP

La dirección IP combina un número de red y un número de dirección (interface). Para asegurar que las miles de redes que componen Internet tengan una dirección única InterNIC (Centro de Información de Red Internet) diseño que el byte de mayor orden identifique a la red y los tres de menor orden a la interface de red (computadora anfitrión). Además el software de Internet interpreta a un campo de sólo 1 como de "todos" o destinado a todas las interfaces de red (computadoras), de igual manera un campo de sólo 0 (ceros) como destinado a "esta" red o interface, por tanto estas direcciones son reservadas.

3.2.3 Clases de Direcciones

Para mayor facilidad en la administración de las redes (que con el anterior diseño eran pocas redes -255- con muchos anfitriones), se ideo un nuevo sistema de codificación simple pero efectivo: las direcciones IP no usarían mas el byte de mayor orden sino los primeros bits del primer byte para identificar una *clase de dirección*. Esta

especifica cuantos bytes utiliza la dirección como número de identificación de dirección de la red.

3.3 interface de sockets

Es una interface de programas de aplicación (API) para redes TCP/IP. Una API es un grupo de funciones que emplean los programadores a fin de desarrollar aplicaciones para un ambiente de cómputo específico. A ésta API también se la conoce como **Interface de Sockets de Berkeley.**

Un socket es un extremo de la comunicación por red. El socket es una representación abstracta del extremo en el proceso de comunicación. Involucra a dos computadoras anfitriones o procesos que se pasan datos entre si a través de la red. Cuando un programa utiliza la interface de sockets para una comunicación en red, necesita un socket en cada extremo de la comunicación para poder " conversar ", el un extremo es el local y el otro es el remoto. Analizando este proceso desde el punto de vista del modelo cliente/servidor, se necesita un socket para la aplicación que inicia la búsqueda activa (ciiente) y otro para la aplicación pasiva (servidor). Cuando se necesita un socket se define sus características y se utiliza una API para solicitar al software de red un identificador que reconozca al socket especificado. Los programas

basados en sockets crean un socket y entonces como paso aparte, lo conectan a un extremo destino. Los sockets pueden ser orientados a conexión o sin conexión. Un proceso de comunicación usando sockets utiliza el modelo básico de abrir - leer - escribir - cerrar.

3.3.1 Creación de un Socket

Para crear un socket, el sistema llama a la función **socket()** como se muestra a continuación:

```
Identificador_socket = socket(familia_protocolos, tipo–socket, protocolo);
```

Esta función devuelve un identificador (llamado también descriptor) del socket, con el cual se lo referencia para su configuración y en el proceso de comunicación, este identifica a un registro en la tabla de descripción. De hecho "crear un socket" significa reservar espacio de almacenamiento para la estructura del socket. Como desarrollador de aplicaciones no es necesario preocuparse tanto por detalles como tablas descriptoras, estructuras de datos internas y lo relacionado con la asignación de memoria.

A continuación analizamos los parámetros de la función socket, los parámetros que se encuentran con un visto son los que se usan en el desarrollo de nuestra aplicación.

Fada–protocolos:

- ✓ PF_INET Familia de protocolos de TCP/IP
- PF_NS Familia de protocolos de XNS
- PF_UNIX Familia de protocolos de UNIX

Tipo–socket (tipo de servicio de información):

- ✓ SOCK_STREAM Para flujo de bytes
- SOCK_DGRAM Para datagramas

Protocolo:

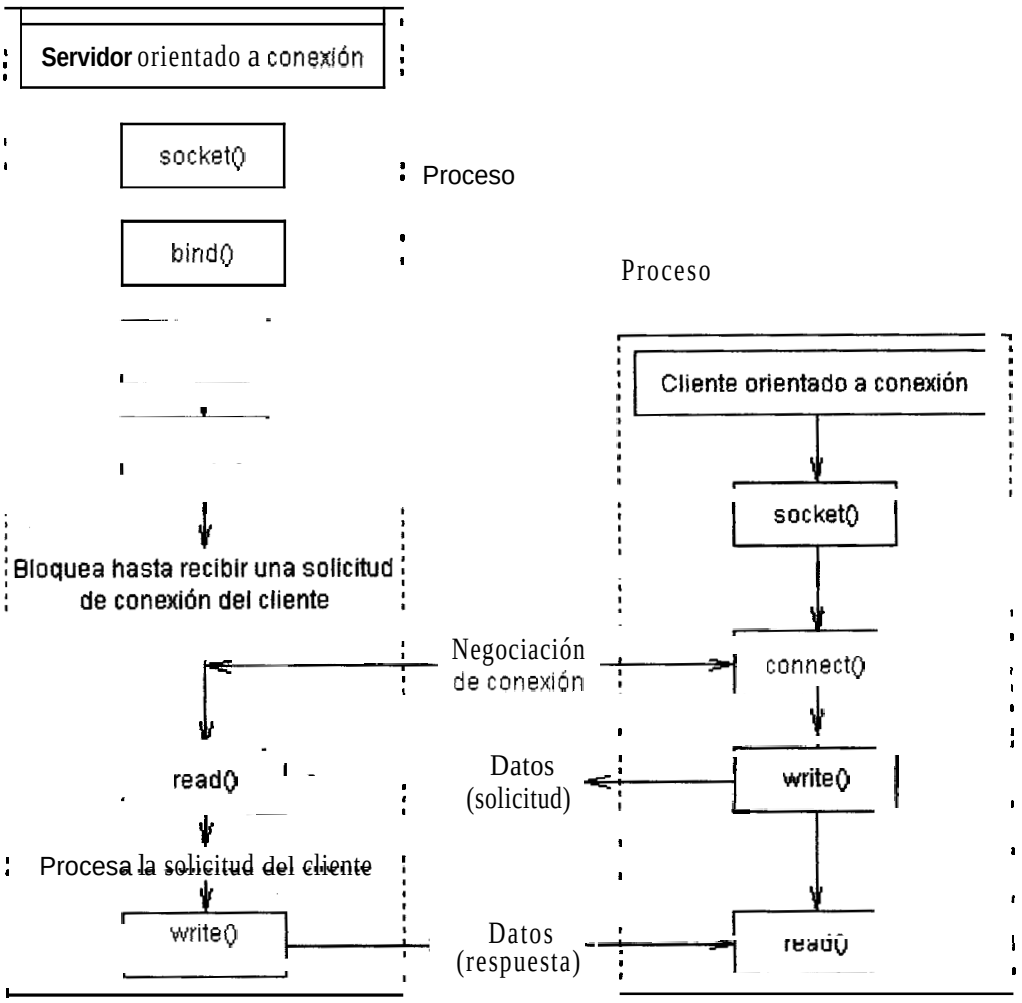
- IPPROTO_TCP Protocolo TCP
- IPPROTO_IP Protocolo IP
- IPPROTO_UDP Protocolo UDP
- IPPROTO_ICMP Protocolo ICMP

J 0 Si no se desea especificar un protocolo en particular

Como etapa independiente en la creación y configuración de un socket, la aplicación debe utilizar la función connet() a fin de conectarse a un extremo específico. Además la API de sockets incluye, para cada función de recepción una de transmisión que le corresponde.

3.3.2 Panorama del proceso de transmisión de datos por un socket

Uso de sockets con un protocolo orientado a conexión Figura 3.2.



3.4 La API Windows Sockets (Winsock)

En los Últimos años el ambiente de programació que ha crecido con mayor rapidez es el de Microsoft Windows en sus versiones para PC (Windows 95) y de red (Windows NT). Conectarse a Intemet desde un PC basado en Windows es prácticamente un estándar hoy en día.

3.4.1. Como se adecua WinSock en el ambiente Windows

La API Winsock proporciona una biblioteca de funciones (WINSOCK.DLL) que pueden usar los programas para lograr tareas específicas. Winsock organiza la biblioteca en tres grupos:

- Funciones de los sockets de Berkeley incluidas en el API.
- Funciones de bases de datos para obtener información de Internet acerca de nombres de dominios, servicios de comunicaciones y protocolos.
- Las extensiones específicas de Windows a las rutinas de sockets de Berkeley.

3.4.1.1 Las funciones sockets o de conexión.

Las identificamos como de *bloqueo* y de *no bloqueo*. Una función de bloqueo evita que un programa llame a cualquier otra función de Winsock hasta que la propia

función de bloqueo termine sus operaciones en la red. Una función de no bloqueo termina de inmediato su operación, en otras palabras "no espera'' hasta que la operación concluya. Es común que las funciones de E/S que utilizan bloqueo sean funciones estilo Berkeley, se debe notar que cualquier función que ejecute o dependa de E/S en la red puede bloquear, casi siempre otras funciones de Winsock.

Las principales funciones de bloqueo son: accept, closesocket, connet, recv, recvfrom, select, send, sendto. Entre las funciones tipo Berkeley que no bloquean en la **API** Winsock tenemos: bind, getpeername, getsockname, getsockopt, htonl, htons, inet_addr, inet_ntoa, ioctlsocket, listen, ntohl, ntohs, setsockopt, shutdown, socket.

3.4.1.2 Las funciones de base de datos

Los programas basados en Internet funcionan con diferentes tipos de direcciones. Winsock proporciona una serie de funciones apropiadas para trabajar con este tipo de información, como son nombres de dominios, servicios de comunicación y protocolos. Estas funciones se las utiliza principalmente en el procesamiento de direcciones IP (165.155.22.103), recuperación de nombres de dominio (espol.edu.ec), etc.

En la versión síncrona de estas funciones tenemos: gethostbyaddr, gethostbyname, gethostname, getprotobyname, getprotobynumber, getservbyname, getservbyport. La API Winsock incluye versiones asíncronas específicas de windows, de todas estas funciones excepto de gethostname, para nombrarlas anteponemos el prefijo WSAAsync. Varias de estas funciones devuelven apuntadores *(o* estructuras) de datos volátiles. La información volátil que contienen las áreas de buffer de Winsock sólo es válida hasta la próxima llamada que haga su programa a una función Winsock.

3.4.1.3 Las funciones de extensión específicas de Windows

Son versiones asíncronas especiales de ciertas funciones socket para poder aprovechar las características de despliegue de mensajes dentro de Windows.

Estas funciones asíncronas son: WSAAsyncSelect, WSACancelAsyncRequest, WSACancelBlockingCall, WSACleanup, WSAGetLastError, WSAIsBlocking, WSASetBlockingHook, WSASetLastError, WSAUnhookBlockingHook, WSAStartup.

3.4.2 Diferencias entre sockets de Berkeley y de Winsock

La interface de sockets de Berkeley que es una API para trabajar con múltiples sistemas operativos, la API Winsock se dirige exclusivamente a la familia de sistemas operativos de Microsoft Windows.

La interface se sockets de Berkeley implementó la **API** como parte del sistema operativo (UNIX), Winsock implementa la interface de sockets como una biblioteca de enlace dinámico (DLL) independiente del sistema operativo.

En los sockets de Berkeley un programa de red debe incluir varios encabezados de archivo, en Winsock solo necesitan un encabezado de archivo *winsock.h*

Los sockets de Berkeley rigen mayoritariamente el desarrollo de software TCP/IP en el mundo UNIX. Winsock en el ambiente de Microsoft Windows.

Las funciones de los sockets de Berkeley son sincrónicas las de Winsock asincrónicas.

3.4.3 Cómo maneja Winsock las operaciones de bloqueo en Windows

Puesto que Windows es un ambiente multitarea, es muy importante comprender como el bloqueo afecta a la ejecución de un programa. Una función de bloqueo no permite que su programa llame a otra operación hasta que la función termina su tarea. Ejemplo de esto puede ser una llamada a una función que hace una lectura de disco, en efecto esta llamada no permite o bloquea la ejecución de cualquier otra instrucción dentro del mismo módulo. En terminología de sockets, a esta acción se le conoce como *bloqueo.*

A la inversa, una operación de no bloqueo termina de inmediato. Ejemplo un screen saver. En un sistema multitarea las aplicaciones lo que hacen es compartir el CPU, sin embargo Windows y UNIX no comparten el procesador de la misma manera. Windows **3.1** espera y requiere cooperación de programas que comparten el CPU. Por contrario, UNIX interrumpe un programa para obtener control del CPU, lo mismo sucede con Windows 95 y Windows NT.

Los sistemas operativos de multitarea interrumpida y cooperativa manejan las opciones de bloqueo de manera diferente. Cada sistema operativo debe luchar con el problema del bloqueo. Dentro de Windows 95 o Windows NT, una operación de

bloqueo para todas las demás actividades para la tarea que ocasionó la operación de bloqueo. En un programa de Winsock, una llamada a una función de E/S en la red casi siempre inicia una operación de bloqueo.

Para crear un socket de no bloqueo, un programa primero crea y conecta un socket usando las funciones socket y connect. Después llama a la función Winsock que cambia el socket de bloqueo a no bloqueo. Seguidamente puede usar cualquier función de Berkeley o de Winsock sin iniciar una operación de bloqueo. Winsock no restringe el uso solo de funciones asíncronas para las operaciones de no bloqueo. De hecho podemos colocar un socket de Berkeley como de no bloqueo usando la función ioctl(), Winsock provee para esta tarea a la función ioctlsocket().

Muchas veces se emplea los términos no bloqueo y asíncrono como sinónimos, pero esta especificación es extremadamente engañosa.

3.4.4 Síncrono y Asíncrono

La E/S en UNIX es síncrona, lo que significa que las operaciones de E/S ocurren al mismo tiempo que las llamadas a funciones. Aunque las operaciones

asíncronas son muy parecidas a las de socket de no bloqueo, hay una diferencia importante. Cuando se ejecuta una operación en un socket de no bloqueo y esta no puede concluirse de inmediato, no hace nada excepto enviar un mensaje de error. Cuando se llama a una función asíncrona este hecho no causa un error. Una llamada a una función asíncrona solo es responsable de iniciar la operación, no espera a que esta concluya, a cambio el sistema operativo monitorea la conclusión de esta operación.

3.4.5 Cómo usar las funciones clave de Winsock específicas de Windows

La API Windows Sockets requiere dos funciones específicas de Windows, *WSAStartup* y *WSACleanup*, para todos los programas basados en Winsock, deben llamar a la función WSAStartup antes de que llame a cualquier otra de las funciones de Winsock. Por cada llamada a la función WSAStartup un programa debe incluir después una llamada correspondiente a la función WSACleanup.

3.4.5.1 WSAStartup

Permite que un programa especifique que versión de la API Windows Sockets necesita y recopila detalles sobre la implementación de Winsock. Cuando un

programa llama WSAStarup ocurre una negociación entre él y WINSOCK.DLL, en donde el programa especifica la versión mínima de Winsock que requiere y WINSOCK.DLL la versión más reciente que soporta.

3.4.5.2 WSACleanup

Es la función compañera de WSAStartup. Por cada llamada que un programa haga a WSAStartup también debe hacer una llamada correspondiente a WSACleanup. Para hacer el seguimiento del número de llamadas, Winsock mantiene un contador interno. Por cada llamada a WSAStartup se incrementa el contador y lo decrementa si llama a WSACleanup; cuando se realiza la llamada que establece el contador en cero, WINSOCK.DLL realiza operaciones de limpieza de registros y dice a Winsock que el programa ya no lo necesita para administrar ningún recurso. Cuando Winsock realiza operaciones de limpieza de registros, libera todos los búferes para datos utilizados en forma interna, así como cualquier otro recurso que utilizó el programa.

3.4.6 Descriptor de Socket

Winsock define un tipo de datos sin **signo** llamado SOCKET, que lo utiliza como identificador *o* descriptor de socket. Este es un número entero de baja

denominación. Winsock emplea la constante INVALID_SOCKET para identificar a los sockets inválidos. Un identificador de socket válido tiene valor del rango [0,INVALID_SOCKET-1], por tanto el Único entero que identifica a un socket inválido es 0xFFFF. Un valor negativo se interpreta como un tipo de información no asignada.

3.4.7 Como maneja Winsock los errores

Winsock define la constante SOCKET_ERROR (de valor -1) como el valor que identifica los errores del socket. Cuando sucede un error de socket, Winsock llama a la función WSAGetLastError para identificar la condición concreta que causó el error. En Windows 95 y Windows NT WSAGetLastError llama a la función GetLastError que nos devuelve el estado del error para todas las funciones de Windows de *32* bits tarea por tarea.

3.4.8 Como entender la función WSAASYNCSELECT

WSAAsyncSelect es una función clave para las operaciones de socket de no bloqueo en Winsock. WSAAsyncSelect es el fundamento para las clases de sockets

MFC (CasyncSocket y CSocket). La función WSAAsyncSelect es la única función asíncrona específica de Windows en la especificación de Winsock que usa un identificador de socket (descriptor) como un parámetro. Para monitorear los sockets WSAAsyncSelect debe efectuar una llamada por cada socket que desea monitorear ya que solo acepta un identificador de socket a la vez como parámetro.

WSAAsyncSelect tiene dos efectos: primero colocar a un socket en modo de no bloqueo y segundo preguntar que mensajes se ponen en una ventana cuando ciertos eventos ocurren. El prototipo de una llamada de este tipo es:

int Pascal WSAAsyncSelect(SOCKET s, HWND hWnd, u_int wMsg, long lEvent);

Esto se puede traducir de la siguiente manera: cuando cualquiera de los eventos descritos en *lEvent* ocurre en el socket *s*, muestra el mensaje *wMsg* en la ventana *hWnd*. Los posibles valores para *lEvent* y lo que representan es como sigue:

- FD_READ: datos están disponibles para lectura.
- FD_WRITE: el socket esta listo para escritura (se puede escribir en él).
- FD_OOB: datos necesitan ser leídos de una manera urgente.
- FD_ACCEPT: hay un requerimiento de conexión por parte de un cliente.

- ✓ FD_CONNECT: se ha completado satisfactoriamente un requerimiento de conexión.
- ✓ FD_CLOSE: aviso de terminación de socket.
- ✓ FD_ALL_EVENTS: significa que se activa para cualquiera de los eventos descritos.

En el presente proyecto monitoreamos los eventos que están con un visto. Cada llamada a WSAAsyncSelect sobrescribe la información de la llamada previa. Por tanto si deseamos que nuestro programa monitoree múltiples enventos, como frecuentemente sucede, se debe combinar los valores mostrados con una operación OR. Suponiendo que queremos que Windows nos envíe un mensaje cuando el socket este listo para enviar información o cuando lleguen los datos, el valor de *lEvent* sería (FD_READ | FD_WRITE), si colocamos cero (0) en *lEvent* se cancelan las notificaciones para el socket.

Cuando se requiere enviar una notificación en una ventana sobre un evento ocurrido, *wParam* contiene el número de socket, *lParam* contiene el código del evento y cualquier error que puede haber ocurrido. Si el código de error es cero, la operación se realizó satisfactoriamente.

No es necesario reinicializar WSAAsyncSelect; esta es automáticamente reactivada cuando se habilita una llamada a esta función. WSAAsyncSelect debe ser llamada únicamente cuando se necesita recibir notificación sobre el monitoreo de diferentes eventos, o cuando se desea cancelar todas las notificaciones.

3.4.8.1 Uso de WSAAsyncSelect() en el lado del Servidor

Esta es una secuencia típica de los eventos que se dan en una aplicación del Server.

1. Crea un socket y lo asigna o enlaza a una dirección.
2. Llama a WSAAsyncSelect() y realiza un requerimiento de notificación para el evento FD_ACCEPT.
3. Llama a listen() y se coloca la aplicación servidora en estado de escucha (acción pasiva).
4. Cuando llega un requerimiento de conexión, la ventana de notificación recibe el mensaje y la notificación FD_ACCEPT. Responde limado por accep() or WSAAcept() para completar la conexión.
5. Llama a WSAAsyncSelect() para realizar requerimientos de notificación de FD_READ | FD_OOB | FD_CLOSE del socket creado por accept().

Estas notificaciones pueden ser enviadas al cliente cuando envía datos o cuando cierra el socket que estamos usando.

6. Cuando recibimos la notificación de FD_READ o FD_OOB, se llama a ReadFile(), read(), recv(), recvfrom(), WSARecv(), o WSARecvFrom para recuperar los datos.
7. Responde a la notificación FD_CLOSE llamando a closesocket() con el socket retornado por accept().

Normalmente el cliente inicializa el proceso de cerrar la conexión ya que él conoce cuando ya no necesita los servicios que el servidor le ofrece.

Capítulo 4
Base de Datos

4.1 Acceso a Base de Datos con ODBC

La característica principal del WebServer, es la posibilidad de accesar a Base de Datos y mostrar información dinámica.

Como se menciona en el titulo de este documento y en muchos párrafos anteriores el WebServer que se implemento tiene acceso a Base de Datos, pero ***algo*** importante es: que este acceso es *"tanto local y externo al* ***mismo*** *tiempo a través* ***de*** *ODBC* ".

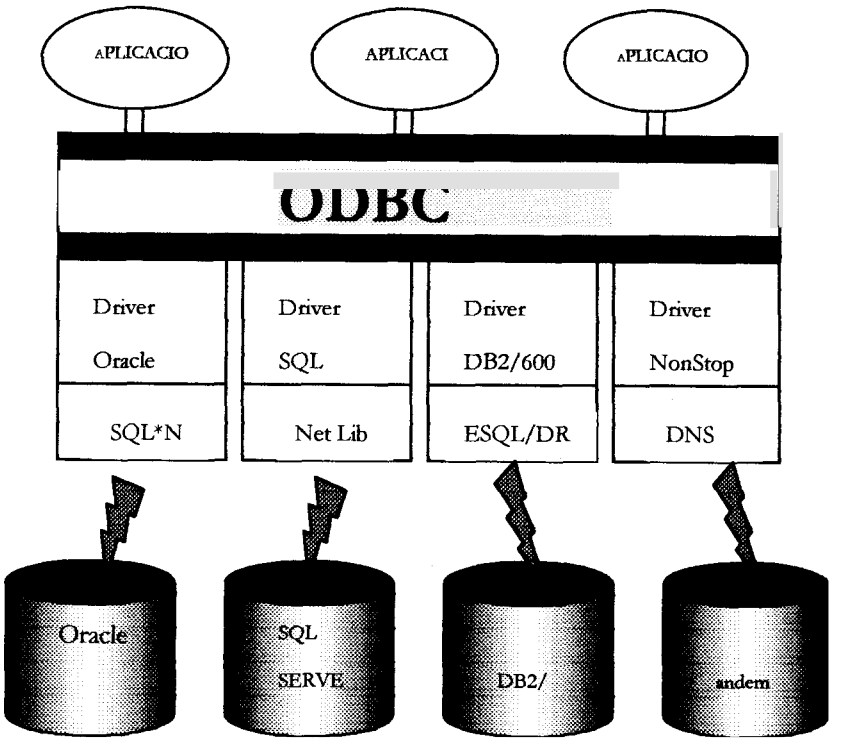


Figura 4.1

Lo único que se necesita para accesar a una Base de Datos por ODBC es que exista el controlador para ese DBMS. Todos estos controladores deben estar instalados en el equipo donde corre el WebServer, ya que es este el que accesará a las Bases de Datos clientes

Es importante indicar que ODBC es ahora un estándar y la mayoría de **los** fabricantes de DBMS también fabrican los controladores de ODBC para acceso a la Base de Datos fabricada por ellos.

El acceso local que hace el WebServer es a la Base de Datos con la que trabaja directamente, podríamos llamarla **"propia"** la **cual** está elaborada en *Access versión* 7.0 *para Windows 95 y* toma el nombre de ***"empresas"***. Esta Base de Datos guarda información acerca de las empresas, clientes, consultas que ofrecen, sentencias SQL para retornar estas consultas.

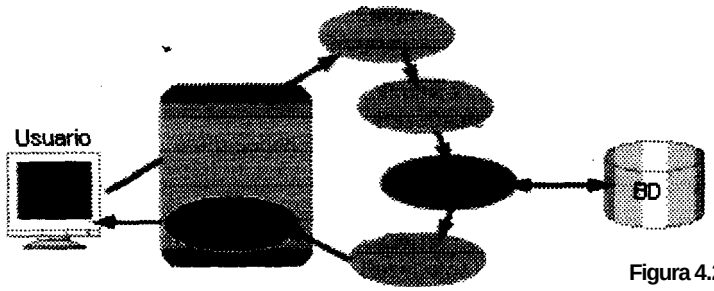


Figura 4.2

El acceso externo que hace el WebServer es justamente ejecutando las sentencias SQL obtenidas de la base de datos local (empresas), para este acceso se necesita tener registrado el nombre de la Base de Datos, user y password. La Base de Datos cliente debe estar en red con el equipo donde corre el Webserver.

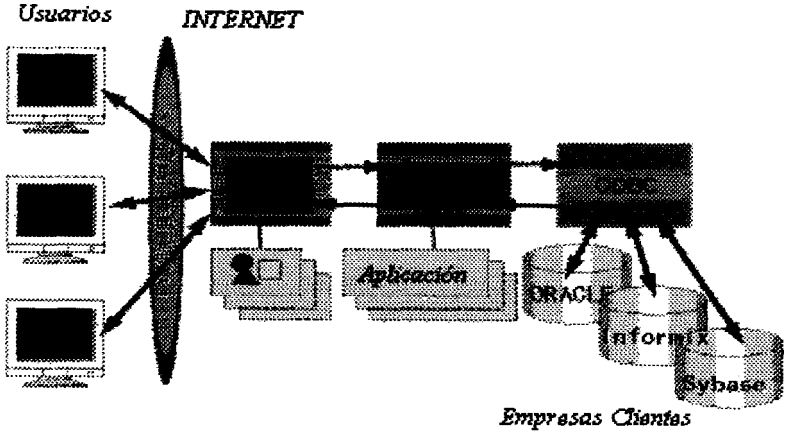


Figura 4.3

La forma en que se accede y las bases de datos se explica en la sección **4.1.2.**

4.1.1 La forma en que el WebServer accesa a la Base de Datos

Una vez que el usuario final, a través de la Internet, solicite un requerimiento de una consulta, el WebServer captura su requerimiento y envía un mensaje **al** Objeto *AccesoBase* con los parámetros necesarios, entre los cuales se encuentra el URL, para que pueda accesar a la Base de Datos propia del WebServer y ejecutar la consulta solicitada. Este URL es el identificador, el cual identifica a las Bases de Datos que se accesa, la consulta que se debe ejecutar, los nuevos URL que deben regesar en la consulta, todo esto es realizado mediante las funciones implementadas en código. Recuerde que todo este acceso lo hace por ODBC a dicha Base y extrae el resultado

del query que solicitó el usuario, a este resultado se le agrega una cabecera con los datos de que Base de datos se está consultando, luego se añade los nuevos URL, que se necesitarán en la siguiente página y que vienen como resultado de la consulta y actualizado por el URL anterior, los que se colocarán del lado izquierdo del archivo, de esta forma se genera dicho archvo con datos del Cliente accediendo a la Base propia del WebServer y resultado del query accediendo a la Base de Datos del Cliente, de esta forma el archivo es generado con un nombre aleatorio.

Luego este archivo es enviado a la función que lo convierte en un archivo HTML y una vez convertido a HTML es regresado al WebServer para que lo regrese al usuario.

El procedimiento para el acceso a las Bases de datos foráneas, es muy parecido, puesto que recoge el requerimiento del usuario (Visualizador cliente), y envía un mensaje al Objeto *AccesoBase* con los parámetros necesarios, entre los cuales se encuentra el URL, para que pueda accesar a las Bases de Datos propia del WebServer y ejecutar la consulta solicitada. Si ese URL indica un acceso a una Base de Datos foránea, entonces se procede a ejecutar primero el query a la base local (propia), para obtener el query, el usuario, la clave, el DSN, etc. , que necesita para

accesar a la Base de Datos cliente, y al obtener el resultado, continua los **mismos** pasos descritos anteriormente.

Para la demostración de las ventajas de ODBC, hemos creado bases de datos de empresas clientes en ORACLE, MSAcces, y SQLServer; y se han configurado sus respectivos accesos en ODBC. Para mayor información de lo que hace el WebServer se ha escrito un capítulo especial del füncionamiento de una aplicación llamada ICARO en el Capitulo 5.

4.1.2 Para instalar los controladores ODBC

Como está explícito en capítulos anteriores el software que se elaboró está realizado para plataforma Windows 95 y Windows NT, es decir, se necesita tener el software Administrador de ODBC para estas plataformas.

En esta sección sugerimos pasos para instalar los controladores bajo Windows NT Server:

- Si no instaló la opción Administración y controladores ODBC, ejecute el programa de instalación para Windows NT de nuevo haciendo clic en el icono

Instalar de Internet Information Server en el grupo de programas Microsoft Internet Server. Necesita el disco compacto de Windows NT Server *o* un directorio de **red** que contenga el contenido completo del disco compacto.

- Haga clic en el botón Aceptar.
- Haga clic en el botón Agregar/Eliminar.
- Haga clic en el botón Aceptar.
- Seleccione la opción Administración y controladores ODBC.
- Haga clic en el botón Aceptar.
- Aparecerá el cuadro de diálogo Instalar controladores.
- Para instalar el controlador deseado solo seleccione el controlador de la lista **de** Controladores ODBC disponibles y haga clic en el botón Aceptar.

El programa de instalación completará la copia de archivos.

4.1.3 Para crear los orígenes de datos del sistema

Una vez instalado los controladores de ODBC y los drivers necesario, se necesita crear los orígenes de los sistemas, es decir, donde está la Base de Datos

ubicada. Recordemos que la Base de Datos no importa dónde está ubicada, ya sea como local o en una red.

Los orígenes se crearán para los drive que estén instalados y se los agregará de esta forma:

Para accesar a una Base de Datos :

- Haga clic en el icono Panel de control del grupo de programas Principal del Administrador de programas.
- Haga doble clic en el icono ODBC.
- Aparecerá el cuadro de diálogo Orígenes de datos ODBC.
- Puede que en la lista haya otros orígenes de datos si hubiera instalado otros controladores ODBC anteriormente.
- Elija el botón DSN del sistema.
- Importante Asegúrese de hacer clic en dicho botón. El Conector de bases de datos de Internet sólo funciona con DSN del sistema.
- Aparecerá el cuadro de diálogo Orígenes de datos del sistema.
- Haga clic en el botón Agregar.

- Aparecerá el cuadro de diálogo Agregar origen de datos.
- Seleccione un controlador de ODBC en el cuadro de lista y haga clic en Aceptar. .
- Aparecerá un cuadro de diálogo específico para su controlador.
- Escriba el nombre del origen de datos.
- El nombre del origen de datos es un nombre lógico utilizado por ODBC para hacer referencia al controlador y a cualquier otras información necesaria para tener acceso a los datos, como por ejemplo el nombre real del servidor *o* la ubicación de la base de datos. En los archvos del Conector de bases de datos de Internet, el nombre del origen de datos se utiliza para informar a Internet Information Server acerca de dónde se puede tener acceso a los datos.
- Haga clic en el botón Aceptar.
- Aparecerá otra vez el cuadro de diálogo Orígenes de datos del sistema, pero ahora presentando el nombre del origen de datos.
- Haga clic en el botón Cerrar para cerrar el cuadro de diálogo Orígenes de datos del sistema.
- Haga clic en el botón Cerrar para cerrar el cuadro de diálogo Orígenes de datos.
- Haga clic en el botón Aceptar

4.2 Otros métodos de acceso a Base de Datos por Intemet

Existe variedad de métodos que para accesar a Base de Datos a través de Internet, algunos del lado del servidor, otros del lado del cliente. **A** continuación describimos algunos de los más populares.

4.2.1. Interfaz de puerta de enlace o gateway común (CGI).

Es un conjunto de especificaciones para transferir información entre el explorador de un cliente Web, un servidor Web y una aplicación CGI. El explorador de un cliente Web puede iniciar una aplicación CGI completando un formulario HTML *o* haciendo clic en un vínculo de una página HTML del servidor **Web.** La aplicación CGI puede aceptar información escrita por el usuario y tratarla de cualquier modo que se pueda programar, y después devolver los resultados en una página HTML *o* enviar información a una base de datos. Como las aplicaciones CGI sencillas a menudo están escritas con lenguajes de archivos de comandos como Perl, a las aplicaciones CGI también se las conoce como "archivos de comando".

La siguiente ilustración muestra cómo intercambian información un explorador, un servidor y una aplicación CGI utilizando CGI.

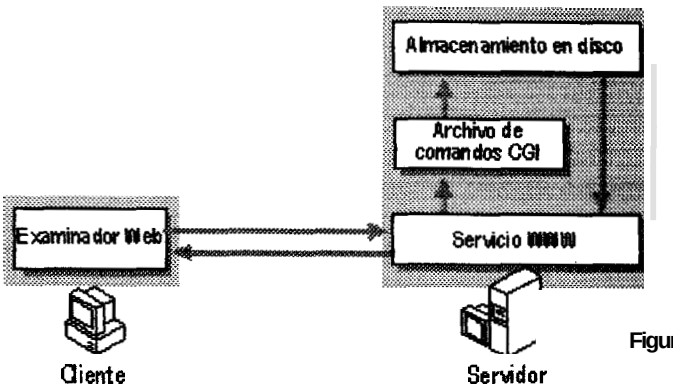


Figura 4.4

El proceso consta de cinco partes:

- El cliente envía una petición.
- El servidor recibe la petición.
- El servidor pasa la petición a la aplicación.
- La aplicación CGI devuelve los datos al servidor.
- El servidor devuelve los datos al cliente.

4.2.2. API de Internet Server

ISAPI para Windows NT se puede utilizar para escribir aplicaciones que los usuarios de Web pueden activar completando un formulario HTML o haciendo clic

en un vínculo de una página HTML de su sitio Web. La aplicación remota puede aceptar información introducida por el usuario y tratarla de cualquier modo que se pueda programar, y después devolver los resultados en una página HTML o enviar la información a una base de datos.

ISAPI se puede usar para crear aplicaciones que se ejecuten como DLL *en* su servidor Web. Encontrará que las aplicaciones ISAPI tienen un mejor rendimiento que los archivos de comandos CGI porque se cargan en memoria durante la ejecución del servidor. Requieren menos tiempo de espera porque cada petición no inicia un proceso distinto.

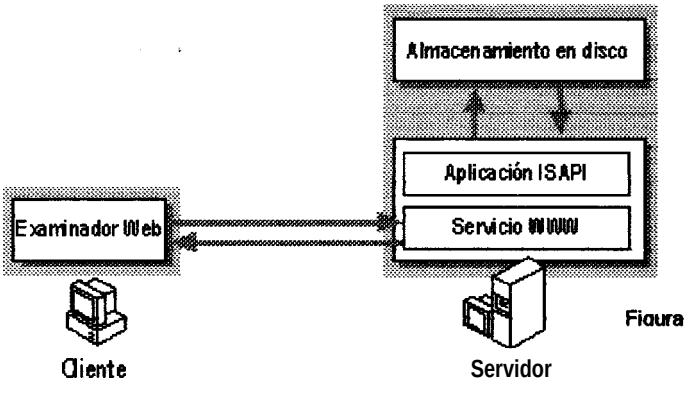


Figura 4.5

Otra característica de ISAPI permite el preprocesamiento de peticiones y el postprocesamiento de respuestas, permitiendo la administración de peticiones y respuestas con el Protocolo de transferencia de Hipertexto (HTTP) que sean

específicas del sistema. Puede usar filtros ISAPI en aplicaciones como autentificaciones personalizadas, accesos o registros.

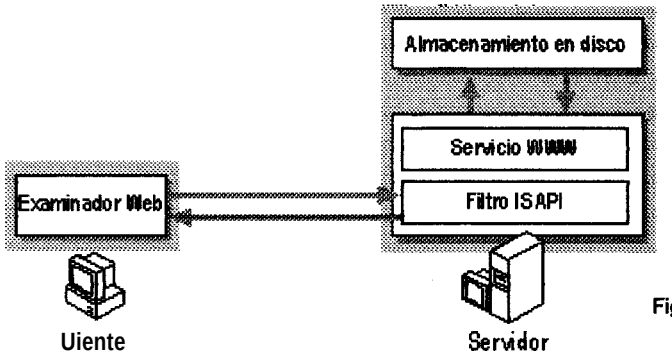


Figura 4.6

Pueden crear sistemas muy complejos usando filtros y aplicaciones ISAPI. También puede combinar extensiones ISAPI con el Conector de bases de datos de Internet para crear sitios altamente interactivos.

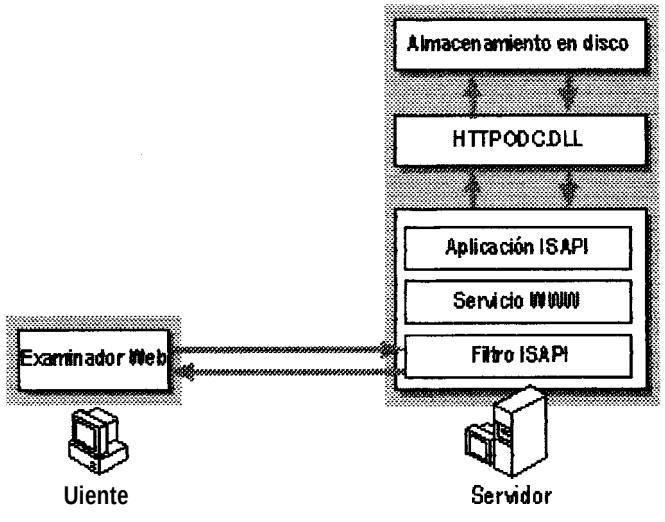


Figura 4.7

4.2.2. JAVA (JDBC)

El JDBC **API** define clases Java para representar conexiones a Base de Datos, sentencias SQL, conjunto de resultados, etc. Esto permite al programador ejecutar sentencias SQL y procesar los resultados. JDBC es la primera API para acceso a Base de Datos en Java.

El JDBC **API** es implementado vía manejador de drive que **soporta** múltiples drive de conexiones a diferentes base de datos. Los drive JDBC pueden ser enteramente escritos en Java que pueden ser bajados como parte del applet, o pueden ser implementados usando métodos para puentear con librerías de acceso de las bases de datos existentes.

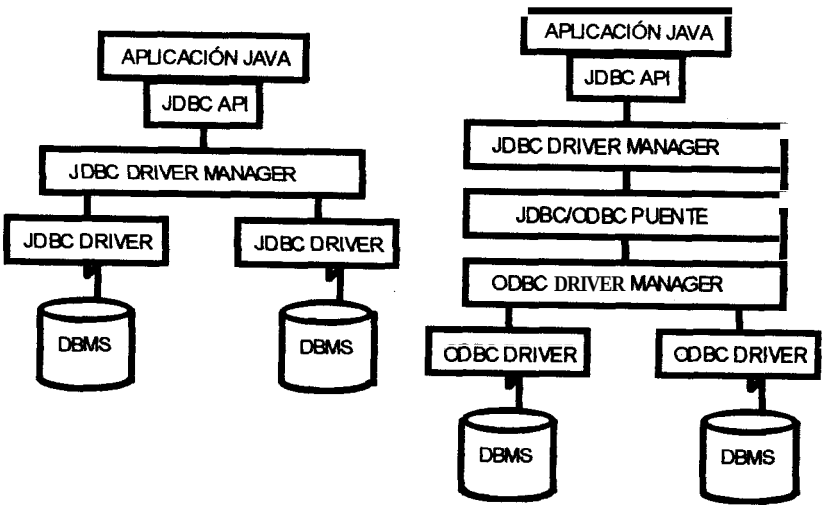

APLICACIÓN JAVA
JDBC API
JDBC DRIVER MANAGER
JDBC DRIVER
JDBC DRIVER
DBMS
DBMS
APLICACIÓN JAVA
JDBC API
JDBC DRIVER MANAGER
JDBC/ODBC PUENTE
ODBC DRIVER MANAGER
ODBC DRIVER
ODBC DRIVER
DBMS
DBMS

Figura 4.8

Capítulo 5
Aplicación Centro de Servicios ICARO

5.1 Generalidades

5.1.1 introducción

El proyecto analizado y diseñado anteriormente "WebServer orientado a base de datos", puede formar parte de Internet. Para mostrar su aplicación en el mercado crearemos una empresa denominada ***"Centro de Servicios ICARO",*** la cual tendrá como corazón al Web Server denominado ICARO, podrá estar disponible en la Web y su objetivo será mostrar información actualizada ya que su información es consecuencia de accesos a Bases de Datos.

El presente capítulo tiene como propósito estimar las ventajas técnicas y económicas que se deriven de poner en marcha el proyecto de Tópico de Graduación, para que asignando ciertos recursos de un grupo de jóvenes entusiastas, podamos dar prestación de servicios.

Consideramos primordial e importante mostrar la aplicación de nuestro proyecto en el mundo real ya que nos da riqueza profesional para poder defender nuestras ideas y al mismo tiempo aporta a nuestra experiencia en el verdadero campo de guerra.

El proyecto no podrá contener los detalles relativos a todos los elementos que inciden en él, ni prever todas las dificultades que habrá que resolver en el terreno mismo, en cuanto a organización, puesta en marcha y funcionamiento. Pero el proyecto representa la base racional de la decisión de montar la empresa, y elio explica la necesidad de que esté lo mejor estudiado posible.

Esta empresa representa para las empresas privadas y estatales el camino de mejorar el servicio a sus usuarios, informando y realizando sus transacciones mediante Internet.

En este documento se dará las generalidades del proyecto, el Estudio de Mercado, el Estudio Técnico, Estudio Organizacional, y por Último, el Estudio Financiero.

Con toda la información con que contemos al realizar todo el análisis del proyecto, esperamos tener una visión más amplia y los fundamentos necesarios para decidir si es factible o no incursionar en este mercado.

5.1.1.1 Antecedentes

Ya que el boom del momento es el WEB, consideramos apropiado escoger como proyecto de Tópico de Graduación el "WEB Server con acceso a Base de Datos".

Acogidos a este requerimiento, ahora lo importante es definir que servicio dará el WebServer y que información será recuperada de las bases de datos.

El Web Server debe detectar los requerimientos del usuario y ese requerimiento ser interpretado, se le retorna al usuario el archivo HTML que contiene la información que el usuario solicita. En el mercado existen compañías que brindan servicio del Web, otorgando cuentas para Internet y elaborando WebSites (páginas HTML). Pero estas páginas son estáticas ya que deben actualizarlas cada cierto periodo de tiempo.

Analizando estos requerimientos la pregunta era a que tipo de negocio o institución se enfocara el servicio de ICARO, ya que de esto dependía el análisis y diseño del proyecto.

La meta es generar información dinámica, y para esto **analizamos** el **flujo** de información de algunas empresas.

Conocemos que algunas empresas tienen la misma estructura de información ya que se dedican a la misma actividad.

Por ejemplo todas las universidades registran a sus estudiantes, necesitan hacer el cobro de matrículas, permitir que los estudiantes puedan ver sus deudas, puedan escoger sus horarios, etc. y todo esto necesita un acceso a sus bases de datos, sea solo para consultar o para actualizar la información. Todas estas transacciones se pueden realizar en el Web.

De **igual** forma las empresas que ofrecen los servicios básicos de luz, teléfono, agua, etc. podrían ofrecer a sus usuarios consultas de deudas. Estas empresas son estatales y la mayoría de ellas no tienen una buena estructura en el área informática por falta de equipos, personal y porqur difícilmente invierten en tecnología.

¿Por **qué** *Consultas a las bases de datos de las empresas estatales?*

Todos conocemos que las empresas estatales ofrecen un mal servicio al usuario, algunos usuarios que van a cancelar sus planillas no lo pueden hacer, ya que no conocen sus códigos para poder consultar sus deudas, los recaudadores de las empresas estatales no tienen paciencia para atenderlos y exigen las planillas de aviso. Esta actitud ha generado total desagrado y desconfianza en los usuarios, **mirando** con total pesimismo el futuro de tales empresas.

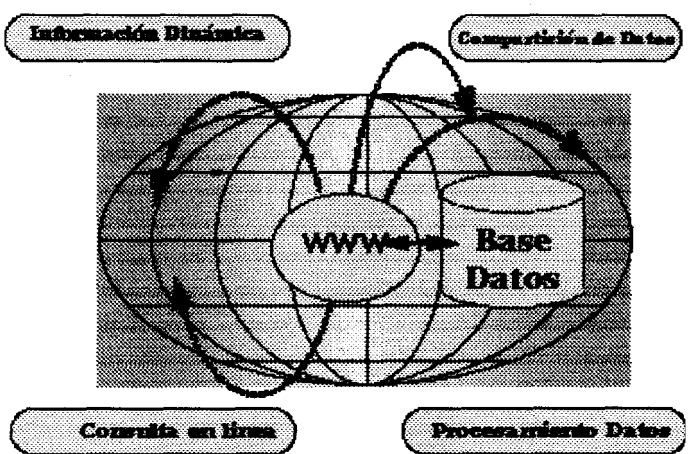


Figura 5.1

Hay usuarios que por razones de trabajo o familiar, les agradaría poder tener un contacto con la empresa desde su casa; lo que actualmente hacen es **llamar** a la empresa para que se les consulte sus deudas *o* la fecha máxima de pago, pero no son atendidos ya que no existe personal asignado a este servicio; entonces existe una gran

demanda de que se mejore el servicio al cliente, y que este servicio sea eficiente y oportuno.

Actualmente existen ciertos paliativos a estos problemas, tales como el envío de las planillas de aviso donde se les indica las deudas y la fecha máxima de **pago** de estas, pero por razones desconocidas algunos usuarios informan que nunca les llegó, u otros simplemente aceptan que la perdieron; también han implementado la consulta de deudas por medio del teléfono, donde el usuario marca un número de teléfono de consulta, un contestador le pregunta el código y le da la información de sus deudas, pero en este sistema la interacción hombre-máquina es muy rígida.

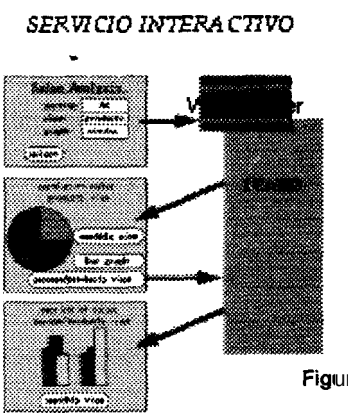


Figura 5.2

Lo que ofrece ICARO es un servicio interactivo, flexible, con interfaces gráficas, sonido y animación, con opción a consultar en varias compañías por un mismo medio y sacar un impreso de las consultas realizadas. Todo esto a través del WEB ya que es un medio hipermedio donde cada día aumenta el número de navegadores.

5.1.1.2 Aspectos legales

ICARO será una asociación voluntaria de personas que crearán un fondo patrimonial común para colaborar en la explotación de la empresa, con el ánimo de obtener un beneficio individual, participando en el reparto de las ganancias o pérdidas que se obtengan. Será constituida como una Sociedad Anónima, ya que estará formada por las aportaciones de los socios, divididos en acciones, y funcionará bajo el principio de falta de responsabilidad de los socios por las deudas sociales. Es decir, independientemente de las deudas que la sociedad pueda contraer, los socios sólo responderán con la aportación de capital que habrán hecho a la Sociedad.

5.1.1.3 Tipo de negocio

ICARO se dedicará a prestación de servicio **para** Internet, ser un medio de comunicación entre usuarios y empresas; entre ellos, diseño de sus **páginas,** promoción vía a la Red, generación de páginas dinámicas, reportar el número de usuarios que utilizan el servicio.

Contaremos con un Servidor donde estará instalado el Web Server ICARO (servidor Web) el cual estará enlazado a la red de Internet, y será uno de los servidores que proporcione páginas HTML dinámicas. Tendremos acceso a las bases de datos de nuestros clientes y es esta información la que será mostrada a los usuarios.

Nuestros clientes podrán estar enterados de las irregularidades a las que hallan sido sometidos sus usuarios.

5.1.2 ¿Quiénes Somos?

5.1.2.1 Misión:

- Mostrarles a nuestros clientes el estudio técnico con el que podemos respaldarnos para afirmar que la tecnología que ofrecemos es una puerta al

cambio en especial a las empresas estatales, las cuales están en una etapa de modernización. Sin olvidar que para que la tecnología sea realmente una herramienta productiva debe ser bien utilizada, y para esto se necesita un buen estudio técnico.

- Somos una empresa privada, enfocada a ser los líderes en la prestación de servicios como un WebServer que genera páginas Web dinámicas cuya información es obtenida de las bases de datos de nuestros clientes, brindándole al país la oportunidad de encaminarse a la nueva era de la información; coadyuvando así a la integración con el mundo actual. *Y* es así que nuestro objetivo es que cuando la gente piense en Internet piense en nosotros.
- Buscamos satisfacer todas las necesidades de todas las personas, empresas e instituciones que deseen integrarse a esta nueva tendencia de comunicación, información y comercialización con todo el mundo, teniendo siempre presente el brindar un servicio de primera acorde con el avance tecnológico.
- Es por eso que damos una especial atención a la capacitación permanente de los miembros de la compañía, buscando su bienestar en todo sentido, incentivar la cooperación, el compañerismo y la unión, para lograr nuestro objetivo.

- Nuestra empresa puede abarcar muchas áreas de información lo único que necesitamos es que la empresa se registre y nos permita accesar a su base de datos y mostrar la información que él desee mostrar o actualizar.

5.1.2.2 Objetivo:

- Mostrar las ventajas técnicas y económicas de un Web Server con acceso a base de datos enfocado a brindar servicio a empresas privadas o estatales, que necesiten de la tecnología actual como una herramienta para mejorar el servicio al usuario y ganar esa confianza, además mostrar a nuestros clientes las razones técnicas por lo que el proyecto es bueno, en esta era de la información.
- Estamos empeñados en das un servicio de primera, eficiente, económico, y con asesoría permanente.
- Abarcar todo el mercado, ya que somos los únicos que prestaríamos este servicio en Ecuador y el Tercero en el Mundo.
- Se busca ofrecer un servicio totalmente flexible, en donde se pueden agrupar empresas de un mismo tipo de actividad económica y que tengan la total libertad de decidir que tipos de consultas y procesamientos van a permitirle que los

usuarios de la red realicen, además que tenga total independencia del ambiente operativo de cada empresa así como el DBMS que utilicen.

- Todos los diseños de las páginas Web se realizarán siempre pensando en el usuario ya que ellos serán el parámetro que evaluará nuestro servicio, **por** esto utilizar como herramienta de diseño de interfaces a interacción hombre-máquina.

5.1.2.3 Fortalezas:

- Egresamos de una Universidad donde su característica es la excelencia, nuestro escuela el esfuerzo y nuestra herramienta la tecnología.
- Estamos capacitados para dar este tipo de servicio.
- Contamos con una parte del equipo computacional para el desarrollo de las aplicaciones de los usuarios (páginas WEB, etc.).
- Poseemos experiencia en el diseño e implementación de páginas WEB.
- Somos un grupo de estudiantes entusiastas y ambiciosos que estamos empeñados en llevar adelante este proyecto y alcanzar este gran objetivo
- Contamos con toda la asesoría necesaria para la implementación del proyecto.
- Es ideal si el usuario pudiera pagar mediante Internet pero esta transacción debe ser bien estudiada con las empresas estatales para todos los controles y

seguridades es por esto, que no actualizamos en ningún momento sus bases de datos. Pero si lo podemos hacer.

5.1.2.4 Debilidades:

- No somos conocidos en el mercado.
- No contamos con el dinero necesario para la compra de los servidores y otros equipos y tampoco contamos con el local.
- Primera vez que buscamos dar servicio al cliente.

5.1.3 Servicio

Buscamos brindar servicio de información, es decir permitir a los usuarios de nuestros clientes poder consultar deudas, datos, códigos, etc. y que al **mismo tiempo** puedan notificar de posibles problemas a través de la Red Global Mundial (WWW); participar en grupos de noticias, la creación de los Home-Pages de nuestros clientes, difundir revistas a todos los usuarios.

Nuestros clientes contarán con un verdadero y efectivo **grupo** de profesionales que le brindarán servicio permanente las **24** horas, tendrán acceso todo el día, y contarán con un período de prueba y una cancelación sin problemas.

Ahora la tendencia es que en la mayoría de los hogares exista una computadora, entonces una parte de los clientes de las empresas estatales tienen el equipo para poder ingresar al servicio del WEB.

Nuestra empresa considera necesario brindar servicio de consulta de deudas de los servicios básicos como luz, teléfono, **pago** de impuestos, agua potable, y que el usuario pueda notificar de posibles problemas *o* reclamos a la empresa y seremos nosotros el medio que permita que ambos interactuen.

Además toda empresa que se registre en nuestro servicio debe tener un responsable en el área informática que estará registrado en nuestra base de datos y quien será responsable de que la información de la empresa sea correcta, exacta y oportuna.

5.1.4 Clientes

5.1.4.1 ¿Quiénes son?

Las empresas Públicas *o* Privadas, las Universidades y Negocios, que deseen tener un medio donde puedan mostrar su información en Internet, y en especial aquellos que deseen mostrar información de sus bases de datos.

5.1.4.2 ¿Dónde están?

Se encuentran distribuidas a lo largo del país.

5.1.4.3 ¿Qué satisfacciones busca el cliente?

El cliente privado busca tener un medio por el cual pueda informar **a** sus usuarios de sus productos, servicio que ofrecen, sucursales y ser parte de la red de información más completa, actualizada y de temas muy diversos en cuestión de segundos; desea que sus productos sean conocidos por todos; mantener su Status, no quedar rezagados de lo nuevo y actual; y por último, contar con medio ***eficaz*** y efectivo de hacer negocios.

Las empresas estatales buscan poder informar a sus usuarios de las deudas, cambios de precios, subsidios, etc. y poder conocer de forma inmediata alguna anomalía técnica *o* las irregularidades a las que son sometidos sus usuarios.

En definitiva nuestros clientes buscan ver y ser visto.

5.1.4.4 ¿Cuál es el Poder del cliente?

El cliente tiene el poder de elegir si desea *o* no nuestro **servicio** y **siempre será** dueño de los datos que mostremos mediante el Web Server ya que la información que mostramos es absolutamente obtenida de sus bases de datos, **y** solo **será** modificada después de hacer un buen estudio con sus representantes y que legalmente sea aceptada la transacción de actualización.

5.1.5 Competencia

A nivel nacional no existe ninguna competencia en el mercado, ya que las empresas que prestan servicio de elaborar páginas Web a empresas lo hacen de forma estática y no tienen acceso a las bases de datos de sus clientes.

Nuestra competencia a nivel mundial son las siguientes compañías: WebBase (Japón) y DynaBase (Alemania), pero el alcance de nuestro proyecto al **inicio *es*** a nivel nacional por lo cual no tenemos competencia alguna.

WebBASE es una empresa que ofrece un programa para construir y proveer un servicio de búsqueda en base de datos sobre el Web. Soporta información comercial y científica en base de datos. Hace consulta a las bases de datos de sus clientes y además permite que los usuarios de Internet puedan crear sus propias tablas.

DynaBASE es una empresa que proporciona páginas Web dinámicas con acceso a base de datos.

5.1.5.1 Fortalezas:

- Conocidas en el mercado Internacional.
- Cuentan con experiencia en dar el servicio al cliente.
- Ofrecen un servicio único a nivel mundial ya que solo existen dos empresas.
- WebBase ofrece productos de software a clientes y usuarios.

- WebBASE tiene registrados un gran número de empresas que cuentan con ese servicio.
- WebBase vende el software que es instalado en cada empresa **que desea** poner **su** información en el Web.

5.1.5.2 Debilidades:

- En nuestro país no es conocido.
- La distancia, ya que la compañía esta ubicada en Japón.
- Se dan a conocer solamente en un cierto sector del mercado
- El cliente los busca por el servicio.

5.1.6 Tendencias

5.1.6.1 Políticas:

Existe un grado de descontento debido a la no regulación de transacciones comerciales vía Internet en el país.

Las medidas que tome el gobierno en turno para cubrir el déficit de líneas telefónicas en el país, por lo que es el medio de comunicación que es usado para el acceso a Internet.

Las trabas que ponga el gobierno para el desempeño de esta actividad, tales como: impuestos, garantías, etc.

5.1.6.2 Tecnológico:

La investigación permanente ha hecho que se descubran y creen nuevos medios de transmisión cada vez más veloces, capaces de enviar por un mismo medio voz, datos y video y con una gran confiabilidad, lo cual nos hace prever que el futuro es muy prometedor para el área de comunicaciones.

El costo del hardware cada vez disminuye más, brindándonos la oportunidad de adquirir más y mejores equipos a menor precio, haciendo accesible principalmente a que todas las personas puedan adquirir un computador, de esta manera se convierten en un futuro usuario de nuestro servicio.

Mejoramiento continuo de las seguridades en la red, nuevos y mejores métodos para que la información que circula por la red y que es de seguridad extrema no sea descifrada e interceptada por los "piratas' de la red.

5.1.6.3 Social:

En el mundo actual donde la tendencia es: "El dueño de la información es el dueño del poder", se hace necesario contar con un medio rápido, efectivo y eficiente; por el cual se debe **estar** al tanto de diversos temas de diferente índole.

El conocimiento cada vez mayor de este medio de comunicación, información, comercialización y entrenamiento; está creando una explosión de intereses por tener acceso a esta Red de Redes que es Internet.

El gran número de usuarios alrededor del mundo que accesan a esta red está haciendo volcar los ojos de todos para realizar negocios de todo tipo.

El incremento cada vez mayor de los centros educativos que imparten conocimientos de computación en todas sus especialidades, hace que cada vez un

mayor número de personas conozcan de este servicio, y por lo tanto sigue incrementando el mercado.

5.2. Estudio Técnico

5.2.1 Introducción

El proyecto a estudiar es un "WEB Server con acceso a base de datos". Porque el Web es la tecnología actual en información ya que es un medio de investigación, comercialización, intercambio de ideas, etc. Es decir afecta las entidades que son productivas para el país, sean estas intelectuales o económicas.

En el presente documento se describe el problema y se detalla el estudio técnico de la solución, recomendándole a usted estimado lector que no pase **por** alto la lectura del punto 1.1 y 1.2 ya que allí esta la base del porque consideramos ideal este proyecto.

5.2.2 Descripción del problema

El Web Server siempre debe estar sensando si un usuario requiere servicios. Una vez que el usuario se conecta a la gran red de información de Internet, y en su navegación podrá hacer un enlace a nuestro Web Server, y este debe responder de acuerdo a 1requerimiento.

El Web Server le debe enviar el homepage y a través de sus enlaces el usuario podrá consultar sus datos *o* deudas en la empresa que desee.

Las empresas clientes de ICARO deben estar registradas en una base de datos y cada que se agregue una empresa cliente, automáticamente se actualizará en la lista de empresas clientes mostradas en el Web, pero para esto necesitamos que de alguna forma la página HTML muestre automáticamente los nuevos cambios sin necesidad de cambiar el código HTML.

Estas empresas deben ofrecer a sus usuarios algún tipo de consulta las cuales también deben ser almacenadas en base de datos, estas consultas se ejecutaran directamente en los servidores de las empresas clientes. Las cuales retornaran un resultado que también debe ser generado en HTML dinámicamente.

Es decir debemos tener una base de datos donde registremos a las empresas clientes con sus datos y consultas que ofrecen. Además necesitamos accesar a **sus** bases de datos para poder obtener información actualizada de sus usuarios.

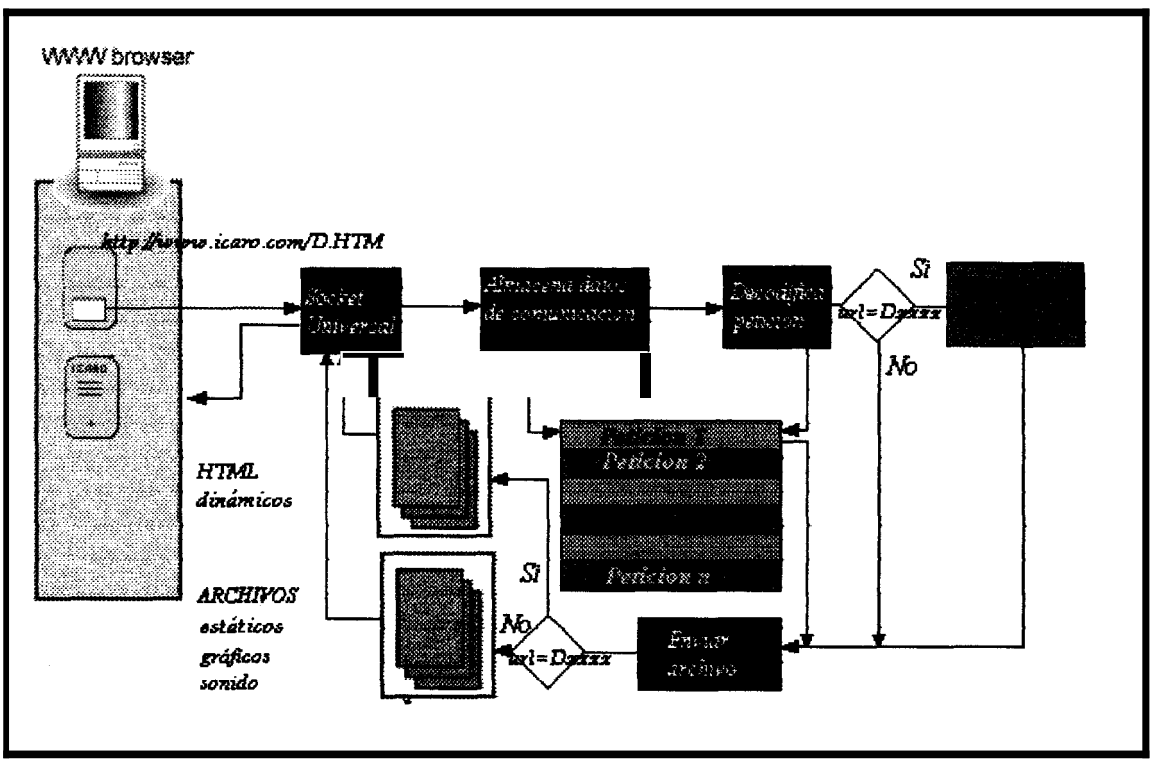
Para todo esto se necesita generar un HTML dinámicamente después de hacer las consultas a las bases de datos vía ODBC.

5.2.3 Descripción del proyecto

A continuación se describe el servicio que dará el WebServer y que información será recuperada de las bases de datos.

- ICARO tiene su propia base de datos, donde tiene toda la información de la empresa clientes tal como nombre, tipo de servicio, tipo de consultas, etc. **Y** lo más importante el nombre de la base de datos donde podemos accesar y una contraseña de seguridad. (Ver ANEXO #1, Diseño de la base de datos de ICARO).
- Para poder proporcionar reportes y estadísticas a nuestros clientes también se registra en la base de datos de ICARO, el IP del usuario, la hora de entrada y salida, fecha, etc.

Lo que sucede al levantar ICARO es:



- Existe un archivo ICARO.INI donde están seteados los parámetros iniciales con el que se levantara el Centro de Servicios ICARO. En este archivo se puede configurar los siguientes parámetros: la actualización de la **pantalla,** esto es la mensajería que despliega el WebServer para que el administrador del servicio pueda observar si esta funcionando normalmente; si se desea que el WebServer este activo *o* no, en caso de estar desactivado, ICARO enviara un

mensaje de servicio desactivado; el directorio donde hará referencia y el número de peticiones que podrá atender el WebServer al mismo tiempo.

- Este número de usuarios esta limitado dentro del código a **20** usuarios máximo es decir, solo aceptará hasta 20 usuarios conectados a la vez, si en el archivo ICARO.INI es configurado un número mayor que el permitido el WebServer actualizara este valor al máximo permitido, dando a conocer al administrador de tal cambio.

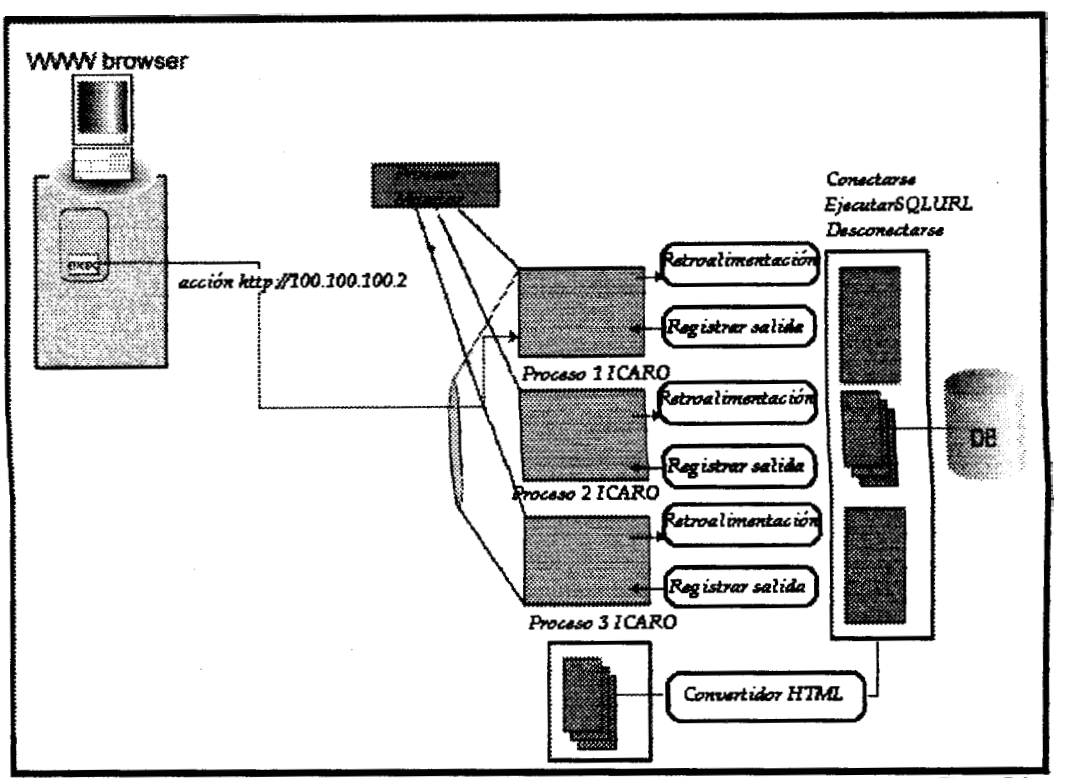


Figura 5.3

- Cuando el usuario escribe en su browser el URL: http://www.icaro.com o la dirección del servidor, en este caso: http://100.100.100.2 , la aplicación WebServer es alertado por medio de la mensajería que implementa Windows, lo cual es reflejado dentro la función miembro principal, con lo cual el WebServer detecta la validez del URL enviado y dependiendo si la petición necesita información de la Base de Datos Local o no (plantillas, gráficos,

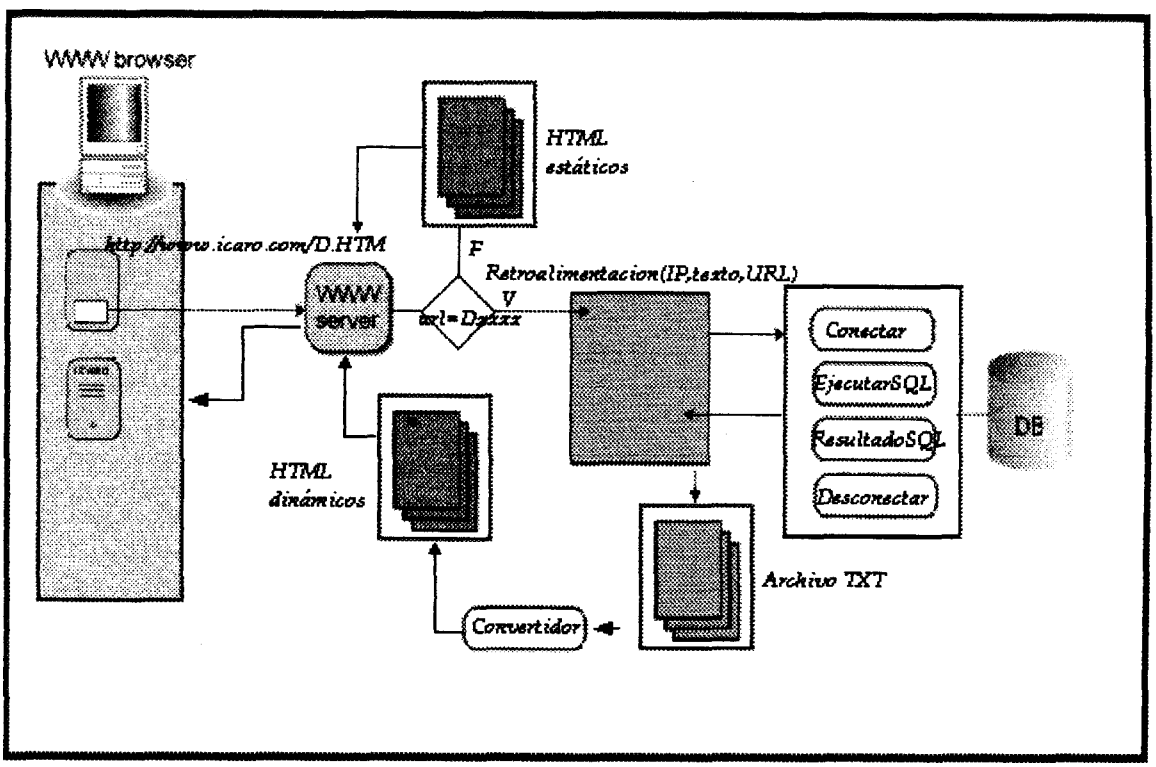


Figura 5.4

sonido, home, etc.), llama a la función retroalimentación de Base de Datos la cual devuelve un *.htm, con lo cual estamos listos para enviarlo **al** usuario que pidió la petición.

- Se crea un archivo log donde se registran todas las acciones que desarrolla el Web Server en el momento que detecta una conexión, esto es dirección IP del cliente solicitante, la fecha y la hora de la petición, y el URL solicitado.
- El WebServer obtiene el URL, y la dirección IP, la primera vez que el usuario se conecta el WebServer retornara el homepage de nuestra empresa (ICARO).
- Pero si tiene antepuesto el caracter D significa que se va a realizar una consulta a la base de datos.
- En el homepage hay dos enlaces ‘‘Inicio ” y “Acerca de ICARO”, si el usuario escoge el enlace “Inicio” se le dará un bienvenida y se encontrara con 2 enlaces:
- ‘‘Nuestros clientes” y “Acerca de ICARO”.

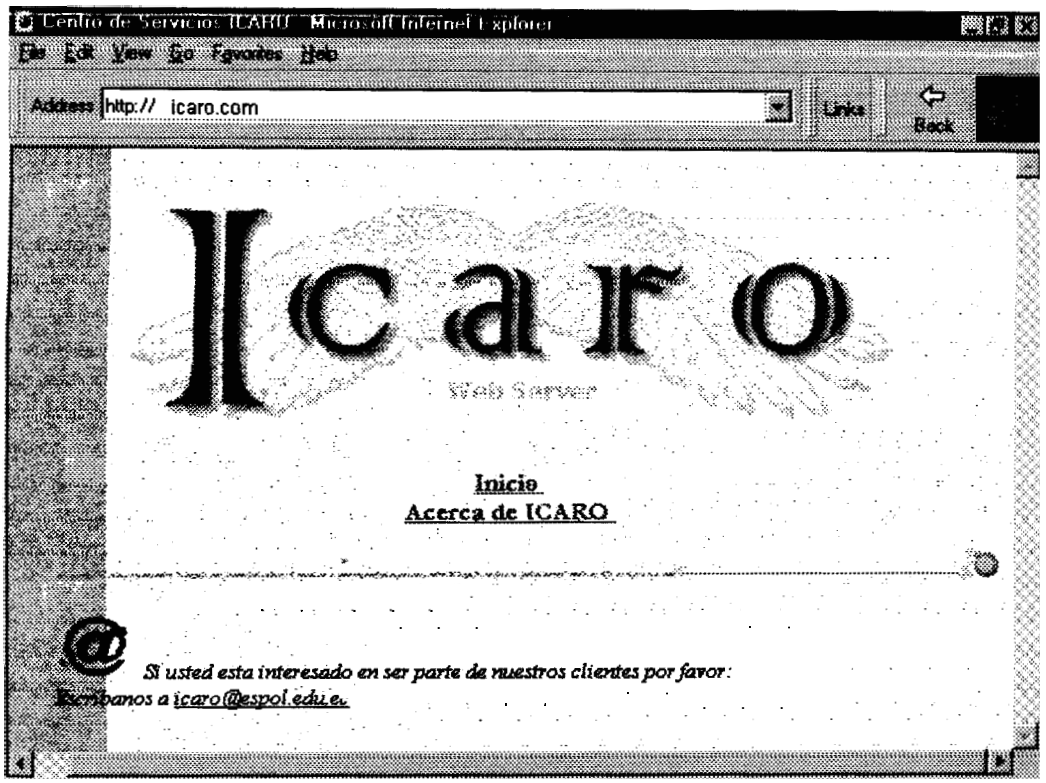

Centro de Servicios ICARO - Microsoft Internet Explorer
File Edit View Go Favorites Help
Address http:// icaro.com
Links
Back
Icaro
Web Server
Inicio
Acerca de ICARO
Si usted esta interesado en ser parte de nuestros clientes por favor:
Escribanos a icaro@espol.edu.ec

Figura 5.5

Si es escoge el enlace "Nuestros clientes'' será registrado en nuestra base de datos, ya que es la primera llamada a una consulta a la base de datos por lo tanto ***se*** genera un HTML dinámico, es decir la prirnera llamada a la función retroalimentación registrará el IP de la máquina que se conecta, hora y fecha de ingreso del sistema. Esta función recibe tres parámetros (URL, dato, usuario, IP) y retorna un **archivo.**

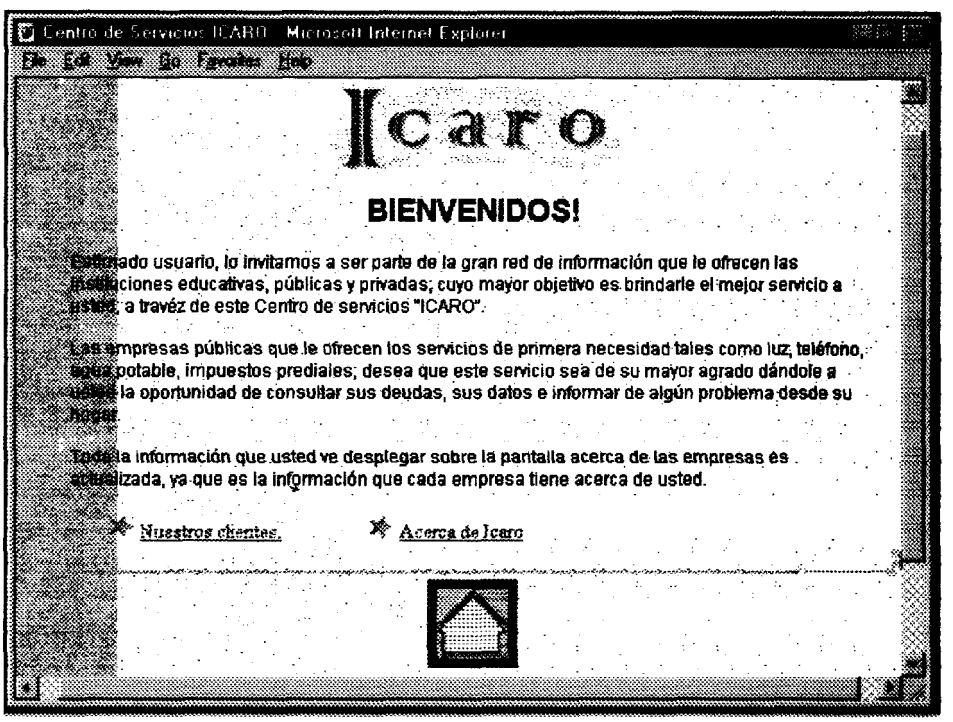


Figura **5.6**

Se preguntarán que significa?

Si el usuario escoge el enlace "Acerca de" el Web Server retornara una pagina HTML estática. Ya que algunas páginas como el homepage, acerca de, la bienvenida y

la lista de desarrolladores son páginas estáticas pre diseñadas. Pero si el usuario escoge el enlace "nuestros clientes", significa que requiere conocer que empresas están registradas en nuestro servicio por lo que es necesaria la consulta a la base de datos de ICARO.

El Web Server obtiene el URL, y la dirección IP y con estos datos ilama a la función retroalimentación la cual es *una función pública de nuestra clase "CAccesoBaseDatos"* y la que permite el envío de mensajes entre el Web Server y la base de datos y la que controla como se genera el nuevo archivo HTML. Es de esta forma que retroalimentación es considerada la función más importante ya que interpreta el URL y lo traduce y de acuerdo a esto genera un archivo con extensión txt.

La estructura del URL es:

- XXXXXX .html
- El primer carácter puede tener dos valores E (HTML estático) o D(HTML dinámico).
- Los siguientes dos caracteres tienen el código de la empresa en la cual se quiere hacer la consulta.

- Los dos siguientes caracteres tienen el código de la consulta, pero este código puede ser "EM" el cual nos ayuda a reconocer que lo que se solicita es la **página** de mail para que el usuario pueda reportar algún problema.
- El último carácter puede tener dos valores S *o* N, será **S** cuando la página web tenga un dato el cual es ingresado por el usuario a través de una caja de texto y N cuando la página no pida datos al usuario. Esto es para identificar que en el archivo HTML se debe generar una caja de texto.

El archivo TXT, contiene la información para la generación del archivo HTML. Contendrá el nombre de la plantilla que se llenará con el resultado de la consulta a la base de datos.

La estructura del archivo TXT *es:*

- prirnera línea: nombre de la plantilla
- segunda línea: nombre empresa
- tercera línea: nombre del logotipo
- cuarta línea: nombre de la consulta
- quinta línea: mensaje
- a partir sexta línea: filas retornadas de la consulta con el siguiente formato:

URL\tX\tfilas retornadas (separadas por \t)\n

X es el Status de la empresa si es un cliente viejo *o* nuevo (O *o* N) esto es para que en el HTML se ponga un indicador de new.

El archivo TXT es enviado a la función convertidor la cual debe **interpretar** cada línea y ponerle el formato indicado.

EJEMPLO:

La función retroalimentación recibe URL, dato ingresado por el **usuario** e IP de la máquina. El enlace "Nuestros Clientes" tiene por default la llamada **al** archivo D.HTM . Retroalimentación interpreta el URL y consulta nuestra base de datos **para** saber en que plantilla debe ser retornado el resultado del query. Como el URL que llega es "D" sabe que es el primer HTML dinámico a generar y que debe hacer **una** consulta a la base de datos de nuestra empresa y seleccionar las empresas que tenemos registradas y en que plantilla sera generado el resultado (específicamente de la tabla empresa obtiene nuestros clientes y de la tabla pantalla–plantilla obtiene la plantilla de acuerdo al URL). Retornará un archivo el cual contendrá en la primera línea el nombre de la plantilla ("inicial.htm"), de la segunda a la quinta línea serán

vacías ya que va a generar todas las empresas y a partir de la sexta línea contendrá el resultado del SQL

Inicial.htm

D01\tO\tEMPRESA ELECTRICA LOS RIOS\t\n
D02\t\O\tEMPRESA TELEFONICA\t\n
D03\t\O\tMUNICIPIO DE BABAHOYO\t\n

- El convertidor debe traducir en la primera línea: D01 como el nuevo enlace **para** el texto sensitivo "EMPRESA ELECTRICA LOS RIOS" y O **indicara que la** empresa ya es cliente antiguo. Si en vez de O estaba N indica que es un cliente nuevo y que se le va a poner un blink new en el archivo HTML.
- El nombre de este archivo siempre es diferente ya que resulta de la concatenación de poli + (variable hora). Es decir en el directorio **c:\icaro** todos los archivos poli*.txt han sido retornados por la función retroalimemntación, pero cuando ya es interpretado el archivo TXT **por** la función convertidor este es **borrado.**
- Luego este archivo es enviado a la función convertidor el cual coge este archivo y lo transforma en html. Para hacer esto lo primero que hace es leer el archivo y conoce que la primera línea le va a dar el nombre de la plantilla (HTML) donde

debe insertar el resultado del query. Además conoce que siempre al inicio de cada registro del query (inicio de cada línea a partir de la sexta) esta el URL, es decir *el* archivo html que debe llamar cuando cada registro retornado de la base de datos va a ser sensitivo. Es decir en el archivo ya se le envía los nombres de archivos html que debe generar.

- Es de esta forma que retroalimentación es considerada la función que interpreta y traduce que nuevo html hay que generar y si el texto retornado del query va a ser sensitivo y si el HTML a generar va a obtener un dato del usuario. Como nos damos cuenta es independiente del número de usuario que accesan, ya que se crea un objeto por cada usuario.

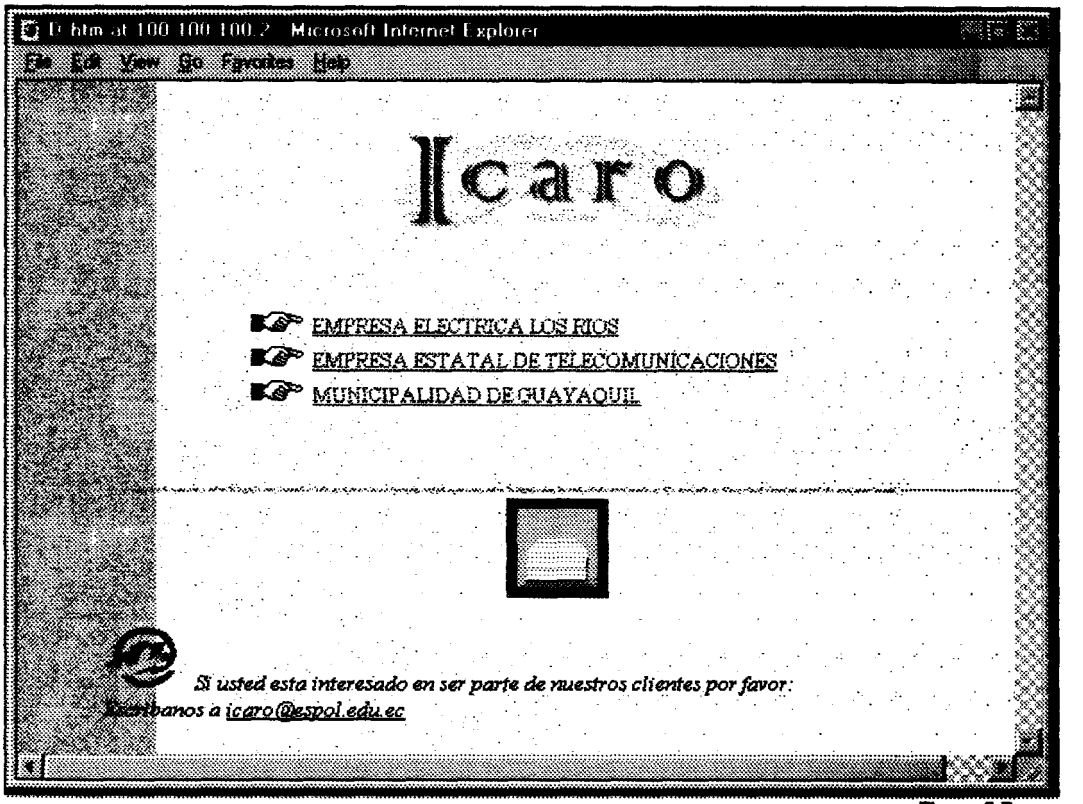


Figura 5.7

- Si escoge el enlace **EMPRESA** ELECTRICA LOS RIOS, se llama al enlace D01.HTM y se creara el siguiente archivo TXT :

```
Inicial.htm
EMPRESA ELECTRICA LOS RIOS
Emelrios.bmp

CANCELE SUS PLANILLAS AL DIA
D0130\tO\tCONSULTA DE CUENTA POR CEDULA\t\n
D0231\t\O\tCONSULTA DE CUENTA POR NOMBRE\t\n
D0332\t\O\tCONSULTA DE DEUDA POR CEDULA\t\n
D0133\tO\tCONSULTA DE DEUDA POR CUENTA\t\n
D0234\t\O\tCONSULTA DE DEUDAS VENCIDAS POR CUENTA\t\n
D03EM\t\O\tNOTIFICACIONES\t\n
```

Figura 5.8

- El archivo HTML que se generara es el siguiente. Ver lo que hace el indicador N en la segunda y cuarta consulta.

- Pero estas consultas pueden ser retornadas en la misma plantilla o en diferentes plantillas. Las plantillas son archivos HTML que tienen un contenido por default, ejemplo el encabezado y pie de página. *Y* en el body existe un indicador [*$contenido] que permite detectar en que lugar debe ser insertado el texto del archivo que contiene el resultado del query, (este archvo es retornado por retroalimentación).

- Consideramos indispensable el desarrollo de un programa que **permita dar** mantenimiento a nuestra base de datos, esto es registrar a nuestros clientes, actualizar sus datos, eliminarlos si ya no necesitan nuestro servicio *o* **por** mora. **Y**

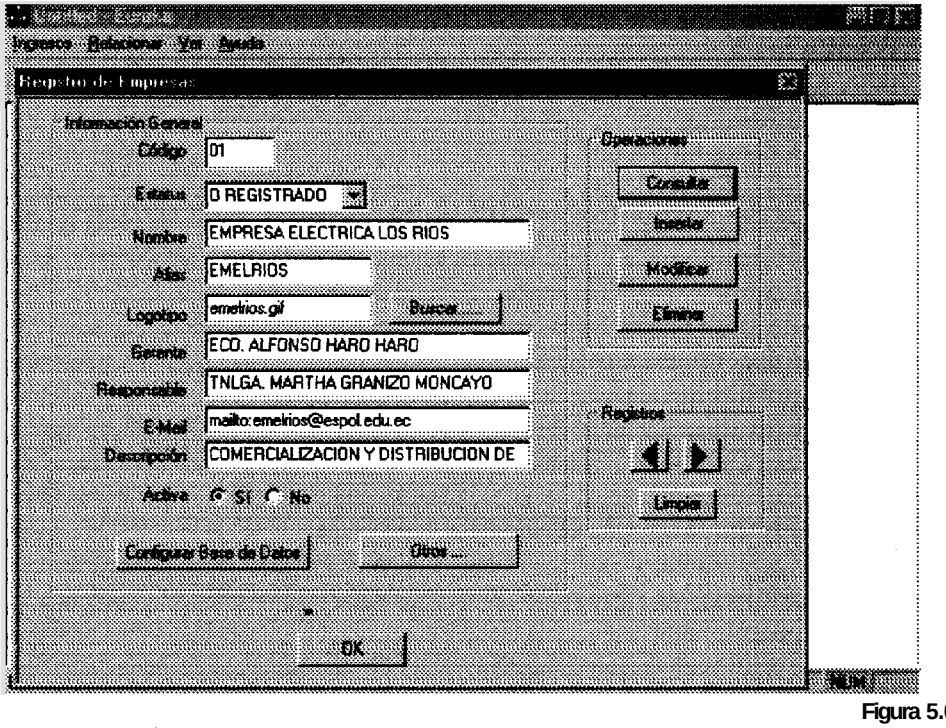


Figura 5.0

lo más i nportante es registrar el acceso DI ` user y word q nos permitirá accesar a su base de datos. El programa de mantenimiento se llama EUREKA y presenta la siguiente interface. *Ver* ***manual*** *del* ***usuario de*** *EUREKA* para mayor información. Como se puede ver en el siguiente gráfico EUREKA es el software que permitirá registrar a los clientes, ponerles un Status, etc.

- Como podemos notar a parte de registrar a las empresas también registramos consultas y plantillas por lo siguiente.

Cada empresa puede tener las consultas que desee dar a sus a usuarios y en el formato que más le guste escogiendo las plantillas.

5.2.3.1 Puntos a considerar en la implementación

- La Interacción Hombre-Máquina debe ser bien diseñada ya que se desea que los usuarios de las empresas estatales sientan una real atención a sus consultas y reclamos, y además hay que considerar que la mayoría serán usuarios inexpertos.
- Las transacciones que se realicen sobre la base de datos de nuestros clientes deben ser bien estudiadas con sus representantes y ser aceptadas mediante un documento, ya que son ellos los que decidirán si solo se realizará consultas *o* si desean un servicio de cobro, registro de abonados, actualización de datos de sus abonados, etc. Ya que esto necesitará un esfuerzo extra de nuestra empresa para tomar medidas de seguridad como puede ser uso de password, actualización en una tabla temporal, encriptación de la información, etc. Medidas que deben ser estudiadas.

- Si la empresa requiere actualizaciones sobre sus bases de datos, existe un acceso restringido en el uso del software con respecto a que tiene password y debe quedar registrado quien de nuestros técnicos realiza una actualización y así no hallan irregularidades por parte de nuestra empresa.
- Además es nuestro cliente el que nos dará un user y password para poder accesar a las tablas que ellos consideren que podemos accesar. Ya que para accesar por ODBC necesitamos estos parámetros."

5.2.4 Descripción técnica

- Retornando al tema que ICARO posee un acceso a Bases de Datos de otras empresas ubicadas en cualquier lugar, a través de ODBC, se puede aclarar que entonces NO interesa la Base de Datos que posea la empresa que solicite el servicio que presta ICARO. También es conocido que lo único que se necesita para accesar a una Base de Datos por ODBC es que exista el drive para ese DBMS, entonces la Base de Datos cliente deberá tener un drive para ODBC. Todos estos drive deben estar instalados en el Servidor de ICARO, ya que es este el que accesará a las Bases de Datos clientes"

- **Para** que una empresa sea cleinte de ICARO debe existir en el mercado el controlador ODBC de se base de datos. Aprovechando la mención de los driver, es importante indicar que ODBC es ahora un estándar y la mayoría de los fabricantes de DBMS también fabrican los driver de ODBC para acceso a la Base de Datos fabricada por ellos.

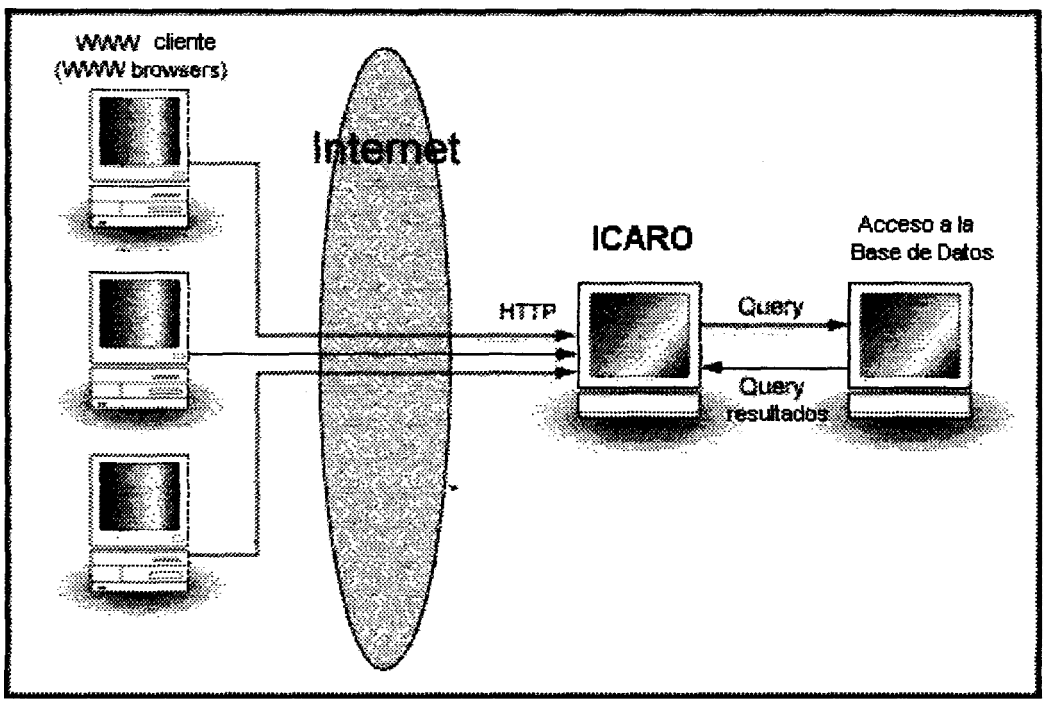


Figura *5.6*

- Una vez que ICARO y la empresa cliente encuentran el driver para ODBC, el siguiente paso es que la empresa cliente indique qué consultas desean ofrecer a sus abonados en el Internet. Para esto dicha empresa cliente debe permitir el acceso a la Base de Datos que ellos poseen, y lo que tienen que facilitar para que

ICARO acceda es: el nombre de la Base de Datos y asignar a ICARO un usuario o user, con su clave respectiva. Además ICARO solo solicita las sentencias **SQL** que permitan extraer la información que ellos quieran mostrar en las consultas anteriormente establecidas.

- Una vez establecido lo anterior se procede a crear la empresa nueva en la Base de Datos de ICARO, guardar las consultas y asignarle un código para que pueda ser accedida desde la Intemet, todo ese proceso de la asignación, se lo hace a través del software de mantenimiento de ICARO llamado EUREKA.
- La Base de Datos de ICARO está construida en Access versión 7.0 para Windows 95, entre otras cosas esta Base de Datos toma el nombre de "empresas" y es accedida por ICARO también a través de ODBC.
- Utilizamos el ODBC de Microsoft para poder configurar los accesos a las bases de datos de las empresas.
- Esto significa que nuestros clientes pueden tener bases de datos ORACLE, INFORMIX, SQL SERVER, SYBASE, MSACCESS. Lo único que debe cumplir la base de datos es que debe tener los controladores ODBC. Físicamente puede estar a cualquier distancia, esto significa que la base de datos puede ser remota; es decir puede estar en un servidor que no es parte de nuestra intranet..

- Por lo descrito en los párrafos anteriores, las empresas a las cuales le brindamos el servicio *(nuestros clientes)* deben formar una *Red privada*, exclusivamente entre estas e ICARO.
- *Red Privada:* La comunicación que formen *nuestros clientes* con ICARO dependerá del costo y de la distancia que se encuentre el local de ICARO y *nuestros clientes*, es decir, puede ser por línea dedicada, por enlaces de radio, por microonda, por fibra óptica, por CDPD, por cableado estructurado (con cables UTP). Ver el punto 5.4 donde se detalla el estudio técnico de la red privada.

Ventajas:

- Nuestro WebServer permite un servicio flexible ya que puede accesar a bases de datos que corren sobre cualquier plataforma.
- Las bases de datos son efectivas para proveer grandes cantidades de información al usuario a través del Web. La información esta en línea y fácilmente compartida por medio de redes privadas.
- Actualmente existen ciertos paliativos para informar a los usuarios de sus deudas, tales como el envío de las planillas donde se les indica las deudas y la fecha máxima de pago, también existe la consulta de deudas por teléfono a través de un código, pero la interacción hombre-máquina es muy rígida por lo que

consideramos que a estas alternativas es mejor la consulta de deudas por medio del Web, ya que este ofrece un mundo hipermedia, es decir texto, imágenes y sonidos. Con estás características podemos llamar la atención del usuario y proporcionarle una interface que le brinde confianza .

- Las páginas HTML serán diseñadas considerando todas las recomendaciones de interacción hombre-máquina, y es necesario recalcar que actualmente se hacen consultas a las bases de datos vía ODBC con sentencias SQL, pero que no sería difícil cambiar este esquema para que en la búsqueda se utilicen los algoritmos de búsqueda de inteligencia artificial, es por esto nuestra continua capacitación en las siguientes ramas:
  - Interacción Hombre-Máquina
  - Inteligencia Artificial
- El Web Server en general queda abierto a futuras mejoras, y las clases que se crearon tanto para el Web Server y el acceso a la base de datos pueden ser re-utilizados en otros proyectos (Ventaja de la programación orientada a objetos).

5.2.5 Ubicación

El local donde funcionará ICARO debe estar ubicado dentro de los sectores comerciales y buscando la cercanía de las compañías que pueden ser clientes de

ICARO, además el sector donde se ubique, debe estar provisto de los servicios básicos como son: Agua, Energía Eléctrica, Alcantarillado, y por supuesto líneas telefónicas.

El lugar donde se ubique a ICARO debe tener el espacio **aéreo** NO muy copado, para que el ruido NO afecte a la transmisión de datos entre clientes e ICARO, si se la realiza por medio de enlaces de radio *o* microondas, utilizando como medio de transmisión el aire.

El local debe tener un área de alrededor de unos 100m2, y brindar un sistema de ventilación y aire acondicionado, esto es imprescindible para los equipos.

5.2.6 Capacidad de Servicio

Para que ICARO funcione en óptimas condiciones, se necesita comprar un equipo servidor que soporte al número de clientes proyectados, ya sea **para** el **inicio** y para el **futuro,** es decir, inicialmente se espera que los clientes sean todas las empresas estatales en las cuales los usuarios de estas mantengan deudas, y todo aquello que se ha explicado en las etapas anteriores; aproximadamente solo en la ciudad de

Guayaquil tenemos cinco empresas estatales tales como Municipio, Consejo Povincial, Gobernación, Empresa de Agua Potable y Alcantarillado, Empresa de telecomunicaciones y otras que se podrían añadir en el futuro como por ejemplo Instituto Ecuatoriano de Seguridad Social (IESS), Instituto Ecuatoriano Crédito Estudiantil (IECE), los colegios, Universidades y Escuelas Politécnicas. Los clientes inicialmente serán aquellos que están ubicados a nivel local, es decir, en la misma ciudad; pero en el futuro ICARO se difundirá a nivel Nacional e Internacional.

Y en cuanto a Usuarios se refiere, tenemos a todos los usuarios de estas empresas, que son todos los habitantes de la ciudad y Provincia, y *en* el futuro todos los usuarios. Dado que el servicio que presta ICARO es solo pedir requerimientos y no mantener una conexión permanente, soporta cualquier cantidad de usuarios accesando a las páginas, solo veinte (20) accesando al mismo tiempo.

Es así que se necesitarán adquirir los siguiente equipos:

- Servidor Web (Compaq ProLiant 5000 6/2000)
- Server de comunicaciones
- Hub
- Router

- Enlaces de radio, líneas dedicadas, cableado estructurado, fibra óptica
- Módem Satelital, para enlace a Internet
- Firewall para protección de la información de nuestros clientes
- Computadoras (Pentium **166** Mhz, **16** M *o 32* M de RAM, equipo multimedia 10X), para mejorar continuamente ICARO
- Impresoras (HPDesk Jet 820 Cse), *(No* son necesarios)

5.2.7 Programación del equipamiento requerido

5.2.7.1 Presupuesto de las Inversiones

A continuación mostramos el detalle de las inversiones que se realizarán a **lo** largo del primer año, con cinco (5) empresas clientes hasta tener en funcionamiento ICARO, el costo está dado en dólares americanos.

SOFTWARE:	
MSAcces 7.0	30
	0
Visual C++	
	300
Windows 95	
	300
Netscape	
	100
PARCIAL	10
	00

Tabla 1.

HARDWARE:	
Equipos para brindar el servicio:	
Equipo Servidor (1)	
	10000
Server de comunicaciones	8750

(1)	
Concentrador Grande(1)	3000
Concentrador Pequeño (1)	450
Router (1)	1750
Módem satelital (1)	850
Firewall (1)	4000
Computadoras (5)	1000
	0
Cables y demás materiales para	5600
la red de desarrollo e Internet .	
Impresoras (1)	300
Permiso de funcionamiento	500
Otros	1000
PARCIAL	4620
	0

Tabla II.

Muebles de oficina	
Escritorios (2)	500
Mesas para computadoras (6)	1200
Líneas telefónicas (2)	400
Teléfonos (2)	120
Sillas	2000
PARCIAL	4220

Tabla III.

SOFTWART	1000
HARDWARE	46200
Muebles de oficina	4220
TOTAL	51420

T IV.

Como podemos apreciar, el monto total a invertir durante en el primer año *es* de 51.420 dólares.

Para la implementación de la ***Red*** *Privada* se hace necesario un estudio de comunicaciones mas profundo, buscando exclusivamente el medio a bajo costo, que

proporcione excelente seguridad, haya disponibilidad en el mercado local, sea factible implementarlo en los lugares físicos donde estén ubicados *nuestros clientes.*

Cabe señalar que estas inversiones se volverán a realizar dentro de tres años en **por** el mismo monto y en la misma secuencia, debido a que **los** equipos de computación NO estarán con tecnología de punta, es decir, ya estarán discontinuados.

5.2.7.2 Presupuesto de arranque

Para el inicio de la puesta en marcha del proyecto está planeado que arrancará con una sucursal en la ciudad de Guayaquil con un total de cinco (5) clientes, los elementos que se necesitarán para el arranque del proyecto son:

EQUIPOS:	
Equipos para brindar el servicio:	
Equipo Servidor Web (1)	10000
Server de comunicaciones (1)	8750
Concentrador Grande(1)	3000
Concentrador Pequeño (1)	450
Router (1)	1750
Módem satelital (1)	850
Firewall (1)	4000
Computadoras (5)	10000
Software (Sistemas Operativos)	1000
Cables y demás materiales para la red de desarrollo e Internet	5600
Impresoras (1)	300
Permiso de funcionamiento	500
Otros	1000
PARCIAL	47200

Tabla V.

Muebles de oficina	
Escritorios (2)	500
Mesas para computadoras (6)	1200
Líneas telefónicas (2)	400
Teléfonos (2)	120
Sillas	2000
PARCIAL	4220

Tabla V.

Equipos y Software	47200
Muebles de oficina	4220
TOTAL	51420

ab V.

De este detalle calculamos que la inversión inicial a realizar es de **51.420** dólares. En el presente proyecto se realiza el análisis tomando en menta si se financia *o* no este rubro.

5.3 Estudio del mercado

5.3.1 Características de la demanda

Nuestros potenciales clientes son aquellas empresas que desean formar parte de la red de Internet y que no cuentan con los recursos para desarrollar en sus propias aplicaciones para el Web.

Se encuentran distribuidas a lo largo del país, pero con mayor concentración en las provincias del Guayas, Pichincha y Azuay.

El cliente busca proveer al usuario un medio por el cual pueda tener acceso a la información más completa y actualizada en cuestión de segundos; desean que sus productos sean conocidos por todos; mantener su Status, no quedar rezagados de lo nuevo y actual; y por último, contar con un medio eficaz y efectivo de hacer negocios. En definitiva lo que busca el cliente es ver y ser visto.

Nuestro cliente contará con un verdadero y efectivo grupo de profesionales que le brindarán servicio y asistencia permanente las 24 horas, los usuarios tendrán acceso durante todo el día. Además todos nuestros clientes

contarán con espacio en nuestro servidor (sin costo extra) brindándoles la oportunidad de incluir elementos multimedios y gráficas en su sitio.

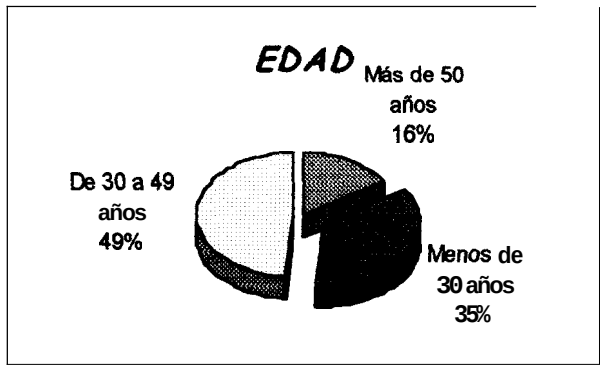
Los factores que inciden en la demanda son, el precio de los computadores, del servicio, de la comunicación, del alquiler del satélite, del tipo de servicio que brindan las compañías de teléfono, entre otros.

Hoy todo el mundo habla de la Internet. Los usuarios se multiplican geométricamente, las empresas y personas particulares pugnan por su presencia en ese mundo electrónico con el interés de ver y ser vistos.

Establecer las cifras reales de la red es tarea muy compleja. Se dice que conocer ***algo*** real es prácticamente imposible por la cantidad de personas que accesan **al mismo** tiempo y porque los contadores que algunas páginas presentan, pueden estar arreglados deliberadamente para aumentar las realidades. Una razón para ellos es que el negocio de la publicidad en Internet ha arrancado a pasos agigantados. Se calcula que después de tres años, en el nuevo siglo, la publicidad ***en*** la red esté facturando cifras superiores a los 200 millones de dólares **al** año.

Pese a esta baja confiabilidad en las estadísticas, se ha demostrado y para ello no hace falta ser un experto, que la red es el medio de comunicación más personalizado que existe. Para quien presente su información, el intentar mantener una vinculación real y positiva con sus clientes es absolutamente posible. El usuario se sienta frente a su computadora, se abstrae de su entorno y, virtualmente, ingresa a ese mundo mágico que le pone todo al alcance de sus sentidos.

Muchas son las instituciones, especialmente universidades que hacen esfuerzos por presentar ante el mundo datos cercanos a la realidad sobre quiénes son las personas a las que los recursos de la Internet puedan ser dirigidos. El Instituto de Tecnología de Georgia, de Estados **Unidos,** por ejemplo, ha demostrado que el 17 por ciento de la población norteamericana ya está conectada a la red, o como ellos lo llaman están "on line". Casi la mitad de ellos son personas entre los **30** y **49** años. Aunque en el Ecuador no tenemos todavía un estudio similar, de un muestreo realizado por la Revista Vistazo deducen que la tendencia de los suscriptores de Internet es similar.



Información de la revista Vistazo de 1996

Figura 5.10

Hasta hace poco, el comercio en el Web era más un rumor que una realidad, no obstante, ahora los compradores están adquiriendo sus billeteras en iínea.

Hasta ahora, el principal impedimento para el comercio en Web ha sido el riesgo potencial de transmitir su número de tarjeta de crédito u otra información de transacciones a través de una red inherentemente insegura.

El Web es una billetera, o al menos lo será muy pronto. Con estas premisas se proyecta un futuro muy prometedor para quienes se vean inmersos en este mercado, se estima que este mercado se encuentra en plena etapa de crecimiento, y que continuará así por lo menos una década.

5.3.2 Variaciones de la demanda

La demanda se ve afectada principalmente por las tendencias tecnológicas, políticas y sociales.

5.3.2.1 Tecnológico:

- La investigación permanente han hecho que se descubran y creen nuevos medios de transmisión cada vez más veloces, capaces de enviar por un mismo medio **voz** y datos y con una gran confiabilidad, lo cual nos hace prever que el futuro es muy prometedor para el área de comunicaciones.
- El costo del hardware cada vez disminuye **más**, brindándonos la oportunidad de adquirir más y mejores equipos a menor precio, haciendo accesible principalmente a que todas las personas puedan adquirir un computador, de esta manera se convierte en un futuro cliente de nuestro servicio.
- La era donde la información en el Web, ha llevado a que la nueva tecnología este orientada a compartir información alrededor del mundo.

5.3.2.2 Social:

- En el mundo actual donde la tendencia es: "El dueño de la información es el dueño del poder", se hace necesario contar con un medio rápido, efectivo y eficiente; por el cual estar al tanto de diversos temas de diferente índole.
- El conocimiento cada vez mayor de este medio de comunicación, información, comercialización y entrenamiento; está creando una explosión de intereses por tener acceso a esta Red de Redes que es Intemet.
- El gran número de usuarios alrededor del mundo que accesan a esta red está haciendo volcar los ojos de todos para realizar negocios de todo tipo.

5.3.2.3 Políticas:

- Existe un grado de descontento debido a la no regulación de transacciones comerciales vía Intemet en el país, debido a que aún no se emiten leyes con respecto al encriptamiento de datos, especialmente aquellos datos de los clientes que son confidenciales como lo es el número de tarjeta de crédito que vendría a ser el medio principal de compra vía Internet.

- Las medidas que tome el gobierno en turno para cubrir el déficit de líneas telefónicas en el país, por lo que es el medio de comunicación que es usado para el acceso a Internet.
- Las trabas que ponga el gobierno para el desempeño de esta actividad, tales como: Impuestos, garantías, etc.
- El incremento cada vez mayor de los centros educativos que imparten conocimientos de computación *en* todas sus especialidades, hace que cada vez un mayor número de personas conozcan de este servicio, y por lo tanto incrementando el mercado.

5.3.2.4 Otros datos:

Mucho interés a generado la posibilidad de utilizar los recursos de Internet sobre la cantidad de servicios especializados que dentro de la Red se encuentran.

En un principio es interesante ver como ha crecido el número de empresas dedicadas a los servicios de contenido WEB *o* contenido para la telaraña mundial (World Wide Web). En los últimos nueve meses los Web empresariales y comerciales también han proliferado debido a la rápido difusión de los beneficios de la telaraña de Internet (principalmente el obtener presencia instantánea de la Red Mundial), que

ha su vez ha creado una creciente demanda de diseño, desarrollo y renta de espacio en servidores (homepage hosting) para lo que se conoce como páginas WEB (homepages).

Esta demanda ha creado un creciente número de empresas que se dedican al desarrollo y diseño de paginas WEB, por lo que podemos decir que la necesidad de contar con diseñadores, que además de estar preparados en su área tengan conocimiento de tecnología, va adquiriendo relevancia.

Existen páginas WEB sencillos que contienen atractivo texto y una sencilla gráfica; o por el contrario, existen las más elaboradas como la página o servicio WEB empresarial que puede tener cuatro o veinte o más páginas con atractivas imágenes, botones interactivos, bases de datos y aplicaciones multimedia vía ligas o hiperenlaces que son activadas con el click de un ratón. El costo así depende de la necesidad de información y promoción de los bienes y servicios del cliente.

Sin embargo al investigar los estándares de cotización de los desarrolladores de páginas WEBs muestran "que no existen estándares" en el emergente negocio, pero que los diseñadores estudian, primero profundamente las necesidades de información y los requerimientos del cliente antes de dar una cotización y que puede

variar desde 300,400 hasta 5,000 dólares por el desarrollo de una "Internet" (incluida una página WEB), *o* sea el desarrollo de una red WEB interna en la **empresa para *en*** intercambio de datos, documentos e información y aplicaciones "groupware".

Una página WEB sencilla con texto se puede cotizar en un promedio de 425 dólares, y por otro lado un servicio WEB de cuatro páginas se cotiza según la necesidad del cliente entre poco más de 550 y 1,000 dólares. Todos estos precios mencionados son **por** seis meses. También investigamos que el costo, de poner el servicio de una revista con todo e imágenes puede alcanzar entre 6,000 y *6,600* dólares.

5.3.3 Estrategia de mercadeo

5.3.3.1 Producto

Se busca brindar servicio de acceso a la Red Internet, permitiendo curiosear a través de la Red Global Mundial (WWW), la creación de los Home-Pages de los usuarios, soporte telefónico, instalación del módem si es necesario, instalaciones a domicilio, dar capacitación a todos los clientes que se suscriben a nuestra empresa, difundir revistas a todos los clientes, entre otras.

Buscamos brindar servicio de información, es decir permitir a los usuarios de nuestros clientes poder consultar deudas, datos, códigos, etc. Y que al mismo tiempo puedan notificar de posibles problemas a través de la Red Global Mundial (WWW), participar en grupos de noticias, la creación de los Home-Pages de nuestros clientes, , difundir revistas a todos los usuarios.

5.3.3.2 Promoción

Sabemos que la imagen que proyecta una empresa es la realidad que perciben los clientes y competidores, por eso hay que cuidarla especialmente; y es eso lo que buscaremos, proyectar la imagen de que somos una empresa seria preocupada por el cliente, hacer sentirlo que tendrá asistencia técnica especializada en el momento en que la necesite. Nos daremos a conocer por anuncios publicitarios en diarios y revistas, principalmente.

El servicio se lo ofrecerá por medio de agentes vendedores que visitarán especialmente a las empresas, centros educativos, instituciones y profesionales en todas las áreas dándoles a conocer todos los beneficios y ventajas de integrarse a este red de información.

5.3.3.3 Lugares de venta

Como lo hemos dicho anteriormente, contaremos con una sucursal una en la ciudad de Guayaquil, las oficinas se encontraran ubicadas en un lugar preferentemente comercial de cada ciudad.

5.4 Red Privada de ICARO y las empresas clientes.

Por ofrecer el Centro de Servicios ICARO un servicio dinámico y actualizado, es de suponerse que necesitará de un Sistema de conexión con sus empresas filiales.

Para esta conexión se propone establecer una RED **PRIVADA** conformada por ICARO y sus empresas filiales.

Como se especifica en este documento, ICARO es posible implementarlo, por esto es necesario plantear una alternativa de solución para dicha red. También es necesario indicar que la opción más adecuada será acogida *en* el momento de la implementación.

Los requerimientos planteados para la red privada **son:**

- Transmisión digital, debido a que es comunicación de datos.
- Rapidez con un promedio de **64** Kbps, debido a que el acceso a la Base de Datos se lo hace a través de ODBC y en tiempo real.
- Seguridad, puesto que las empresas que deseen ser filiales de ICARO deben tener la plena confianza en que sus datos no sean alterados.
- El enlace en lo posible debe mantenerse, mientras existan clientes accesando **a** ICARO

A continuación se describe brevemente las alternativas que solucionan la instalación de la Red y que además existen ***en*** el mercado nacional para establecer conexiones punto a punto entre ICARO **y** sus filiales.

- CDPD
- **VSAT**
- Líneas dedicadas analógicas y digitales
- Enlaces de Radio/Microondas
- Enlaces con fibra óptica

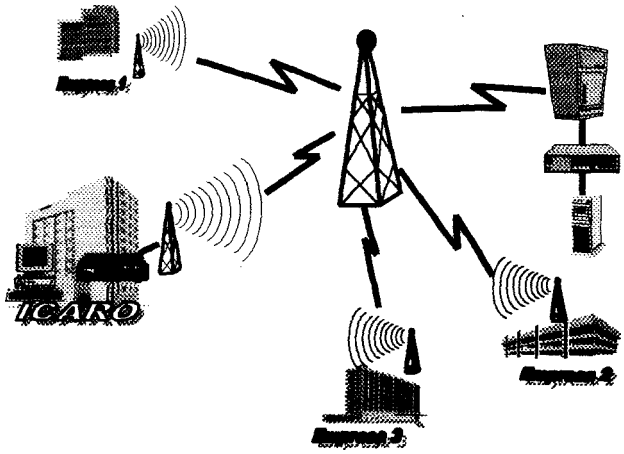
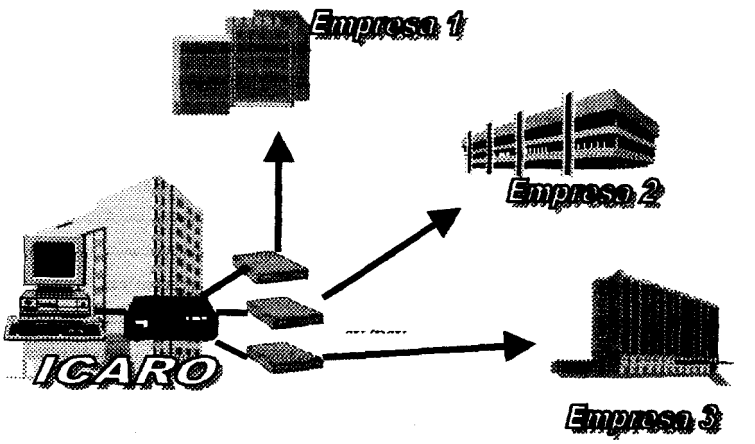


Figura 5.1.1

El enlace por **CDPD** (Celular Digital Packet Data) es realizado por la empresa de telecomunicaciones CONECELL, esta conexión emplea los tiempos NO usados de la red celular (analógica o digital), lo más importante de esta tecnología es que los costos son solo por los paquetes transmitidos, es decir, que si no consumes o no transmiten NO paga y además el Hardware que se necesita para este es un módem CDPD conectado a la computadora de cada empresa filial y en la computadora donde funcionará ICARO.

El **enlace satelital compartido** por **VSAT** es realizado en el Ecuador por compañías como Teleholding e Impsat, y consiste en un enlace compartido entre un grupo de usuarios por medio del satélite con otros puntos ya sea dentro o fuera del país a **través** de una antena VSAT que ***es*** pequeña (1.2m a 1.8m) ubicada en cada uno

de los usuarios, los costos se reducen para cada usuario por la compartición de un mismo canal. Utilizando un canal para compartir entre las empresas filiales a *ICARO*, se podrían distribuir los costos.



Fig

Las líneas dedicadas provista por la empresa estatal de comunicaciones (EMETEL) a través de Teleholding de una manera indirecta, son también otra alternativa, ofrece velocidad máxima de **64** Kbps, que para el acceso a Base de Datos no es tan buena como otras alternativas, no es muy confiable por ser insegura, y además siempre paga una tarifa mensual.

Los enlaces de radio analógicos y digitales son también una buena opción **para** la transmisión de datos entre ICARO y las empresas. En este tipo de enlace existen involucrados costos de frecuencia, costos de instalación que **son** relativamente altos y

además como es conocido no se puede transmitir desde un mismo punto a la **misma** frecuencia, entonces se deben alquilar tantas frecuencias como el doble de las empresas que sean filiales de ICARO.

Un enlace con fibra óptica es rápido, eficaz y seguro, pero la inversión que se debe hacer es muy cara en lo que se refiere a la adquisición del hardware, costos de instalación y de mantenimiento.

La alternativa a escogerse dependerá de la tecnología que exista en el momento en que se lo implemente, puesto que hasta el momento de la implementación pueden invadir nuevas tecnologías al Ecuador, como son PCS, xADL, conexión punto a punto por la red de TVCABLE. Lo importante es no olvidar los requerimientos que se plantean en esta sección.

Si asumimos que se debe implementar en este momento la alternativa puede ser CDPD, por el bajo costo de instalación y su costo solo por datos transmitidos.

Conclusiones y Recomendaciones

Sobre la base de lo estudiado y a los objetivos planteados inicialmente, se pueden obtener las siguiente concluliones y recomendaciones.

Conclusiones

- Fue un proyecto difícil de modelarlo a Objeto, debido al enfoque estructurado que mantuvimos en el estudio de nuestra carrera y también por no palpar tangiblemente los objetos que intervenían en el tema.
- Al encontrar y disitinguir los objetos en nuestro análisis, no fue **fácil** creer que tan pocos objetos se involucren en este proyecto. Sin embargo al solidificar nuestros conocimientos en modelamiento de Objetos aclarábamos que estábamos por buen camino
- El análisis y diseño orientado a objetos resultó natural debido a que aplicación corre en una plataforma Windows. Aunque este proyecto no explota tanto la herencia de las clases fue un punto en contra pero no radical.

- El modelamiento a objetos y a la programación orientada **a** objetos permitió independencia entre los dos importantes objetos como son: el programa servidor web y la base de datos.
- Nuestro programa servidor torna ventaja de poder manipular las peticiones del cliente de acuerdo a estándares establecidos o a especificaciones propias lo que nos permite flexibilidad dentro del servidor. Se pueden implementar seguridades de acuerdo a las necesidades, se puede modificar fácilmente para que actúe como servidor de otros protocolos además de HTTP, y como ejemplo práctico nosotros lo modificamos para generar paginas dinámicas de acuerdo a nuestros estándares. Otros programas servidores (comerciales) ofrecen acceso a base de datos pero mediante métodos predefinidos que involucran en muchos casos aprendizaje de los mismos.
- La generación de páginas dinámicas reduce el esfuerzo de mantenimiento de sitios, especialmente cuando la información cambia constantemente. No obstante, se necesita una aplicación que me permita mantener información como plantillas de encabezado y pie, criterios de consulta de las paginas, etc. , en general mantener la base de datos del sitio en cuestión
- La implementación de la aplicación "Centro de Servicios ICARO", que publica las deudas se tienen con empresas de servicio público, fue la opción adecuada de

acceso a base de datos, dado que este tipo de servicio no se ha implementado en nuestro medio.

- El acceso a las bases de datos mediante ODBC, fue una excelente alternativa por la independencia del DBMS fuente de los datos. El soporte a sentencias SQL estandar es bueno pero no completo. La desventaja radica en tiempos de respuesta muy largos ya que aumenta una capa adicional a las consultas realizadas.
- Para configurar ODBC, solo es necesario tener los drivers que manejan la base de datos a accesar. La configuración varía según el drivers, justamente porque se considera que para cada base de datos se debe tomar diferentes consideraciones. Se debe remitir a la documentación proporcionada por el fabricante para su correcto funcionamiento.
- No hay dificultad en conseguir drivers **ODBC,** puesto que estos están en muchas ocaciones están disponibles ya sea gratis o través de una compra en la Internet.

Recomendaciones

a) De utilización

La funcionalidad que provee ICARO Web Server nos **permite divisar** un sin número de campos en el que puede ser utilizado. **A** continuación describimos algunos de ellos:

- Comercial.- Se puede utilizar ICARO para manejar un Centro Comercial ***en*** el que se permita a los visitantes hacer consultas sobre el stock de un artículo determinado y sus características.
- Pública.- Sería de gran ayuda que la FIEC en su nueva división CDP[1] comercialize el servicio en empresas públicas, tomando ventaja de los derechos que posee sobre todo proyecto presentado por los estudiantes.

b) De búsqueda de información

Si desea tener mayor conocimiento sobre alguno de los temas expuestos en este documento diríjase a la sección Bibliografía.

c) De mejoramiento de la aplicación

[1] Centro de Desarrollo de Proyectos

Sin duda alguna todo Sistema de Computación necesita ser mejorado dependiendo de las necesidades de quien lo vaya a utilizar. ICARO Web Server no es una excepción. Para esto sugerimos ciertos puntos que deberían ser considerados para futuras versiones:

- Implementación de seguridades.
- Servicio de actualización de datos.
- Rediseño de plantillas, explotando los controles Active X.
- Mejoramiento de la Interface de ICARO Web Server

d) De equipos de computación y comunicaciones.

- Se necesitan por lo menos 2 tipos de computadores, uno servidor y otro cliente, y una infraestructura de comunicaciones como se detalla a continuación:

- **Servidor:** Para tener un buen tiempo de respuesta de la aplicación servidor hacia los requerimientos de los clientes, se recomienda que el equipo

principal Windows NT Server sea un computador de las siguientes características **mínimas:**

- Equipo multiprocesador.
- Procesadores de **64** bits con velocidades de 300 Mz.
- Memoria RAM de **256** Mb y de acceso aleatorio menor a 70 ns.
- Disco duro tipo **SCSI** de **3 Gb y** de acceso aleatorio menor a 10 ms.
- Monitor SVGA **.28** de **17"**
- Tarjeta de video de **4Mb** con soporte para Windows NT
- Capacidad Multimedia.
- Tarjeta de interface de red de 100 **Mbps**

- **Cliente:** No requiere gran potencial de procesamiento y puede tener las siguiente características:

- Equipo monoprocesador.
- Procesadores de **32** bits con velocidades de **233** Mz. Tipo Pentium.
- Memoria RAM de **64** Mb y de acceso aleatorio menor a 70 ns.
- Disco duro de **1** Gb y de acceso aleatorio menor a **10** ms.
- Monitor SVGA .28 de **15"**

- Tarjeta de video de 4Mb con soporte para Windows NT
- Capacidad Multimedia.
- Tarjeta de interface de red de 100 Mbps

Comunicaciones:

- Medio físico de transmisión de fibra óptica
- Tarjeta de interface de red de 100 Mbps
- Concentradores / Ruteadores de 100 Mbps
- Modems de 64 kbps

e) De intalación fisica y eléctrica

Para evitar daños físicos en el hardware se recomienda un exámen exahustivo en de las instalaciones como se detalla a continuación y tomar en cuenta las presentes recomendaciones:

Instalación fisica:

- Evitar que los computadores esten cerca de una vía de mucha circulación, sobre todo si hay tráfico pesado

- Evitar que los computadores esten en lugares de mucha vibración, como los pisos superiores de un edificio, cerca de construcciones grandes, etc.
- Trate de que el lugar esté a una temperatura de **10** grados centígrados.
- Evite lugares húmedos
- Evite exposición directa al sol o a cualquier fuente de calor intensa.

Instalación eléctrica:

- Revice la polarización y carga por linea o fase eléctrica
- Revice si la línea a tierra esta bien conectada

f) De software

En general hoy en día las bases de datos "grandes" (ORACLE, INFORMIX, SQL SERVER, SYBASE) son muy seguras, y no existe ningún problema en utilizar cualquiera de ellas.

Con respecto a Windows NT, se debe preferir utilizar la versión **4.0** o superior.

g) De **dificultades**

En el desarrollo de esta aplicación surgieron diversos contratiempos que los describimos a continuación, esperamos que sean considerados por otros grupos de desarrollo en el momento de hacer un cronograma.

- Asesorarse acerca de la instalación de software desconocido para Ud.
- No sea demasiado optimista con su cronograma de trabajo.
- Asesorese correctamente sobre la herramienta y la versión que va ha utilizar para el desarrollo.
- Si separa el proyecto no subestime el tiempo **que** le dedicará a la unión del mismo.
- Trate de que las habilidades de los integrantes del grupo de trabajo estén bien distribuidas de **tal** foma que sus conocimientos sean explotados al **máximo.**

ANEXOS

Anexo 1

Base de datos empresas –Modelo E-R

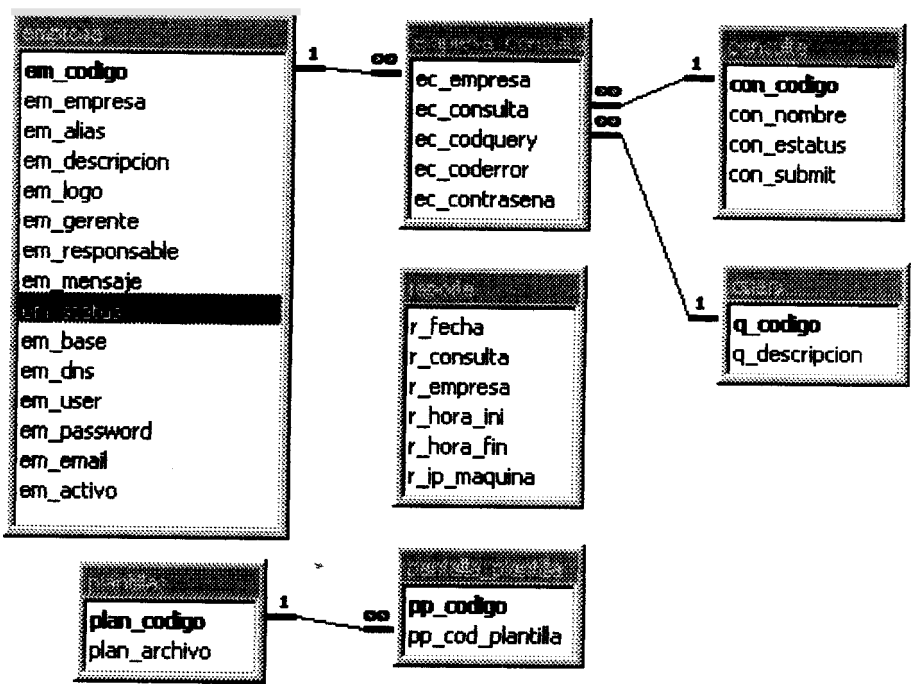


Tabla: consulta

Columnas

Nombre	Tipo	Tamaño
con_codigo	Texto	2
con–nombre	Texto	50
con–estatus	Texto	1

con_submit	Texto	30

Relaciones

consultaempresa_consulta

consulta		empresa_consulta
con–codigo	1 ∞	ec_consulta

Atributos: Exigir, Actualizaciones en cascada

Atributos: Uno a varios

Índices de tabla

Nombre	Número de
PrimaryKey	1
Campos:	con_codigo, Ascendente

Base de datos empresas

Tabla: empresa

Columnas

Nombre	Tipo	Tamaño
em_codigo	Texto	2
em_empresa	Texto	50
em_alias	Texto	30
em_descripcion	Texto	100
em_logo	Texto	15
em_gerente	Texto	50
em_responsable	Texto	50

em_mensaje	Memo	
em_status	Texto	1
em_base	Texto	15
em_dns	Texto	15
em_user	Texto	10
em_password	Texto	10
em_email	Texto	40
em_activo	Texto	1

Relaciones

empresaempresa_consulta

empresa			empresa_consulta
em_codigo	1	∞	ec_empresa

Atributos: Exigir, Actualizaciones en cascada

Atributos: Uno a varios

Índices de tabh

Nombre Número de

PrimaryKey 1

Campos: em_codigo, Ascendente

Base de datos empresas

Tabla: empresa–consulta

Columnas

Nombre	Tipo	Tamaño
ec_empresa	Texto	2
ec_consulta	Texto	2
ec_codquery	Texto	3
ec_coderror	Texto	10
ec_contrasena	Texto	1

Relaciones

consultaempresa_consulta

consulta		empresa_consulta
con_codigo	1 ∞	ec_consulta

Aíributos: Exigir, Actualizaciones en cascada

Aíributos: Uno a varios

empresaempresa_consulta

empresa		empresa–consulta
em_codigo	1 ∞	ec_empresa

Atributos: Exigir, Actualizaciones en cascada

Atributos: Uno a varios

queryempresa_consulta

query		empresa_consulta
q_codigo	1 ∞	ec_codquery

Atributos:	Exigir, Actualizaciones en cascada
Atributos:	Uno a varios

Índices de tabla

Nombre	Número de
consultaempresa–consulta	1
Campos:	ec_consulta, Ascendente
empresaempresa_consulta	1
Campos:	ec_empresa, Ascendente

Base de datos empresas

Tabla: empresa–consulta

queryempresa_consulta	1
Campos:	ec_codquery, Ascendente

Base de datos empresas

Tabla: pantalla_plantilla

columnas

Nombre	Tipo	Tamaño
pp_codigo	Texto	15
pp_cod_plantilla	Texto	3

Relaciones

plantillaspantalla_plantilla

plantillas		pantalla_plantilla
plan_codigo	1 ∞	pp_cod_plantilla

Atributos: Exigir, Actualizaciones en cascada, Eliminaciones en cascada

Atributos: Uno a varios

Índices de tabla

Nombre	Número de
plantillaspantalla_plantilla	1
Campos:	pp_cod_plantilla, Ascendente
PrimaryKey	1
Campos:	pp_codigo, Ascendente

Base de datos empresas

Tabla: plantillas

Columnas

Nombre	Tipo	Tamaño
plan–codigo	Texto	3
plan–archivo	Texto	15

Relaciones

plantillaspantalla_plantilla

plantillas		pantalla_plantilla
plan_codigo	1 ∞	pp_cod_plantilla

Atributos: Exigir, Actualizacionesen cascada, Eliminaciones en cascada

Atributos: Uno a varios

Índices de tabla

Nombre	Número de
PrimaryKey	1

Campos: plan–codigo, Ascendente

Base de datos empresas

Tabla: query

Columnas

Nombre	Tipo	Tamaño
q_codigo	Texto	3
q_descripcion	Memo	

Relaciones

queryempresa_consulta

query		empresa_consulta
q_codigo	1 ∞	ec_codquery

Atributos: Exigir, Actualizaciones en cascada

Atributos: Uno a varios

Índices de tabla

Nombre	Número de
PrimaryKey	1

Campos: q_codigo, Ascendente

Base de datos empresas

Tabla: reporte

Columnas

Nombre	Tipo	Tamaño
r_fecha	Texto	50
r_consulta	Texto	2
r_empresa	Texto	2
r_hora_ini	Texto	20
r_hora_fin	Texto	20
r_ip_maquina	Texto	15

Base de datos empresas

Relaciones: Todo

Relaciones

consultaempresa_consulta

consulta		empresa_consulta
con_codigo	1 ∞	ec_consulta

ActivarOrden: Uno a varios

Atributos: Exigir, Actualizaciones en cascada

empresaempresa–consulta

empresa		empresa_consulta
em_codigo	1 ∞	ec_empresa

Atributos: Exigir, Actualizaciones en cascada

Atributos: Uno a varios

plantillaspantalla_plantilla

plantillas		pantalla_plantilla
plan_codigo	1 ∞	pp_cod_plantilla

Atributos: Exigir, Actualizaciones *en* cascada, Eliminaciones en *cascada*

Atributos: Uno a Vanos

queryempresa_consulta

query		**empresa_consulta**
q_codigo	1 ∞	**ec_codquery**

Atributos:	**Exigir**, Actualizaciones *en* cascada
Atributos:	**Uno** a vanos

Anexo 2

Manual Centro de Servicios ICARO

Bienvenido a nuestro Centro de Servicios ICARO en la Internet, en este lugar encontrará información sobre sus deudas a las empresas de Teléfonos, Luz, Municipio, etc. así como los datos relacionados con los servicios que estas empresas ofrecen al público.

Este manual intenta dirigir al Visitante a explotar todo el potencial de este Centro de Servicios, para ello describiremos cada una de las opciones que se ofrecen.

Esperamos que el material aquí expuesto preste toda la claridad y facilidad para el uso de este Centro de Servicios.

Conceptos básicos

Requerimientos de Hardware.-

ICARO Centro de Servicios no requiere de un dispositivo adicional de hardware además de los que se necesita para navegar por la Internet y disfrutar de todo su potencial. Estos requerimientos son:

- Un computador Pentium de **133** Mhz
- 16 Mb de RAM
- Un módem de 28.8 Mbps
- Un Kit Multimedia o Sound Blaster

Si Ud. no cuenta con una estructura mínima de hardware como la descrita su conexión a la Internet será muy lenta y poco eficiente.

Requerimientos de Software.-

Los requerimientos en cuanto a software para el funcionamiento de ICARO son:

- Windows'95, Windows NT Workstation **4.0** ó Windows NT Server **4.0** como Sistema Operativo.
- Un Browser (Netscape, Intemet Explorer, **u otro).**
- Una Cuenta Internet de cualquier proveedor local [1]

[1] **Ecuanet,** Telconet, **Satnet, enire** otros.

Funcionamiento

Inicio

Para accesar a nuestro Centro de Servicios ICARO primeramente debemos dar inicio al Browser[2] para conectarnos a Internet.

Una vez inicializado el Browser escribimos la dirección http://icaro.com e ingresamos al Home Page del Centro de Servicios ICARO Figura 1. y estamos listos para empezar la navegación por el Centro de Servicios.

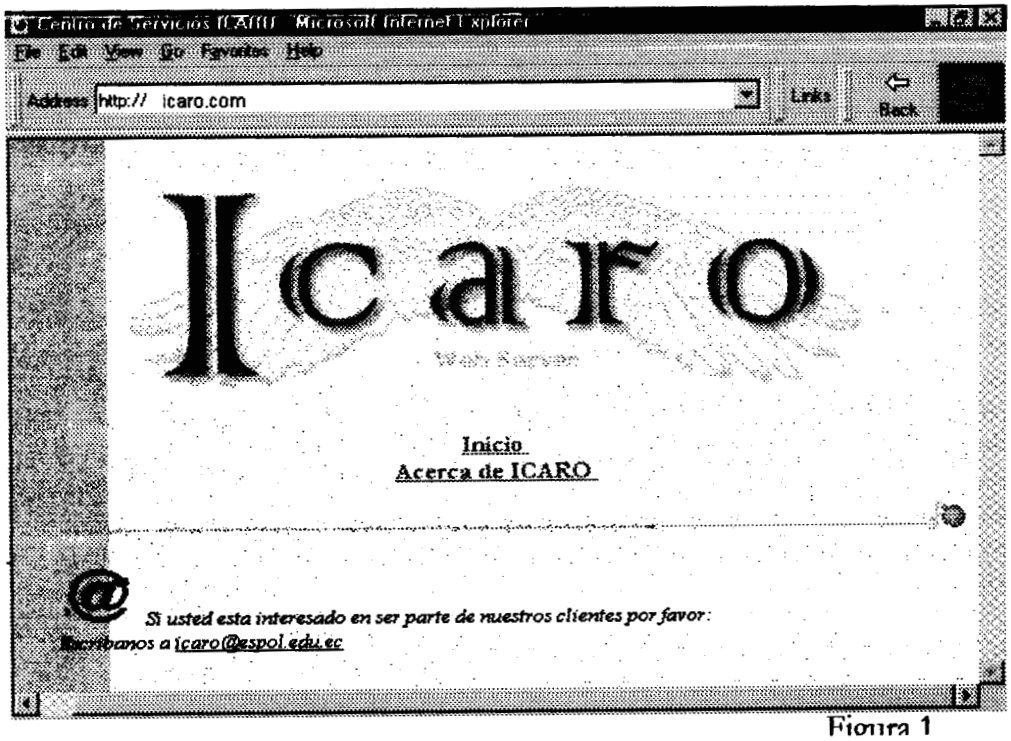


Figura 1

Para el desarrollo de este manual usaremos el Intemet Explorer **como** Browser.

Formato de las Páginas de ICARO.-

Las páginas del Centro de Servicios ICARO tienen el mismo formato aunque los tipos de plantilla [3] pueden diferir por cada empresa, esto se debe al hecho de que cuando una empresa se subscribe a nuestro Centro de Servicios tiene la libertad de escoger el tipo de plantilla en el **cual** se van a mostrar sus datos.

El formato de las páginas construidas es el siguiente:

- En la parte superior siempre estará el logotipo del Centro de Servicios.
- En la 1era iínea estará el título de la empresa seguido por su logotipo.
- En la 2da línea estará el título de la consulta, de existir este.
- En la 3era línea estará un mensaje para el usuario, sino se ha hecho una consulta esta línea se constituirá en la segunda.

[3] archivo HTML predefinido usado como modelo.

- El resto de información mostrada tiene que ver con el tipo de consulta que se hizo.

Sin embargo hay la página resultante de escoger el ítem **NOTIFICACIONES** en esta página sale la dirección Internet de las empresa para enviar sus inquietudes o sugerencias por medio del servicio e-mail.

En la parte inferior de las páginas hay un gráfico que nos permite enlazamos a

la primera página Figura. 1. El gráfico es el siguiente:

Navegación en CARO.-

En el Home Page de ICARO Centro de Servicios nos muestra dos textos sensitivos.

Como resultado de escoger la opción Inicio se nos muestra una pantalla de Bienvenida al Centro de Servicios tal como lo muestra la Figura. 2

Esta pantalla muestra opciones, si escogemos la opción **Nuestros Clientes** tendremos el listado de todos las empresas públicas que utilizan nuestro Centro de Servicios para mostrar las deudas de sus clientes y otra información que sus clientes deben saber.

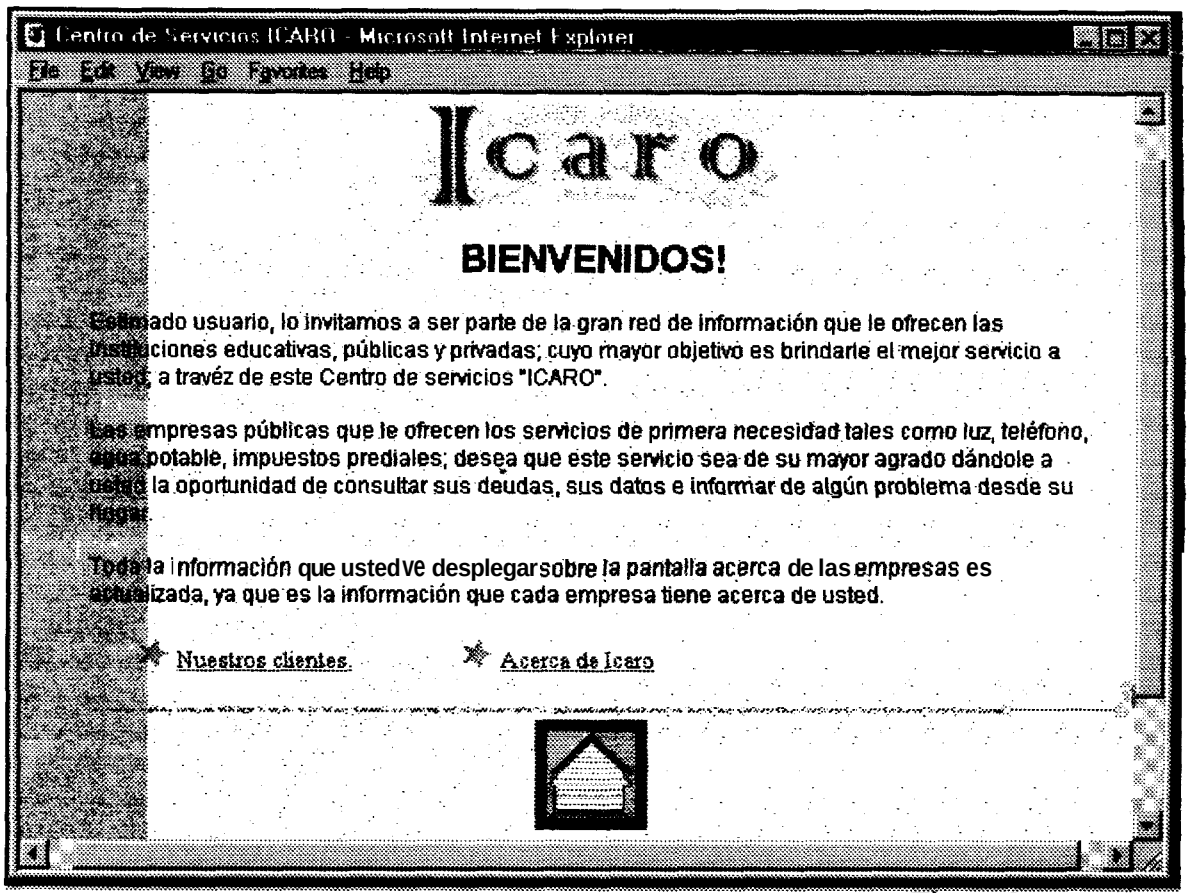


Figura 2

La otra opción en esta pantalla y en la anterior es **Acerca de Icaro** en el cual se da una breve explicación sobre ICARO y sus desarrolladores.

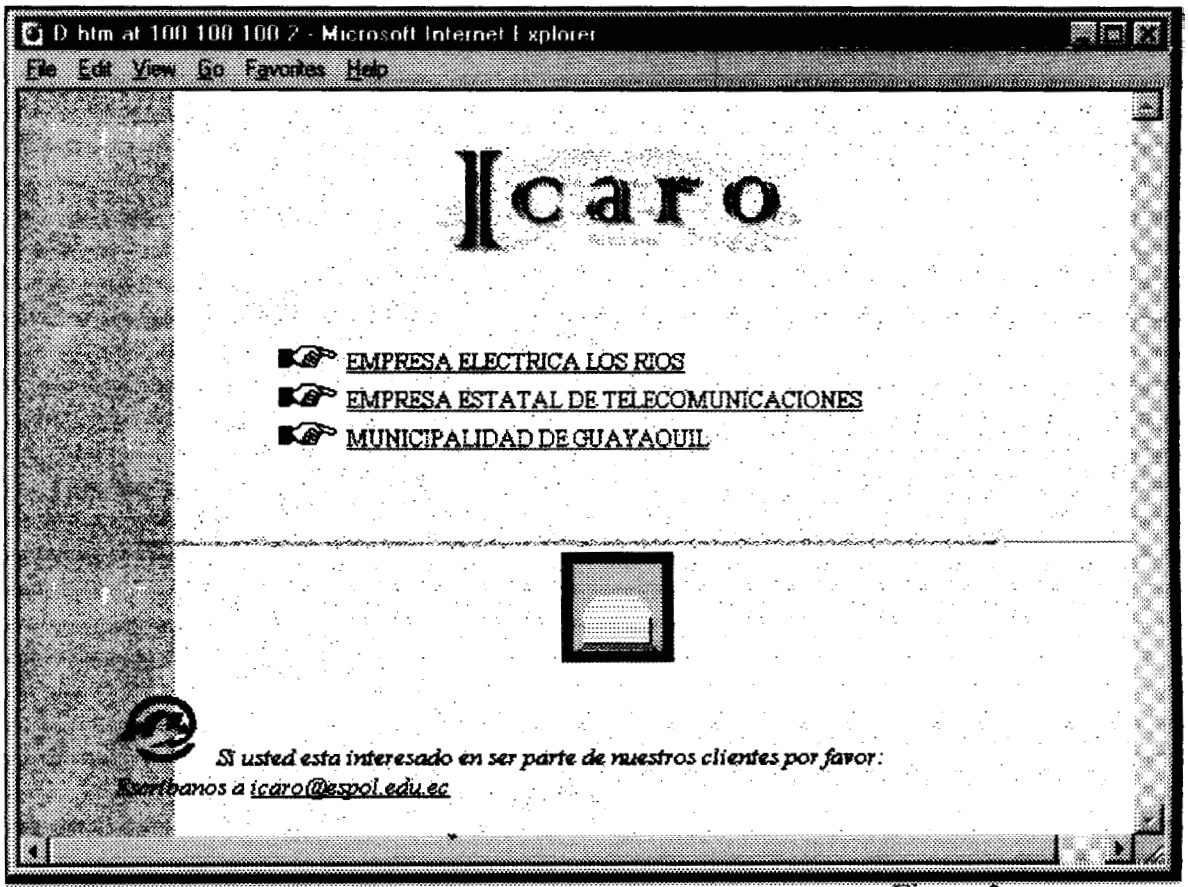


Figura 3

La Figura. 3 nos muestra la pantalla resultante de la acción anterior

En el estado inicial de publicación de este manual, el Centro de Servicios ICARO consta sólo de tres empresas tal como lo muestra la Figura. **3.** En adelante la cantidad de empresas presentes en esta pantalla podrán aumentar dependiendo de que estas se subscriban a nuestro servicio, cuando exista un ítem nuevo en una de las páginas se mostrará un gráfico como el siguiente:

Para efectos de este manual haremos la demostración de una de **las consultas** por cada empresa.

Empresa Eléctrica de Los Ríos.-

Al seleccionar esta opción se nos muestra una información de las consultas que podemos hacer a la Base de Datos de esta empresa, la pantalla resultante lo muestra la Figura. **003.**

Figura **4**

Podemos hacer consultas sobre las deudas que tenemos en esta empresa, estas consultas pueden ser hechas de diferentes maneras. Para consultas similares pero con método de búsqueda diferente el resultado final será el mismo esto se lo hace para dar mayores facilidades.

En esta página tenemos dos tipos de consultas: de DEUDAS y de CUENTAS.

Consultas de DEUDAS se refiere a las deudas pendientes con la empresa y de CUENTAS se refiere a la identificación del cliente en la empresa y toda la información referente a la misma. Estas consultas pueden ser hechas por el nombre del dueño de la cuenta o por su número de cédula.

De esta manera cada usuario del servicio ofrecido por esta empresa podrá saber la cantidad que adeuda y la fecha de pago máxima sin tener que acercarse a la oficina de la empresa a preguntar.

Seleccionemos la opción **CONSULTA DE CUENTA POR CEDULA** como resultado de esto tenemos una pantalla tal como lo muestra la Figura. 5

En el campo de **texto** que se muestra en esta pantalla se escribe el número de cédula del abonado a la empresa y se presiona el botón **Ejecutar Consulta** para obtener el resultado de la misma. La Figura. ***6*** muestra la pantalla resultante de esta acción.

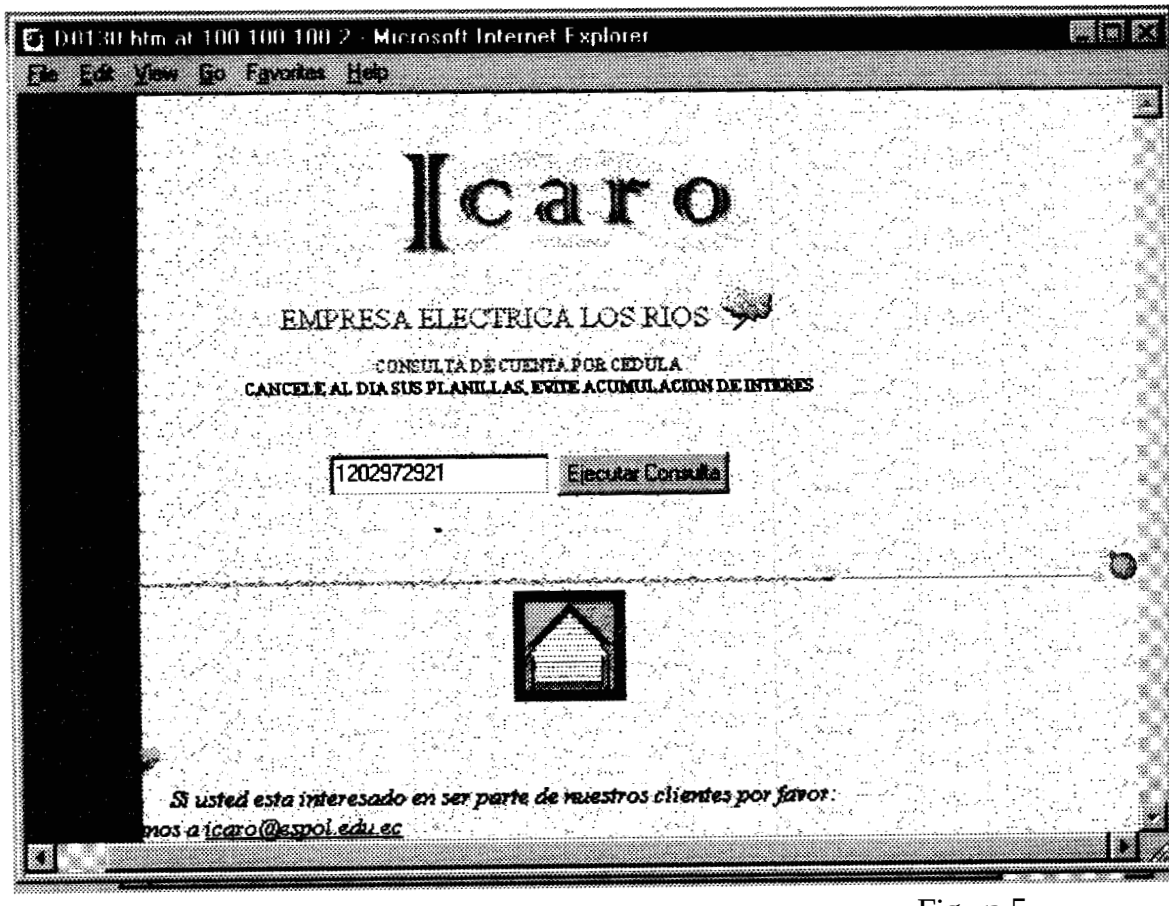


Figura 5

Esta pantalla contiene la(s) cuenta(s) que tiene el abonado con la empresa, esta información está clasificada en nombre, no. cuenta y categoría.

Figura 6

Si el Browser nos permite podemos regresar a la pantalla anterior o en caso contrario podemos enlazarnos a la página principal del Centro de Servicios ICARO por medio del gráfico ubicado en la parte inferior de la página (HOME).

Una vez en las páginas de las empresas vamos a escoger la empresa de telecomunicaciones para ver otro tipo de consulta.

Empresa Estatal de Telecomunicaciones.-

Al seleccionar esta opción se nos muestra una información de las consultas que podemos hacer a la Base de Datos de esta empresa, la pantalla resultante lo muestra la

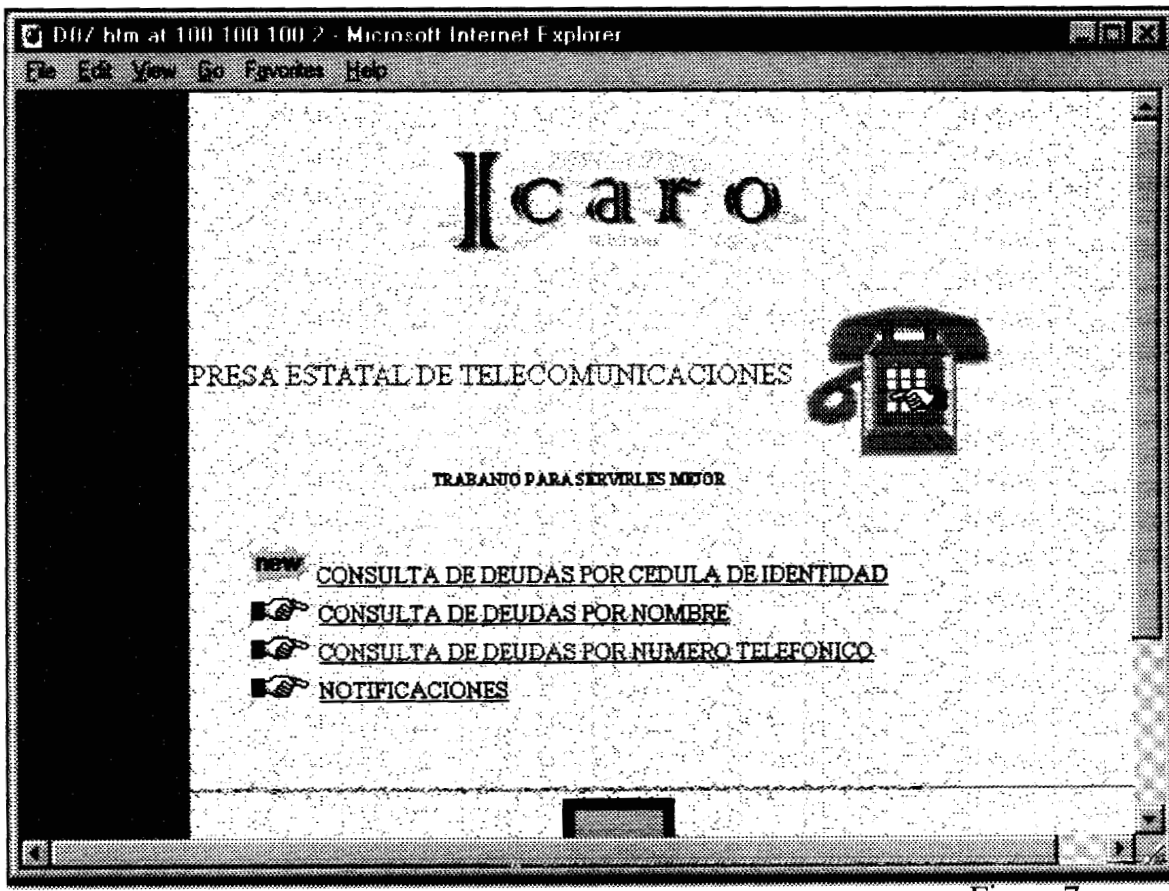


Figura 7

En esta página tenemos consultas: de DEUDAS. Las consultas de DEUDAS se refiere a las deudas pendientes con la empresa. Esta consulta puede ser hecha por el nombre del abonado, por su número de cédula *o* por número telefónico.

Escojamos la opción **CONSULTA DEUDA POR NOMBRE** y como resultado de esta acción tenemos una pantalla tal como lo muestra la Figura. 8

Figura 8

En el campo de texto que se muestra en esta pantalla se escribe el nombre del abonado a la empresa y se presiona el botón Ejecutar Consulta para obtener el resultado de la misma. La Figura. **9** muestra la pantalla resultante de esta acción.

Las consultas también las podemos hacer utilizando el comodín " *Yo* ", **para** lo cual escribimos parte de la cadena de búsqueda seguida por el comodín, por ejemplo:

" CADENA DE BUSQUEDA% "

En la pantalla resultante de la acción anterior tenemos la información de deudas del abonado a la empresa, si utilizamos el comodín tenemos todos los abonados que relacionados con la cadena de búsqueda. La información esta clasificada en nombre, valor de la deuda y fecha máxima de **pago.**

Figura 9

Si el abonado no tiene deudas no se muestra información y podemos hacer navegar a otra consulta o empresa.

Municipalidad de Guayaquil.-

Al seleccionar esta opción se nos muestra una información de las consultas que podemos hacer a la Base de Datos de esta empresa, la pantalla resultante lo muestra la Figura. **10**

Figura 10

En esta página tenemos consultas: de DEUDAS. Las consultas de DEUDAS se refiere a todo tipo de deudas pendientes con la empresa. Estas consultas puede ser hecha por el nombre del dueño del predio o nombre de la compañía, por código catastral y por número de RUC.

Cualquiera de estas consultas mostrará la información necesaria que busca el cliente.

El Centro de Servicios ICARO tiene como finalidad principal dar a los abonados de las empresas Telefónicas, Eléctricas y Municipalidades un acceso fácil y rápido a las deudas y datos de cada uno de ellos en estas empresas.

La información que es mostrada en pantalla es una información en línea, es decir, cualquier cambio realizado en una de las empresas aquí mostradas se reflejara inmediatamente en nuestro Centro de Servicios.

Esperamos que este Sitio en la Internet sea muy visitado por Ud. y si tiene sugerencias háganos llegar su mail a la siguiente dirección icaro@espol.edu.ec

Anexo 3

Manual de Uso: WebServer ICARO 1.0

Bienvenido a ICARO 1.0 WebServer. ICARO 1.0 le permitirá prestar un servicio eficiente para que sus clientes puedan mantener una completa información de los datos de sus empresas en la INTERNET.

Este manual introduce los conceptos básicos para el manejo y administración de ICARO 1.0 así como la generación de archivos de control del **mismo.**

Esperamos que el material aquí expuesto sea lo suficientemente claro para que Ud. opere correctamente el WebServer ICARO.

Conceptos Básicos



Requerirnientos de Hardware.-

El funcionamiento óptimo del WebServer ICARO depende del tipo de computador sobre el cual será instalado. Los requerimientos básicos **para** un buen funcionamiento son:

- Un computador Pentium de **133 Mhz**
- 16 Mb de RAM
- Un módem de 28.8 Mbps
- Adaptador de Red de 10Mbps

Si Ud. no cuenta cŏn una estructura mínima de hardware como la descrita el funcionamiento de ICARO será poco eficiente.

Requerimientos de Software.-

Los requerimientos en cuanto a software para el funcionamiento de ICARO son:

- Windows'95, Windows NT Workstation **4.0** ó Windows NT Server 4.0 como Sistema Operativo.

- Un Browser (Netscape, Internet Explorer, u otro) [opcional].
- No tener otro servicio de WebServer habilitado.
- Instalar un administrador de ODBC.
- Instalar los controladores de Base de Datos a las que va a accesar ICARO.

Instalación.-

Una vez que se ha comprobado que el computador en el cual va ha funcionar ICARO cumple los requisitos de hardware y software descritos anteriormente, procedemos a la instalación.

Los pasos para la instalación son:

1. Introduzca en el *drive 3%* el disco *#1* etiquetado como **Inicio** de Instalación.
2. Seleccionar la opción **Eiecutar** del menú **Inicio** de Windows y en la caja de texto escriba **a:\instalar** ver fig. 000.

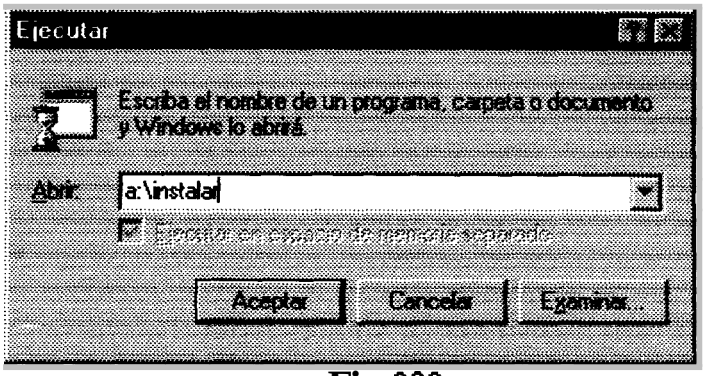


Fig. 000

3. Presione el botón **Aceptar** de la ventana para que se ejecute la instrucción.
4. Ingrese los discos conforme el programa de instalación los vaya requiriendo.
5. Una vez terminada la instalación de ICARO debe reiniciar su equipo.

Como resultado de la instalación se ha creado la carpeta ICARO en la raíz del disco duro y en su interior todos los archivos y carpetas necesarias **para** ICARO.

En caso de ocurrir un error durante la instalación de ICARO, el programa de instalación borrará todos los archivos relacionados con la **misma** y le permitirá volver a intentar la instalación de ICARO.

En caso de persistir el error contáctese con nuestro departamento de soporte al cliente mailto:icaro@espol.edu.ec.

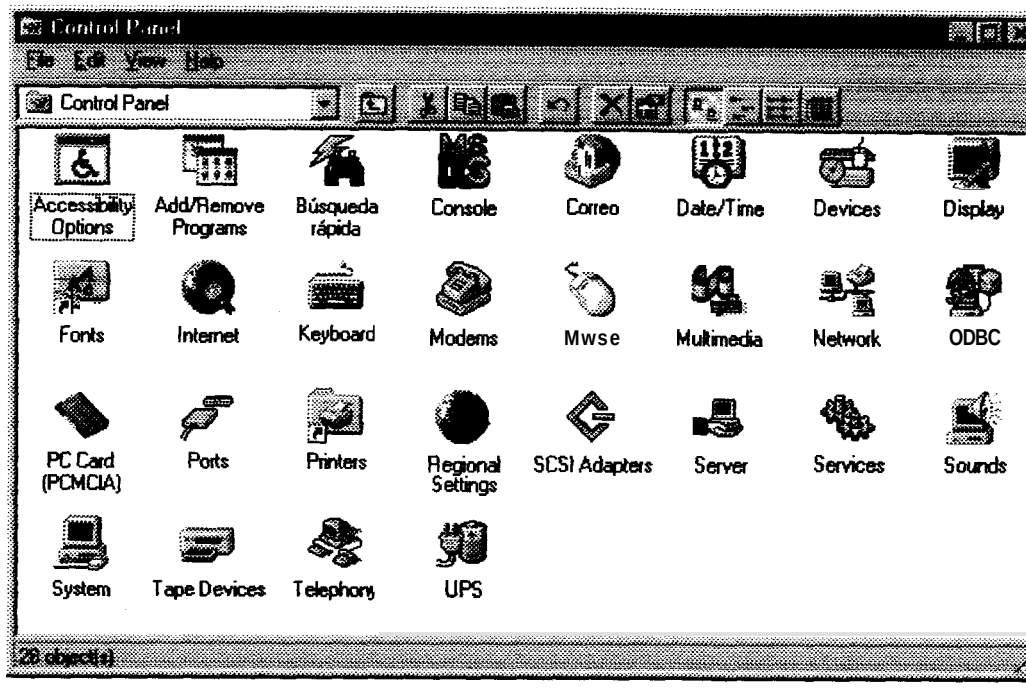
Funcionamiento

Configuraciones

Una vez terminada la instalación de ICARO y esta haya sido exitosa, procedemos a configurar los controladores de las Base de Datos a las cuales se va a accesar.

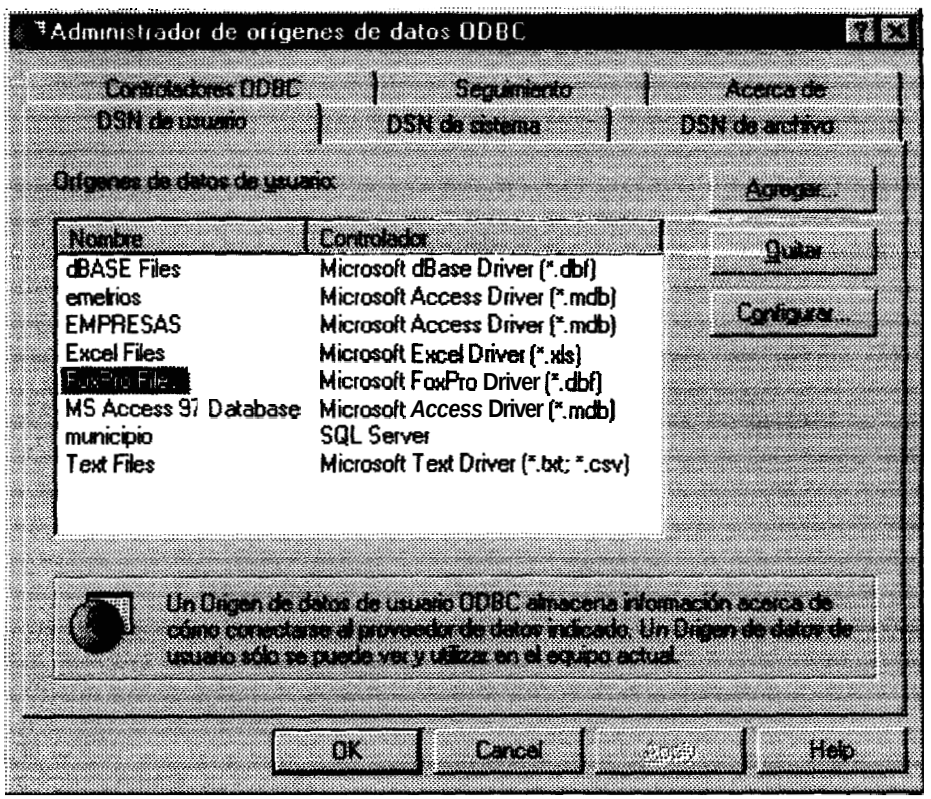
Esta configuración la hacemos desde el administrador de ODBC de Windows, los pasos a seguir para configurar estos controladores son:

1. Seleccionar la opción **Panel de Control** en el ítem **Configuración** del menú **Inicio** de Windows.
2. En la ventana que se genera como resultado a la acción anterior fig. 001 seleccione el ítem titulado como ODBC.



3. El resultado de la acción anterior muestra la ventana de configuración de los controladores **fig.** 002. **Para** más información sobre la configuración de estos controladores consulte la documentación de Windows.

Fig. *002*



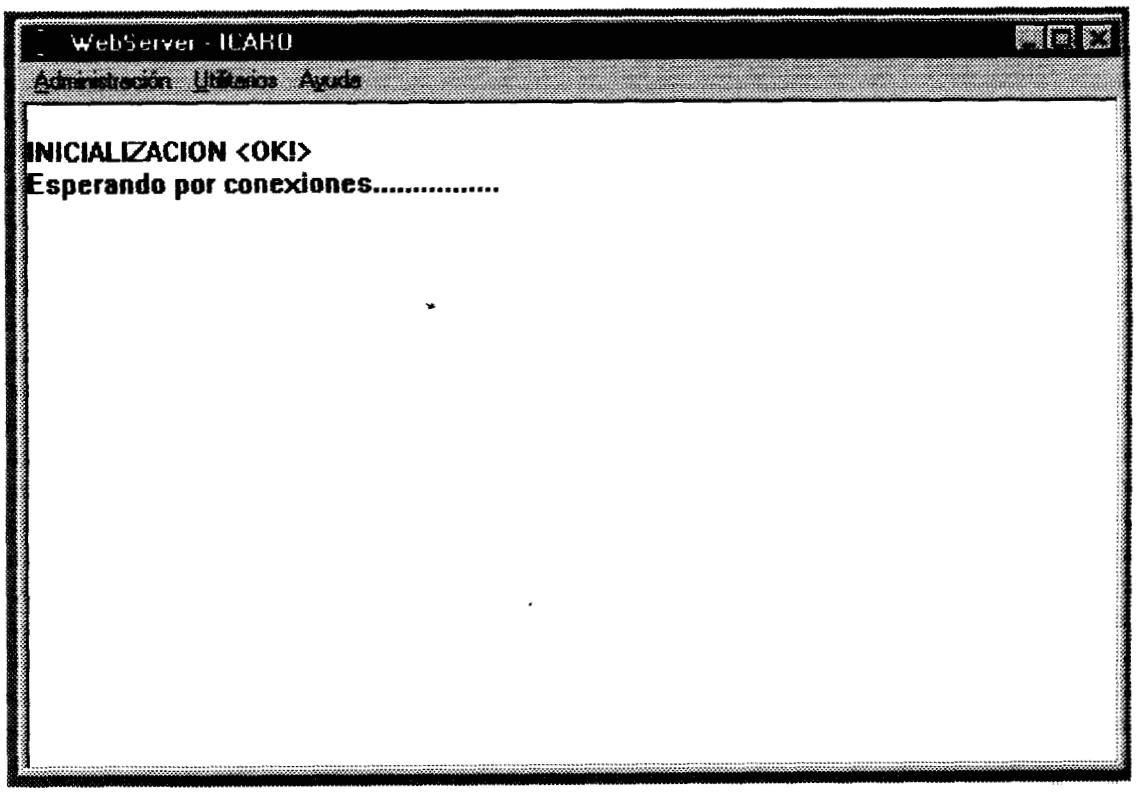
4. Una vez configurados los controladores de Base de Datos, se debe detener los servicios de WebServer que trae Windows (en caso de haber sido instalados). Para ello refiérase a la opción **Microsoft Internet Server** del ítem **Pro——as** del menú **Inicio.**

Inicio

Una vez que se han realizado las configuraciones necesarias, estarnos listos para iniciar ICARO para esto seleccionarnos el archivo ejecutable que se encuentra **en** el directorio **C:\ICARO\icaro.exe**.

Como resultado de la ejecución de este archivo se activa el WebServer ICARO y se nos muestra una pantalla similar **al de** la fig. 003

Fig. 003



Esta pantalla nos indica que ICARO esta listo para atender requerimientos de los usuarios del servicio.

ICARO nos da la posibilidad de cierta manipulación por medio de un menú que tiene incorporado, los ítems de este menú se detalian en el siguiente punto.

Menú

El menú de ICARO esta compuesto principalmente por tres ítems, a saber:

Administración

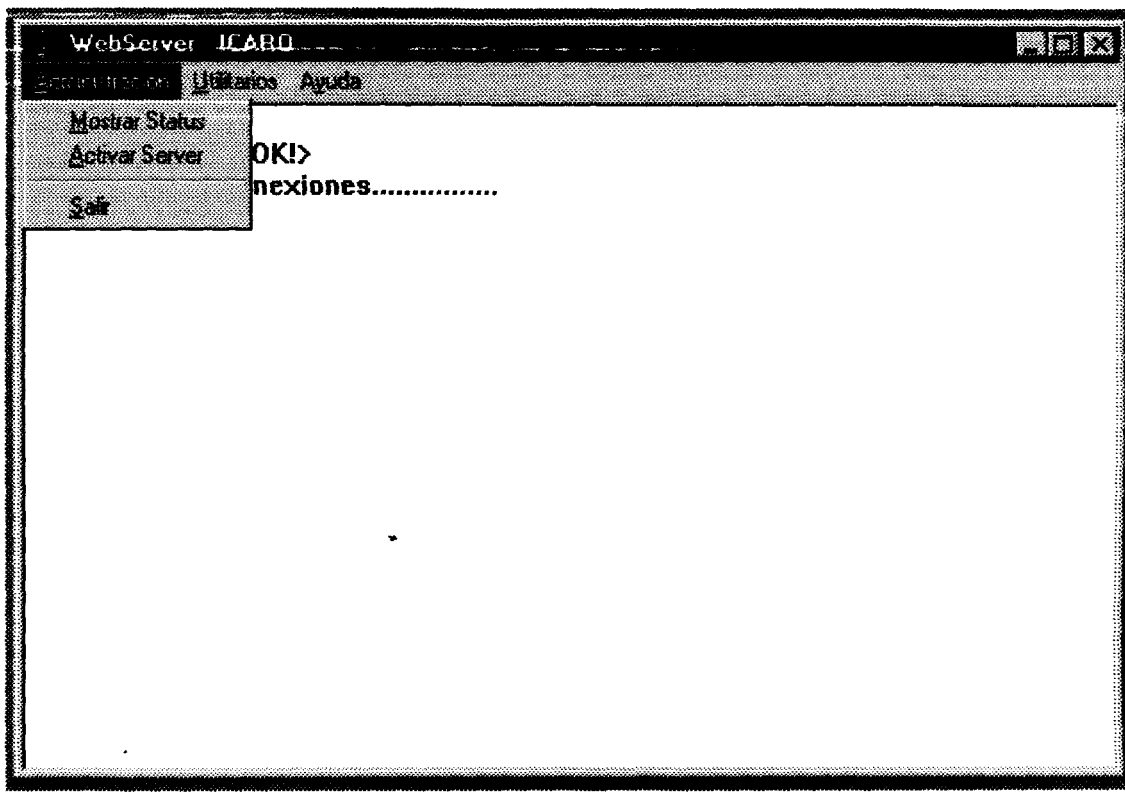
Esta opción permite variar los estados por omisión de ICARO sin necesidad de cerrar el servicio fig. 004.

Mostrar Status

Si esta opción esta activada por omisión y **permite** monitorear en pantalla todos los requerimientos hechos por los usuarios, si esta opción es desactivada no se monitoreará los requerimientos del usuario.

Activar Server.-

ICARO por omisión se inicializa en estado activo, es decir, listo para recibir conexiones. Sin embargo ICARO nos permite cambiar este estado y desactivar el servicio sin necesidad de que se cierre, esto permitirá dar mantenimiento a alguna Base de Datos.



Utilitarios

Ver Archivo Log

Esta opción permite visualizar un archivo LOG el cual contiene la dirección IP del usuario que **hizo** un requerimiento, la fecha, hora y los archivos requeridos. Estos archivos son permanentes y se encuentran en C:\ICARO\LOG\ .

Estos archivos solo podrán ser eliminados explícitamente por el usuario.

Ayuda

Contenido

Presenta una ayuda acerca de las opciones mostradas en el menú y de como se deben utilizar.

Acerca de

Da una información sobre los Ingenieros que desarrollaron el WebServer ICARO versión 1.0

Errores

Un posible error en ICARO sería cuando haya una inicialización incorrecta. Cuando esto ocurre se muestran los siguientes mensajes fig. 005 y fig. 006.

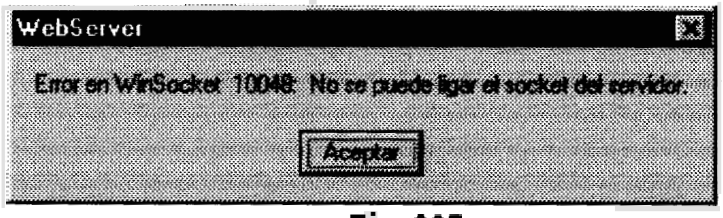


Fig. 005

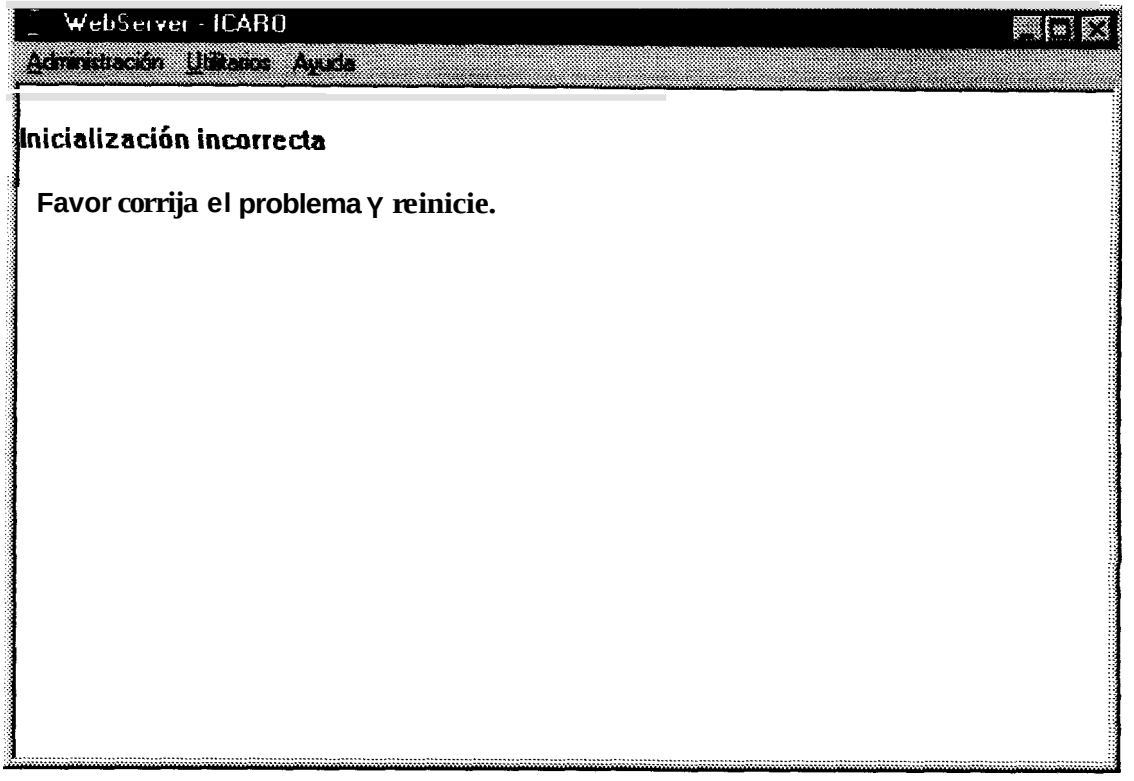


Fig. 006

Este problema se presenta cuando hay **otro** WebServer ejecutándose en la misma máquina y el sistema operativo de la **misma** es Windows NT Server.

Para solucionar el problema refiérase **al** punto **4** del ítem configuraciones en la sección Funcionamiento.

ICARO 1.0 es el único WebServer a nivel nacional que nos permite el acceso a múltiples Bases de Datos de diferentes tipos y en diferentes plataformas, **sin** que esto haga que su administración sea complicada.

El manejo de la Base de Datos propia del sistema ICARO se lo realiza en un sistema denominado EUREKA y para su manejo refiérase a su manual de uso, adjunto a este manual.

Esperamos que nuestro sistema ICARO 1.0 sea completamente de su agrado, cualquier problema con el sistema consulte con nuestros Ingenieros icaro@espol.edu.ec.

Anexo 4

Manual de Uso: EUREKA

Bienvenido a EUREKA, esta aplicación esta orientada a la administración y mantenimiento de la Base de Datos de ICARO WebServer.

Este manual intenta servir de ayuda al administrador de **ICARO en** el ingreso, modificación y consulta de las empresas que requieren de servicio.

Esperamos que el material aquí expuesto preste toda **la** claridad y facilidad para el uso de EUREKA.

Conceptos Básicos

Requerimientos de Hardware.-

El funcionamiento óptimo de EUREKA depende del tipo de computador sobre el cual será instalado. Los requerirnientos básicos para un buen funcionamiento son:

- Un computador Pentium de **133** Mhz
- 16 Mb de RAM
- Adaptador de Red de 10Mbps

Si Ud. no cuenta con una estructura mínima de hardware como la descrita el funcionamiento de EUREKA será poco eficiente.

Requerimientos de Software

Los requerimientos en cuanto a software **para el** funcionamiento de EUREKA son:

- Windows'95, Windows NT Workstation 4.0 ó Windows NT Server 4.0 como Sistema Operativo.
- Instalar un administrador de ODBC.
- Instalar **los** controladores de Base de Datos a las que va a accesar EUREKA..

Instalación

Una vez que se ha comprobado que el computador en el cual va ha funcionar EUREKA cumple los requisitos de hardware y software descritos anteriormente, procedemos a la instalación.

Los pasos para la instalación son:

6. Introduzca en el drive 3½ el disco #1 etiquetado como Inicio de Instalación.
7. Seleccionar la opción **Eiecutar** del menú **Inicio** de Windows y en la caja de texto escriba **a:\instalar** ver fig. 000.

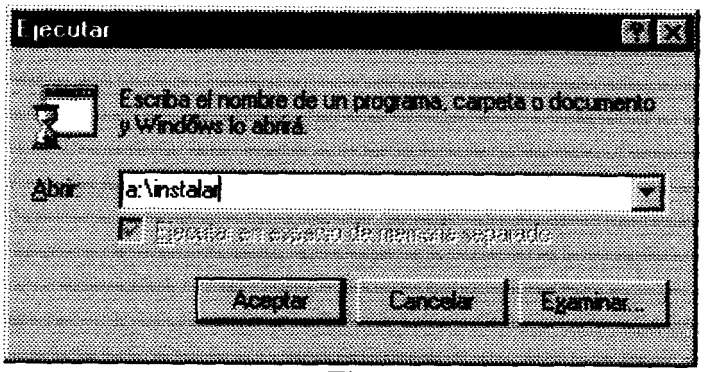


Fig. 000

8. Presione el botón **Aceptar** de la ventana para que se ejecute la instrucción.
9. Ingrese los discos conforme el programa de instalación los **vaya** requiriendo.

10. Una vez terminada la instalación de EUREKA debe reiniciar su equipo.

Como resultado de la instalación se ha creado la carpeta EUREKA en la raíz del disco duro y en su interior todos los archivos y carpetas necesarias para EUREKA.

En caso de **ocurrir** un error durante la instalación de EUREKA, **el** programa de instalación borrará todos los archivos relacionados con la misma y le permitirá volver a intentar la instalación de EUREKA.

En caso de persistir el error contáctese con nuestro departamento de soporte **al** cliente eureka@espol.edu.ec

Funcionamiento

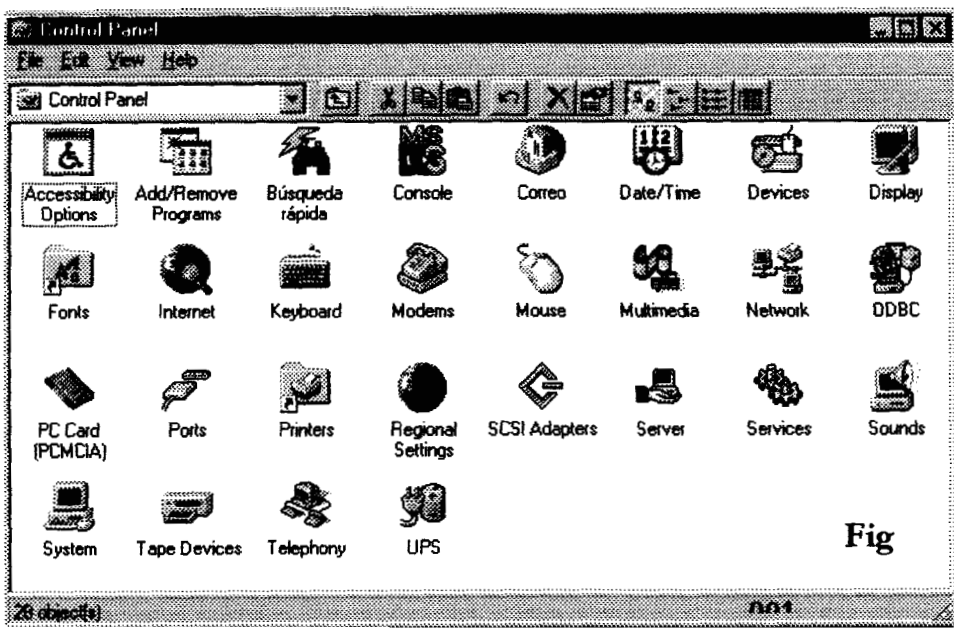
Configuraciones

Una vez terminada la instalación de EUREKA y esta haya sido exitosa, procedemos a configurar los controladores de las Base de Datos a las cuales se va a accesar[1].

[1] Es posible que estas configuraciones ya haya sido hechas en la instalación de ICARO WebServer

Esta configuración la hacemos desde el administrador de ODBC de Windows, los pasos a segur para configurar estos controladores son:

5. Seleccionar la opción **~~Panel de Control~~** en el ítem **~~Configuración~~** del



Fig

menú **Inicio** de Windows.

6. En la ventana que se genera como resultado a la acción anterior fig. 001 seleccione el ítem titulado como ODBC.
7. El resultado de la acción anterior muestra la ventana de configuración de los controladores fig. 002 . Para más información sobre la configuración de estos controladores consulte la documentación de Windows.

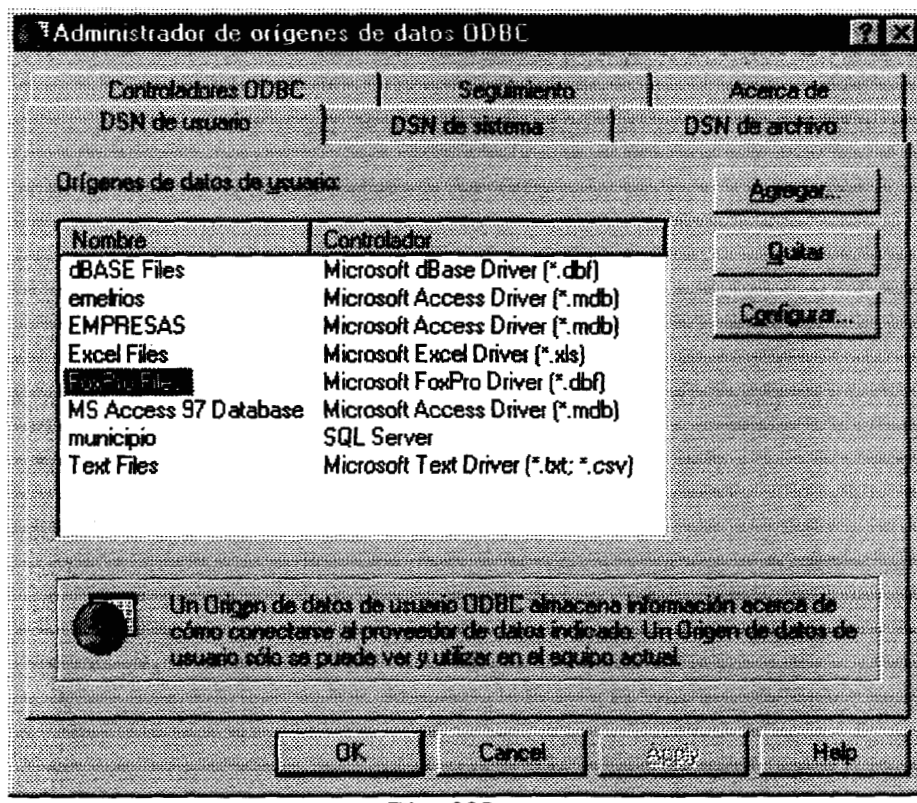

Administrador de orígenes de datos ODBC
Controladores ODBC
Seguimiento
Acerca de
DSN de usuario
DSN de sistema
DSN de archivo
Orígenes de datos de usuario:
Agregar...
Quitar
Configurar...
Nombre
Controlador
dBASE Files
Microsoft dBase Driver (*.dbf)
emetrios
Microsoft Access Driver (*.mdb)
EMPRESAS
Microsoft Access Driver (*.mdb)
Excel Files
Microsoft Excel Driver (*.xls)
FoxPro Files
Microsoft FoxPro Driver (*.dbf)
MS Access 97 Database
Microsoft Access Driver (*.mdb)
municipio
SQL Server
Text Files
Microsoft Text Driver (*.txt; *.csv)
Un Origen de datos de usuario ODBC almacena información acerca de cómo conectarse al proveedor de datos indicado. Un Origen de datos de usuario sólo se puede ver y utilizar en el equipo actual.
OK
Cancel
Apply
Help

Fig. 002

Inicio

Una **vez** que se han realizado las configuraciones necesarias, estamos listos **para** iniciar EUREKA para esto seleccionamos el archivo ejecutable que se encuentra en el directorio **C:\EUREKA\eureka.exe**.

Como resultado de la ejecución de este archivo ***se*** inicia EUREKA y se nos muestra una pantalla *similar* **al** de la fig. **003**

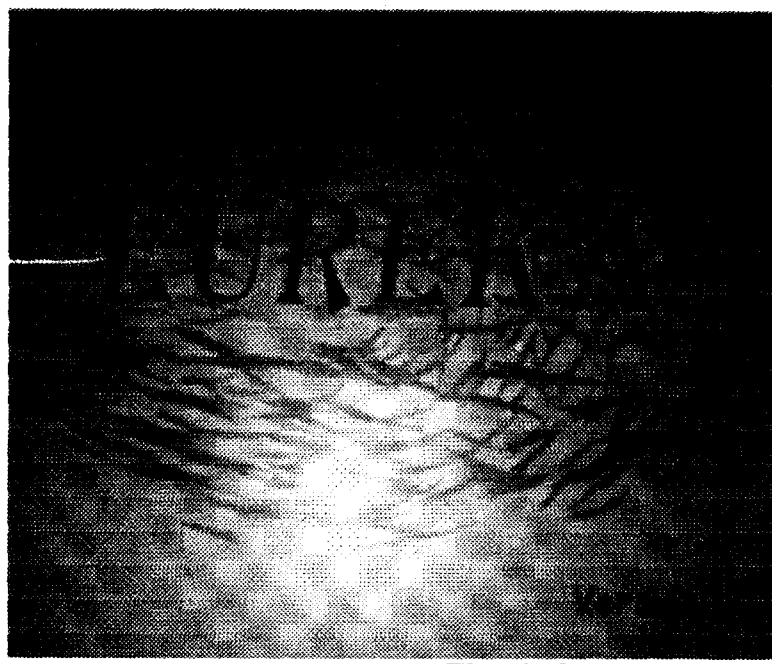


Fig. 003

Esta es la pantalla inicial de ingreso a EUREKA la cual tarda uno pocos segundos mientras se prepara el sistema para trabajar. A continuación se muestra en pantalla el menú mediante el cual vamos a administrar la base de datos de ICARO WebServer fig.004.

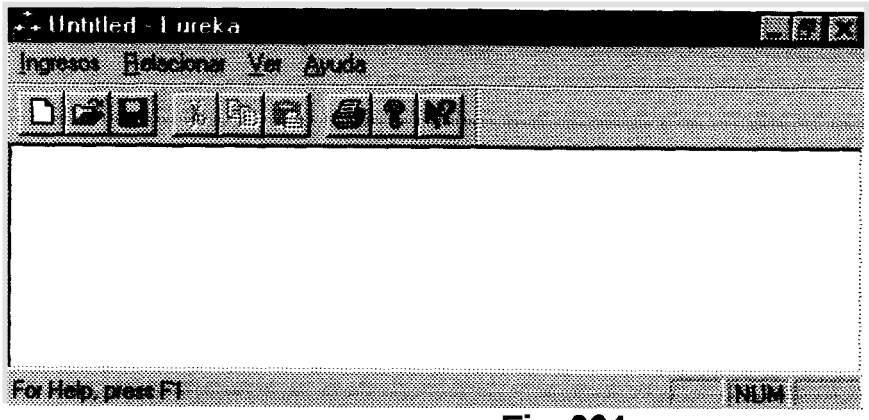


Fig. 004

Menú.-

El resultado de la acción anterior muestra el menú de EUREKA en la pantalla, cada uno de los ítems cumple una función específica y lo procedemos a explicar a coontinuación:

tngresos

Al seleccionar esta opción del menú se muestra una pantalla tal como lo muestra la fig. 005.

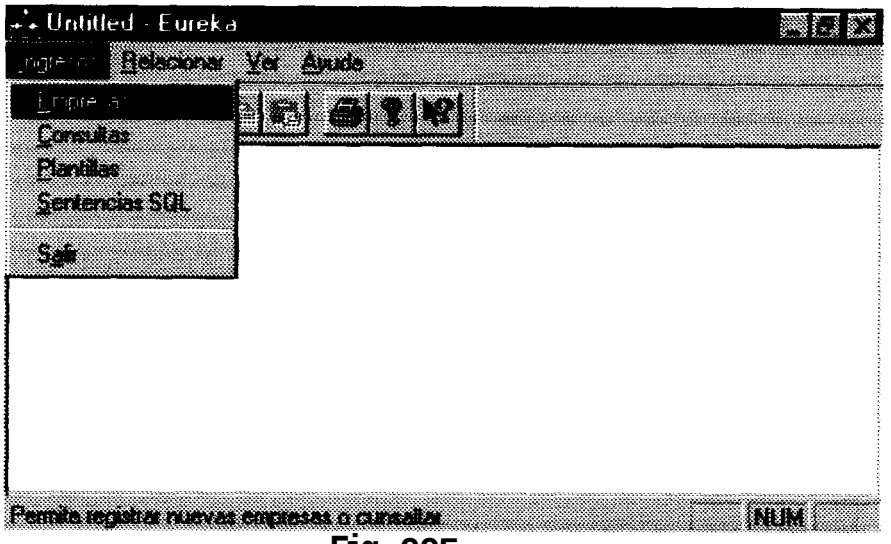


Fig. 005

En este sub **menú** se ingresan los datos de las Empresas ***fig.006,*** Consultas ***fig.007,*** Plantillas fig.008 y Sentencias SQL ***fig.009.***

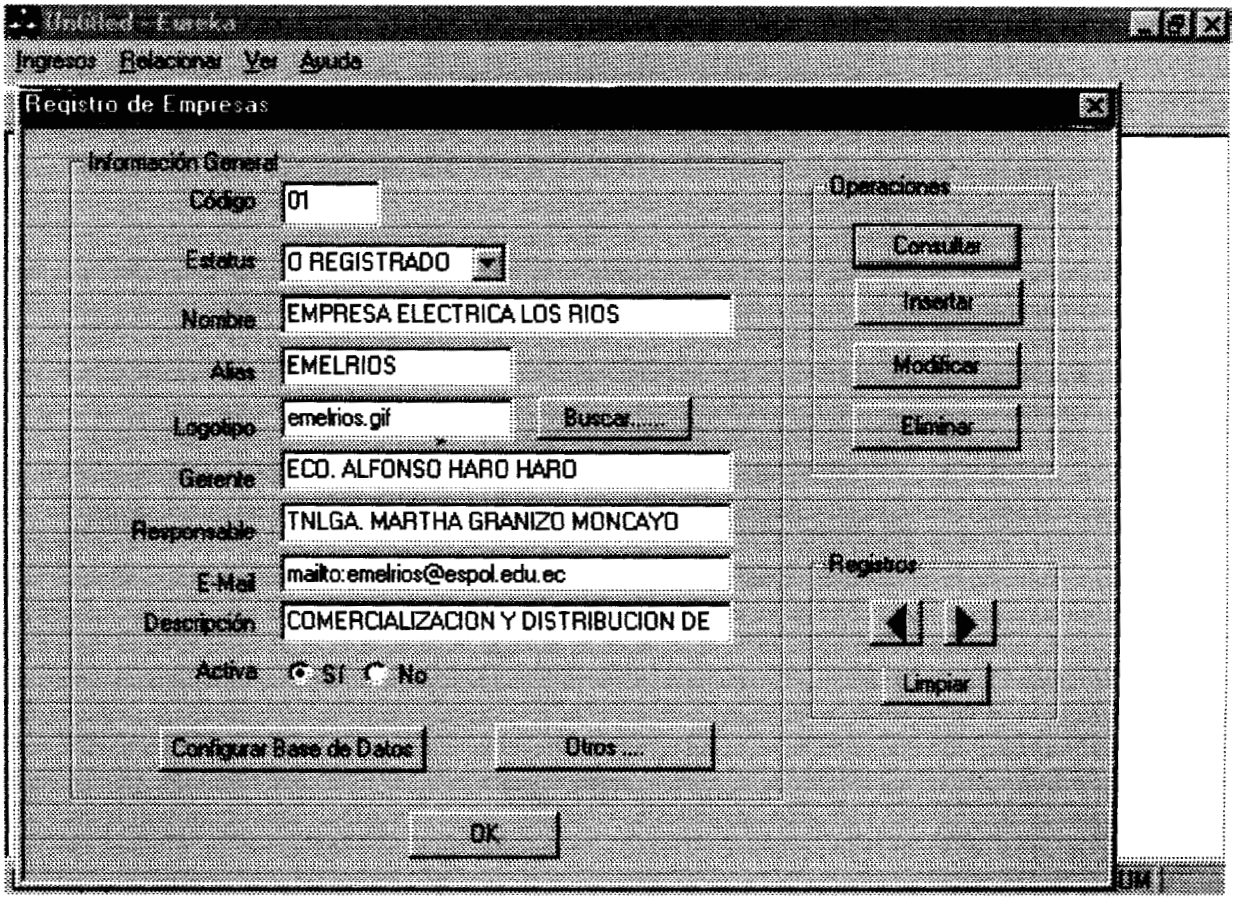
La fig.006 se muestra que tenemos la opción de hacer consultas, modificaciones, eliminaciones e ingresos de empresas, además de la opción de activar esta empresa para que aparezca en el Centro de Servicios ICARO. En esta ventana relacionamos la empresa con el Origen de Datos de la Empresa Cliente.

La *fig.007* nos muestra una pantalla de ingreso del nombre de las consultas que las empresas ofreceran a los usuarios de sus servicios.

La ***fig.008*** muestra una pantalla en la cual el usuario **podrá** escoger la plantilla en la cual se mostrarán sus datos.

La fig. **009** nos muestra la pantalla en la ***cual*** relacionamos la sentencia SQL coon el nombre de la consulta de una empresa predeterminada.

Fig.006



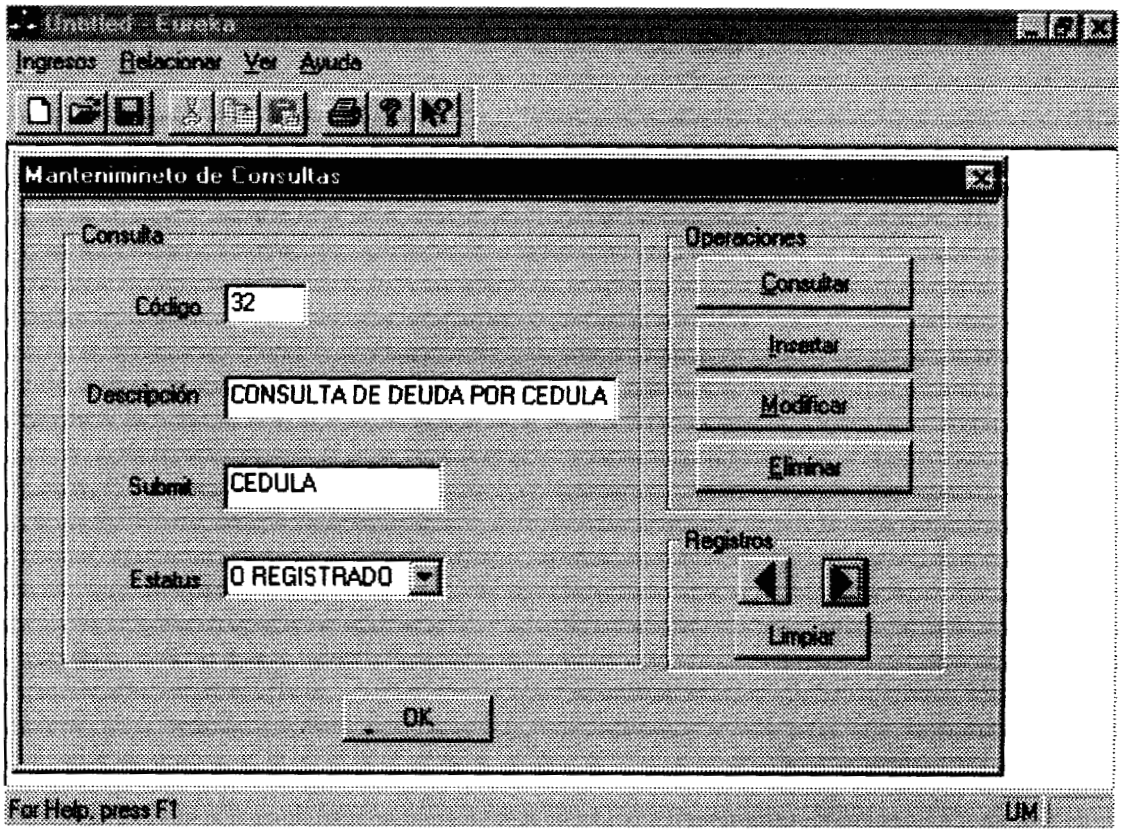

Untitled - Eureka
Ingresos Relacionar Ver Ayuda
Mantenimineto de Consultas
Consulta
Código 32
Descripción CONSULTA DE DEUDA POR CEDULA
Submit CEDULA
Estatus 0 REGISTRADO
Operaciones
Consultar
Insertar
Modificar
Eliminar
Registros
Limpiar
OK
For Help, press F1
UM

Fig. 007

Fig. 008

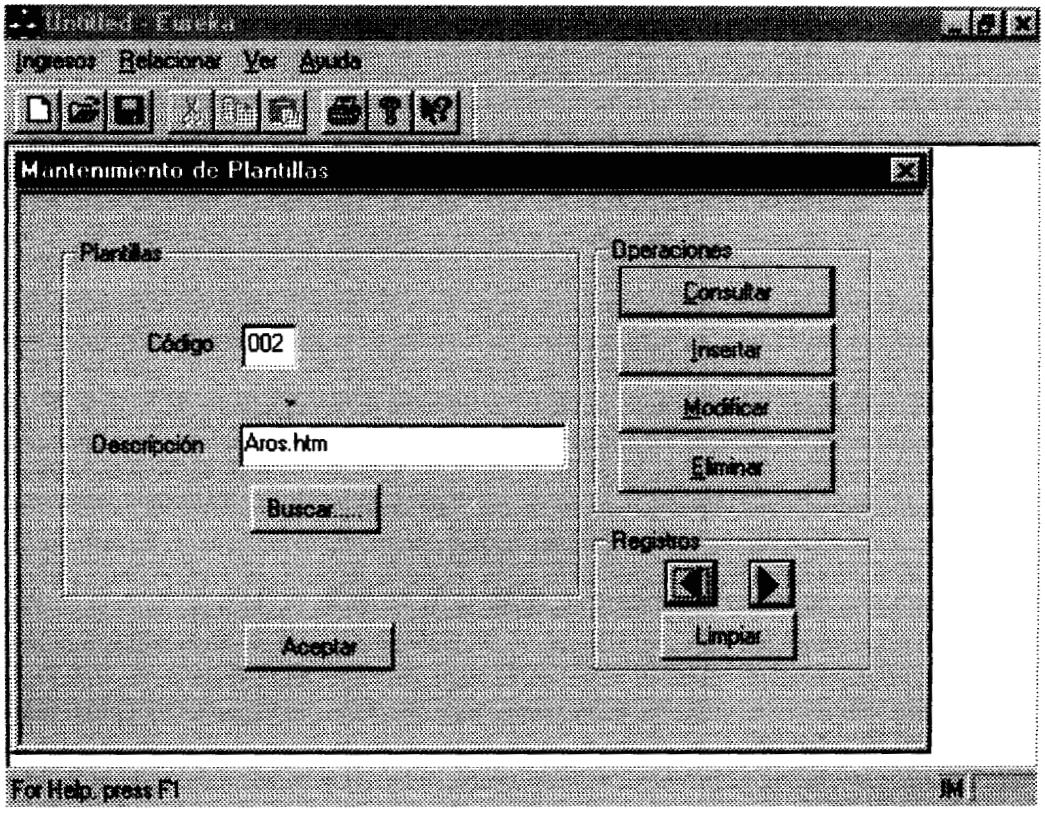
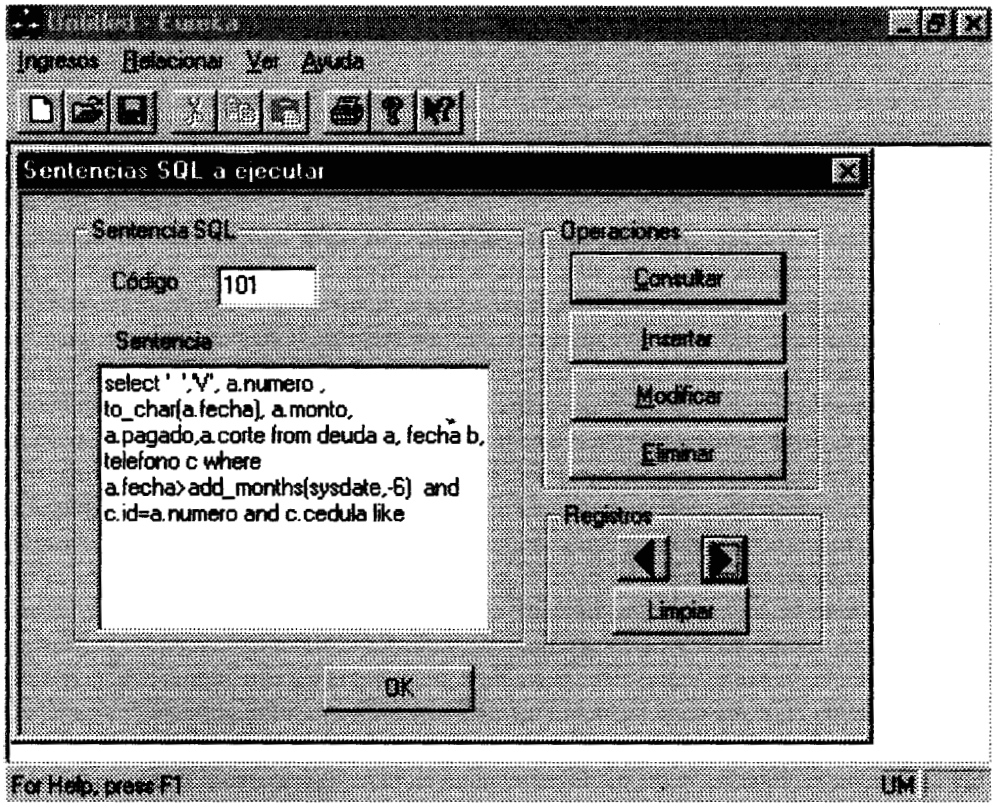
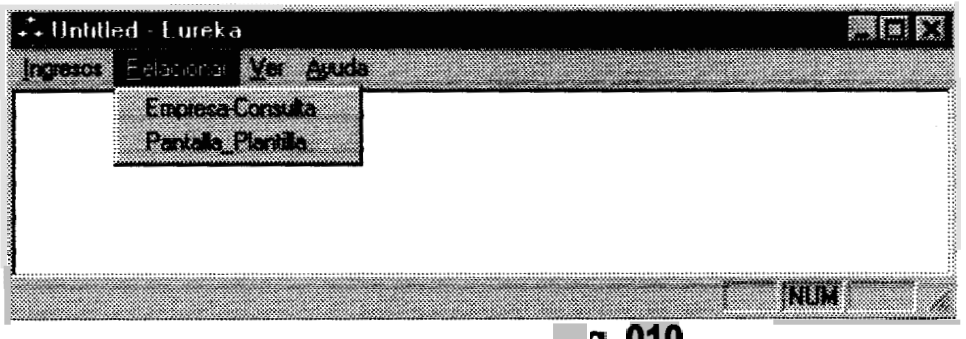

Ingresos Relacionar Ver Ayuda
Mantenimiento de Plantillas
Plantillas
Código
002
Descripción
Aros.htm
Buscar....
Aceptar
Operaciones
Consultar
Insertar
Modificar
Eliminar
Registros
Limpiar
For Help, press F1

Fig. 009


Ingresos Relacionar Ver Ayuda
Sentencias SQL a ejecutar
Sentencia SQL
Código 101
Sentencia
select ' ','V', a.numero ,
to_char(a.fecha), a.monto,
a.pagado,a.corte from deuda a, fecha b,
telefono c where
a.fecha>add_months(sysdate,-6) and
c.id=a.numero and c.cedula like
Operaciones
Consultar
Insertar
Modificar
Eliminar
Registros
Limpiar
OK
For Help, press F1
UM

Relacionar.-

Al seleccionar esta opción del menú se muestra una pantalla tal como lo muestra la ***fig.*** 010.



g 010

El ítem Empresa-Consulta nos permite relacionar el nombre de la empresa **con** una consulta definida anteriormente fig.011 y el ítem Pantalla-Plantilla nos permite fig.012.

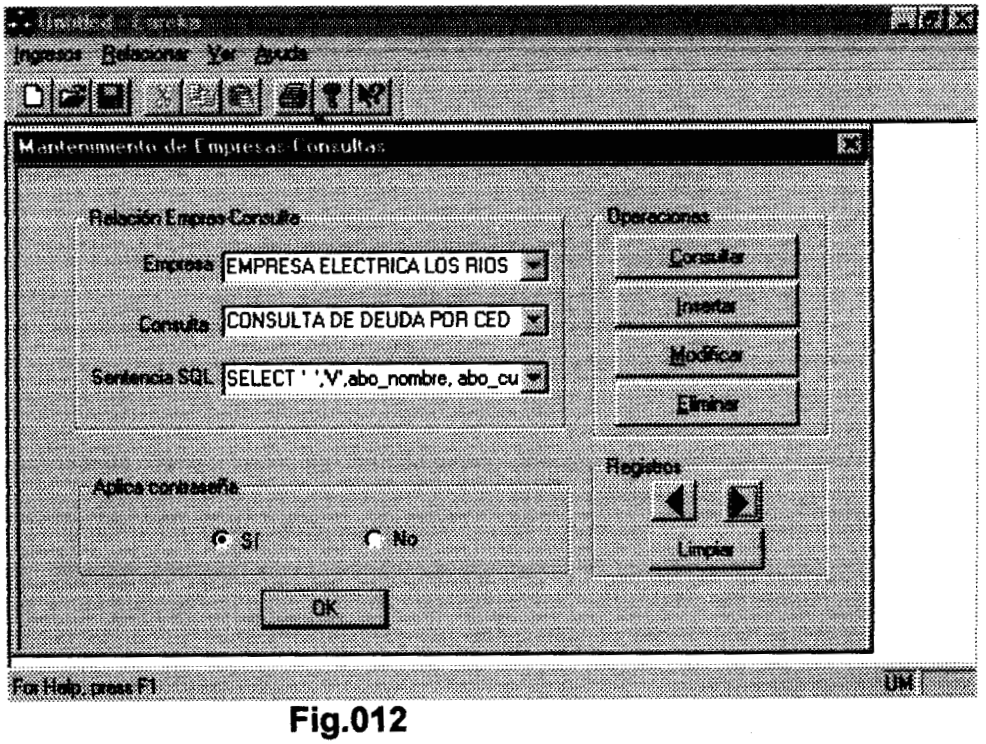


Fig.012

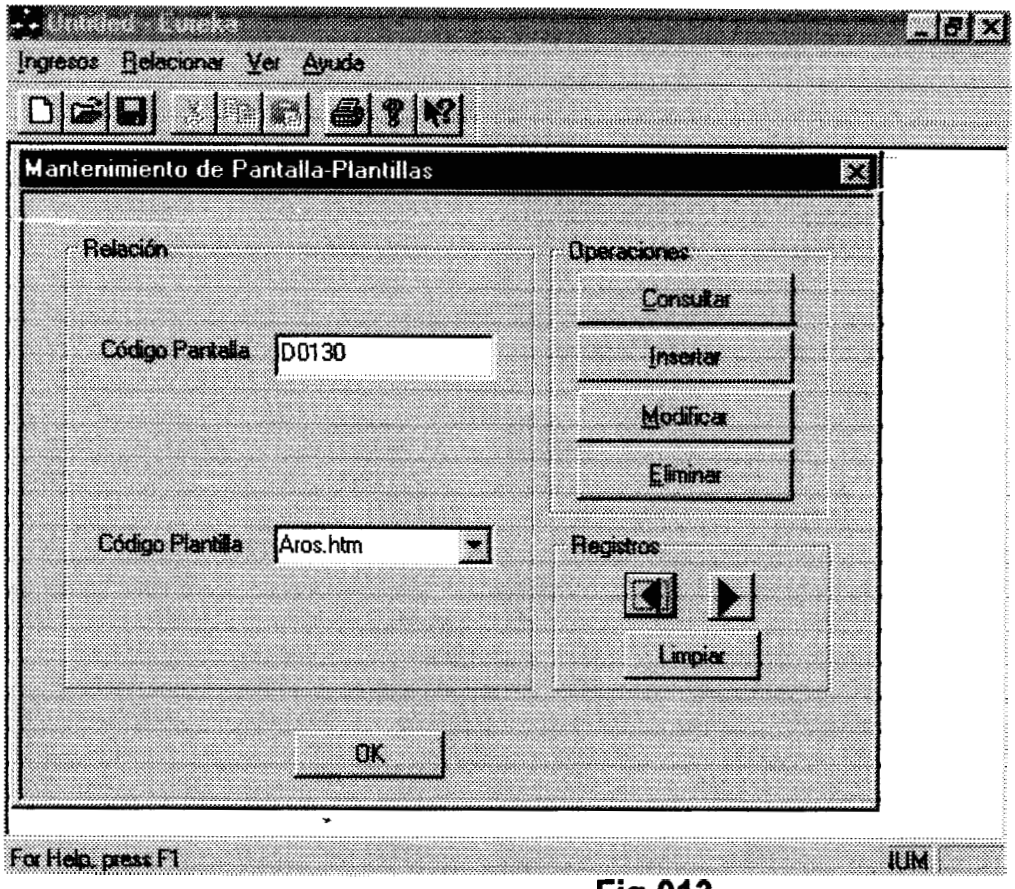


Fig.013

Eureka constituye un software adjunto a ICARO WebServer, este se orienta a la administración y mantenimiento de la Base de Datos del mismo.

Eureka presta toda las facilidadas para su administración de tal forma que la persona que opere el mismo no tiene que ser un experto en computación ni en sentencias SQL.

Cualquier problema con la aplicación comuniquese con nuestro departamento técnico eureka@espol.edu.ec