59

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES

CAMPUS ARAGON

"IMPLEMENTACION DE UN SISTEMA INFORMATICO PARA EL CONTROL DE BECARIOS PARA LA DIRECCION GENERAL DE DIVULGACION DE LA CIENCIA (DGDC)"

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A :

MARCO ANTONIO SALAZAR FRANCO

ASESOR DE TESIS:
M.EN C. MARCELO PEREZ MEDEL

MEXICO, 2002

Gracias por darme la vida y permitirme conocerte.

Gracias por hacerme sentir que siempre estás a mi lado.

Gracias por escuchar mis problemas.

Gracias por estar pendiente de cada momento de mi vida.

Gracias por dar alivio a mis angustias

y consuelo a mi alma.

Gracias por permitirme sentirte en mí,

cuando más te necesito.

Gracias MAMÁ

A Dios, por prestarme vida y permitirme tener la familia y los amigos que he conseguido a lo largo de ella...

A mi madre, por su amor, apoyo y sacrificios que han sido indiscutiblemente parte importante para que me encuentre en el lugar en el que estoy...

A mi padre, cuyos consejos y ejemplo me han ayudado a distinguir lo bueno y lo malo y con ello formarme como persona...

A mis hermanos, por soportarme, comprenderme y ayudarme durante mis estudios y durante toda mi vida...

A mis amigos, que gracias a Dios son muchos y verdaderos, por su apoyo, sus observaciones y críticas constructivas sobre mi trabajo y mi persona...

A la coordinadora del área de becarios del museo UNIVERSUM, Tita Pérez (Física) y al jefe del área de Innovación Tecnológica, el Doctor Alfredo Careaga, por permitir tomar como tema de tesis mi labor como becario.

Al M. en C. Marcelo Pérez Medel por aceptar dirigirme el trabajo de tesis, por sus sugerencias, por compartir su experiencia, y por el apoyo que me ha brindado tanto personal como profesionalmente...

A los ingenieros Gladis Fuentes Chávez, Cesar Germán Rosas, Alejandro Ponce y Agustín Vargas Padilla por aceptar ser revisores de este trabajo y por todo lo que aportaron para enriquecerlo...

A la Universidad Nacional Autónoma de México, donde ha transcurrido mi formación profesional y donde he podido desarrollar mis inquietudes académicas...

A todos ustedes

GRACIAS

Índice

Índice de Tablas

INTRODUCCIÓN

La Dirección General de Divulgación de la Ciencia (DGDC) tiene la necesidad de llevar un control sobre aquellas personas que se encuentran adscritas a su Programa de Becarios. Actualmente este control se realiza utilizando en conjunto programas de procesamiento de datos tales como Microsoft Word 97 y Excel 97, siendo este un proceso laborioso y costoso tanto en tiempo de procesamiento de los datos como en tiempo de vinculación entre las distintas aplicaciones arriba mencionadas, además de presentar inconvenientes a la hora de solicitar reportes.

EL Dr. Alfredo Alejandro Careaga, quien actualmente está a cargo de la Subdirección de Innovación Tecnológica, se ha preocupado por dar apoyo a este departamento, solicitando para ello la elaboración de un sistema de base de datos que sea capaz de realizar esta tarea. El sistema requerido debe prestar las mayores facilidades posibles al usuario de tal forma que éste sea capaz de realizar el trabajo que hace con los programas mencionados anteriormente en una sola aplicación, mejorando así el tiempo de procesamiento de los datos, así como el desempeño del trabajo.

Se desea que una primera versión del sistema sea de tipo local, es decir, para uso directo en la Jefatura de la Coordinación de Becarios, lo que ayudará a que el personal a cargo se ajuste al cambio en la forma de trabajo, así como también dará las bases para el uso del posterior sistema en Internet, para lo cual se deben considerar las medidas de seguridad que sean adecuadas. Para tal efecto se ha pensado en utilizar básicamente dos herramientas en la elaboración del sistema local: Access 97 como el motor de la base de datos, y Visual Basic 6 como el gestor del Sistema Informático. Para el sistema que funcionará en Internet se ha pensado en utilizar las herramientas del Personal Web Server (PWS) como el servidor de contenido web, y el uso de la tecnología de ASP (Active Server Pages) para realizar las páginas dinámicas que trabajen con el sistema. El poder utilizar el sistema a través de Internet es de suma importancia y realmente resulta fácil de

implementar. Esto debido al gran crecimiento que ha tenido Internet en los últimos años, así como las herramientas utilizadas para generar contenido dinámico. Cada vez hay más portales que utilizan las últimas tecnologías para entregar este tipo de contenido al usuario, permitiendo que el mismo tenga mayor control sobre los resultados que pretende obtener al manipular datos. El permitir que el sistema opere en Internet amplia la funcionalidad del mismo. Todas las personas que lo requieran pueden acceder al sistema y trabajar con el, no importando si trabajan bajo Windows, Linux[ii], Sun[iii], Mac[iv] o cualquier otra plataforma; basta con que cuenten con un navegador de Internet. Debemos considerar que al estar el sistema disponible en Internet pudiera ser que cualquier persona ajena al mismo entrara y alterara la información, lo cual no será posible. El acceso al sistema es restringido, como sucede en muchos sitios en Internet (como servidores de correo, por ejemplo). Por tanto, el tiempo de vida del sistema será mientras cumpla con su función, y como la misma no cambia y el Internet seguirá creciendo, realmente tendrá un largo futuro. A continuación se presenta una breve descripción de lo que se tratará en cada capítulo.

En el capítulo 1 se hace un estudio de viabilidad y se realiza el análisis de los requerimientos del sistema. Se considera el modelo SSADM para completar el análisis del sistema de información. Se estudia la forma en que se ha venido trabajando en la Coordinación de Becarios, además de que se estudian las posibles alternativas de solución y se eligen las herramientas necesarias con base a los requerimientos del sistema y del espacio físico donde funcionará el mismo. También se establecen ciertos criterios sobre la información, tales como: los campos que contendrá el sistema, la jerarquía de los mismos, la estructura que debe(n) tener el(los) reporte(s) requerido(s), etc.

En el capítulo 2 se realizan diferentes versiones hasta llegar al prototipo final del sistema local. Se especifica las características de las herramientas utilizadas, así como el hardware necesario para su funcionamiento. Se describe la interfaz del

sistema, y se muestra el código de cada módulo, al menos la parte importante. También hay un apartado para el tratamiento de errores.

En el capítulo 3 se describen los resultados obtenidos luego de presentar el sistema a la Coordinación de Becarios con el fin de detectar y corregir los posibles errores, así como también realizar las mejoras posibles al sistema prototipo obtenido del capítulo 2. Con todo esto obtendremos la versión final del sistema local.

En el capítulo 4 se explicará cada uno de los archivos utilizados para el funcionamiento del sistema. También se muestra la instalación del mismo, pantalla por pantalla. Se hace la recomendación de los requerimientos mínimos para el correcto funcionamiento del sistema. Se muestra además un manual de usuario, que sirve de capacitación para la Coordinación de Becarios.

En el capítulo 5 se trata el acceso a la información por Internet. Se describen las herramientas empleadas para publicar datos en la red, así como también se da una breve visión de lo que es el lenguaje ASP, empleado para trabajar con los datos en Internet. Una parte importante es la seguridad de la información, por lo que se muestra como proteger no sólo la base de datos, sino el acceso a las páginas del sistema.

En el Apéndice A se aborda lo referente a la instalación del IIS 3 y 4 bajo Windows NT, ya que el sistema final fue instalado en una PC con ese sistema operativo. También se especifican las características de software y de hardware necesarias para el funcionamiento del sistema. Es bueno hacer la recomendación de que quien consulte este material tenga ciertos conocimientos de programación para que se le facilite el entendimiento de los ejemplos que aquí se muestran.

[i] Word, Excel, Access, Visual Basic, PWS, ASP y Windows son productos registrados de Microsoft Corporation
[ii] Linux es un producto gratuito de OSDN (Open Source Development Network)
[iii] Sun es un producto registrado de Sun Microsystems
[iv] Mac es un priducto registrado de Apple Computer, Inc.

1. ANÁLISIS DE LOS REQUERIMIENTOS INFORMÁTICOS

INTRODUCCIÓN

En este capítulo se darán los primeros pasos para la elaboración del sistema. Como todo sistema de bases de datos, el punto de partida es el entendimiento del problema, lo cual permitirá conocer el mejor camino para su solución.

En este caso, para conocer el problema y vislumbrar su solución, mucho del proceso es hacer un "trabajo de campo", debido a que se realizarán numerosas pláticas con el personal de la DGDC[1], básicamente con el encargado de la Coordinación de Becarios que son quienes utilizarán en mayor medida el sistema, aunque no serán los únicos.

El estudio es extenso ya que se tienen que analizar muchos aspectos como son las formas en papel, la organización que realizan, los métodos que emplean, etc. Todo ello para determinar la mejor forma de alcanzar el objetivo.

Hay que mencionar que, debido a que este será un sistema desarrollado para la DGDC, hay aspectos que deben permanecer fuera de este trabajo por la seguridad del mismo, tales como la forma de implementar la seguridad, que si bien no se dará la forma precisa de cómo se realiza sí se presenta una forma que pueda servir de ejemplo para cualquier otra persona que desee utilizar estas herramientas en el desarrollo de un sistema.

Así pues, este trabajo cumplirá dos objetivos a la vez. Uno es el de desarrollar un sistema para la Coordinación de Becarios de la DGDC. El otro se refiere a la aportación que se hará al apoyar la difusión de las herramientas empleadas.

[1] Dirección General de Divulgación de la Ciencia

1.1 Metodología SSADM

Antes que nada, debemos utilizar un modelo de análisis y desarrollo que nos lleve a la solución del problema. En la bibliografía consultada se encuentran diversos de estos métodos, de los cuales se tomó el que se muestra a continuación, el cual se explica mejor en el libro *"Elementos y Herramientas en el Desarrollo de Sistemas de Información"* de Mario G. Prattini. En 1980, el Gobierno Británico creó la metodología SSADM (Structured Systems Analysis and Design Method), resultado de una búsqueda de un método que solucionara los problemas de los departamentos informáticos. Desde su origen ha tenido varias modificaciones, y se le fueron incorporando diversas fases hasta llegar a las que se verán en esta parte de la Tesis. El SSADM proporciona un conjunto de especificaciones y procedimientos para realizar el análisis y el diseño de un sistema. Trata de dar el mayor grado de especificaciones, sin entrar en la construcción del código. La figura 1.1 muestra la organización SSADM.

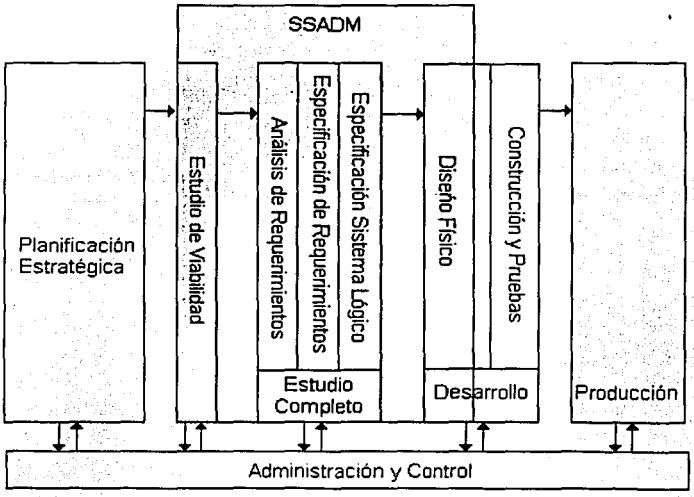

Figura 1.1 Metodología SSADM

Los módulos de la metodología SSADM son los siguientes:

Módulo FS (Feasibility Study) – Estudio de Viabilidad

Se recomienda realizar un estudio de viabilidad antes que el análisis. El objetivo principal es poder establecer un sistema de información que responda satisfactoriamente a los requerimientos de la organización. Se especifica el alcance funcional de las distintas opciones del sistema, y ayuda a planificar los recursos necesarios en el estudio. Se obtiene el informe de viabilidad.

Etapa 0 Viabilidad

- Paso 010 Preparar el estudio de viabilidad
- Paso 020 Definir el Problema
- Paso 030 Seleccionar Opciones
- Paso 040 Componer el informe de viabilidad

Módulo RA (Requirements Analysis) – Análisis de Requerimientos

Se trata de definir el alcance del proyecto, establecer la integración de los componentes del sistema con el resto de las necesidades de la organización, según los requerimientos de los usuarios, y conseguir una visión global del costo y los beneficios que traerá el nuevo sistema. En la Etapa 1 se describe el sistema actual y los nuevos requerimientos que se considera se le deben agregar. La Etapa 2 proporciona un conjunto de soluciones posibles que respondan a los requerimientos planteados. Al final se obtienen: la descripción del sistema actual, el catálogo de requerimientos y la descripción de la solución elegida.

Etapa 1 Investigación del sistema actual

- Paso 110 Establecer el Marco de Trabajo
- Paso 120 Investigar y definir requerimientos
- Paso 130 Investigar procesos del sistema actual
- Paso 140 Investigar datos del sistema actual
- Paso 150 Obtener una visión lógica del sistema actual

Paso 160 Componer los resultados de la investigación

Etapa 2 Estudio de opciones del sistema

Paso 210 Definir opciones del sistema

Paso 220 Seleccionar una de las opciones

Módulo RS (Requirements Specification) – Especificación de Requerimientos

Proporciona una descripción detallada del sistema futuro. Se especifican las descripciones de datos, procesos y requerimientos lo más detallado posible. El SSADM recomienda mostrara al usuario la forma en que la nueva aplicación cubrirá los requerimientos planeados, con el objetivo de detectar cualquier error en las especificaciones. Se obtienen al final la estructura de diálogos y menús, el prototipo y la especificación de requerimientos.

Etapa 3 Definición de Requerimientos

Paso 310 Definir Procesos del sistema requerido

Paso 320 Definir el modelo de datos requerido

Paso 330 Obtener las funciones del sistema

Paso 340 Refinar el modelo de datos requerido

Paso 350 Desarrollar prototipos

Paso 360 Desarrollar especificaciones de procesos

Paso 370 Confirmar objetivos del sistema

Paso 380 Componer la especificación de requerimientos

Módulo LS (Logical System Specification) – Especificación del Sistema Lógico

En la Etapa 4, la solución seleccionada y la especificación de requerimientos se traducen en un conjunto de opciones de implementación técnica (lenguajes, entornos de desarrollo, plataformas de producción) que serán evaluadas económicamente para seleccionar alguna de ellas. En la Etapa 5, la especificación de requerimientos se transforma en componentes que definen la funcionalidad del sistema según sus operaciones. De las funciones se obtiene una descripción de los procesos de

actualización y consulta. Se define la interfaz de usuario mediante el diseño de diálogos. Los requerimientos de acceso a la información se incorporan en la definición del modelo de datos. Obtenemos la opción técnica seleccionada, la descripción del entorno técnico y el diseño lógico de datos y procesos.

Etapa 4 Opciones del sistema técnico
- Paso 410 Definir opciones técnicas
- Paso 420 Seleccionar opción técnica

Etapa 5 Diseño lógico
- Paso 510 Definir diálogos de usuario
- Paso 520 Definir procesos de actualización
- Paso 530 Definir procesos de consulta
- Paso 540 Componer el diseño lógico

Módulo PD (Physical Design) – Diseño Físico

Esta etapa está influenciada por el entorno de implantación utilizado, es decir, el hardware a utilizar para que la solución elegida pueda funcionar correctamente. El objetivo aquí es especificar los datos, procesos, entradas y salidas de acuerdo al entorno físico elegido, incorporando a todo ello los estándares de la instalación. El diseño resultante debe proporcionar todo lo necesario para decidir cómo construir la aplicación. Se obtienen los estándares de desarrollo y el diseño físico de datos y procesos.

Etapa 6 Diseño físico
- Paso 610 Preparar el diseño físico
- Paso 620 Crear diseño físico de datos
- Paso 630 Crear mapa de implementación de funciones
- Paso 640 Optimizar el diseño físico de datos
- Paso 650 Completar la especificación de funciones
- Paso 660 Consolidar interfaces de proceso de datos
- Paso 670 Componer el diseño físico

Entre las ventajas que se le atribuyen a este método se encuentran:

1. La participación del usuario en los modelos asegura que el sistema diseñado coincida con el sistema requerido.
2. La separación de conceptos lógicos y físicos facilita en cierto modo la implementación del sistema analizado en distintos entornos.
3. Produce gran cantidad de documentación a lo largo de todas sus etapas, lo que es una ayuda importante para la producción y el mantenimiento del sistema diseñado.

Los inconvenientes que se mencionan son que, debido a su origen británico tiene un carácter rígido, lo que se observa en la definición rigurosa de cada paso de cada etapa.

Como se puede notar, este modelo refleja el sentido rígido de los británicos. Por tanto, no consideraremos para el presente trabajo el sentido estricto de las fases del método, sino que se tomará la idea general de cada uno, concentrándonos en los resultados principales que arroja cada uno de ellos. A continuación iniciaremos el estudio del sistema con base a estos puntos.

1.2 Estudio de Viabilidad y Análisis de Requerimientos

Primeramente tenemos que definir el objetivo principal, el cuál es Implementar un Sistema Informático que sirva para el Control de Becarios de la DGDC, el cual pueda ser utilizado vía Internet. Esto trae consigo que, en primer lugar, cambie la forma en que se realizan las actividades para el control de los becarios, lo que ya se comentaba al inicio de este trabajo.

Entonces, primeramente debemos evaluar si es posible cambiar la forma en que se ha estado trabajando en el procesamiento de la información de los becarios, para vislumbrar alguna posible solución. Como punto de partida, debemos determinar cuáles serán los datos que el sistema requiere. En reuniones sostenidas con el personal a cargo del área de la Coordinación de Becarios, analizamos la forma en como han venido trabajando para conocer más sobre el problema en cuestión.

Partimos del estudio de las formas en papel que cada becario llena al solicitar la beca de la DGDC. Estas formas se reciben por cada becario, además de otros documentos necesarios para su solicitud. Lo importante aquí es la información que se recopila de las solicitudes. Si el becario es aceptado, estas formas son guardadas y toda la información se capturaba en Excel, donde se almacenaba toda en conjunto, dividiéndola por museos : Museo de la Luz y UNIVERSUM. De acuerdo a los requerimientos de la Coordinación de Becarios de la DGDC, cuando se requería cierto tipo de información como la relación de becarios por sala, por Tutor, etc., se tenía que buscar en la información primaria almacenada en documentos de Excel y de ahí se separaba para obtener los reportes deseados. Con el sistema a desarrollar, se mejorará el desempeño.

Con todo esto, llegamos a la conclusión de que la información podía dividirse en tres categorías diferentes:

- Datos Personales.
- Preparación Académica, y
- Área de Adscripción

Esta división no es jerárquica, sólo es la forma de distinguir la información. Dentro de los Datos Personales, tenemos los campos siguientes: Nombre (que incluye Apellido Paterno y Apellido Materno), Fecha de Nacimiento, Edad, Dirección, Colonia, Delegación o Municipio, Código Postal, Teléfono, Correo Electrónico, y otros dos campos para el tratamiento de posibles emergencias: a quien avisar en caso de que el becario sufra algún accidente y el teléfono al cuál dirigirse en dado caso.

Los datos de la Preparación Académica comprenden: Institución de Procedencia, Facultad o Escuela, Carrera, Otros Estudios (por si tiene alguna otra carrera), Año de Ingreso a Licenciatura, No. De Cuenta o Matrícula, Porcentaje de Créditos, el Promedio, y dos campos por si el becario tiene conocimientos de otro idioma diferente del Español, con las cantidades correspondientes del porcentaje que el becario habla, lee y escribe según sea el caso del idioma.

Finalmente, los datos del Área de Adscripción están formados por: el Área a la cual pertenece, el Jefe del Área, el Tutor a cargo del becario, la Fecha de Ingreso, la Fecha de Egreso, un campo que indique si el becario concluyó con su ciclo, y otro campo para el casillero, si es que el becario cuenta con uno.

Como ya se mencionó, a la fecha toda esta información se concentra primero en papel, por cada becario, para después ser vaciada en la computadora a Excel. Esta forma de tratar la información presenta varias desventajas, ya que el trabajo en Excel es largo y tedioso, debido a que la información tiene que capturarse primero, luego separarse por museo, ordenarse, y para cada tipo de reporte necesario se tiene que buscar manualmente en Excel la información requerida y organizarla, para finalmente poder utilizarla.

Al dividir y organizar la información jerárquicamente en estas tres partes, se tiene un mejor control sobre la misma, ya que su localización es más rápida.

En la página 9 se muestra la forma en papel que se emplea cuando un estudiante pide su ingreso como becario ante la Coordinación de Becarios de la DGDC. El procesamiento de la información se puede ver en el diagrama de la figura 1.2.

Dirección General de
Divulgación de la Ciencia
UNAM

COORDINACION DE BECARIOS

SOLICITUD DE BECARIO

NO._____

NOMBRE: _____
Apellido Paterno Materno Nombre(s)

_____
Fecha de Nacimiento Edad Estado Civil

DOMICILIO:_____
Calle y No. Colonia

_____
Delegación C.P. Teléfono

PREPARACIÓN ACADEMICA

Universidad de Procedencia: _____

Facultad o Escuela:_____ Carrera:_____

Número de Cuenta (matrícula):_____ Créditos (%):_____ Promedio: _____

Otros estudios: _____

Idiomas:_____ Habla _____% Lee _____% Escribe _____%

_____ Habla _____% Lee _____% Escribe _____%

Otras actividades que realizas actualmente: _____

_____

_____

Sala o Área de preferencia: _____

Horario disponible: L_____ M_____ M_____ J_____ V_____ S_____ D_____

Ciudad Universitaria a, _____ de _____ de _____

_____
Firma

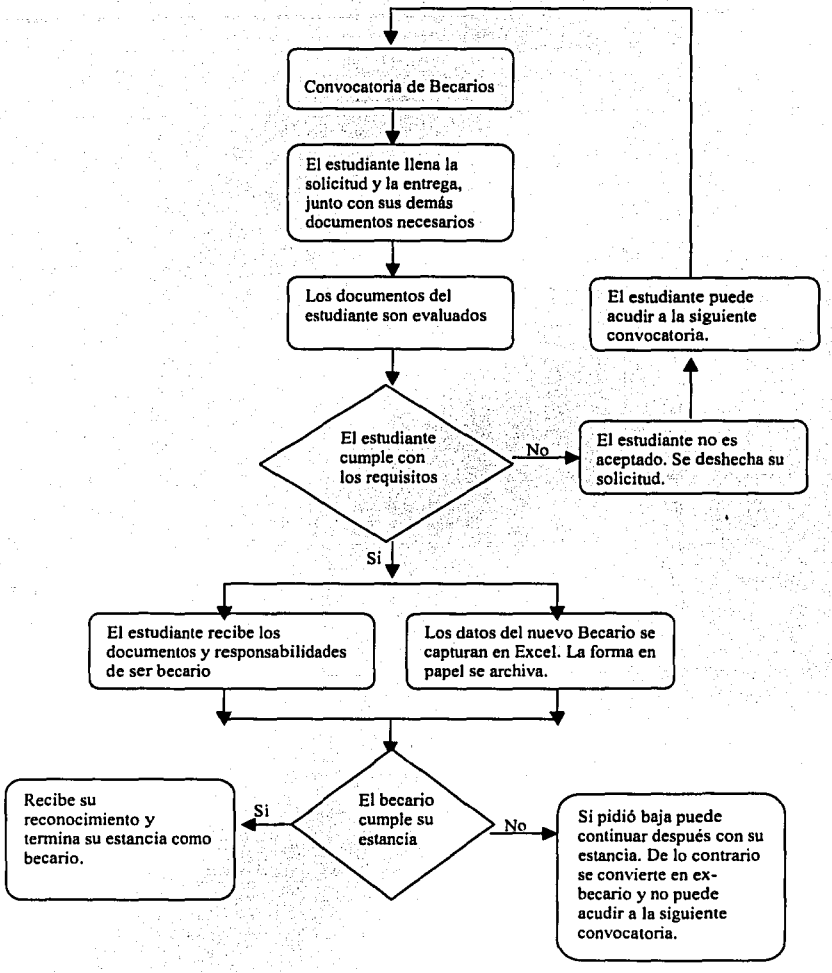

Figura 1.2.b Diagrama de Flujo de Datos

El sistema a desarrollar debe cumplir con las siguientes características:

- Permitir el ingreso de nuevos becarios.
- Permitir que se den de baja becarios por diferentes razones: cumplimiento de ciclos, baja temporal, baja definitiva, etc.
- Poder consultar en cualquier momento los datos de todos o alguno de los becarios.
- Poder hacer correcciones sobre los datos ya almacenados por si hubo errores al momento de capturar, o por si el becario cambia de área, por ejemplo.
- El sistema debe ser capaz de proporcionar diferentes tipos de reportes, según lo requiera la DGDC: por área, por tutor, por generación, etc.
- Además, el sistema debe permitir el acceso a la información sólo al personal autorizado, protegiendo con ello la integridad de los datos.

Como una primera aproximación se ha pensado en una versión del sistema que sea de tipo local, es decir, que sea de uso interno en la Coordinación de Becarios. Esto permitirá que los usuarios del mismo se adecuen al cambio de pasar de su forma manual de trabajo a la forma automática que represente el sistema deseado.

Para esta primera aproximación del sistema se consideraron distintos aspectos concernientes a la ubicación física. La Coordinación de Becarios está en proceso de cambios, lo que afecta las computadoras con las que trabajan; han planeado en sustituir todas las computadoras que trabajan con sistema operativo Windows 98 por computadoras Mac con su correspondiente sistema operativo, Mac OS.

Mientras se realiza el cambio de hardware, dejarán una computadora con Windows 98 para que sigan trabajando en la Coordinación de Becarios, y es en esa donde funcionará el sistema local. Por tanto, las herramientas necesarias tienen que trabajar sobre dicho sistema operativo. Se eligió a Microsoft ® Access como el motor de bases de datos, puesto que la información que ya tienen en la Coordinación de Becarios está en formato de documentos de Excel, los cuales se pueden migrar a una base de datos tipo Access.

Por otro lado, se pensó en cual sería el lenguaje de programación empleado para desarrollar la interfaz del sistema. Se consideraron en un principio Fox Pro, Visual C y Visual Basic. Finalmente se eligió Visual Basic, debido a que en lo concerniente a bases de datos es la herramienta con la que más se trabaja en las empresas (en el ámbito local), además de otras áreas referentes al desarrollo de sistemas.

Posteriormente el sistema debe ser utilizado desde Internet. Otro aspecto importante a considerar para el desarrollo del sistema fue emplear herramientas de bajo costo económico y que no implicara grandes cambios de hardware, debido al proceso de actualización de equipo que se realizaría en la DGDC, involucrando también a todo el UNIVERSUM que es en donde opera la Coordinación de Becarios. Así, para el sistema local se contaba con una PC con Windows 98 dentro de la Coordinación, y para el sistema en Internet con una PC con Windows NT 4.0 localizada en el área de administración de servicios de la Subdirección de Innovación Tecnológica.

Para el sistema en Internet se buscaron las posibles herramientas que permitieran por un lado establecer un servidor de Web para las páginas del sistema, y por otro lado aquellas que permitieran manipular la información de las bases de datos. En lo referente a los servidores de Web se encontró el Apache Server en su versión 1.3, así como el Personal Web Server (para Windows 9x) y el Internet Information Server (para Windows NT), de Microsoft ®. Los tres son de libre distribución, por lo que no implican ningún costo adicional.

Dentro de los lenguajes para realizar páginas dinámicas que trabajen con datos se encontraron el PHP, Perl, ASP (Active Server Page) y JSP (Java Server Page). Los dos primeros presentan la desventaja de que sólo trabajan con el servidor Apache, tienen cierto grado de dificultad al configurarse y se requiere de tiempo para familiarizarse con la sintaxis del lenguaje.

Por otro lado, ASP y JSP son lenguajes que en poco tiempo pueden ser comprendidos sin mayor problema. El código ASP no es interpretado en Apache. El código JSP puede ser interpretado en Apache utilizando un complemento extra como puede ser TomCat. Si bien es cierto que el servidor Apache puede instalarse en plataformas Windows o

Linux, en el primer caso presenta la desventaja de que es más inestable, y por otro lado el departamento donde se instalará el sistema sólo cuenta con computadoras cuyo sistema operativo es Windows NT. Por tanto, el servidor tendrá que ser el Personal Web Server o el Internet Information Server, lo que implica la utilización de ASP que es soportado directamente por cualquiera de ambos servidores; es el lenguaje dinámico por omisión en ambos. Además el ASP es muy similar al lenguaje de Visual Basic, por lo que su entendimiento es prácticamente directo y no presenta mayores problemas para quienes hayan tenido contacto con este lenguaje de programación. Considerando que trabajaremos con Visual Basic para desarrollar el sistema local, el acercamiento al lenguaje ASP será más fácil, por lo que los JSP's se dejan entonces de lado.

A partir de aquí en el presente trabajo se muestran los aspectos que se refieren a los requerimientos planteados por los sistemas propuestos, así como su implantación.

1.3 Características de los datos

En esta parte hay que tener un especial cuidado a la hora de trabajar con los datos, ya que la información que maneje cada campo determinará el tipo de variable a utilizar, lo cuál a su vez determina el espacio requerido para almacenar dicha información, así como el tiempo requerido para su procesamiento. Dadas las características de los equipos actuales esto no debe representar mayor problema, sin embargo un buen planteamiento puede beneficiar el desempeño del sistema.

Debemos considerar que el tamaño de cada campo sea lo más adecuado posible. Es decir, hay que tratar de evitar, en la medida de lo posible, que no sea tan limitado como para que no pueda contener la información requerida, y al mismo tiempo cuidar el que no sea más grande de lo que se pueda necesitar.

Por ejemplo, para el campo de la Edad, es notorio que el dato es de tipo numérico (aunque también podría ser tipo texto) y de tamaño 2. Para el caso de las fechas, se utilizará el formato dd/mm/aa, por lo que el tamaño será de 8, y el tipo de dato puede ser de fecha o de texto.

Por otro lado, hay que considerar todas las posibles alternativas que se pueden dar a la hora de capturar la información. Por ejemplo, para el caso del código postal se podría pensar que el tipo de dato es numérico con una longitud de 5 dígitos aproximadamente, que es lo más común. Sin embargo, en la práctica las cosas son distintas. En las reuniones con el personal a cargo del área también se trataron estas posibilidades, y se observó que hay casos en que el código postal (siguiendo con este ejemplo) incluye un cero al inicio, lo cual no podría ser posible si fuera de tipo numérico.

Hay que considerar también que el tipo de dato de cada campo tiene un papel importante en el momento de la captura de los datos, ya que si, por ejemplo, se introducen letras en campos de tipo numérico, esto provocaría un error, el cuál debe ser previsto y manejado correctamente. Por otro lado, también se debe tener en cuenta que el tipo de dato reserva el espacio necesario en memoria para procesarlo, además de que la información almacenada (en Access) crecerá y con ello también crecerá el tamaño que la base de datos reservará para almacenar un tipo de datos numérico o de texto.

En las reuniones sostenidas se observó que hay campos como el No. de Cuenta que, si bien en la mayoría de los casos es numérico, también se pueden dar casos en que incluya letras; y puede ser también que el tamaño del dato capturado sea más grande de lo que un tipo de dato numérico puede alcanzar.

Por todo esto se convino en que la información en general sea de tipo texto, lo que trae varias ventajas como que el espacio reservado (tanto en memoria como físicamente) sea menor que si fuera para un tipo de dato numérico, la posibilidad de error se elimina porque no hay problemas con la longitud del dato o si se mezclan letras y números en un campo.

En la tabla 1 se muestran las características de los datos.

Campo	Tamaño	Campo	Tamaño
Nombre Completo	50	Otros estudios	30
Fecha de Nacimiento	8	% de Créditos	5
Edad	2	Promedio	5
Dirección	60	Idiomas 1 y 2	12 c/u
Colonia	40	% que Habla, Lee y Escribe	3 c/u
Delegación o Municipio	40	Año de Ingreso a Licenciatura	4
Código Postal	10	Área	50
Teléfono	16	Jefe de Área	50
E-mail	35	Tutor	50
Avisar a	35	Año de Ingreso	2
Otro teléfono	16	Mes de Ingreso	2
Institución de Procedencia	45	Año de Término	2
Facultad o Escuela	35	Mes de Término	2
Carrera	40	Estado	15
No. De Cuenta	22	Casillero	3

Tabla 1 Longitud de los datos del sistema

1.4 Estructura de la información en Access

La información se organizará de la siguiente manera: la base de datos se creará en Access. ¿Por qué en Access y no en algún otro motor de bases de datos? Pues debido a que los usuarios (en su mayoría) trabajan bajo el sistema operativo Windows, de ahí que trabajen con Excel. Quienes no utilizan PC utilizan computadoras Macintosh, también con Excel. Además, el tamaño de la información no será tan grande como para pensar en utilizar motores como Oracle. Por otro lado, al hacer la primer aproximación del sistema se tomó en cuenta que quienes trabajarían manipulando la información

usarían PCs, por ello se optó por hacerlo en Visual Basic 6. El servidor de web, el Personal Web Server (que se verá más adelante) trabaja bajo Windows y puede acceder a diferentes bases de datos, entre ellas Access.

La información se distribuirá en una tabla dentro de la base de datos. Serán dos bases de datos, una para el Museo de la Luz y otra para el Universum. La tabla contendrá los campos necesarios : 33 para el museo de la Luz y 36 para el Universum. Los datos no se dividen en tablas como se dividió la información (Datos personales, Preparación Académica y Datos de Adscripción) puesto que no es una gran cantidad de campos que podrían confundir si se concentraran en una sola tabla, además de que se debe considerar que al ser utilizada por Internet deben reducirse lo más posible los tiempos de acceso a los datos y su procesamiento, y si se tuvieran varias tablas se aumentarían estos tiempos por las búsquedas necesarias.

Un aspecto importante es considerar los límites que Access tiene en cuanto a espacio de almacenamiento, y consultando la ayuda del mismo programa, se sabe que puede manejar archivos de hasta un máximo de 1 GB, por lo que es suficiente para manejar los datos de la DGDC, que no pasan de unos cuantos MB.

Hemos terminado con el estudio de viabilidad y el análisis de los requerimientos del sistema, el análisis de la forma en que se ha venido trabajando en la Coordinación de Becarios así como el estudio de las características de Access y se establecieron las características de los datos. Considerando los resultados obtenidos el paso que sigue es diseñar una primera aproximación local del sistema para ir vislumbrando lo que será la versión final del mismo, considerando los aspectos necesarios respecto del hardware y software, como se verá en el capítulo siguiente.

2. DISEÑO DEL SISTEMA

INTRODUCCIÓN

Esta parte fue la más larga del desarrollo, aunque a primera vista el presente capítulo no lo refleje tal cual. Esto debido a que en cada una de las reuniones sostenidas con el personal correspondiente de la Coordinación de Becarios, siempre se le añadían más cosas al sistema. Esto es bueno en la medida en que se van estableciendo todas las partes necesarias; sin embargo, trae la desventaja de que no se podía llegar a una versión preliminar final porque siempre se le estaban añadiendo cosas, aunque también se iban mejorando otras partes del sistema. Por elle se llegó a la conclusión de que se fijarían las partes más importantes del sistema, y se trabajaría sobre ellas en el desarrollo de la versión preliminar.

En esta parte de desarrollo se empleará la herramienta de programación Visual Basic 6 para desarrollar la interfaz del sistema. Se eligió esta herramienta debido a que la versión local en una PC con Windows 98. Se debe entender como versión local a la versión del sistema que no estará disponible en Internet, es decir, es de uso local para la DGDC. Se podría pensar que Visual Basic no es la única herramienta de programación que funciona bajo Windows, y es cierto. Sin embargo, en muchas empresas este lenguaje de programación es de los más utilizados para trabajar con bases de datos, debido a las prestaciones que trae consigo, que ayudan en mucho al desarrollo de un sistema.

Como ya se ha mencionado, debido a que es un sistema de carácter privado todo esto se esbozará de manera general sin mucho detalle, salvo aquellas partes que lo requieran. Esta primera aproximación se creará en una PC con Windows 98, con Visual Basic 6 como ya se mencionó, contando con 32 MB de memoria RAM en un monitor con una resolución de 800x600 pixeles con una profundidad de color de 32 Bits. Por esto mismo, se recomienda que la computadora donde opere el sistema posea por lo menos las mismas características, sobre todo en cuanto memoria se refiere. En cuanto a espacio de almacenamiento, se recomienda un mínimo de 500 MB.

2.1Estructura de los datos y diseño de la interfaz gráfica

La forma en que se organizarán los datos y el diseño de la interfaz para acceder a ellos es muy importante. Partamos entonces del diseño de la base de datos. Como se observó en el capítulo anterior el tipo de datos es general para cualquier información en la BD, que es de tipo texto. Esto presenta las ventajas ya comentadas en la página 14. El tamaño de los campos será el definido en la tabla 1 de ese mismo capítulo y se mantendrá tanto para el sistema local como el que será utilizado por Internet, ya que se utilizarán las mismas BD's en ambos casos. Además debemos considerar que no debe haber la posibilidad de que se repitan valores en la BD ya que se podría afectar a registros no deseados, como pudiera ser en el caso de que el nombre de un becario estuviera repetido y al momento de querer eliminarlo afectemos a otro becario con el mismo nombre, aunque ese caso pareciera muy raro podría llegar a darse. Para ello se convino en utilizar el número de cuenta del becario como la clave que lo identifica, ya que es prácticamente imposible que otro becario tenga el mismo número de cuenta. Otro aspecto importante es establecer si pueden existir campos con valores nulos, lo cuál es especialmente útil si por alguna razón no se pudieran ingresar los datos de un becario en un mismo momento. La convención fue que los campos del nombre y del número de cuenta eran necesarios, es decir, no nulos o sin datos, mientras que el resto de los campos pueden ser de tipo nulo. En la figura 2 se muestra la forma de acceder las bases de datos.

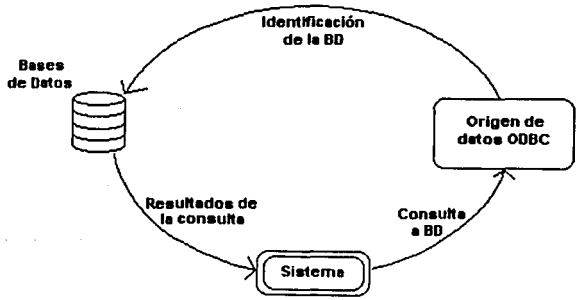

Figura 2 Acceso a los datos

En el capítulo 4 se explica más sobre ODBC, por ahora basta que sepa que es la forma que nos permite distinguir en la PC una base de datos de otra mediante un nombre clave para poder acceder a ella desde el sistema sin tener que pasar por el motor de bases de datos (Access).

Ahora veamos lo referente al diseño de la interfaz del sistema. Si nos remontamos a la forma que se aprecia en la página 9 del capítulo 1 (la solicitud de un nuevo becario) podemos distinguir tres grandes áreas en ella: una es la del inicio que contiene los datos personales del becario, la segunda es la que se refiere a su preparación académica y la última es la que se aprecia al final de la página y que corresponde al área de adscripción donde el becario realizará su estancia. Retomando la tabla 1 de ese mismo capítulo podemos darnos cuenta que en la tabla hay datos que no aparecen en la solicitud y que sin embargo sí aparecen al momento de capturar la información.

Es importante que el sistema pueda manejar todos los datos señalados en la tabla 1, la cual si contiene los datos de la solicitud a excepción del horario ya que es un aspecto que varía mucho durante la estancia del becario y realmente no tiene caso manejarlo en la base de datos. Por todo esto la interfaz que se planeó utilizar es la que aparece en la figura 2.1 donde, como se puede observar, el área principal de la pantalla del sistema se divide en las tres áreas que necesitamos para que se manejen todos los datos requeridos.

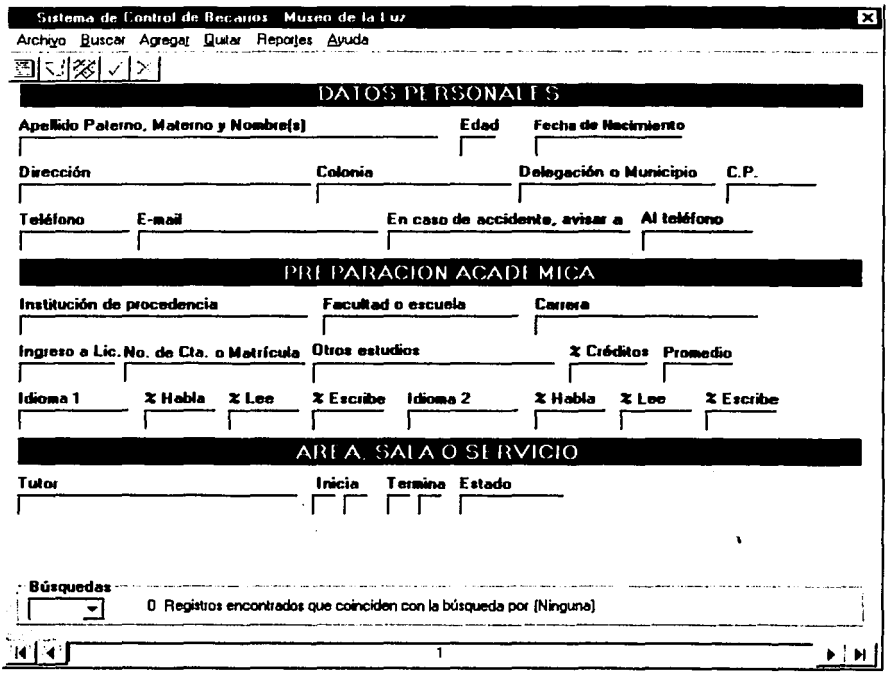

Figura 2.1Interfaz para el sistema del Museo de la Luz

Como se puede observar, hay 6 menús: 5 correspondientes a las opciones del sistema en sí, y uno más que corresponde a la Ayuda y que contiene lo referente a los tópicos clásicos de Acerca de y la Ayuda para el usuario. La parte central del sistema es la que mostrará la información de la base de datos. El sistema del Museo de la Luz y del UNIVERSUM son casi iguales, salvo tres campos que el del UNIVERSUM tiene adicionalmente, como ya se vio en el Capítulo 1.

Antes de entrar a la descripción de cada menú, hay que mencionar un aspecto que es notorio, y es el referente a los cinco iconos que se encuentran debajo de los menús. Estos iconos son como aquellos que podemos encontrar en muchas aplicaciones como Excel, y que representan una forma rápida de realizar las operaciones básicas del

sistema de la base de datos. De izquierda a derecha son: Alta, Baja, Editar, Aceptar y Cancelar.

El menú de Archivo tiene las opciones principales de manipulación de los datos.

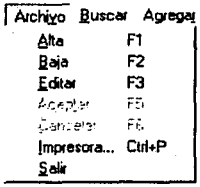

Figura 2.2 Opciones del Menú Archivo

Alta: Permite agregar un nuevo registro a la Base de Datos.
Baja: Permite borrar el registro actual de la base de datos. Antes de eliminar aparece un cuadro de diálogo para confirmar la acción.
Editar: Permite que los campos de texto del sistema se habiliten para modificar los datos del registro actual. Normalmente los campos de entrada de datos están deshabilitados para evitar cambios no deseados en los datos.
Aceptar/Cancelar: Realiza o cancela los cambios provocados por las opciones Nuevo o Modificar.
Impresora: Permite configurar la impresora predeterminada, así como seleccionar los campos y los registros a imprimir.
Salir: Cierra el sistema. Un cuadro de diálogo pregunta para confirmar la acción.

El menú de Buscar, como lo muestra la figura 2.3, permite hacer búsquedas por cualesquiera de los elementos listados.

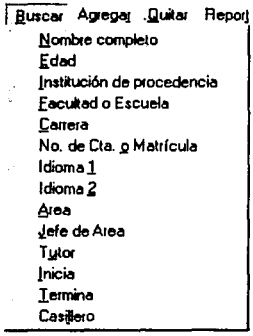

Figura 2.3 Opciones del Menú Buscar

La figura 2.4 muestra el menú de Agregar, que se emplea para llenar las listas de opciones que aparecen cuando se está editando o agregando un nuevo registro a la base de datos. A su vez, el menú de Quitar se emplea para retirar de dichas listas algún elemento de ellas que ya no se utilizarán más. Las opciones de cada lista se guardan en un archivo de texto, el cual sirve para alimentar las listas cuando sea requerido, como cuando se está editando un registro o agregando uno nuevo.

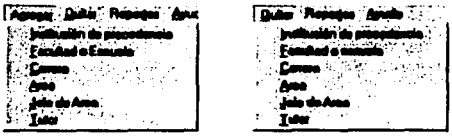

Figura 2.4 Opciones del Menú Agregar y del Menú Quitar

El menú de Reportes (figura 2.5) es otra parte importante del sistema. Permite abrir un documento de Excel que importa datos del sistema utilizando criterios previamente establecidos y con ello elaborar cada uno de los cuatro tipos de reportes listados.

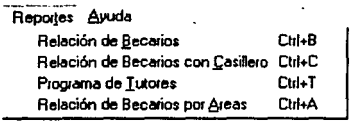

Figura 2.5 Opciones del Menú Reportes

El menú de Ayuda se muestra en la figura 2.6 y tienes dos opciones.

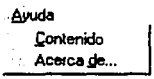

Figura 2.6 Opciones del Menú Ayuda

Contenido: Muestra una ventana con todos los tópicos de ayuda que puedan auxiliar al usuario del sistema.
Acerca de: Muestra un cuadro de diálogo con los créditos y la licencia del programa.

El recuadro de **Búsquedas** (figura 2.7) muestra los resultados obtenidos al realizar la última búsqueda. La lista desplegable de la izquierda contiene todos los registros coincidentes con la búsqueda, y al pulsar en ella mueve el apuntador al registro correspondiente.

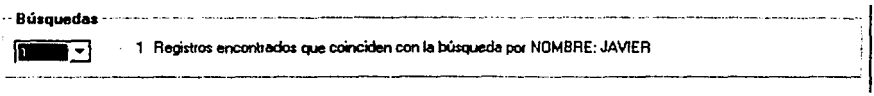

Figura 2.7 Área de Búsquedas

En la parte inferior se observa el control de desplazamiento por los registros de la base de datos (figura 2.8). La parte central muestra el registro actual o la operación en curso, por ejemplo, modificando un registro.

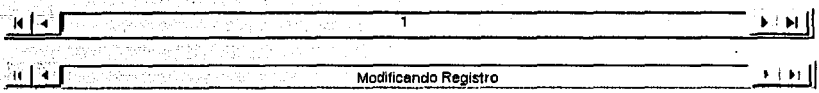

Figura 2.8 Barra de desplazamiento

Teniendo ya cada parte de la interfaz, procederemos a realizar la programación de cada una de las partes del mismo.

2.2 Codificación de los módulos requeridos

Aquí mostraremos la programación de cada parte del sistema, no de todo sino de aquellas partes que puedan aportar algo a quien se interese en diseñar un sistema de información. El método que se seguirá es por orden en cada menú. Es recomendable que quienes piensen en utilizar estos segmentos de código, estén familiarizados con el lenguaje de programación de Visual Basic, para una mejor comprensión. Para una mejor lectura y entendimiento, cada parte de código que deba explicarse se hará mediante una línea de texto identificada por el signo ', que en Visual Basic hace referencia a un comentario. A continuación mostramos el código de las partes del menú.

Menú de **Archivo**

Alta

```
Private Sub Alta_Click()          'Inicia la rutina de esta opción
 cmdAdd_Click      'Llama al procedimiento que le indica al sistema que se está
                   'agregando un nuevo registro
  FunEditar (True)  'habilita los cuadros de texto y muestra las listas de opciones para
                   'introducir los nuevos datos
  SubAlta.Enabled = False 'De aquí en adelante se habilitan o deshabilitan los
elementos
                          'de cada menú (así como los iconos debajo de los menús)
```

```
                        'que deben o no estar disponibles mientras
                        'se agrega un nuevo registro
  SubBaja.Enabled = False
  SubEditar.Enabled = False
  SubAceptar.Enabled = True
  SubCancelar.Enabled = True
  MenuBuscar.Enabled = False
  MenuAgregar.Enabled = False
  MenuQuitar.Enabled = False
  MenuReportes.Enabled = False
  MenuImpresora.Enabled = False
  Toolbar1.Buttons(1).Enabled = False
  Toolbar1.Buttons(2).Enabled = False
  Toolbar1.Buttons(3).Enabled = False
  Toolbar1.Buttons(4).Enabled = True
  Toolbar1.Buttons(5).Enabled = True
End Sub             'termina el procedimiento
```

Baja

```
Private Sub Baja_Click()
 cmdDelete_Click   'elimina el registro actual de la base de datos
                   'note que no hay confirmación, los datos se eliminan
                   'inmediatamente después de elegir esta opción
End Sub
```

Editar

```
Private Sub Editar_Click()
 cmdEdit_Click     'indica al sistema que se está editando el registro actual
 FunEditar (True)  'Igual que en otros casos
 SubAlta.Enabled = False
 SubBaja.Enabled = False
 SubEditar.Enabled = False
 SubAceptar.Enabled = True
 SubCancelar.Enabled = True
 MenuBuscar.Enabled = False
 MenuAgregar.Enabled = False
 MenuQuitar.Enabled = False
 MenuReportes.Enabled = False
 MenuImpresora.Enabled = False
 Toolbar1.Buttons(1).Enabled = False
 Toolbar1.Buttons(2).Enabled = False
```

```
 Toolbar1.Buttons(3).Enabled = False
 Toolbar1.Buttons(4).Enabled = True
 Toolbar1.Buttons(5).Enabled = True
End Sub
```

Aceptar

```
Private Sub Aceptar_Click()
 cmdUpdate_Click  'actualiza la base de datos
  FunEditar (False)  'deshabilita los cuadros de texto y oculta las listas de opciones para
                     'evitar que se modifiquen los datos
  SubAlta.Enabled = True  'habilita de nuevo las opciones que deben estar disponibles
                          'y deshabilita las que no deben de estarlo
  SubBaja.Enabled = True
  SubEditar.Enabled = True
  SubAceptar.Enabled = False
  SubCancelar.Enabled = False
  MenuBuscar.Enabled = True
  MenuAgregar.Enabled = True
  MenuQuitar.Enabled = True
  MenuReportes.Enabled = True
  MenuImpresora.Enabled = True
  Toolbar1.Buttons(1).Enabled = True
  Toolbar1.Buttons(2).Enabled = True
  Toolbar1.Buttons(3).Enabled = True
  Toolbar1.Buttons(4).Enabled = False
  Toolbar1.Buttons(5).Enabled = False
End Sub
```

Cancelar

```
Private Sub Cancelar_Click()
 cmdCancel_Click
 FunEditar (False)  'deshabilita los cuadros de texto y oculta las listas de opciones para
                    'evitar que se modifiquen los datos
  SubAlta.Enabled = True  'habilita de nuevo las opciones que deben estar disponibles
                          'y deshabilita las que no deben de estarlo
  SubBaja.Enabled = True
  SubEditar.Enabled = True
  SubAceptar.Enabled = False
  SubCancelar.Enabled = False
  MenuBuscar.Enabled = True
  MenuAgregar.Enabled = True
  MenuQuitar.Enabled = True
  MenuReportes.Enabled = True
  MenuImpresora.Enabled = True
```

```
 Toolbar1.Buttons(1).Enabled = True
 Toolbar1.Buttons(2).Enabled = True
 Toolbar1.Buttons(3).Enabled = True
 Toolbar1.Buttons(4).Enabled = False
 Toolbar1.Buttons(5).Enabled = False
End Sub
```

Impresora : Sólo se muestra la parte que permite configurar la impresora.

```
Private Sub Impresora_Click()
On Error GoTo Fallo        'si ocurre algún error, pasa a la línea con esta etiqueta
Dim m As Integer   'variable para guardar el resultado de la respuesta
'muestra el mensaje que se ve al final de este código
m = MsgBox("Las modificaciones que haga afectarán a TODAS las aplicaciones" &
Chr(13) & "que utilicen esta misma impresora. ¿Desea continuar?", vbExclamation +
vbYesNo, "Advertencia")
If (m = 6) Then        'si se pulsa SI
 CDPrinter.ShowPrinter       'muestra un cuadro que configura la impresora
predeterminada
                              'CDPrinter es el nombre de un control Commond Dialog
End If
Exit Sub       'termina el procedimiento y evita que se continúe con el código
               'siguiente que sólo debe ejecutarse cuando surja un error

Fallo:  'etiqueta para el control de errores
'muestra un mensaje con la descripción del error
 MsgBox Err.Description, 16, "Error de Impresora"
End Sub
```

El atributo Description del objeto Err hace referencia a la descripción del error que sucedió. El número siguiente (16 u otro) permite que el mensaje que aparece tenga un icono. El mensaje que aparece es el de la figura 2.9.

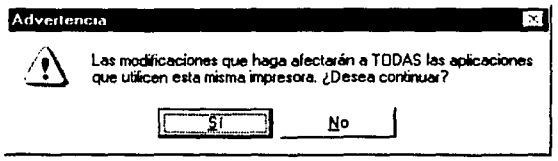

Figura 2.9 Mensaje de Impresora

Salir: Consta de dos partes, al pulsar esta opción aparece la ventana de la figura 2.10, debido a la ejecución del código siguiente

```
Private Sub Salir_Click()
 Unload Me          'llama a la propiedad UNLOAD del formulario principal
End Sub
```

Que a su vez llama a este código, que es especialmente útil para evitar que se cierre el sistema por equivocación.

```
Private Sub Form_Unload(Cancel As Integer)  'Cancel es la variable de Visual Basic
                                            'que determina si un formulario se cierra o no
Dim m As Integer
'pregunta si en verdad se quiere cerrar el sistema
m = MsgBox("¿Salir del Sistema del Museo de la Luz?", vbYesNo + vbQuestion,
"Cerrar")
If (m = 7) Then        'si la respuesta es NO
 Cancel = 1  'cancela cerrar el sistema (regresa al sistema)
Else   'de lo contrario
 Screen.MousePointer = vbDefault        'regresa el puntero del mouse a su forma
normal,
                                        'por si fuera necesario
 End          'termina la ejecución del programa
End If
End Sub
```

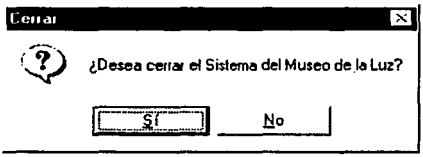

Figura 2.10 Confirmación de Cerrar el Sistema

Menú de **Buscar**: El código es similar para cualquier tipo de búsqueda. Primero, se solicita el dato a buscar:

```
Private Sub QueSeBusca_Click()
'QueSeBusca es cualquiera de los elementos listados por los que
'se puede buscar: Nombre, Carrera, etc.
Dim cadena As String
```

```
'cadena guarda el dato que se busca
cadena = InputBox("Escriba el dato a buscar", "Buscar")
If (cadena <> "") Then
 FunBuscar 0, cadena      'llama a la función de búsqueda
End If
End Sub
```

Para esta parte se realizó una función que podía hacer búsquedas por cualquiera de los elementos listados. Al seleccionar alguno de ellos, aparece un cuadro de diálogo que pregunta por el dato a buscar, como se ve en la figura 2.11.

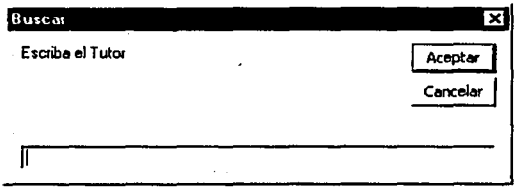

Figura 2.11 Cuadro de diálogo del submenú Buscar Tutor

Luego, ese dato se envía a dicha función, junto con un identificador que determina cuál campo es el que se empleará para la búsqueda. Los resultados se muestran en el recuadro de Búsquedas, como ya se vio en la figura 2.7.

Menú de **Agregar**: El código es el mismo para cada opción de este menú. Lo que cambia es el nombre del procedimiento, correspondiente a la lista de opciones ala que se haga referencia.

```
Private Sub Lista_Click()  'Lista puede ser cualquiera de los nombres
                           'que contiene este menú
On Error GoTo Fallo
Dim linea As String        'guarda el valor del nuevo elemento de la lista
Dim fileno As Integer, m As Integer, ya As Boolean
linea = InputBox("Escriba la Institución de Procedencia a añadir." & Chr(13) & "(Máximo
40 caractéres)", "Nuevo valor")
If (linea <> "") Then
 If (Len(linea) > 40) Then
  MsgBox "Demasiados caractéres.", 16, "Error"
 Else
```

```
 m = MsgBox("Se dispone a añadir este dato:" & Chr(13) & Chr(13) & linea & Chr(13) &
Chr(13) & "¿Es correcto?", 36, "Nuevo valor")
  If (m = 6) Then
   ya = False
   For m = 0 To Val(LblInstitucionI.Caption) 'primero lo buscamos a ver si esta
    If (UCase(ComboInstitucion.List(m)) = UCase(linea)) Then
     ya = True 'SI se encuentra
     m = Val(LblInstitucionI.Caption)
    End If
   Next m
   If (ya = False) Then 'solo lo escribe si no esta ya en el combo
    Open "C:\dgdc\misc\institucion.cta" For Append As #1
     Print #1, linea
    Close #1
    ComboInstitucion.AddItem linea
    LblInstitucionI.Caption = Val(LblInstitucionI.Caption) + 1

   Else 'si no, manda un mensaje de que ya existe
    MsgBox "Ese dato ya existe.", 16, "Error"
   End If
  End If
 End If
End If
Exit Sub

Fallo:
 MsgBox Err.Description, 16, "Error"
End Sub
```

Menú de **Quitar**: Elimina un elemento de alguna de las listas de opciones. Sigue la misma fórmula del menú Agregar.

```
Private Sub SubInsttitucionMenos_Click()
Dim indice As Integer, m As Integer
Dim linea As String
linea = InputBox("Escriba la Institución de Procedencia a quitar." & Chr(13) & "(Máximo
40 caractéres)", "Quitar valor")
If (linea <> "") Then
 If (Len(linea) > 40) Then
  MsgBox "Demasiados caractéres.", 16, "Error"
 Else
  m = MsgBox("Se dispone a quitar este dato:" & Chr(13) & Chr(13) & linea & Chr(13) &
Chr(13) & "¿Es correcto?", 36, "Quitar valor")
  If (m = 6) Then
   indice = -2
   For m = 0 To Val(LblInstitucionI.Caption) 'primero lo buscamos a ver si esta
    If (UCase(ComboInstitucion.List(m)) = UCase(linea)) Then
```

```
    indice = m 'si se encuentra, obtenemos el indice donde esta
    m = Val(LblInstitucionI.Caption)
   End If
  Next m
  If (indice >= 0) Then 'si esta, entonces lo quita del combo y del archivo
   ComboInstitucion.RemoveItem (indice)
   LblInstitucionI.Caption = (LblInstitucionI.Caption) - 1
   Open "C:\dgdc\misc\institucion.cta" For Output As #1
   For m = 0 To Val(LblInstitucionI.Caption)
    Print #1, ComboInstitucion.List(m)
   Next m
   Close #1

  Else 'si no, manda un mensaje de que no esta
   MsgBox "Ese dato no se encuentra.", 16, "Error"
  End If
 End If
End If
End If
End Sub
```

Menú de **Reportes**: Igual que el anterior. El código es el mismo para todos, cambiando sólo el nombre del archivo. Aquí lo importante es que el documento de Excel que se abre tiene establecidos criterios que permiten obtener y filtrar los datos de la información que maneja el sistema, presentando sólo aquellos que son requeridos, de acuerdo al tipo de reporte seleccionado.

```
Private Sub Relacion_Click()        'Relación representa al elemento seleccionado del
menú
On Error GoTo Fallo
Dim c As Integer
Dim strFic As String, strParam As String
'establece la ruta donde se encuentra el programa Excel, de lo contrario no funcionará
strFic = "C:\Archivos de programa\Microsoft Office\Office\EXCEL.EXE"

'abre el archivo de Excel que se seleccionó
strParam = "reporte.xls"
Shell strFic & " " & strParam, vbNormalFocus
End If
Exit Sub
Fallo:
 MsgBox Err.Description, 16, "Error"
End Sub
```

Hay que remarcar que no basta con abrir el documento de Excel; previamente se debió de crear dicho archivo con los criterios necesarios para filtrar la información. Para hacer esto, hay que hacer lo siguiente

Crear un nuevo documento de Excel, ir al menú de

Datos->Obtener datos externos->Crear nueva consulta

Esto hace que aparezcan las ventanas de las figuras 2.12 y 2.13

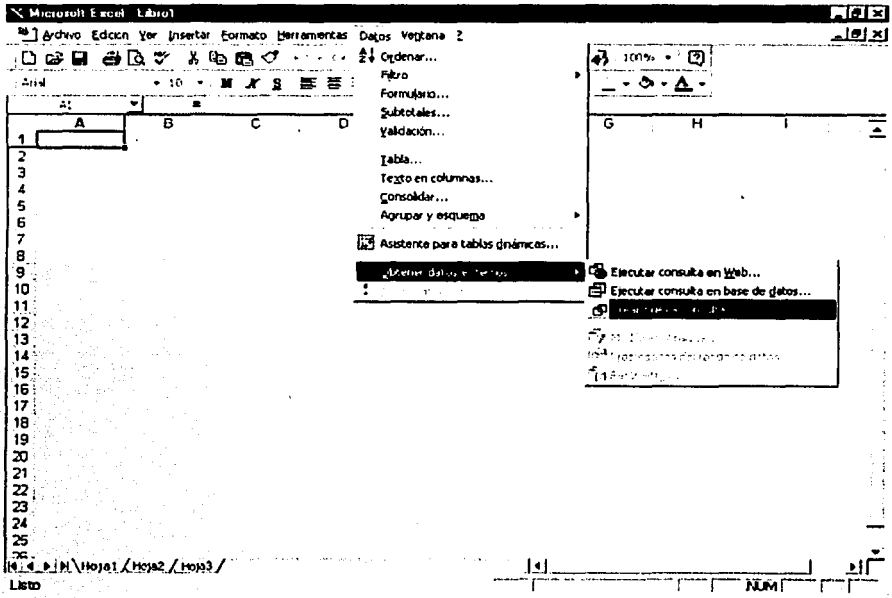

Figura 2.12 Crear nueva consulta en Excel

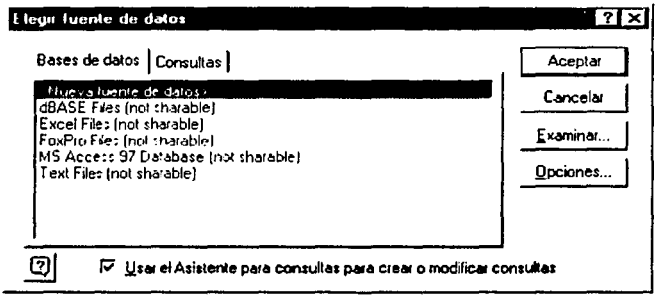

Figura 2.13 Elegir fuente de datos

En la figura 2.13 elegimos la opción por omisión, Nueva fuente de datos. Aparece la ventana de la figura 2.14 que nos pide los datos de la BD de la que crearemos la consulta. El botón conectar muestra una ventana donde seleccionamos la BD. También seleccionamos la tabla de la que obtendremos los datos. La nueva fuente de datos aparece en la lista de fuentes disponibles, como se ve en la figura 2.15.

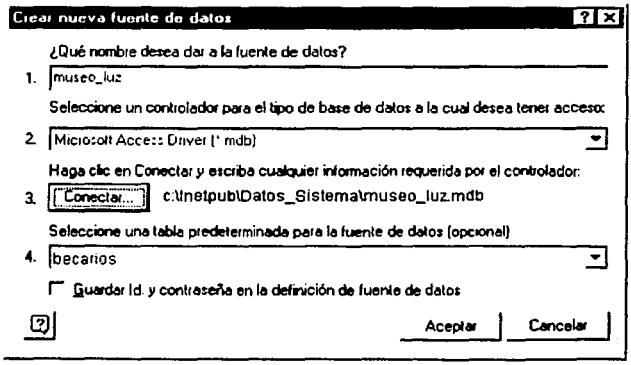

Figura 2.14 Nueva fuente de datos

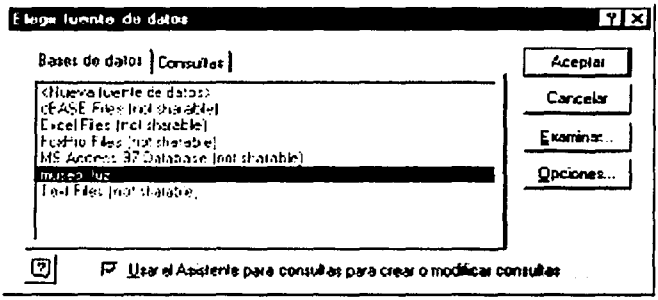

Figura 2.15 La nueva fuente de datos fue creada

Presionamos Aceptar y aparece el asistente para tablas (figura 2.16) que nos pide las tablas y los campos de los que obtendremos los datos.

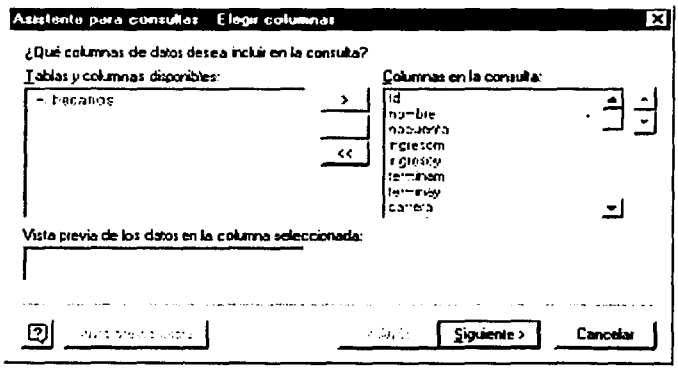

Figura 2.16 Elegir tablas y campos

Podemos establecer filtros para los datos que deseamos ver, como se aprecia en la figura 2.17.

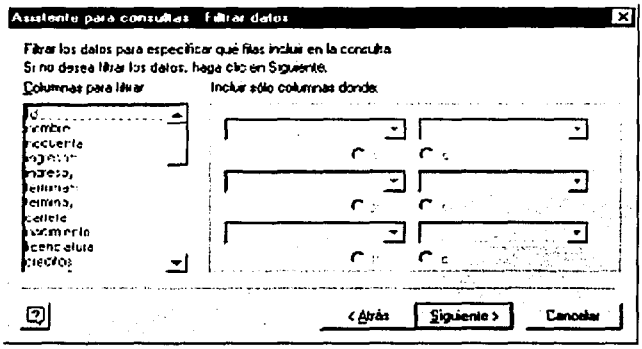

Figura 2.17 Filtrar datos

Seleccionamos el campo por el cual se van a ordenar los resultados.

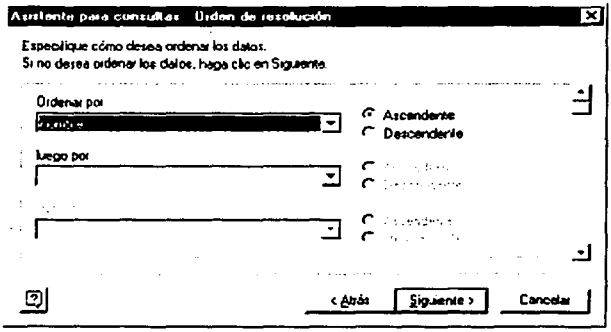

Figura 2.18 Orden de los datos

Para terminar, dejamos seleccionada la opción de Devolver datos a Microsoft Excel, como se ve en la figura 2.19 y pulsamos Finalizar.

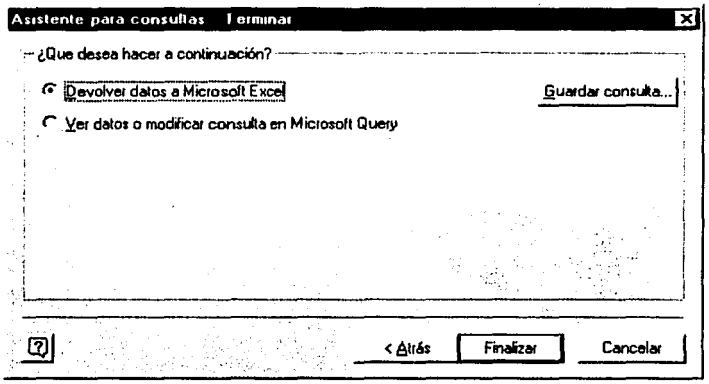

Figura 2.19 La consulta fue creada

El asistente nos pregunta dónde queremos que comiencen a aparecer los datos de la consulta. Podemos dejar la opción que tenga por omisión, como se ve en la figura 2.20.

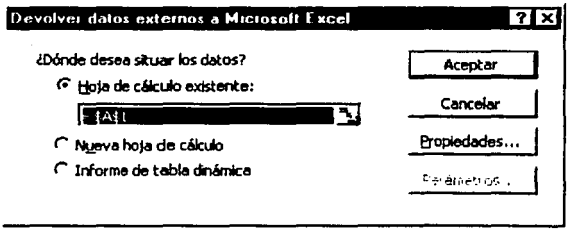

Figura 2.20 Posición de los datos de la consulta

Podemos seleccionar la opción de Propiedades para ajustar otros detalles de la consulta, como se ve en la figura 2.21.

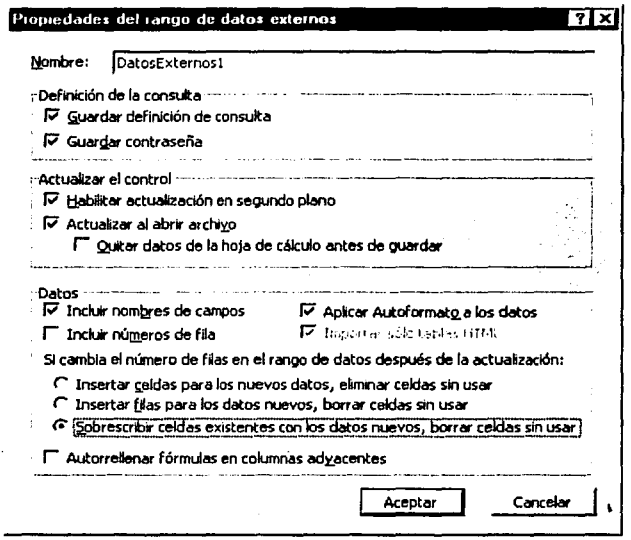

Figura 2.21 Propiedades de la consulta

Pulsamos Aceptar y los datos de la consulta aparecen en la hoja de Excel. Obviamente no mostramos esos datos, por la seguridad e integridad del sistema.

Menú de **Ayuda**

Contenido: El código siguiente


```
Private Sub SubContenido_Click()
 FormAyuda.Show
End Sub
```

Aparece la ventana de la figura 2.22. Nótese que no es como la ayuda que se puede encontrar en cualquier programa comercial. Esto se solucionará más adelante.

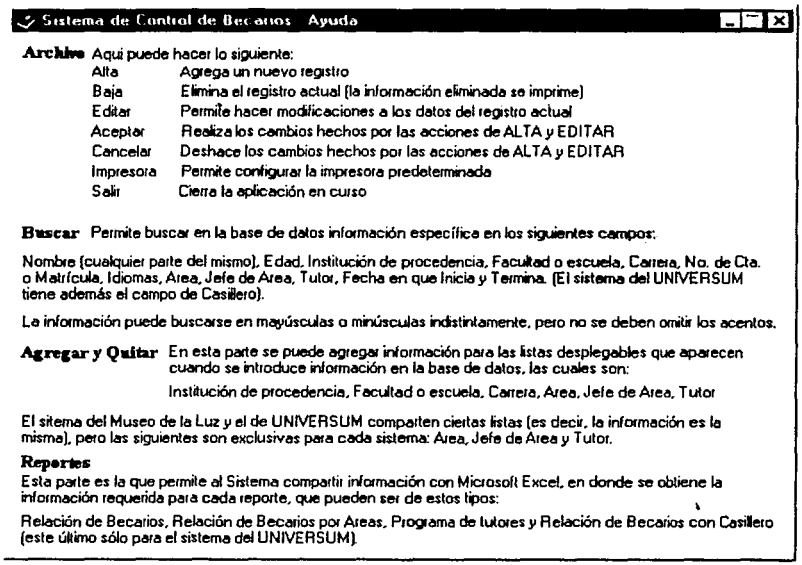

Figura 2.22 Ventana de Ayuda

Acerca de: Código

```
Private Sub SubAcercaDe_Click()
 frmAbout.Show
End Sub
```

Muestra la ventana de la Figura 2.23

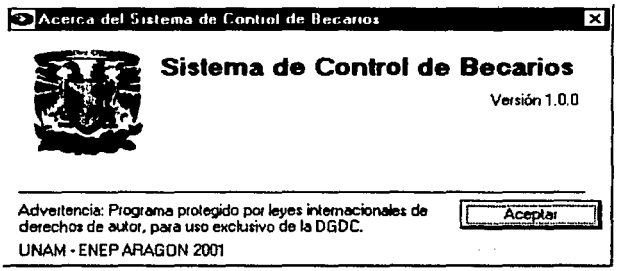

Figura 2.23 Ventana de "Acerca de...."

Ya debe haber notado que se puede acceder a los elementos de los menús más usados de maneras diferentes. Una es con el click normal con el ratón, otra es pulsando la letra que esté subrayada, y otra forma es por medio de las teclas especiales, identificadas por letras que acompañan algún elemento del menú. Por ejemplo, en el menú de Archivo se puede acceder a la opción Alta haciendo click en ella, pulsando la letra A (mayúscula o minúscula), o presionando la techa F1.

Teniendo ya la interfaz y la programación de cada parte del sistema, llegamos a la versión preliminar del mismo. Aunque aquí se muestra solo una versión preliminar, en realidad se tuvieron varias versiones anteriores a esta, pero por las razones ya mencionadas al inicio de este capítulo, y debido a que en lo general la mayoría de ellas permanecían igual, sólo se presenta la versión final, a manera de ejemplo, nunca reflejando la realidad completa del sistema por la confidencialidad hacia el mismo. Así mismo se especificaron las características de las herramientas utilizadas, así como el hardware necesario para su funcionamiento. Se describió la interfaz del sistema, y se mostró la parte importante del código de cada módulo.

Se realizaron varias pruebas al sistema, desde el punto de vista del desarrollador, las cuales fueron satisfactorias. Lo que sigue ahora es su utilización por parte del personal correspondiente de la Coordinación de Becarios, quienes nos proporcionarán los posibles errores o mejoras que se puedan agregar al sistema para su correcto funcionamiento. El resultado de esto se aborda en el capítulo siguiente.

3. EVALUACIÓN DEL SISTEMA

INTRODUCCIÓN

Como se mencionó en el capítulo anterior, el proceso para llegar a esta versión final fue muy largo. La realización de esta versión final del sistema local para la Coordinación de Becarios se llevó casi un año en su elaboración.

Fueron muchas reuniones para poder determinar, poco a poco, lo que realmente debía de contener el sistema. La versión preliminar funcionó correctamente, lo que permitió continuar con el paso siguiente, que fue agregar todas las opciones restantes para que estuviera completo. Además se realizaron cambios en la programación y se mejoró la interfaz, acercándose en mucho a aplicaciones comerciales, como se verá en el desarrollo de este capítulo.

Continuamos empleando la herramienta de programación Visual Basic 6; con el mismo hardware. Se mantiene la privacidad del código, como en el capítulo anterior.

Para esta versión final la Coordinación de Becarios trabajará con una PC con Windows 98, con las librerías necesarias que se instalan junto con el sistema, contando con 48 MB de memoria RAM en un monitor con una resolución de 800x600 pixeles con una profundidad de color de 32 Bits. Por el espacio de almacenamiento no hay problema ya que dicha PC cuenta con más de 1 GB de espacio disponible.

Es en esta PC donde el sistema local estará operando mientras se desarrolla la versión para Internet. Por tanto, por el momento sólo el personal de la Coordinación de Becarios podrá utilizarlo y posteriormente la versión en Internet permitirá que cualquier persona que lo requiera pueda acceder y utilizarlo.

3.1 Presentación del sistema al área a cargo para su utilización y detección de errores o mejoras

El sistema preliminar desarrollado en el capítulo anterior se instaló en una PC del área de la Coordinación de Becarios de la DGDC. Ya que el sistema en sí no es complejo, y debido a que la interfaz permite que el usuario se relacione rápidamente con el sistema, sólo fue necesaria una breve explicación a las personas (dos por el momento) que utilizarían el sistema. Se mantuvo contacto con el personal por si tenían alguna duda al estar usando ya el sistema. El sistema fue utilizado por el periodo de un mes y medio, observando lo siguiente.

En cuanto al funcionamiento no se encontraron errores. Lo que sí se observó es que había aspectos que no tenían la importancia necesaria para estar, como fueron algunas de las opciones del menú de Buscar. Los menús de Agregar y Quitar se reemplazaron por uno solo, llamado Listas, que permitía hacer ambas funciones. El menú de archivo fue ampliado, agregando funcionalidad al sistema. El menú de Reportes sólo tuvo un agregado. El menú de ayuda cambió un poco y mejoró bastante respecto de su antecesor. Además, a los elementos más usados de cada menú se les agregó un icono, y estos iconos también se agregaron a los cinco ya existentes debajo de los menús, dando un total de once iconos para las operaciones más usuales.

Todos estos cambios le dieron una presentación más profesional al sistema. A continuación se mostrará la mueva interfaz del mismo, así como los cambios hechos en la programación. Partes ya vistas se presentan de nuevo, como el menú Archivo, para mostrar los cambios que tuvieron. Así mismo, se presentan nuevos módulos y la forma de realizarlos, como es la forma de crear ayudas como las que tienen las aplicaciones comerciales.

3.2 Implementación de las correcciones y mejoras pertinentes al sistema

Al igual que en el capítulo anterior, seguiremos el esquema de mostrar la parte gráfica y de programación del sistema. Los resultados producidos por los cambios son notorios, tanto en la interfaz (figura 3.1) como en la programación de los módulos, la cuál se redujo en muchos casos. Para no perder la línea, se continúa utilizando el ejemplo del Museo de la Luz. Ahora sólo se comentará la parte del código que sea nueva. Si lo necesita puede consultar de nuevo el capítulo anterior por si no recuerda alguna parte del código mostrada aquí y que, por haber sido ya explicada, no se comenta de nuevo.

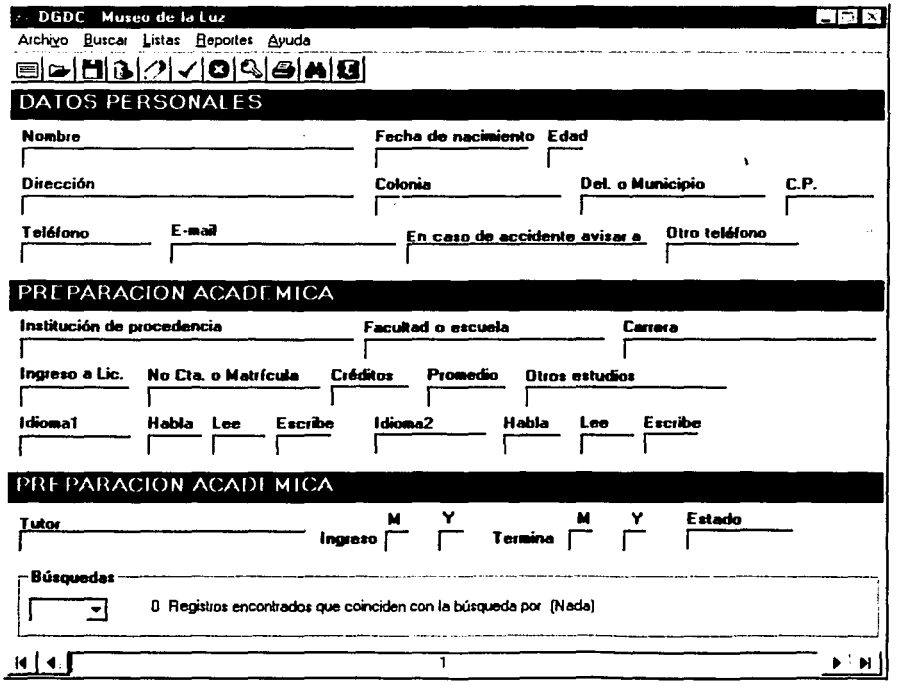

Figura 3.1 Interfaz final para el sistema del Museo de la Luz

Como se puede observar, los menús pasaron de ser 6 a 5. Los iconos que antes eran cinco ahora son once, y que de izquierda a derecha representan las opciones de: Nuevo, Abrir, Guardar como, Borrar, Editar, Aceptar, Cancelar, Ocultar, Imprimir, Buscar y Salir. Como se verá a continuación, cada elemento de un menú que lo requiera, tiene alguno de estos iconos, para que se puedan identificar fácilmente.

El menú de Archivo tiene las opciones principales de manipulación de los datos del sistema. Note que hay opciones nuevas (figura 3.2), y otras ya existentes cambiaron de nombre.

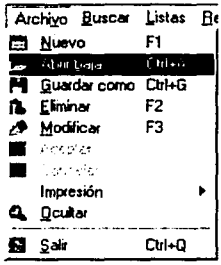

Figura 3.2 Opciones finales del Menú Archivo

Nuevo: Permite agregar un nuevo registro a la Base de Datos.
Abrir bajas: Permite abrir un archivo de texto donde fue guardado un registro que fue eliminado.
Guardar como: Permite guardar los datos del registro actual en un archivo de texto. Se utiliza cuando se va a eliminar algún registro y se quieren conservar los datos para referencias futuras.
Eliminar: Permite borrar el registro actual de la base de datos. Antes de eliminar aparece un cuadro de diálogo para confirmar la acción.
Modificar: Permite que los campos de texto del sistema se habiliten para modificar los datos del registro actual. Normalmente los campos de entrada de datos están deshabilitados para evitar cambios no deseados en los datos.
Aceptar/Cancelar: Realiza o cancela los cambios provocados por las opciones Nuevo o Modificar.
Impresión: Permite configurar la impresora predeterminada, así como seleccionar los campos y los registros a imprimir.
Ver/Ocultar: Muestra u oculta los datos de los registros, así como de las opciones de los menús. Esto es por seguridad, por si el usuario del sistema tiene que dejar por un

momento la computadora o simplemente no quiere que alguien no autorizado pueda ver o manipular la información de la base de datos.
Salir: Cierra el sistema. Un cuadro de diálogo pregunta para confirmar la acción.

El menú de Buscar (figura 3.3) permite hacer búsquedas por cualesquiera de los elementos listados. Sus elementos fueron reducidos al eliminar a los que no eran necesarios y al agrupar otros como fueron los correspondientes a las fechas de inicio y de término.

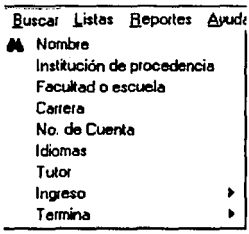

Figura 3.3 Opciones finales del menú Buscar

El menú de Listas (figura 3.4) se emplea para llenar las listas de opciones que aparecen cuando se está editando o agregando un nuevo registro a la base de datos. Este menú reemplaza a los anteriores Agregar y Quitar, cuya programación había sido extensa.

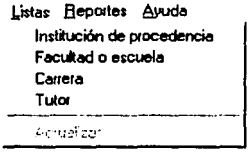

Figura 3.4 Opciones finales del menú Listas

En el menú de Reportes (figura 3.5), los nombres de los reportes disminuyó, pero siguen funcionando exactamente igual. La opción de Cumpleaños se utiliza para buscar en la base de datos todos los becarios que cumplan años en el mes corriente. Esta es una de las ampliaciones finales que se le hicieron al sistema.

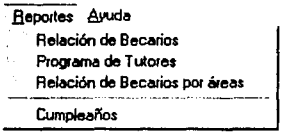

Figura 3.5 Opciones finales del menú Ayuda

La figura 3.6 muestra el menú de Ayuda.

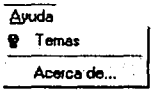

Figura 3.6 Opciones finales del menú Ayuda

Temas: Contiene todos los tópicos de ayuda que puedan auxiliar al usuario del sistema.
Acerca de: Muestra un cuadro de diálogo con los créditos y la licencia del programa.

El recuadro de Búsquedas y el control de desplazamiento permanecen exactamente igual que antes, por lo que ya no se muestran.

Ahora procederemos a mostrar la nueva programación del sistema. Las convenciones para comentar el código es igual que en el capítulo anterior, las partes que ya se hayan explicado anteriormente ya no se explicarán de nuevo. Seguimos el esquema de mostrar el código por el orden en cada menú.

Menú de **Archivo**

Nuevo

```
Private Sub Nuevo_Click()
On Error GoTo Fallo
Dim cuenta As String, linea As String
Dim esta As Boolean                'variable que indica si el registro ya se encontraba
                                   'en la base de datos
cuentata = InputBox("Escriba el Número de cuenta del registro:", "Nuevo")
If (cta <> "") Then
'busca el no. de cta. en la base de datos
'se utiliza la misma función que realiza búsquedas en otro de los menús
```

```
'el primer parámetro identifica cuál campo se emplea para la búsqueda
'el segundo parámetro pasa el dato buscado
FunBuscar 15, cuenta
If esta Then          'si el nuevo becario que se quiere agregar se encuentra ya en la
                      base 'de datos o ya fue becario anteriormente, el sistema lo avisa
                      con un 'mensaje al usuario
  MsgBox "Esa persona ya es o fue becario de la DGDC", 64, "Información"
Else          'si no, puede ser agregado a la base de datos
  cmdAdd_Click      'función que agrega el registro a la base de datos
End If
Exit Sub
Fallo:
 MsgBox Err.Description, 16, "Error"
End Sub
```

Abrir bajas

```
Private Sub Abrir_Click()
On Error GoTo Fallo
Dim c As Integer
'CDBGuardar es el nombre del control CommondDialogBox
CDBGuardar.ShowOpen   'muestra una ventana estándar de Windows para
                      'abrir un archivo
strFic = "Notepad.exe"    'variable utilizada como primer parámetro del Shell
If (CDBGuardar.FileName <> "") Then
'en la siguiente línea establecemos el Shell completo y se ejecuta
 Shell strFic & " " & CDBGuardar.FileName, vbNormalFocus
Exit Sub

Fallo:
 MsgBox Err.Description,48,"Error"
End Sub
```

Guardar como

```
Private Sub Guardar_Click()
On Error GoTo Fallo
CDBGuardar.ShowSave   'muestra un cuadro estándar de Windows
                      'para guardar un archivo
Dim ruta As String
ruta = CDBGuardar.FileName    'obtiene la ruta donde se desea guardar el archivo
Open ruta For Output As #11         'abre el archivo para escritura
                      'el número 11 es para identificar el archivo
 Print #11, DATOS             'escribe en el archivo el contenido de DATOS
 .....
 Close #11                    'cierra el archivo
Exit Sub

Fallo:
```

```
 MsgBox Err.Description, 48, "Error"
End Sub
```

Eliminar

```
Private Sub Eliminar_Click()
On Error GoTo Fallo
Dim c As Integer
'variable que guarda el resultado de la opción pulsada
' al ser un mensaje que tiene botones de SI y NO
c = MsgBox("¿Eliminar este registro?", 32 + vbYesNo, "Eliminar")
'si se pulsa SI, se llama a la función que elimina el registro
'adecuado para prevenir que se borren registros accidentalmente
If (c = 6) Then FunElimina
Exit Sub

Fallo:
 MsgBox Err.Description, 48, "Error"
End Sub
```

Modificar

```
Private Sub Modificar_Click()      'trabaja con los datos del registro actual
 FunMenus  'función que deshabilita los elementos de los menús
           'que no deben de estar accesibles mientras se edita el registro, como
           'pudiera ser el de Nuevo, por ejemplo
 FunHabilitar        'habilita los cuadros de texto para poder editar los datos
 cmdEdit_Click       'le indica al sistema que se están editando los datos
End Sub
```

Aceptar

```
Private Sub Aceptar_Click()              'valida la última acción, como se explicó
anteriormente
cmdUpdate_Click   'actualiza la base de datos
FunMenus          'habilita de nuevo los menús
FunHabilitar      'deshabilita los cuadros de texto
End Sub
```

Cancelar

```
Private Sub Cancelar_Click()  'cancela la última acción, como se explicó anteriormente
cmdCancel_Click   'cancela cualquier cambio en la base de datos
FunMenus          'habilita de nuevo los menús
FunHabilitar      'deshabilita los cuadros de texto
End Sub
```

Impresión : Sólo se muestra la parte que permite configurar la impresora.

```
Private Sub ConfigurarImpresora_Click()
On Error Resume Next     'otra forma de controlar los errores. Cuando ocurre un error,
                         'la ejecución no se interrumpe, sino que se continúa con la
                         'línea siguiente a donde ocurrió el error. Esto evita que se
                         'salga del programa prematuramente
CDBGuardar.ShowPrinter 'muestra la ventana estándar para configurar
                         'la impresora predeterminada, o seleccionar la impresora
                         'a utilizar
End If
End Sub
```

Ocultar: Por ser uno de los aspectos que proveen seguridad su código no se muestra. Basta con decir que evita que personas ajenas trabajen con el sistema, como se explico anteriormente.

Salir: Consta de dos partes, al pulsar esta opción ocurre la ejecución del código siguiente

```
Private Sub Salir_Click()
Unload Me            'llama a la propiedad UNLOAD del formulario principal
End Sub
```

Que a su vez llama a este código, que es especialmente útil para evitar que se cierre el sistema por equivocación.

```
Private Sub Form_Unload(Cancel As Integer)  'Cancel es la variable de Visual Basic
                                            'que determina si un formulario se cierra o no
Dim m As Integer
'pregunta si en verdad se quiere cerrar el sistema
m = MsgBox("¿Salir del Sistema del Museo de la Luz?", vbYesNo + vbQuestion,
"Cerrar")
If (m = 7) Then       'si la respuesta es NO
 Cancel = 1   'cancela cerrar el sistema (regresa al sistema)
Else   'de lo contrario
 Screen.MousePointer = vbDefault        'regresa el puntero del mouse a su forma
normal,
                                        'si fuera necesario
 End          'termina la ejecución del programa
End If
End Sub
```

Menú de **Buscar**: El código es similar para cualquier tipo de búsqueda. Primero, se solicita el dato a buscar:

```
Private Sub QueSeBusca_Click()
'QueSeBusca es cualquiera de los elementos listados por los que
'se puede buscar: Nombre, Carrera, etc.
Dim cadena As String
'cadena guarda el dato que se busca
cadena = InputBox("Escriba el dato a buscar", "Buscar")
If (cadena <> "") Then
 FunBuscar 0, cadena        'llama a la función de búsqueda, como se vio en
                            'la opción Nuevo del menú Archivo
End If
End Sub
```

Para esta parte se realizó una función que podía hacer búsquedas por cualquiera de los elementos listados. Al seleccionar alguno de ellos, aparece un cuadro de diálogo que pregunta por el dato a buscar. Luego, ese dato se envía a dicha función, junto con un identificador que determina cuál campo es el que se empleará para la búsqueda.

Menú de **Listas**: El código es el mismo para cada opción de este menú. Lo que cambia es el nombre del archivo que se abre, que es el que contiene los valores de la lista de opciones.

```
Private Sub Lista_Click()  'Lista es cualquiera de las opciones de este menú
On Error GoTo Fallo
Dim c As Integer, file As String
file = "archivo.txt"       'establece el nombre del archivo a abrir, el cual corresponde
                           'al elemento seleccionado: Institución, Carrera, etc.
strFic = "Notepad.exe"     'establece que el Notepad será la aplicación que
                           'abra dicho archivo
Shell strFic & " " & file, vbNormalFocus 'arma el Shell y lo ejecuta
Exit Sub

Fallo:
 MsgBox Err.Description, 48, "Error"
End Sub
```

Menú de **Reportes**: Igual que el anterior. El código es el mismo para todos, cambiando sólo el nombre del archivo. Aquí lo importante es que el documento de Excel que se abre tiene establecidos criterios que permiten obtener y filtrar los datos de la información que maneja el sistema, presentando sólo aquellos que son requeridos, de acuerdo al tipo de reporte seleccionado.

```
Private Sub Relacion_Click()      'Relación representa al elemento seleccionado del
menú
On Error GoTo Fallo
Dim c As Integer
Dim strFic As String, strParam As String
'establece la ruta donde se encuentra el programa Excel, de lo contrario no funcionará
```

```
strFic = "C:\Archivos de programa\Microsoft Office\Office\EXCEL.EXE"
'abre el archivo de Excel que se seleccionó
strParam = "reporte.xls"
Shell strFic & " " & strParam, vbNormalFocus
End If
Exit Sub
Fallo:
 MsgBox Err.Description, 16, "Error"
End Sub
```

Se mantiene todo para este menú. El procedimiento para crear los documentos de Excel es igual que en el capítulo anterior.

Menú de **Ayuda**

Temas: Un cambio importante: se eliminó la ventana anterior y se sustituyó por un archivo de ayuda muy similar al de cualquier aplicación comercial, como se puede apreciar en la figura 3.7.

```
Private Sub Temas_Click()
On Error GoTo Fallo
Const cdlHelpContents = &H3    'muestra el tema de contenido del archivo de Ayuda
'llama al archivo de ayuda
Call WinHelp(Me.hWnd, "C:\sistema\ayuda.hlp", cdlHelpContents, 0&)
Exit Sub

Fallo:
 MsgBox Err.Description, 16, "Error"
End Sub
```

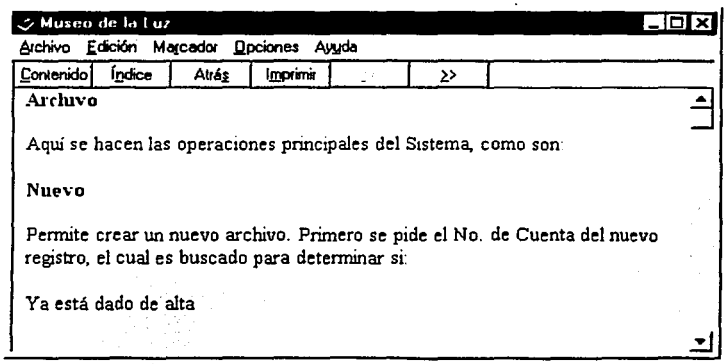

Figura 3.7 Ayuda del Sistema

El cambio fue bastante positivo, aunque es un archivo simple de ayuda da un toque bastante profesional al sistema.

Para los programadores que deseen incorporar un archivo de ayuda para sus aplicaciones, se describe la forma de crear un archivo de ayuda similar a este.

Primero se necesita un compilador de archivos de ayuda, ya sea conseguirlo en Internet o incluso Microsoft trae el suyo en los CDs de instalación de Visual Studio. Los compiladores de ayudas pueden ser muy simples o muy complicados, gratuitos o de prueba. El que se utilizó fue el conocido como HCW (Microsoft Help Workshop), figura 3.8, que es un sencillo compilador de ayudas gratuito de apenas 531 Kb.

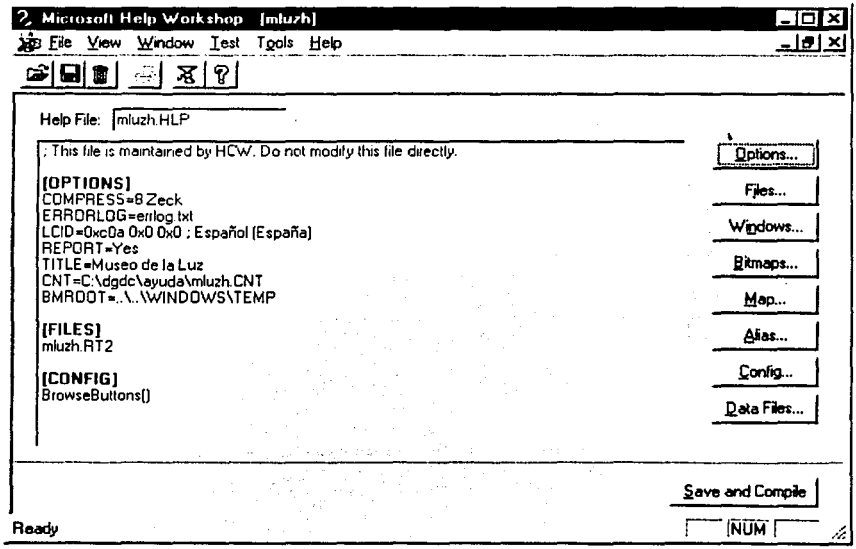

Figura 3.8 Pantalla del HCW

Este es un compilador que por si mismo permite crear el archivo de ayuda que necesitamos, pero de entrada se ve muy complicado. Por eso utilizamos además otra herramienta gratuita también llamada RTF2HLP (figura 3.9), la cual permite (teniendo un compilador, como el HCW) convertir un documento de formato RTF en un archivo de ayuda tipo Windows.

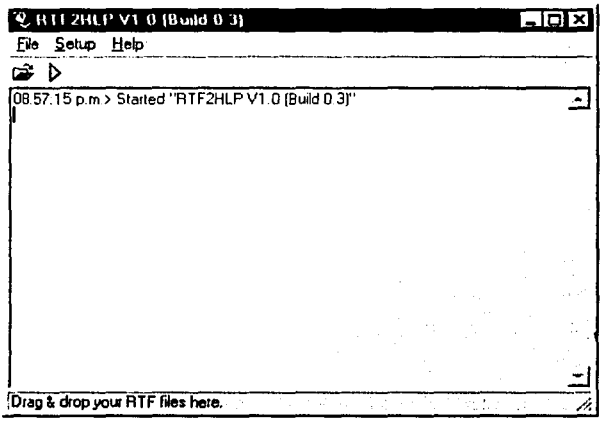

Figura 3.9 Pantalla del RTF2HLP

Teniendo ya estas herramientas, procederemos a crear el documento RTF donde colocaremos todo lo que consideremos necesario para nuestro archivo de ayuda. Debemos insertar un Salto de Línea entre cada uno de los temas para que queden divididos en el archivo de ayuda. Además, es necesario que cada parte del documento que sea algo importante en la ayuda, como pueden ser los nombres de los menús, deben de ser un Marcador, es decir, un lugar de referencia para un tema en específico.

Por ejemplo, consideremos que crearemos un archivo de ayuda cuyo documento RTF es el siguiente:

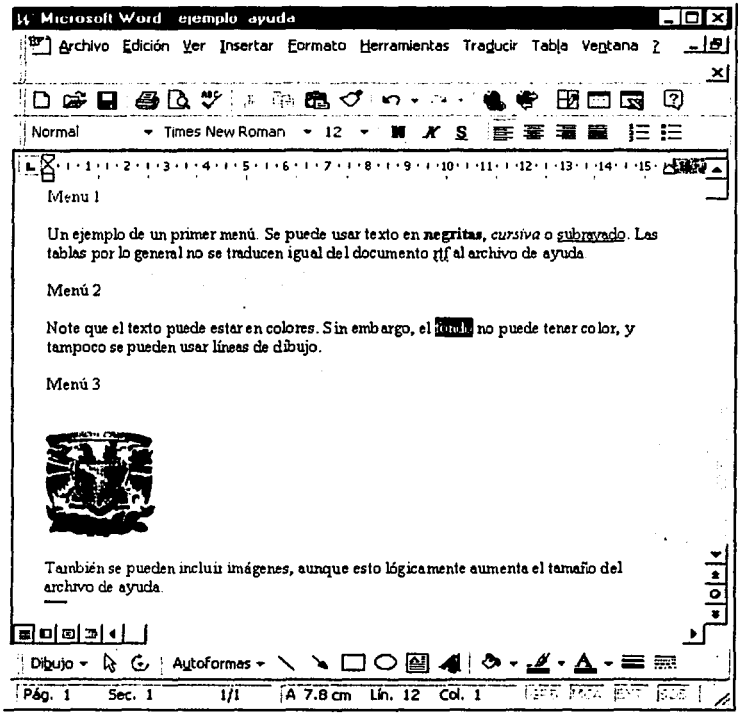

Figura 3.10 Ejemplo de documento RTF

Para personalizar el título del archivo de ayuda vaya al menú Archivo ->Propiedades y edite el título del documento. Para insertar un salto de página, vaya al menú Insertar -> Salto... Salto de página, como se ve en la figura 3.11.

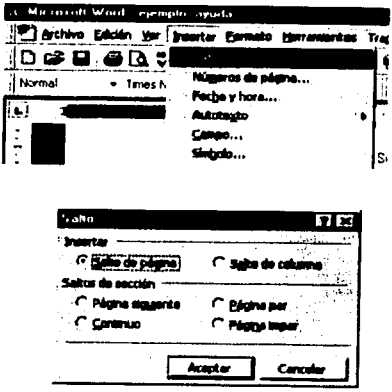

Figura 3.11 Insertar salto

Con esto se divide el texto por páginas. Para insertar un marcador seleccione el texto deseado, vaya al menú Insertar -> Marcador y de un nombre al marcador sin espacios en blanco, es decir, una sola palabra. La figura 3.12 muestra este proceso.

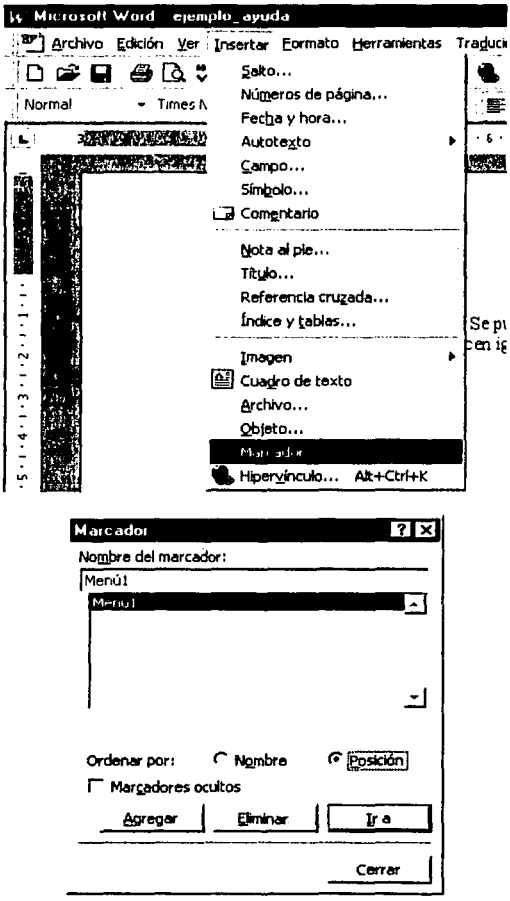

Figura 3.12 Insertar Marcador

Haga esto para cada marcador que desee agregar. Puede decidir cómo es que aparecerán los marcadores en el archivo de ayuda, por Nombre o por Posición, según haya seleccionado al agregar el marcador.

Una vez que ya haya insertado los saltos de línea y los marcadores necesarios guarde el archivo y ciérrelo; a continuación abra el programa RTF2HLP y seleccione el archivo que desea convertir. Antes de hacer esto es necesario que le indique al programa en dónde se encuentra el compilador que debe utilizar (figura 3.13), para ello vaya al menú Setup -> Configure. Las opciones del compilador se pueden dejar como aparecen.

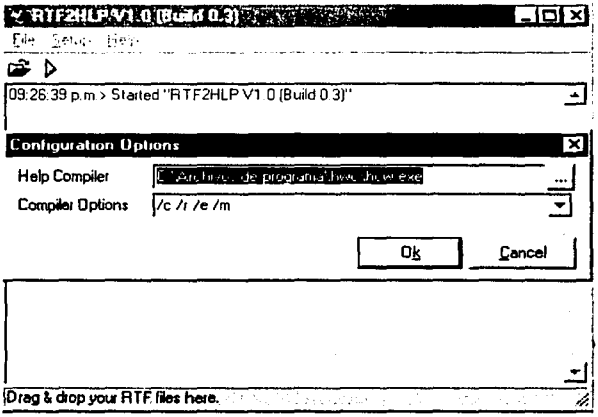

Figura 3.13 Estableciendo la ruta del compilador

Al compilar el archivo de ayuda, la pantalla de programa se torna como la que se aprecia en la figura 3.14.

El tiempo que tarde en hacer la conversión depende del tamaño del archivo y si tiene imágenes incluidas. La ventana del programa muestra el avance y los resultados de la compilación: si hubo errores, el tiempo empleado, etc. Si todo sale bien, al terminar de hacer esto aparece nuestro archivo de ayuda, como el que se ve en la figura 3.15 para el case del ejemplo.

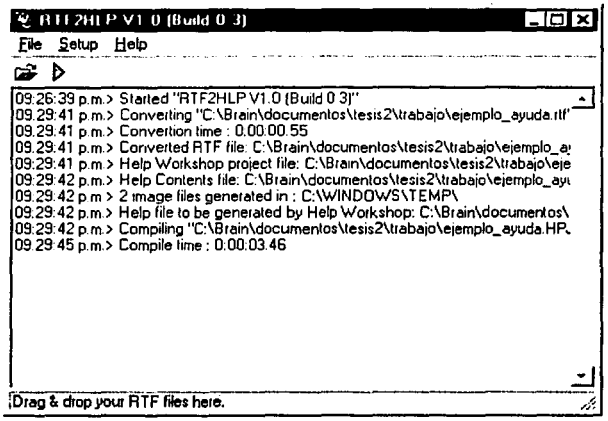

Figura 3.14 Compilación del documento RTF

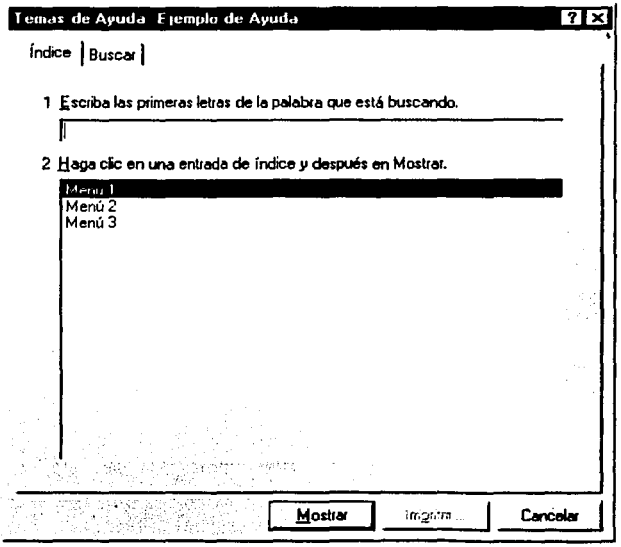

Figura 3.15 Índice del archivo de ayuda de ejemplo

En esta pantalla aparecen los marcadores que hayamos creado, en este caso tres. Al pulsarlos (o al escribir su nombre en el campo de arriba) iremos al tema especificado. Por ejemplo, al seleccionar el tema Menú 3, aparece el contenido de la figura 3.16.

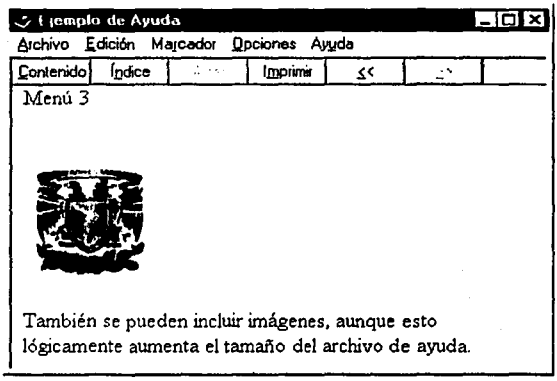

Figura 3.16 Tema 3 de la ayuda de ejemplo

Con esto terminamos con el archivo de ayuda. Realmente son herramientas muy simples para hacer un archivo de este tipo, pero como una primera aproximación es un muy buen ejemplo. Existen más opciones con más características, busque las que le den mejores opciones para su aplicación.

Acerca de: El código

```
Private Sub SubAbout_Click()
On Error Resume Next
Dim X As Long       'variable para reproducir un sonido cuando se muestra la ventana
FAcercaDe.Show   'formulario "Acerca de..."
'ejecuta una vez el sonido en la ruta especificada
X = sndPlaySound("C:\sonido.wav", SND_SYNC Or SND_NODEFAULT)
End Sub
```

Muestra la ventana de la figura 3.17, y se produce un sonido al mostrarla.

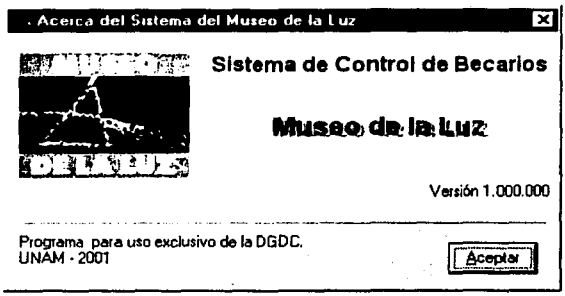

Figura 3.17 Diálogo "Acerca de..."

Para agregar los iconos a los menús, primero debemos declarar el siguiente código en la parte de (General) - (Declaraciones)

```
'declaramos las librerias necesarias para accesar a los menús de nuestra aplicación
Private Declare Function GetMenu Lib "User32" (ByVal hWnd As Long) As Long
Private Declare Function GetSubMenu Lib "User32" (ByVal hMenu As Long, ByVal npos
As Long) As Long
Private Declare Function GetMenuItemID Lib "User32" (ByVal hMenu As Long, ByVal
npos As Long) As Long
Private Declare Function SetMenuItemBitmaps Lib "User32" (ByVal hMenu As Long,
ByVal nPosition As Long, ByVal wFlags As Long, ByVal hBitmapUnchecked As Long,
ByVal hBitmapChecked As Long) As Long
```

Después podemos generar una función que se encargue de poner nuestros iconos. A los iconos se accede así: de izquierda a derecha, en orden ascendente comenzando desde cero representan los menús principales, y lo mismo de arriba hacia abajo para cada menú. Para el sistema del Museo de la Luz tenemos entonces cinco menús principales (de 0 a 4).

```
Public Function FunIconos( )
'Obtenemos el Menu
   hMenu = GetMenu(Me.hWnd)
'Asignamos el primer Submenu
   hSubMenu = GetSubMenu(hMenu, 0)
'Asignamos el icono de alta
   MenuID = GetMenuItemID(hSubMenu, 0)
```

```
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(0).Picture,
PicMenus(0).Picture)
'Asignamos el icono de abrir
    MenuID = GetMenuItemID(hSubMenu, 1)
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(1).Picture,
PicMenus(1).Picture)
'Asignamos el icono de guardar
    MenuID = GetMenuItemID(hSubMenu, 2)
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(2).Picture,
PicMenus(2).Picture)
'Asignamos el icono de eliminar
    MenuID = GetMenuItemID(hSubMenu, 3)
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(3).Picture,
PicMenus(3).Picture)
'Asignamos el icono de config modificar
    MenuID = GetMenuItemID(hSubMenu, 4)
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(4).Picture,
PicMenus(4).Picture)
'Asignamos el icono aceptar
    MenuID = GetMenuItemID(hSubMenu, 5)
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(5).Picture,
PicMenus(5).Picture)
'Asignamos el icono de cancelar
    MenuID = GetMenuItemID(hSubMenu, 6)
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(6).Picture,
PicMenus(6).Picture)
'ocultar
    MenuID = GetMenuItemID(hSubMenu, 8)
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(7).Picture,
PicMenus(7).Picture)
'salir
    MenuID = GetMenuItemID(hSubMenu, 10)
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(10).Picture,
PicMenus(10).Picture)

'Asignamos menu buscar
    hSubMenu = GetSubMenu(hMenu, 1)
'buscar nombre
    MenuID = GetMenuItemID(hSubMenu, 0) 'buscar nombre
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(9).Picture,
PicMenus(9).Picture)

'Asignamos ultimo menu
    hSubMenu = GetSubMenu(hMenu, 4)
'Asignamos el icono de ayuda
    MenuID = GetMenuItemID(hSubMenu, 0)
    X = SetMenuItemBitmaps(hMenu, MenuID, &H4, PicMenus(11).Picture,
PicMenus(11).Picture)
End Function
```

Para bloquear (deshabilitar) el botón de cierre de las ventanas primero se debe declarar el API que controla este evento, lo cual se hace en la parte (General) - (Declaraciones) ya sea del formulario principal o bien desde un módulo externo. La figura 3.18 muestra el botón habilitado.

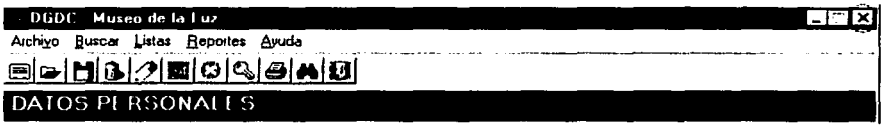

Figura 3.18 Botón de cierre habilitado

El código es el siguiente:

```
'Definición de la API
Declare Function GetSystemMenu Lib "User32" (ByVal hWnd As Integer, ByVal bRevert
As Integer) As Integer
Declare Function RemoveMenu Lib "User32" (ByVal hMenu As Integer, ByVal nPosition
As Integer, ByVal wFlags As Integer) As Integer
'Definicion de la constante
Global Const MF_BYPOSITION = &H400
```

Después basta con poner el siguiente fragmento de código en la parte Load del Formulario:

```
MenuSistema% = GetSystemMenu(hWnd, 0)
Res% = RemoveMenu(MenuSistema%, 6, MF_BYPOSITION)
```

Lo que produce el resultado de la figura 3.19.

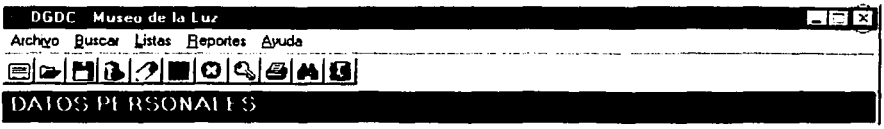

Figura 3.19 Botón de cierre deshabilitado

3.3 Obtención de la versión final del Sistema Local

Después de haber realizado todas las correcciones y mejoras al sistema, las cuales surgieron de la evaluación de la versión preliminar del mismo por parte del personal de la Coordinación de Becarios, llegamos a la versión final del mismo.

Finalmente, para concluir el desarrollo del sistema local para el control de becarios de la DGDC se debe instalar el sistema en el área a cargo. Sin embargo, la versión que operará definitivamente es la que se podrá utilizar vía Internet, por lo que en el tiempo que se tarde en migrar este sistema local a su versión en Internet el personal trabajará con el sistema ya terminado para que se familiarice con el funcionamiento general del sistema, para que el cambio al sistema en Internet sea lo menos accidentado.

En el capítulo siguiente se trata el proceso para crear los paquetes de instalación así como los aspectos relacionados con la instalación de los mismos y la configuración requerida para que funcione el sistema desarrollado.

4. TRABAJANDO CON EL SISTEMA

INTRODUCCIÓN

Bien, finalmente el sistema local creado en Visual Basic ha sido terminado y será instalado en la Coordinación de Becarios para que lo utilicen.

La instalación se realizó finalmente en una computadora con Windows 98, con 64 MB en RAM y espacio en disco duro de 1 GB. El archivo de instalación fue de 15 MB aproximadamente, incluyendo las librerías necesarias para que el sistema funcione, así como los archivos de datos necesarios para el mismo.

La capacitación se realizó en un par de visitas al departamento, debido a que las características del sistema permiten reconocer rápida y fácilmente las tareas que se deben realizar. Además, al realizar todas las pláticas con el personal del departamento para establecer las partes necesarias se utilizaron las versiones preliminares, lo que permitió que el personal se familiarizara rápidamente con el sistema.

El sistema se dejó en funcionamiento en Octubre del 2001 por un periodo de 3 meses, dentro de los cuales el personal lo evaluó y determinó si las características incluidas en el eran suficientes y acertadas para el trabajo que debe desempeñar. Recordemos que la versión final es la que será utilizada vía Internet, sin embargo esta versión local proporcionó la información necesaria para el correcto funcionamiento del sistema definitivo.

4.1Instalación del Sistema

La aplicación está lista, sólo falta crear el paquete de instalación del mismo. Una vez terminado el proyecto en Visual Basic 6, creamos el ejecutable del mismo. El programa del sistema del Museo de la Luz es de 270 Kb, al igual que el del UNIVERSUM (se manejan por separado los sistemas de cada museo). Sin embargo, para que los sistemas funcionen es necesario añadirle a la instalación las librerías necesarias para que funcionen en cualquier PC con Windows, aún cuando no tengan instalado el Visual Basic 6. Para hacer eso utilizamos el Asistente para empaquetado y distribución que acompaña al Visual Studio (figura 4.1).

1. Pulsamos primero el botón de Examinar para localizar nuestro proyecto de Visual Basic (mantenemos el ejemplo del Museo de la Luz).
2. Seleccionamos la opción de Empaquetar, que nos creará el archivo de instalación con todas las librerías necesarias.

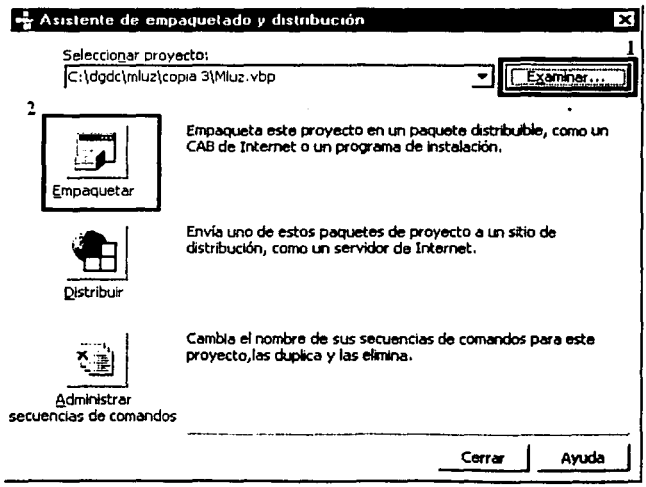

Figura 4.1 Asistente para empaquetado y distribución

Después aparece el asistente, que nos pide el tipo de empaquetado para nuestra aplicación (figura 4.2). Seleccionamos la primera, Paquete de instalación estándar, y pulsamos Siguiente.

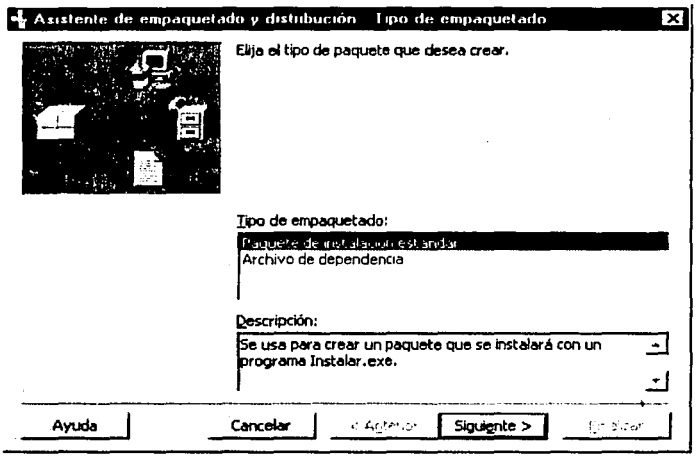

Figura 4.2 Tipo de empaquetado

A continuación debemos especificar un directorio en donde se guardará el paquete creado (figura 4.3). Por omisión el asistente sugiere crear una carpeta llamada Paquete dentro de mismo directorio donde se encuentre nuestro proyecto.

Quizá aparezca una pantalla que nos indica que no ha encontrado dependencias para ciertos archivos (figura 4.4). Esto no representa problema para nuestra instalación, simplemente pulsamos Aceptar.

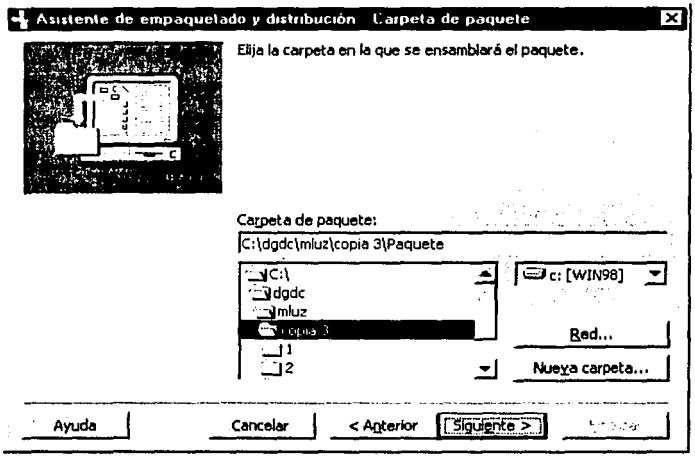

Figura 4.3 Carpeta de paquete

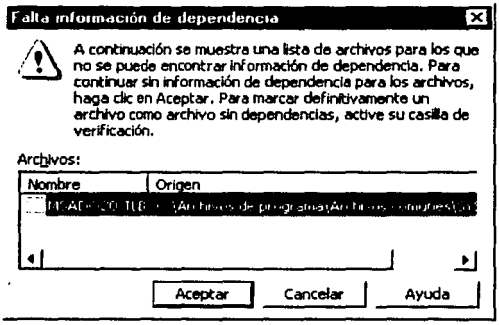

Figura 4.4 Archivos sin dependencias

Ahora aparece una ventana que nos muestra los archivos incluidos en nuestro paquete de instalación (figura 4.5). Es importante señalar que sólo se ha incluido el ejecutable de nuestra aplicación y las librerías necesarias, pero faltan otros elementos como los

archivos de ayuda y las BD del sistema. Pulsando el botón Agregar podemos incluir estos archivos, como lo muestra la figura 5.6.

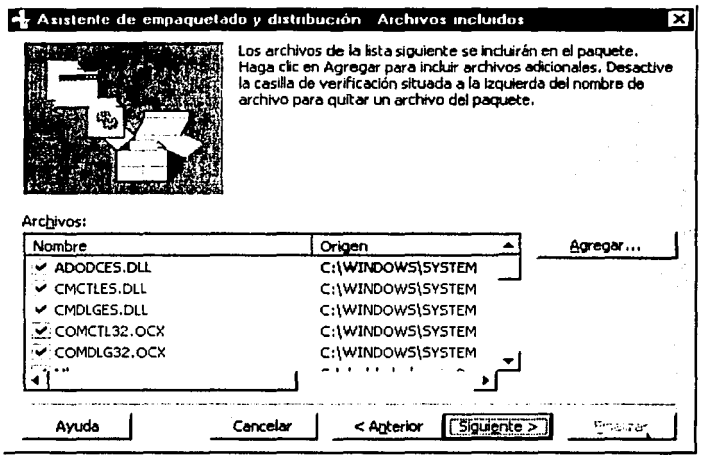

Figura 4.5 Archivos incluidos

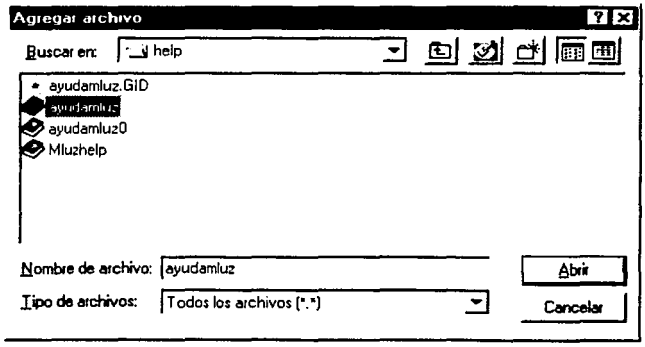

Figura 4.6 Agregar archivos

Como se trata del sistema local, sólo incluiremos las BD que contienen los becarios de cada museo. Seleccionamos el tipo de paquete que deseamos crear (figura 4.7). La opción de un único archivo .cab es recomendable si se cuenta con un grabador de CDs, ya que le archivo suele ocupar más de 10 MB, mientras que la opción de múltiples archivos .cab es por si se desea utilizar disquetes para la instalación, lo cual no es recomendable. En nuestro caso seleccionamos la opción de archivo único.

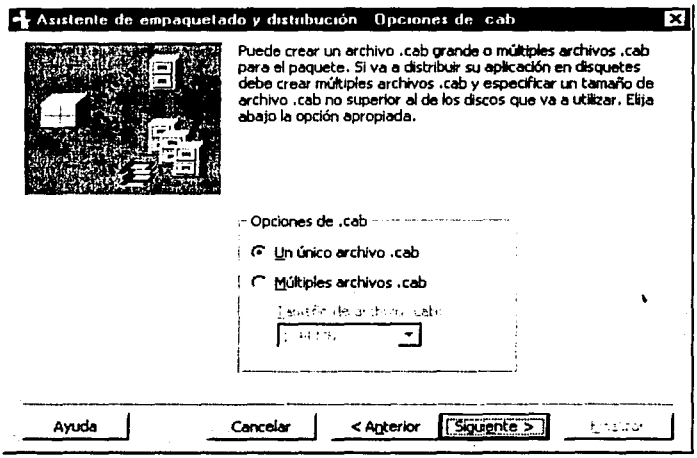

Figura 4.7 Tipo de archivo de instalación

Le damos un título a nuestra aplicación, que es el que se mostrará cuando se esté ejecutando el paquete de instalación, como se ve en la figura 4.8.

La figura 4.9 muestra la pantalla que nos informa dónde se instalaran los accesos directos de la aplicación, dentro del grupo de programas de la computadora.

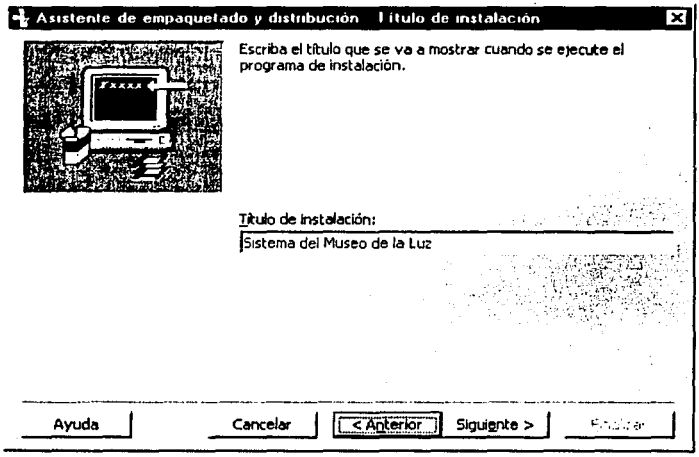

Figura 4.8 Título de la instalación

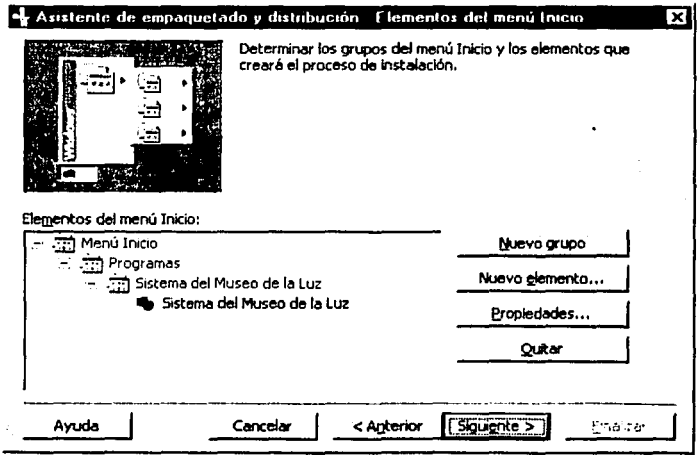

Figura 4.9 Carpeta en el grupo de Programas

Ahora vemos que el asistente nos informa en dónde serán instalados los diferentes archivos que componen nuestra instalación, como se ve en la figura 4.10. Por omisión los archivos propios de la aplicación (ejecutables, ayudas, BD, etc.) se instalan en la carpeta donde quede instalado el ejecutable del sistema, mientras que las librerías se instalarán donde el sistema crea que deben de ir.

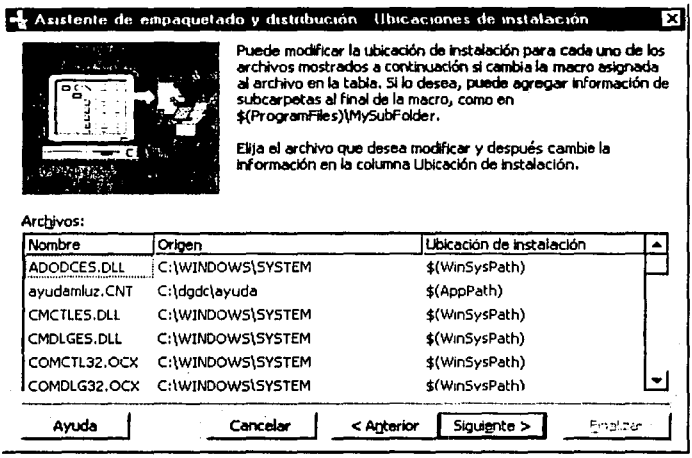

Figura 4.10 Ubicación de los archivos de instalación

WinSysPath hace referencia a la ruta de la carpeta System. AppPath es la ruta de la aplicación, la cual se determina al momento de ejecutar el paquete de instalación.

A continuación se nos pregunta si queremos que los archivos de nuestra aplicación (sin incluir las librerías) se instalen como compartidos (figura 4.11). Mantenemos las casillas sin seleccionar para que se instalen como no compartidos.

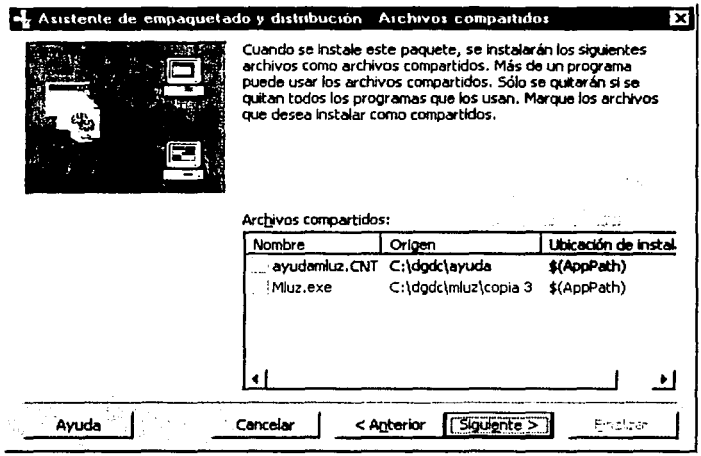

Figura 4.11 Archivos compartidos

El asistente ha obtenido los datos necesarios para crear el paquete de instalación. Se nos pide un nombre para distinguir el paquete que hemos creado. Pulsamos Finalizar y el paquete de instalación se comienza a crear. Las figuras 4.12 y 4.13 muestran la parte final de este proceso.

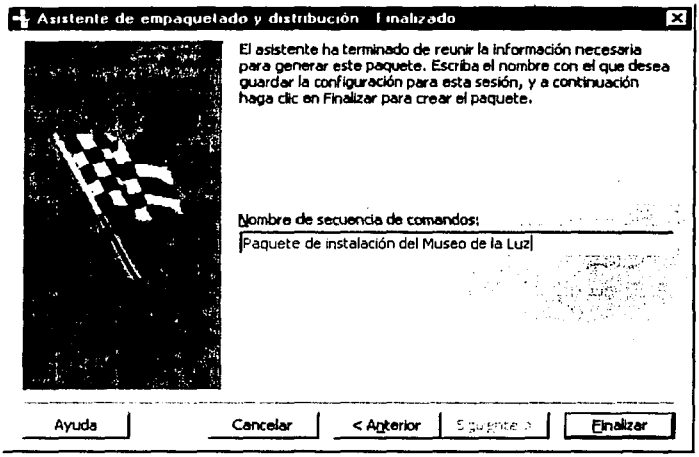

Figura 4.12 Fin de la recolección de datos del paquete de instalación

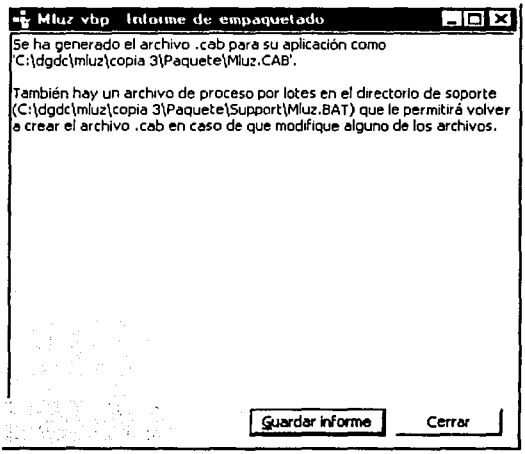

Figura 4.13 El paquete de instalación fue creado

El paquete de instalación fue creado. Cerramos el asistente y ya tenemos el paquete de instalación en la carpeta que especificamos. El contenido de la carpeta Paquete (donde guardamos el paquete de instalación) lo grabamos en un CD para distribuirlo como el disco de instalación.

Ahora procederemos a instalar el sistema. Ejecutamos el archivo Setup.exe que se generó al crear el paquete de instalación. Vemos una pantalla como la que se muestra en la figura 4.14.

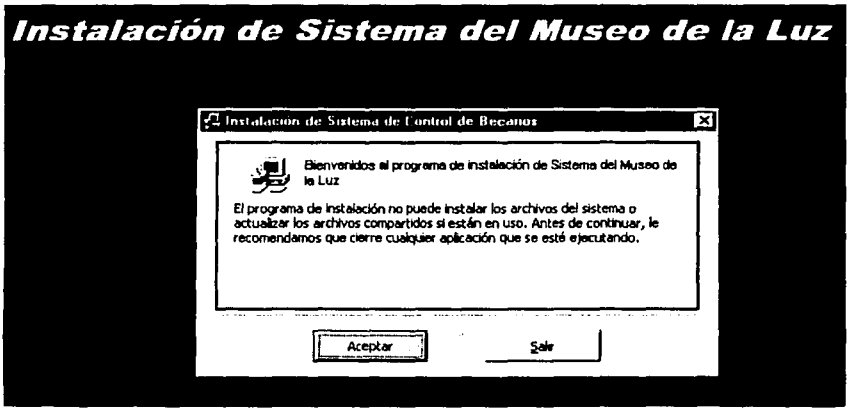

Figura 4.14 Pantalla de inicio de la instalación

Como vemos, muestra el título que le dimos anteriormente al momento de crear el paquete de instalación. Pulsamos Aceptar y nos pide que le indiquemos el directorio donde instalaremos la aplicación (figura 4.15). Pulsamos sobre el icono y comienza la instalación del sistema.

Debemos elegir también el nombre de la carpeta que se creará en *Menú -> Inicio -> Programas* para nuestra aplicación, como lo muestra la figura 4.16.

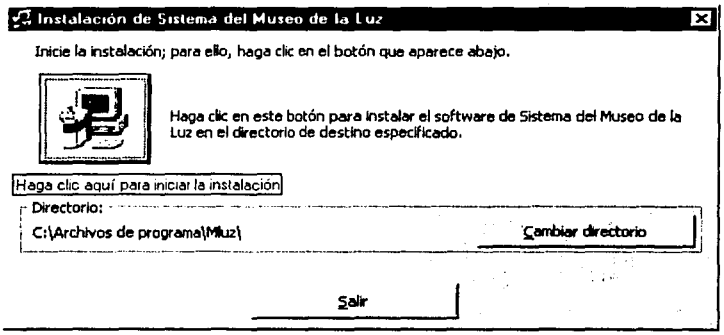

Figura 4.15 Instalación del sistema de becarios del Museo de la Luz

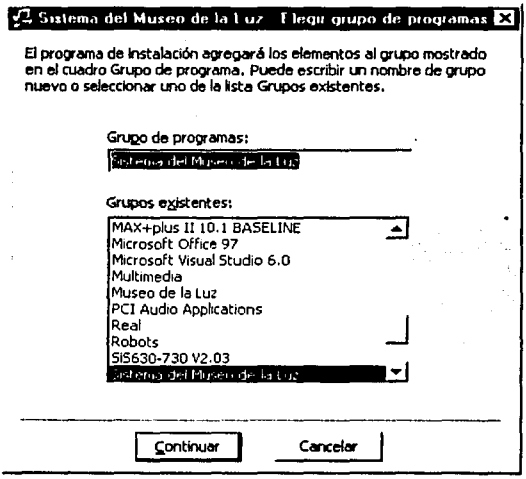

Figura 4.16 Elección de la carpeta para la aplicación

La instalación termina, mostrando la ventana de la figura 4.17.

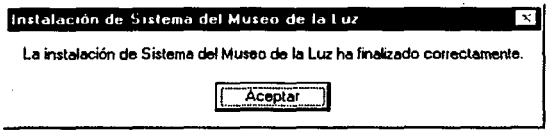

Figura 4.17 El sistema ha sido instalado

La instalación para el sistema del UNIVERSUM es de la misma forma. En el sub-capítulo 4.2 veremos una guía para comenzar a utilizar el sistema.

Ya que hemos creado e instalado el sistema local, podemos crear las bases para el sistema de Internet. Considere que ya ha sido instalado el servidor de web (consulte el capítulo 5). Debido a que el sistema para Internet requiere mayor seguridad, necesitamos BD que contengan la información de los usuarios conocidos, además de los becarios de cada museo. Necesitamos también los nombres virtuales de las bases de datos que el sistema utiliza. Los nombres virtuales son los nombres DSN con los que se identifica en el servidor a cada una de las bases de datos. Para establecerlos, hacemos lo siguiente.

Primero, necesitamos que las bases de datos existan en una carpeta dentro del directorio Inetpub, pero cuidando que no sea dentro de la carpeta wwwroot. Esto es para evitar que las BD puedan ser vistas dentro de un navegador en Internet, al consultar el directorio principal. Realmente las DB pueden estar en cualquier lugar dentro de la PC que funcione como servidor, ya que las referencias de los DSN son de tipo local, aún cuando se puedan acceder por Internet. Las colocamos en el directorio principal, Inetpub, para guardar una estructura, ya que en la carpeta wwwroot es donde almacenaremos las páginas ASP de nuestro sistema.

La conexión con un DSN (Data Source Name o Nombre de la Fuente de Datos) es la forma más cómoda de hacer una conexión, pero sólo se puede utilizar si tenemos acceso al Panel de Control de la máquina servidor. La forma de hacerlo es la siguiente. Creamos nuestra base de Datos en Access y la guardamos en algún directorio de nuestro servidor, en este caso en C:\Inetpub\Datos_Sistema. Luego vamos a Inicio ->

Configuración -> Panel de Control -> Fuentes de Datos ODBC. Encontramos una pantalla que es el administrador de orígenes de datos ODBC de 32bits, como lo muestra la figura 4.18.

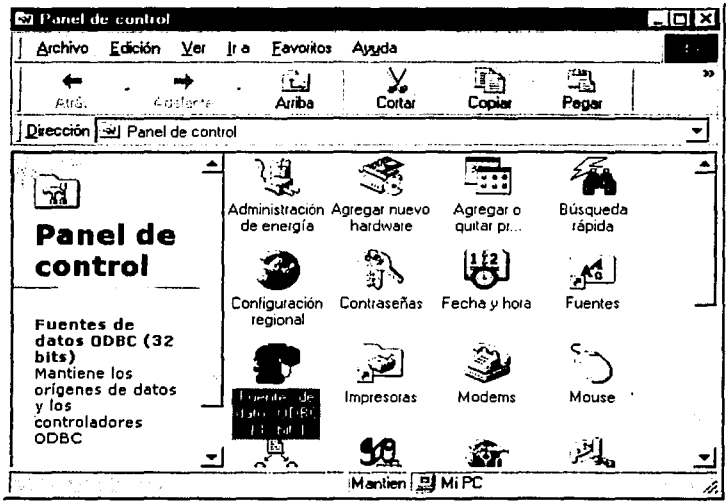

Figura 4.18 Fuentes de Datos ODBC en el panel de Control

Dentro de él, deberemos crear un DSN (Data Source Name) de tipo usuario. Para ello nos colocamos en la solapa correspondiente y seleccionamos Añadir. A continuación se nos pedirá que seleccionemos los controladores de la aplicación que hemos utilizado para crear la base de datos, que en nuestro caso es Microsoft Access Driver (*.mdb), el nombre que le queremos asignar (aquel que empleemos en nuestras páginas) y el camino para encontrarla en el disco duro. Las figuras 4.19, 4.20 y 4.21 muestran este proceso.

Es importante señalar que quien maneje estos orígenes de datos sea la persona que será el administrador del sistema, mismo que podrá hacer modificaciones cuando se requieran.

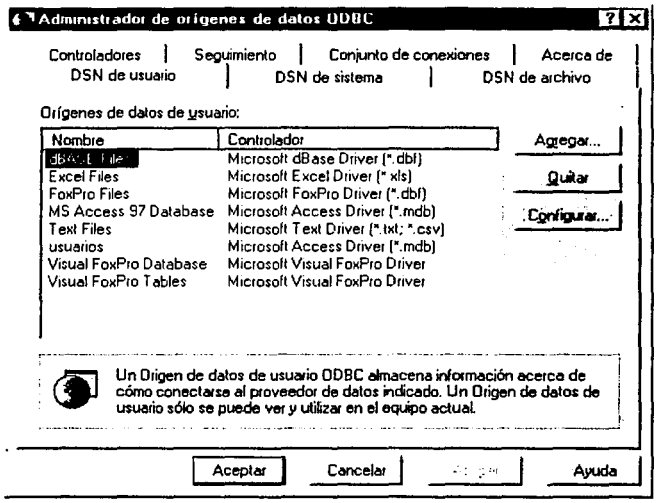

Figura 4.19 Nuevo DSN de usuario

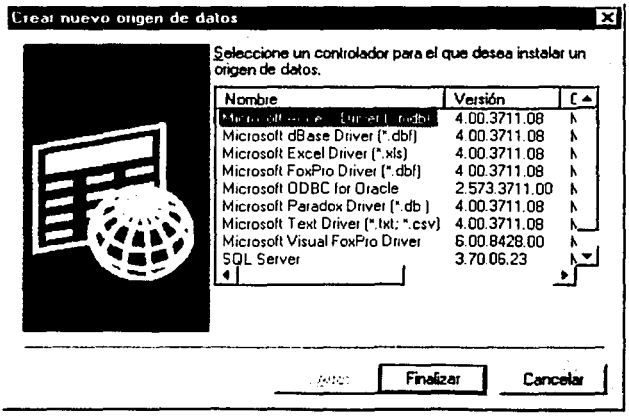

Figura 4.20 Seleccionar controlador para el nuevo DSN

En este ejemplo, estamos creando el DSN para la BD del Museo de la Luz.

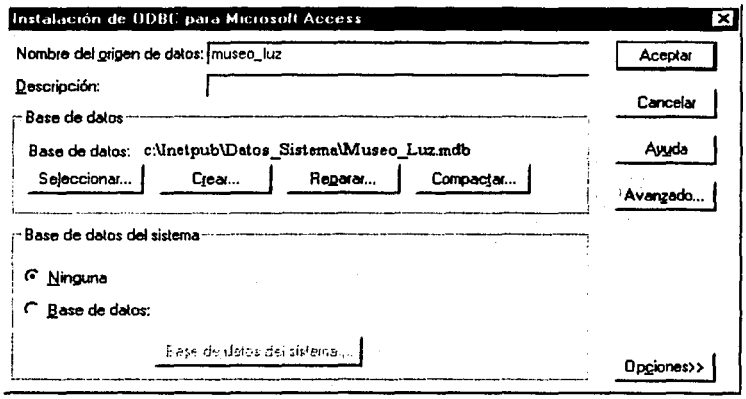

Figura 4.21 DSN para el Museo de la Luz

Si todo salió bien deberá aparecer el nombre en la solapa DSN de usuario y ya tendremos hecha nuestra conexión ODBC. Esta DSN permite definir la BD que será utilizada sin necesidad de pasar por la aplicación que hayamos empleado para construirla, es decir, con simples llamadas y órdenes desde nuestros archivos ASP haciendo referencia al nombre que le asignamos en la solapa de DSN podremos obtener los datos que buscamos sin necesidad de ejecutar Access, el cual no necesariamente debe encontrarse en el servidor donde trabajemos, basta con que cuente con los drivers (controladores) necesarios.

Hacemos lo mismo para el resto de las BD utilizadas: son dos para el Museo de la Luz (para los becarios que entran y los que salen), dos para el UNIVERSUM (para los que entran y salen también), otra para los usuarios que pueden entrar al sistema, y otra más para la parte de avisos de la DGDC (para mayor referencia, consulte el sub-capítulo 5.2.2).

4.2 Puesta en marcha del Sistema y capacitación de la Coordinación de Becarios

Como ya se mencionó al inicio de este capítulo, el sistema fue instalado en Octubre del 2001 y funcionó por un periodo de tres meses aproximadamente. Mientras tanto se desarrolla el sistema para Internet, sin olvidar el sistema local por los errores o fallas que pudiera tener.

Ahora veremos cómo empezar a trabajar con el sistema instalado. Esto es parte de la capacitación que se le dio al personal de la Coordinación de Becarios.

1. El sistema tiene una pantalla de presentación (figura 4.22). Después una ventana nos pide una clave de acceso al sistema (figura 4.23). Si lo recuerda, esta parte no se vió en las etapas de diseño del sistema (capítulo 2) y eso es por la seguridad del mismo. Basta decir que el sistema tiene una serie de claves preestablecidas, y solo mediante ellas se puede acceder al sistema.

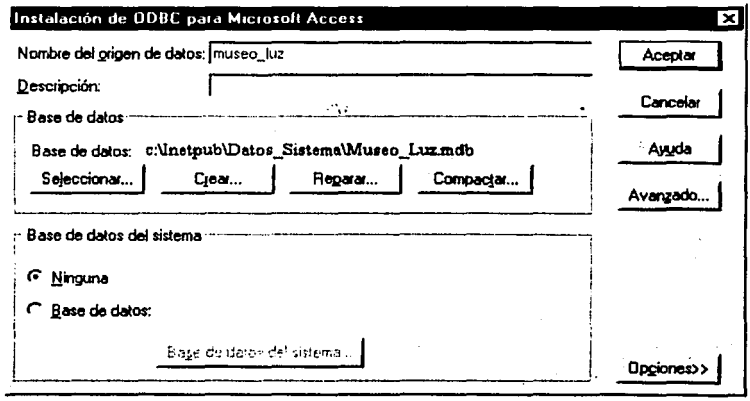

Figura 4.22 Presentación del sistema

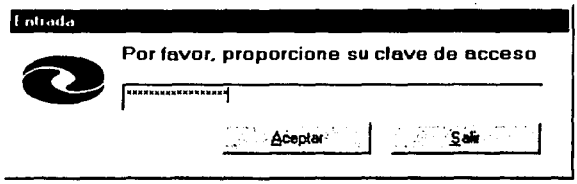

Figura 4.23 Clave para entrara al sistema

Si la clave es incorrecta, el sistema no permite la entrada, como lo muestra la figura 4.24.

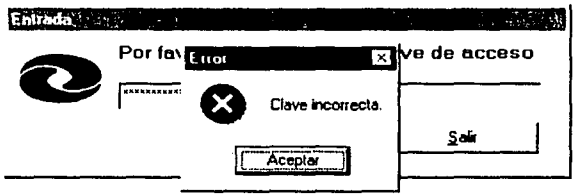

Figura 4.24 Clave incorrecta

Si la clave es correcta, entramos a la pantalla principal del sistema del Museo de la Luz, que ya se vio en el capítulo 3. Por seguridad los campos están deshabilitados. Ahora, supongamos que deseamos agregar un nuevo becario. Para eso vamos a Archivo->Nuevo, o presionamos F1 o bien el icono correspondiente en la barra de tareas como se ve en la figura 4.25.

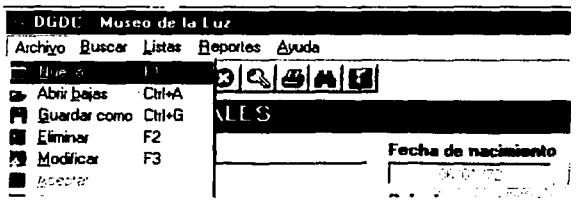

Figura 4.25 Nuevo becario

El sistema nos pide un Número de Cuenta. Esto es para saber si el becario ya fue agregado o todavía no. Si el becario ya está, el sistema lo informa. Las figuras 4.26 y 4.27 muestran esta parte.

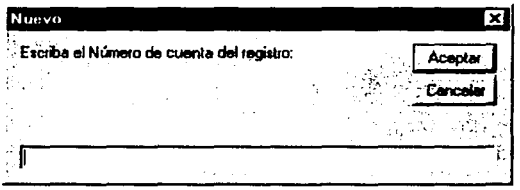

Figura 4.26 Nuevo becario

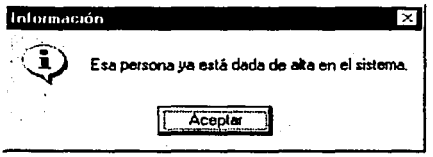

Figura 4.27 Becario existente

Si el becario es nuevo el sistema da la posibilidad de introducirlo en la base de datos, como lo muestra la figura 4.28.

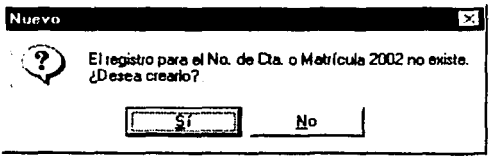

Figura 4.28 El nuevo becario puede ser agregado

Si el becario es agregado, el sistema habilita los campos para que se introduzcan los datos del nuevo becario. Todas las opciones de la barra de iconos y de los menús se

deshabilitan, excepto las opciones Aceptar y Cancelar, como se ve en la figura 4.29. Aceptar hace que el nuevo registro sea introducido y regresa al sistema, habilitando los campos. Cancelar evita que se agregue el nuevo becario y también regresa al sistema.

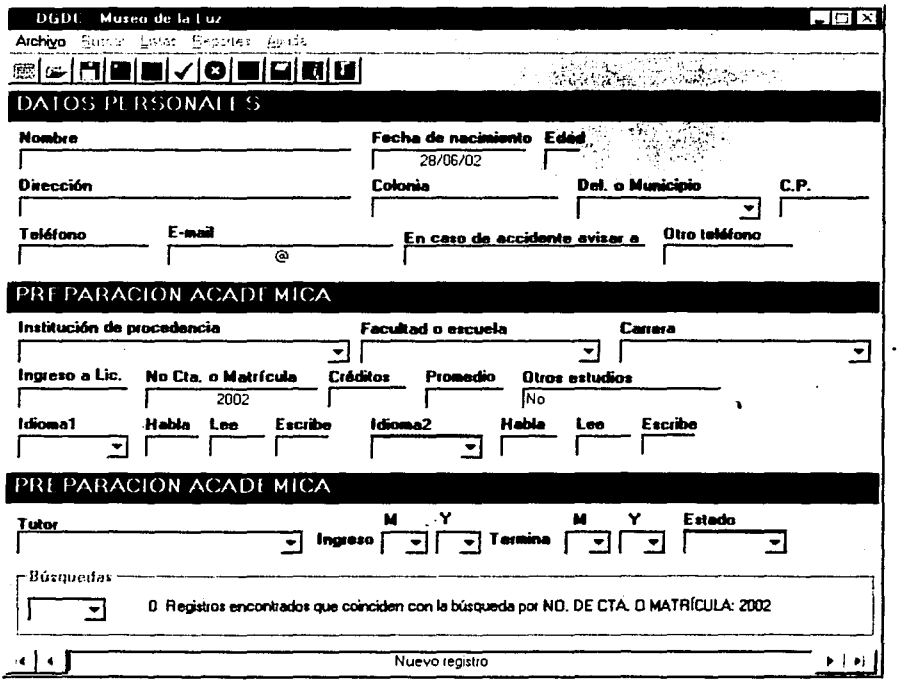

Figura 4.29 Nuevo registro

Si se desea Eliminar un registro (tecla F2), el sistema pide confirmación (figura 4.30). Si se pulsa Si el registro es eliminado de la BD, de lo contrario sus datos permanecen intactos.

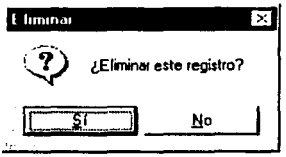

Figura 4.30 Eliminar registro actual

El sistema muestra la ruta donde fueron guardados los datos del registro eliminado (figura 4.31), en un archivo de texto. Esto sirve a la Coordinación para llevar un control de los que ya han sido becarios antes.

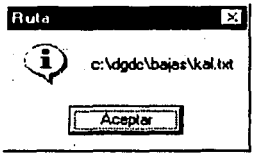

Figura 4.31 Registro eliminado

Si se elige Modificar un registro (tecla F3), los campos se habilitan como cuando agregamos un becario. La opción de Ocultar (figura 4.32) permite ocultar la pantalla del sistema, dejando solo las opciones de Ver de nuevo el sistema o Salir (figura 4.33).

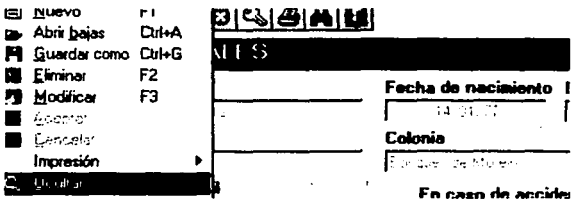

Figura 4.32 Opción Ocultar

Figura 4.33 Sistema oculto

Esto es parte de la capacitación que se le dio a la Coordinación de Becarios sobre el sistema. Obviamente el ponerla aquí sería demasiado extenso, además de que los diversos módulos ya han sido revisados en los capítulos 2 y 3.

En el capítulo siguiente se verán todos los aspectos relacionados con el sistema que funcionará en Internet, tales como: las herramientas empleadas para publicar datos en la red, el lenguaje ASP y la seguridad de la información, por lo que se muestra como proteger no sólo la base de datos, sino el acceso a las páginas del sistema.

5. ACCESO A LA INFORMACION VÍA INTERNET

INTRODUCCIÓN

Este es un capítulo muy importante en el desarrollo de esta Tesis. Realmente aquí es donde radica la importancia de la misma. Por lo general, los sistemas de manejo de bases de datos se diseñan para ser usados solo dentro de un lugar, como pudiera ser una empresa, y por lo general no se tiene acceso desde fuera. Además de que esa es la formación que nos dan en la escuela, bases de datos estáticas (por usarse en un solo lugar, sin acceso desde fuera). Sin embargo, afuera nos damos cuenta de que las necesidades pueden ser otras, más con el auge que ha tenido Internet en los últimos años.

De ahí la importancia de que el sistema desarrollado para la DGDC pueda ser utilizado vía Internet. Las ventajas que tiene son bastantes, destacando que se puede acceder al sistema no importando el sistema operativo ni el navegador que el cliente utilice para entrar al servidor, que será la computadora que gestione todo lo relacionado al sistema, tales como: control de acceso, entrega dinámica del contenido de las bases de datos, etc.

Como ya se vio, los capítulos anteriores no son tan extensos como lo será este. Esto debido a que la importancia de los primeros capítulos radica en todo lo referente a la DGDC (la información, la organización de la misma, etc.), razón por la cual no se puede ahondar tanto, al ser un sistema estrictamente privado, del cual algunas cosas se toman a manera de ejemplo, nunca reflejando la realidad de manera precisa, por la integridad y discreción que se debe tener para la DGDC.

Este capítulo abordará básicamente dos puntos: las herramientas que harán que el sistema funcione en Internet, y cómo será la organización y seguridad del mismo. Como se verá más adelante, se tratará mucho sobre el lenguaje ASP, el cuál se explicará lo más clara y brevemente posible para que se pueda entender de una manera fácil el cómo se puede montar un sistema de bases de datos usando dicha tecnología. No se

abordarán todos los aspectos que puede abarcar el ASP; sólo se tomarán las partes que se puedan emplear para el desarrollo del sistema deseado.

De los ejemplos que se mencionen, algunos son tomados de Internet, y otros serán propios del sistema desarrollado para la DGDC, omitiendo o cambiando obviamente las partes que sean necesarias para la integridad y la confidencialidad del sistema. Al ser este un trabajo de Tesis, uno de los objetivos es el de difundir el conocimiento de esta tecnología para sistemas de bases de datos en Internet, no porque sea el único medio para tal efecto, sino porque es uno de los que existen y será el que se emplee para el objetivo de esta Tesis.

5.1 Herramientas para publicar Bases de Datos en Internet

En los inicios del Internet, la interactividad con los sitios Web era prácticamente nula, debido a que la información que presentaban las páginas era estática, sin que el usuario tuviera forma de manipular la información o hacer algo más en ellas. El HTML comenzó a avanzar y poco a poco incluía más componentes, como fueron la inclusión de contenido multimedia, los applets de Java, scripts, CGI's, etc., por mencionar algunos.

Los sitios comenzaron a incorporar diferentes tecnologías que permitían que el usuario pudiera interactuar con las páginas pasando por el correo electrónico, sistemas de encuestas en línea, los portales, el comercio electrónico, hasta conexiónes con bases de datos (como pueden ser SQL, Access, Oracle, o cualquier otro motor que disponga de un driver, o bien controlador, ODBC), dando paso a la información dinámica. El HTML por si solo no podía brindar todas estas opciones que hoy encontramos fácilmente en la red, por lo que ha tenido que apoyarse en diferentes tecnologías que le permitan realizar este trabajo. La tecnología a emplear se escoge dependiendo del tratamiento que se le dé a la información. En nuestro caso nos enfocaremos a aquellas tecnologías que hacen posible interactuar desde cualquier lugar (que cuente con acceso a Internet) con una base de datos residente en una computadora que sirva como nuestro Servidor Web.

Para poder acceder desde un Navegador de Internet cualquiera a una base de datos residente en un Servidor Web se requiere el uso de las tecnologías que nos permitan lograr nuestro objetivo. En sí, es del lado del Servidor donde estas tecnologías se aplican, procesando la información necesaria y regresando los resultados obtenidos al navegador del usuario.

Hoy en día podemos encontrar un sin fin de portales o sitios en Internet que ofrecen desde un servicio de correo electrónico (yahoo.com.mx, www.hotmail.com,por ejemplo), noticias, clima, entretenimiento (www.elsitio.com, www.unam.mx, etc., por ejemplo),hasta aquellos donde podemos realizar compras en línea (www.amazon.com, www.submarino.com, etc., por ejemplo). En todos ellos podemos encontrar diferentes tecnologías tecleando simplemente el URL del sitio. Anteriormente en la red era común que todas las páginas tuvieran extensión htm o html. Actualmente hay muchas páginas que tienen extensiones diferentes, por ejemplo .php, .asp, etc., incluyendo los CGI's (Common Gateway Interface). Dichas terminaciones obedecen a la tecnología que se emplea para procesar la información.

Hay muchos factores que determinan la tecnología a emplear para este procesamiento de la información. Primero se debe considerar en qué plataforma estará el Servidor Web que contendrá la base de datos, como pueden ser Windows 9x/NT, UNIX, Linux, Mac, etc. Hay que mencionar que si bien no todas las tecnologías pueden implementarse en cualquier Sistema Operativo, los navegadores HTML sí pueden trabajar en cualquier computadora, sea PC, Mac, etc., lo que trae una ventaja considerable. Podemos tener un Servidor Web en una máquina con UNIX y acceder a éste desde una PC, por ejemplo, simplemente porque el navegador HTML nos lo permite. Otra ventaja de las bases de datos en red, es que se pueden manipular desde cualquier punto, simplemente con el navegador. Esto es una ventaja frente a lo que era el tratamiento normal de datos, donde se tenía que hacer una aplicación específicamente para el ambiente de trabajo.

Uno de los lenguajes que se ha extendido bastante es el ASP, que significa Active Server Pages (Páginas Activas de Servidor), desarrollado por Microsoft. ASP no es un lenguaje de programación en sí mismo, ya que los ASP se pueden programar en

VBScript, JavaScript, PerlScript o en otros lenguajes, sino que se puede considerar como una tecnología que permite construir aplicaciones basadas en Internet. La tecnología ASP apareció por primera vez (versión 1.0) con el servidor IIS (Internet Information Server) 3.0 de Microsoft en Diciembre de 1996. La versión 4.0 de IIS (que viene en el Option Pack para NT 4.0) incluye la versión 2.0 de ASP, y la versión 5.0 del IIS, distribuida con Windows 2000, incluye ASP 3.0.

Los predecesores de ASP incluyen CGI y Perl. Las tecnologías de Microsoft predecesoras de ASP incluyen IDC y WebDB. Otras tecnologías que compiten con ASP son ColdFusion (Allaire), JavaServer Pages (Sun Microsystems) y PHP (de libre distribución bajo Open System).

Es común encontrar todavía sitios Web que utilizan CGI en sus Páginas Activas de Servidor. Para ello se han usado lenguaje PERL y C, entre otros. El empleo de CGI's requiere de la ejecución de los programas generados con cada lenguaje por lo que es difícil mantener las páginas que se le regresan al cliente, debido a que las instrucciones HTML se mezclan directamente en el código de los programas CGI, teniendo que recompilar los mismos en cada actualización del código. Además, el proceso de carga del código en memoria que se realiza cada vez que el usuario requiere su ejecución por medio de la página Web que la llama la hace ineficiente en la medida en que es requerida por más de un usuario a la vez. La tecnología ASP se creó para facilitar la creación de sitios Web de una forma más sencilla de cómo se realiza con los CGI's.

Algunas de las características de ASP:

- Es gratuito para Microsoft Windows NT o Windows 95/98.
- El código ASP se puede mezclar con el código HTML en la misma página (no es necesario compilarlo por separado).
- El código ASP se puede escribir con un simple editor de textos, como el Bloc de notas de Windows, o con editores más grandes y especializados como el Ultradev de Macromedia.
- Cómo el código ASP se ejecuta en el servidor, y produce como salida código HTML puro, su resultado es entendible por todos los navegadores existentes.

- Mediante ASP se pueden manipular bases de datos (altas, bajas, consultas, etc.) de prácticamente cualquier plataforma, con tal de que cuente con un driver OLEDB u ODBC, instalados en la máquina que será el servidor.

- ASP permite usar componentes escritos en otros lenguajes (C++, Visual Basic, Delphi), que se pueden llamar desde los scripts (o guiones) ASP.

- Los ASP no se compila, se interpretan, lo que es una gran ventaja frente a los CGI's.

- Los scripts ASP se pueden programar en JScript o VBScript (este último es el más usado porque mas programadores lo dominan), pero también existen otros lenguajes, como Perlscript y Rexx, que se pueden emplear para programar ASP.

- Se ha portado casi al 100% a la plataforma Java por Chili!Soft y Halcyon Software, lo que permitirá que ASP sea usado en casi cualquier sistema operativo.

Las principales ventajas de ASP son:

1. Permite acceder a bases de datos de una forma sencilla y rápida.

2. Las páginas se generan dinámicamente mediante el código de scripts, (guiones).

3. El código de script se ejecuta en el servidor, por lo que no se depende del navegador que se utilice

4. Desde una página ASP se pueden ejecutar servidores OLE en el servidor de Web, lo que abre un abanico de nuevas posibilidades sólo accesibles previamente usando CGI y filtros ISAPI: acceso a base de datos, acceso a ficheros, logging en el sistema, envío de correo, etc.

5. Puede correr en PCs no necesariamente grandes, que tengan Windows 9x/NT y un servidor Web.

6. Se puede acceder desde cualquier computadora que esté conectada a la red y que cuente con un navegador.

7. Es muy fácil de programar, y tiene muchas utilidades que permiten personalizar las páginas a nuestros gustos y necesidades.

Requerimientos básicos para programar en ASP

Para programar ASP se necesita, como mínimo, un servidor de Web (Microsoft Personal Web Server para Windows 95/98 o Internet Information Server 3.0/4.0/5.0 para Windows NT 4.0) y un simple editor de textos como el Bloc de notas de Windows para

escribir los guiones. En adelante, utilizaremos las siglas IIS y PWS para referirnos al Internet Information Server y al Personal Web Server, respectivamente.

Si se quiere acceder a una base de datos, se puede usar una conexión ODBC, una conexión OLE-DB, o una ruta física. Una simple base de datos hecha en Microsoft Access es suficiente, dependiendo de las necesidades del sistema. También se soporta SQL Server, Oracle, DBase, etc, teniendo los controladores necesarios, como se verá más adelante.

Para procesar páginas ASP no hay restricciones del lado del cliente, tanto el Internet Explorer como el Netscape Navigator pueden acceder sin problema a una página ASP. Un aspecto que hay que resaltar es que los scripts de una página ASP se mantienen y se ejecutan en el servidor y este envía el resultado como código HTML al cliente, por lo que se tiene plena integridad y seguridad de los datos. Ningún navegador mostrará el código ASP, ni siquiera si se encuentra en la misma computadora que es el servidor Web. La figura 5.1 ejemplifica este proceso.

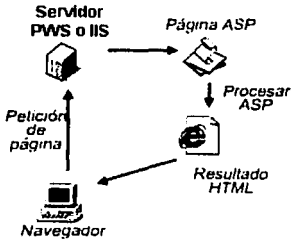

Figura 5.1 Procesamiento de una Página ASP

Una vez que ya tenemos un panorama general de ASP, lo que sigue es instalar nuestro servidor de Web.

5.1.1 INSTALACIÓN DEL SERVIDOR DE WEB

Podemos instalar en nuestra PC un servidor propio. Como ya se mencionó, este servidor distribuido por Microsoft tiene dos versiones diferentes que son utilizadas dependiendo del equipo con el que trabajemos. Para los usuarios de Windows 9X, la versión disponible es el PWS. Si trabajamos bajo sistema Windows NT 4.0, el servidor a instalar es el IIS. Existe también la posibilidad de trabajar en plataformas UNIX empleando en este caso el ChilisoftASP[1], el cual tiene la desventaja que es de pago y que no ofrece todavía un soporte del 100% del código ASP.

Para Windows 95 el PWS se puede conseguir descargándolo desde el sitio de Microsoft (ver referencia en la bibliografía), instalándolo desde el Option Pack de Windows NT 4.0, o bien desde el CD de Windows 98, en la carpeta add-ons/pws. Aún cuando la computadora, con Windows 95, tenga acceso normal a Internet, puede ser necesario que se instale además del PWS una versión de Winsock, debido a que la instalación del mismo no está completa en Windows 95. Para los usuarios de Windows 98 basta con que lo instalen desde el CD, y si no se encuentra ahí (no todos los CDs de instalación de Windows 98 lo traen), basta con bajarlo de Internet. Algunas versiones del PWS anteriores a la 4.0 requieren un archivo adicional, el asp.pexe que se puede bajar del sitio de Microsoft, para poder reconocer páginas ASP.

El PWS podría considerarse como una versión ligera del IIS4. En realidad el PWS no es suficientemente versátil para ejercer de servidor de un sitio de un tamaño mediano aunque si que podría en un momento dado hacerse cargo de un sitio de tamaño reducido y no muy concurrido. De todas formas, la utilidad del PWS radica sobre todo en que nos permite realizar las pruebas del sitio que vayamos a desarrollar en "local" sin necesidad de colocar nuestros archivos en el servidor que alberga nuestro sitio cada vez que queramos hacer una prueba sobre una pequeña modificación introducida. Esto resulta práctico sobre todo si se necesitan hacer pruebas con una relativa frecuencia. Tanto el PWS como el IIS son servidores de Web, la diferencia es que el IIS suele emplearse para sitios pesados e incluye además un servidor de FTP, además de que al

[1] http://www.chilisoft.com/chiliasp/default.asp

ser bajo NT requiere establecer los permisos necesarios para el correcto acceso a los recursos del servidor.
Nosotros nos enfocaremos en el PWS, ya que es el servidor empleado para el sistema desarrollado para la DGDC. La instalación del IIS es similar, teniendo en cuenta las respectivas medidas necesarias de seguridad. Para una visión más detallada del IIS consulte el Apéndice A.

El uso del PWS es extremadamente fácil. Una vez instalado, podemos observar el icono correspondiente en la barra de tareas, lo que nos dará un rápido acceso al administrador de nuestro sitio. Para correr el PWS se necesita el Internet Explorer 4.01 o superior. Para instalar el PWS, se puede descargar el Option Pack para NT desde el sitio de Microsoft, donde también se encuentra una opción para Windows 9x. La descarga completa es de 31Mb. También se puede instalar desde algunos CD's de Windows98. El archivo de instalación puede llamarse Setup.exe o Instalar.exe, dependiendo de si es la versión en Inglés o en Español (figura 5.2).

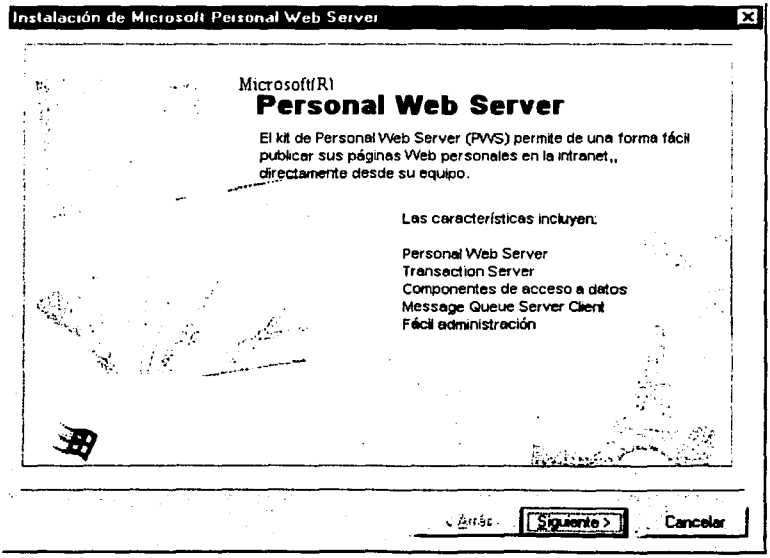

Figura 5.2 Pantalla de instalación del Personal Web Server

Al ejecutar el instalador primero detectará si tenemos instalado Winsock 2.0 o superior. En caso de que no lo tengamos, este será instalado automáticamente. A continuación el asistente nos preguntará cuales herramientas deseamos instalar. Podemos seleccionar la opción típica o escoger bien las opciones en la opción personalizada. Se recomienda que se deje la instalación típica que el asistente señala por defecto ya que si se intenta instalar de forma personalizada hay opciones (como el MTS) que no se pueden omitir. Después de esto, el sistema se debe reiniciar. Ahora podemos ver líneas adicionales al iniciar la computadora, ya que el instalador añadió al autoexec.bat líneas necesarias para que reconozca al MTS. Al iniciar de nuevo, el icono del PWS aparece en la barra de tareas, y al colocar el puntero del ratón sobre el nos muestra una leyenda que dice que el PWS está corriendo (figura 5.3). También se han creado iconos de acceso en la carpeta de Programas del Menú Inicio, así como en el Panel de Control. Al pulsar este icono aparece la pantalla de administración del PWS.

Figura 5.3 Icono del PWS en la barra de tareas

El siguiente paso es crear un directorio virtual dentro del cual alojaremos nuestra página. Decimos que es un directorio virtual debido a que nuestra página puede estar alojada en cualquier parte de nuestro disco duro y con un nombre de directorio que tampoco tiene por qué parecerse al que incluiremos en la URL cuando queramos ejecutar la página.

Para crear un directorio virtual debemos pulsar el icono "Avanzado" el cual nos da acceso a las opciones avanzadas del PWS. Una vez ahí, el siguiente paso es "Agregar" un directorio virtual. Aparecerá una ventana en la que tendremos que introducir el nombre del directorio virtual y especificar en qué carpeta del disco duro tenemos guardados los archivos y carpetas de la página. Vea las figuras 5.4 y 5.5.

Una opción interesante en el menú avanzado es la selección del tipo de archivo que será ejecutado por defecto, esto es, cuando en un directorio de nuestro sitio no escribimos un nombre específico de un archivo, entonces se ejecuta el archivo por defecto que se encuentre en dicho directorio. Aquí podríamos poner archivos con nombre como index.html o index.asp o bien con el nombre default o home en lugar de index.

Para acceder a una página debemos escribir en el navegador de Internet:

http://nombre_sitio/directorio_virtual/archivo

donde

nombre_sitio es la dirección IP de la PC donde se instaló el PWS. Por defecto, la dirección local del sitio es **localhost**, o bien **127.0.0.1**, que es la dirección numérica.

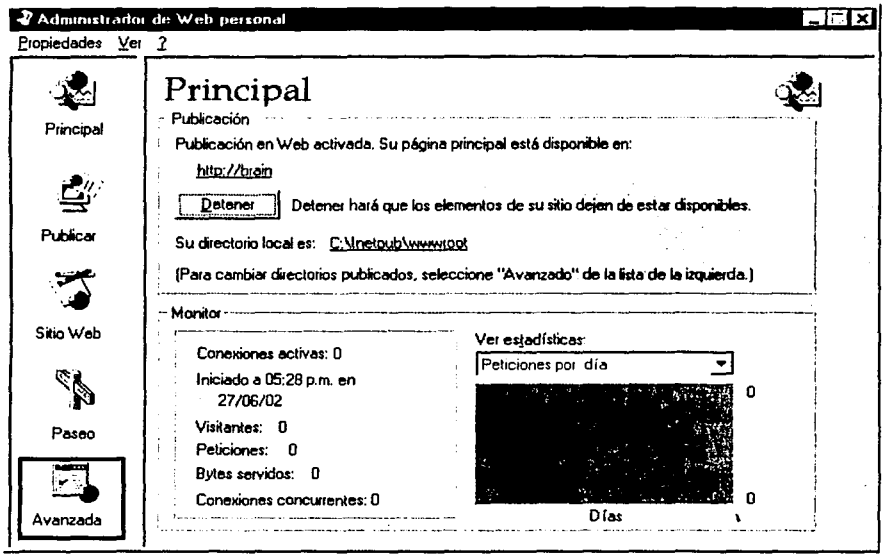

Figura 5.4 Seleccione Avanzada para agregar un directorio virtual

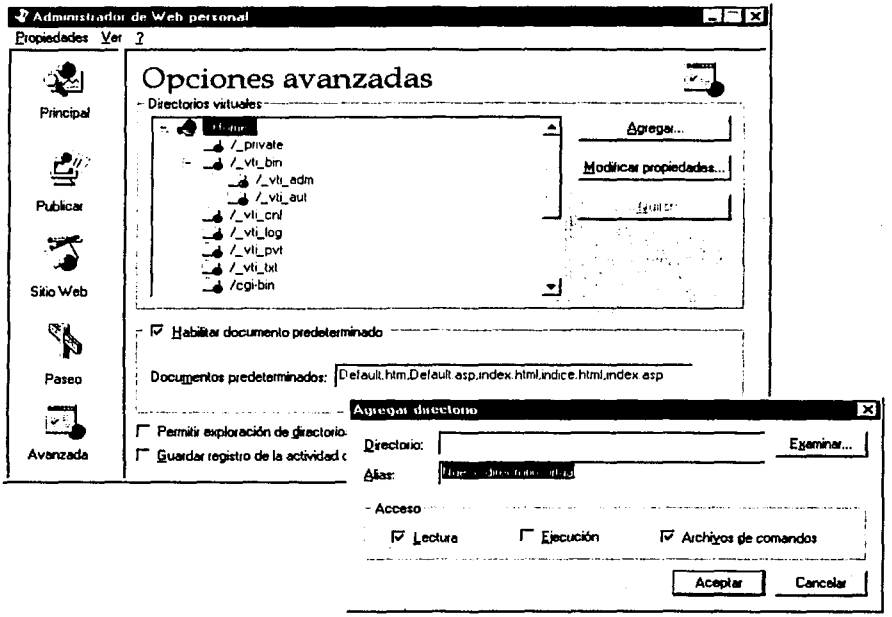

Figura 5.5 Introduzca los datos del nuevo directorio virtual

Estos nombres son para cuando estamos trabajando en nuestro sitio off-line, es decir, cuando no estamos conectados a Internet nos referimos a nuestro sitio como http://localhost. Hay que mencionar que nuestro servidor de Web funciona aún cuando no estemos conectados a Internet, manejando todo como si se tratara de una red local, aún cuando sólo trabajemos en esta PC. Cuando estemos conectados a Internet, para acceder nuestro sitio debemos poner la dirección IP en lugar de localhost o del 127.0.0.1.

directorio_virtual es algún directorio que hayamos agregado a nuestro sitio, como ya se vió

archivo es el nombre de la página a visualizar, ya sea HTML o ASP.

Tanto el PWS como el IIS crean un directorio en nuestro disco duro, llamado Inetpub (figura 5.6), el cual a su vez contiene un directorio llamado wwwroot, que es el directorio cuando hacemos referencia a nuestro sitio con http://nombre_sitio. Por lo tanto, también podemos crear directamente un subdirectorio en la carpeta Inetpub\wwwroot, que es otra forma de agregar directorios virtuales a nuestro sitio. Esto representa un mejor control sobre el mismo, ya que nuestras páginas no están regadas en la PC, sino que las concentramos en el subdirectorio que les corresponde.

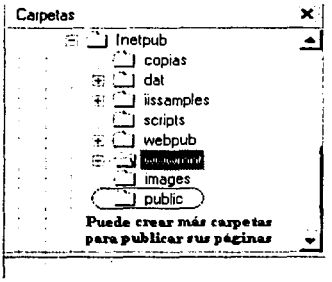

Figura 5.6 Directorios de la carpeta Inetpub

Es posible que durante la instalación del PWS haya tenido algún problema relacionado con el MTS (Microsoft Transaction Server). Lo que se debe hacer en ese caso es bajar e instalar la nueva versión de la librería requerida de la dirección

http://download.microsoft.com/download/transaction/Patch/1/W95/EN-US/Mtssetup.exe

Es recomendable que instale las librerías del MTS antes de instalar el PWS. Aunque la mayoría de los casos funciona indistintamente, en algunas ocasiones no sucede así.

La instalación del IIS no se aborda ya que es similar a la del PWS y no representa mayores problemas. Como ya se ha mencionado, si se instala el IIS, al ser Windows NT se deben de cuidar los aspectos de seguridad que nos permitan compartir nuestras paginas en Internet. La instalación del servidor de Web ha finalizado, por lo que continuaremos con la programación de las páginas de nuestro sitio.

5.1.2 PROGRAMANDO EN ASP

Existen dos formas de incluir código ASP en una página HTML: Una es especificar el segmento que corresponde al script de ASP, delimitándolo como sigue:

```
<SCRIPT LANGUAGE="Lenguaje">
        Escribir el código
</SCRIPT>
```

Donde **Lenguaje** debe cambiarse por Jscript o VBScript, según el lenguaje a usar. Otra forma es escribiéndolo entre los delimitadores <% y %>. Esta forma es la más usada, ya que permite intercalar código ASP y HTML en la misma página. Por defecto, si no se especifica el lenguaje (VBScript o JavaScript) en la primer línea de cada página ASP, el servidor considerará que se empleará el lenguaje VBScript.

Modelo de objetos ASP

ASP incluye seis tipos de objetos nativos que podemos usar en nuestros programas. Los objetos ASP no forman una jerarquía; se relacionan entre sí de forma lógica, no a través de una relación padre-hijo. La figura 5.7 muestra las relaciones.

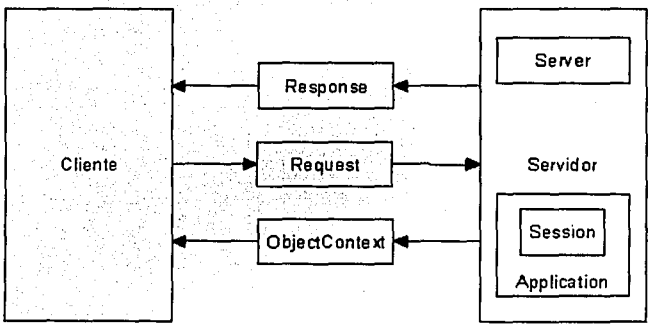

Figura 5.7 Relaciones de los objetos en ASP

Objetos

Los objetos de ASP permiten la interacción entre el servidor y el cliente. No es necesario usarlos todos, pero si hay que recordar que si se abren se deben de cerrar. Los objetos de ASP más utilizados son los siguientes:

Request (Petición): Permite acceder a la información de un formulario, leer cookies, certificados de clientes, leer encabezados HTTP y analizar URLs codificadas. Este objeto nos facilita toda la información sobre la petición HTTP que un usuario realiza a nuestro servidor (informa sobre todo lo que llega al servidor desde el cliente). Puede incluir parámetros obtenidos a partir de un formulario HTML mediante los métodos POST o GET, cookies y certificados que posea el cliente, y las variables del servidor. La siguiente tabla muestra las colecciones, propiedades, y métodos a que podemos acceder con este objeto.

Colecciones	**Propiedades**	**Métodos**
QueryString	TotalBytes	BinaryRead
Form		
ServerVariables		
Cookies		
ClientCertificate		

Response (Respuesta): Este objeto maneja toda la información que se envía desde el servidor al cliente. Incluye el envío de información directamente al navegador del cliente, redirección del navegador hacia otra URL o fijar el valor de las cookies. La siguiente tabla muestra las colecciones, propiedades, y métodos a que podemos acceder con este objeto.

Colecciones	**Propiedades**	**Métodos**
Cookies	Buffer	AddHeader
	CacheControl	AppendToLog
	Charset	BinaryWrite
	ContentType	Clear
	Expires	End
	ExpiresAbsolute	Flush
	IsClientConnected	Redirect
	PICS	Write
	Status	

Application (Aplicación): A través de este objeto se pueden controlar aspectos relativos al inicio y finalización de una aplicación lo mismo que el acceso a información que está almacenada en el servidor. Este objeto se emplea para compartir información entre todos los usuarios de una aplicación (existe un único objeto Application que comparten todos los usuarios). Una aplicación tiene un periodo de vida como cualquier otro programa. Cuando se inicia se ejecuta el evento OnStart y cuando termina se produce el evento OnEnd. La siguiente tabla muestra las colecciones, propiedades, y métodos a que podemos acceder con este objeto.

Colecciones	**Métodos**	**Eventos**
Contents	Lock	OnStart
StaticObjects	Unlock	OnEnd

Server (Servidor): Permite desarrollar funciones de rutina como el mapeo de un path virtual o físico y la creación de una instancia para un componente. Este objeto proporciona propiedades y métodos que están relacionados con el servidor donde se ejecuta nuestra aplicación. Normalmente se usa para crear una instancia de un componente ActiveX o fijar el tiempo de validez de un script. La siguiente tabla muestra las propiedades, y métodos a que podemos acceder con este objeto.

Colecciones	**Propiedades**	**Métodos**
	ScriptTimeout	CreateObject
		HTMLEncode
		MapPath
		URLEncode

Session (Sesión): A cada cliente que solicita una página ASP se le asigna un objeto Session. La información almacenada en este objeto es individual para cada usuario, y se conserva el valor al pasar de una página a otra. Se crea un objeto Session la primera vez que un cliente solicita una página y se destruye, por defecto, veinte minutos después de que se haya realizado la última petición. El tiempo de vida es configurable. Cuando se inicia una sesión se produce el evento OnStart y cuando termina se produce OnEnd. La siguiente tabla muestra las colecciones, propiedades, y métodos a que podemos acceder con este objeto.

Colecciones	Propiedades	Métodos	Eventos
Contents	CodePage	Abandon	OnStart
StaticObjects	LCID		OnEnd
	SessionID		
	Timeout		

ObjectContext (Contexto del Objeto): Este objeto se emplea para gestionar transacciones. La siguiente tabla muestra los métodos y eventos a que podemos acceder con este objeto.

Colecciones	Propiedades	Métodos	Eventos
		SetComplete	OnTransactionCommit
		SetAbandon	OnTransactionAbort

Componentes

Los componentes son controles ActiveX que sirven como interfaz con ASP para simplificar procedimientos comunes. Los más usuales son:

Ad Rotator (Rotador de Publicidad): Es un componente que permite presentar banners en una página de manera aleatoria.

Browser Capabilities (Capacidad del Browser): Permite determinar el navegador que utiliza el usuario y que aspectos soporta.

Collaboration Data Objects (CDO): Teniendo en cuenta que el servidor IIS posea SMTP (Simple Mail Transport Protocol, que es un protocolo de correo simple que se utiliza para la transferencia de e-mail a través de la Red), este componente permitirá enviar un e-mail a través de una página ASP, sin necesidad de enlazar a un script de Perl o cualquier otro CGI.

Content Linking: A través de este componente se puede tejer un sistema de links entre todas las páginas de un sitio de manera que no haya necesidad de escribir los típicos "anterior" o "siguiente". Incluso genera mapas del sitio.

Content Rotator (Rotador de Contenidos): En los sitios comerciales, la necesidad de rotar los contenidos atrae tráfico. Para ello se utiliza este componente que es capaz de añadir contenidos dinámicos a cualquier página sin necesidad de usar una base de datos. Su funcionamiento está basado en fragmentos de código HTML que toma de una archivo de texto y que alterna en un espacio determinado en una página. Cada vez que carga el contenido, el rotador de contenidos presenta aleatoriamente uno de los trozos.

Database Access (Acceso a Datos): A través de este componente se puede enlazar una página a una base de datos y que la página tome sus contenidos de la base.
Además de estos componentes existe una gran variedad de componentes extras desarrollados por otras empresas además de Microsoft destinados a ASP.

Crear una página ASP

Una página ASP es un archivo de texto con la extensión .asp que contiene cualquier combinación de los siguientes elementos:

- Texto
- Etiquetas HTML
- Secuencias de comandos del servidor

Una forma rápida de crear un archivo asp consiste en cambiar la extensión de los archivos HTML (.html o .htm) por la extensión .asp. Si el archivo no contiene funciones ASP, el servidor prescinde del proceso de secuencias de comandos ASP y envía el archivo al cliente. Un aspecto importante es que si bien el código HTML se puede ejecutar o interpretar en un archivo con extensión asp, no sucede lo mismo para este último. Es decir, para que nuestros scripts asp funcionen, el archivo necesariamente debe tener la extensión asp, de lo contrario los scripts serán ignorados por el servidor.

Para crear archivos .asp, se puede utilizar cualquier editor de textos, como el Bloc de Notas de Windows. También se pueden utilizar editores más orientado al ASP, como el Visual InterDev de Microsoft, o el Ultradev de Macromedia. Si no se requiere de editores tan complicados, existen varias alternativas, una de ellas y que es con la que se trabajó

para el desarrollo del sistema para la DGDC, es el EditPlus 2, que es un editor ligero y que soporta diferentes lenguajes (ASP, HTML, C++, Perls, etc.) La ventaja de utilizar un editor más orientado a ciertos lenguajes es que permite distinguir con diferentes colores entre el código HTML, el código ASP (o de otro lenguaje), y el texto normal de una página.

Para publicar el archivo .asp en el sitio Web se debe guardar el nuevo archivo en un directorio virtual del sitio Web (asegúrese de que el directorio tenga los permisos Secuencia de comandos o Ejecución si se encuentra fuera del directorio Inetpub\wwwroot). A continuación ingrese en el explorador la dirección URL del archivo para pedirlo.

Las páginas ASP deben pasar por el servidor, por lo que no puede pedirlas mediante su ruta física. Es decir, debe acceder a su página de la siguiente forma:

http://nombre_sitio/directorio/la_pagina.asp

Como ya se vio antes, si la dirección IP del servidor fuera 182.225.10.8, cuando su página se encuentre en red (su servidor está activo), quien desee acceder a su página debe escribir

http:// 182.225.10.8/directorio/la_pagina.asp

Cuando el archivo se cargue en el explorador del cliente, observará que el servidor envió una página HTML. Pudiera parecer que algo no anda bien, pero hay que recordar que el servidor analiza y ejecuta todas las secuencias de comandos ASP del servidor antes de enviar el archivo. El usuario siempre recibe código HTML estándar. Incluso si un usuario un poco más avanzado intenta ver el código fuente en su navegador, sólo verá el código HTML que el servidor le envía.

Cómo funciona el procesador ASP

Consideremos que ya tenemos nuestro servidor funcionando, y que tenemos un archivo .asp con el siguiente código:

```
<%@ LANGUAGE="VBSCRIPT" %>
<HTML>
<BODY bgcolor="WHITE" text="black">
<% Dim sitio
sitio = "Mi sitio"
%>
<h1 align="center">Bienvenidos a <b><%=sitio%></b></h1>
</BODY>
</HTML>
```

Este archivo lo guardamos con una extensión .asp en algún directorio del servidor. Cuando abrimos nuestro navegador y escribimos en la barra de dirección la URL local del servidor (habiéndolo arrancado antes) se nos presentará una página cuyo código será el siguiente:

```
<HTML>
<BODY bgcolor="WHITE" text="black">
<h1 align="center">Bienvenidos a <b>Mi sitio</b></h1>
</BODY>
</HTML>
```

Cuya apariencia en el navegador será el de la figura 5.8.

Figura 5.8 Ejemplo de ASP

Lo que el navegador de Internet muestra al ver el código fuente es lo que muestra la figura 5.9.

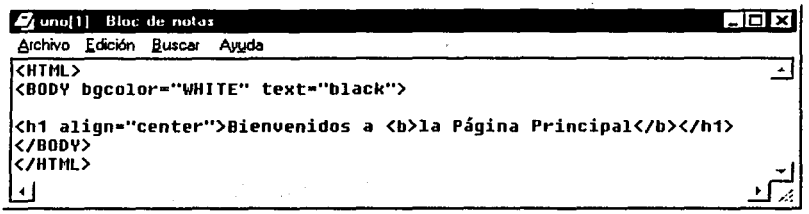

Figura 5.9 Código de la página ASP en el navegador

El parser (librería) ASP.DLL procesa todo el texto situado entre los símbolos <% %> y envía al cliente solo el código resultante. Se debe notar que se utilizó un código de VBScript que incluyó la expresión "dim", y es que ASP permite la creación de funciones y subrutinas de la misma manera que en Visual Basic. Más adelante veremos algunos aspectos referentes a funciones, variables y componentes.

Utilizar variables y constantes

Una variable es una ubicación de almacenamiento con nombre dentro de la memoria del equipo servidor que contiene datos, como un número o una cadena de texto. A los datos contenidos en una variable se les llama valor de la variable. Las variables ofrecen una manera de almacenar, recuperar y manipular valores mediante nombres que ayuden a entender lo que hace la secuencia de comandos. VBScript no necesita declaraciones de variables, pero es conveniente declarar todas las variables antes de utilizarlas. Para declarar una variable en VBScript se utiliza la instrucción **Dim**, **Public** o **Private**. Por ejemplo:

<% Dim NombreUsuario %>

Se puede utilizar la instrucción **Option Explicit** de VBScript al inicio de los archivos .asp para hacer obligatoria la declaración de variables con las instrucciones **Dim**, **Private**, **Public** y **ReDim**. La instrucción **Option Explicit** debe aparecer después de las directivas ASP y antes del texto HTML o de los comandos de la secuencia de

comandos. Esta instrucción sólo afecta a los comandos ASP escritos en VBScript; no afecta a los comandos escritos en JScript.

```
<% Option Explicit %><HTML>
<% Dim strNombreUsuario
Public lngNumeroCuenta %>
```

Al contrario que en otros lenguajes de programación, en VB solo existe un tipo general de datos que se conoce con el nombre de **Variant**. En otros lenguajes existen datos de tipo String (Cadena) para almacenar contenidos alfanuméricos, distintos tipos de datos numéricos enteros y en coma flotante, datos booleanos, etc. Esta característica es muy útil, ya que permite reasignar un valor de un tipo a una variable de otro tipo. En la actualidad es el único lenguaje de alto nivel que implementa esta característica. Los datos se clasifican en subtipos en función del contenido en un momento dado. Así se logra toda la funcionalidad de gestión de datos en lenguajes de alto nivel, pero con una mayor flexibilidad. Para cambiar una variable de un subtipo a otro, es suficiente con asignarle un dato de diferente tipo. Los subtipos de datos en VBScript son las que muestra la tabla 2.

Subtipo	Descripción	Valor regresado por Vartype
Empty	Variable sin inicializar	0
Null	Variable intencionalmente vacía	1
Boolean	Puede ser True o False	11
Byte	Entero entre 0 y 255	17
Integer	Entero entre -32.768 y 32.768	2
Currency	Número entre -922.337.203.685.477,5808 y 922.337.203.685.477,5807	6
Long	Número entre -2.147.483.648 y 2.147.483.647	3
Single	Numero de precisión simple	4
Double	Numero de doble precisión	5
Date	Fecha entre 1-1-100 y 31-12-9999	7
String	Cadena de longitud variable hasta 2.000.000.000 de caracteres	8
Object	Contiene un Objeto	9
Error	Contiene un numero de error 10	10

Tabla 2 Subtipos de datos en VBScript

Para convertir (si se desea, aunque no es necesario) de un tipo a otro de datos, se cuenta con las funciones de la tabla 3.

Función	Convierte la expresión a tipo
Cbool	Boolean
Cbyte	Byte
Cint	Integer
Cing	Long
Csng	Single
Cdbt	Double
Ccur	Currency
Cdate	Date
Cstr	String

Tabla 3 Funciones de conversión de tipos en VBScript

Para convertir, se requiere **Función(variable)**, donde variable es el dato que se desea convertir. También podemos conocer el subtipo de una variable mediante la función **Vartype(variable)** que nos devuelve el valor referenciado en la tercera columna de la tabla anterior de los subtipos de datos.

Por ejemplo. El siguiente fragmento de código daría un error en lenguajes como C++ o Java; sin embargo, en VB es absolutamente correcto:

```
Variable = 1
' más código
Variable = "cadena"
```

Variables

Los datos que se manejan en un lenguaje de programación y cuyo valor puede cambiar durante la ejecución de un programa se conocen con el nombre genérico de variables. Podemos ver a una variable como una caja cuyo contenido cambia a lo largo de la ejecución; la caja tiene un nombre que sirve para identificarla y poder utilizar el contenido o valor. Los nombres de variables deben empezar con una letra y pueden tener letras, números o el signo de subrayado (único signo de puntuación que se admite en el nombre de una variable). Las letras que formen parte del nombre de una variable

deberán ser de la alfabetización internacional, no de la española o específica de algún idioma en particular. Así pues, no deberá haber en un nombre de variable letras como la ñ, letras acentuadas, la ç, etc. Un nombre de variable no deberá contener caracteres especiales (como $,%,&,^,ñ, etc.). Tampoco deberá contener espacios en blanco, puntos, comas, ni ningún otro signo de puntuación. Por supuesto, el contenido de una variable alfanumérica SI podrá contener cualquier cosa que deseemos. Además deberemos tener cuidado de no emplear como nombres de variable las palabras clave del lenguaje. Los siguientes ejemplos de variables serán correctos en VBScript y en VB:

```
Variable = "Esto está en una cadena."
cosa = "*&%$*/87*/&)ñ"
Edad2 = 74
Casado_si_o_no = true
```

Sin embargo, los siguientes ejemplos ilustran declaraciones de variables que el lenguaje no aceptará:

```
72Edad = 28 ' No lo admite por empezar con un número
MsgBox = false ' No lo admite por ser una palabra reservada
Año = 1977 ' No lo admite por contener el nombre una ñ
Ave& = "Esto no vale nada" ' No vale por tener una letra acentuada y un &
```

En otro orden de cosas, para usar una variable es necesario dar dos pasos: declararla e inicializarla. La declaración es la forma de decirle al lenguaje que se va a usar una variable y se hace con la palabra reservada DIM, seguida del nombre de la variable:

```
DIM variable
```

Esto reserva espacio en memoria para la variable. Sin embargo aún no le hemos asignado ningún valor. En realidad tiene un valor de subtipo **Null.** La inicialización de la variable será la que le asigne su primer valor aunque, como ya sabemos, éste podrá cambiar a lo largo de la ejecución. La inicialización es, simplemente una asignación. Por ejemplo

```
variable = "Carro"
```

inicializa la variable declarada anteriormente. Sin embargo, es conveniente realizar las declaraciones de forma manual (escribiendo la instrucción **DIM**), a fin de incrementar el nivel de estructuración de nuestros programas y facilitar la legibilidad de los mismos. La declaración debe ir siempre antes de la inicialización.

Lo correcto es realizar la declaración de todas las variables al principio de nuestro código. Existe una forma de asegurarnos que tengamos que hacer las declaraciones, lo cual se hace incluyendo la instrucción **OPTION EXPLICIT** en nuestro código. Si incluimos esa línea, el programa no podrá usar ninguna variable que no haya sido declarada.

Operadores en VBScript

Se cuenta con operadores (tabla 4) en Vbscript para realizar las operaciones necesarias con cada tipo de datos.

Operador	Operación
+	Suma aritmética o concatenación de cadenas
-	Resta aritmética
*	Multiplicación aritmética
/	División. El resultado está en coma flotante
\	Obtiene la parte entera de la división
^	Potenciación
Mod	Obtiene el módulo de una división
=	Igual que
<>	Distinto de
>,>=	Mayor y mayor o igual que, respectivamente
<,<=	Menor y menor o igual que, respectivamente
&	Concatenación de datos de diferente subtipo
And	Devuelve true si las expresiones son verdaderas
Or	Devuelve true si alguna de las expresiones es verdadera
Xor	Devuelve true si sólo una de las expresiones es verdadera
Not	Cambia una expresión verdadera en falsa y viceversa

Tabla 4 Operadores en VBScript

Alcance de las variables

El alcance o vida de una variable determina qué comandos pueden tener acceso a dicha variable. Una variable declarada dentro de un procedimiento tiene alcance local; la variable se crea y se destruye cada vez que se ejecuta el procedimiento. No se puede tener acceso a ella desde fuera del procedimiento. Una variable declarada fuera de un procedimiento tiene alcance global; su valor es accesible y modificable desde cualquier comando de secuencia de comandos de una página ASP. Al limitar el alcance de la variable a un procedimiento mejorará el rendimiento. Si declara una variable local y una variable global, estas pueden tener el mismo nombre. La modificación del valor de una de ellas no afecta al valor de la otra

Asignar a las variables alcance de sesión o de aplicación

Las variables globales solo son accesibles en un mismo archivo ASP. Para hacer que una variable sea accesible en varias páginas se le asigna a la variable alcance de sesión o de aplicación. Las variables con alcance de sesión están disponibles en todas las páginas de una aplicación ASP que pida un mismo usuario. Las variables con alcance de aplicación están disponibles en todas las páginas de una aplicación ASP que pida cualquier usuario.

Las variables de sesión son una buena manera de almacenar información para un único usuario, como sus preferencias o el nombre o la identificación del usuario. Las variables de aplicación son una buena manera de almacenar información para todos los usuarios de una determinada aplicación, como los saludos específicos o los valores generales necesarios en la aplicación.

ASP proporciona dos objetos integrados en los que puede almacenar variables: el objeto **Session** y el objeto **Application**.

Alcance de Sesión

Para asignar alcance de sesión a una variable, almacénela en el objeto **Session** asignando un valor a una entrada con nombre del objeto. Por ejemplo, los siguientes comandos almacenan dos nuevas variables en el objeto **Session**:

```
<%
Session("Nombre") = "Pepe"
Session("Apellido") = "Pecas"
%>
```

Para recuperar la información del objeto **Session** se pude usar otra variable que tome el valor del objeto Session, o bien tomándola directamente del objeto con los signos <%= **Objeto** %>. Por ejemplo:

Hacer

```
<%Dim nombre
nombre=Session("Nombre")%>
Bienvenido <%=nombre%>
```

Es equivalente a hacer

```
Bienvenido <%=Session("Nombre")%>
```

También se puede utilizar la sentencia

```
<%
Response.Write("Bienvenido " & Session("Nombre"))
%>
```

Por lo general, es conveniente utilizar variables a las que se les asigne el valor de un objeto Session si no se desea que el valor del mismo sea alterado.

Puede almacenar las preferencias del usuario en el objeto **Session** y después tener acceso a dichas preferencias para determinar qué página hay que devolver al usuario. Por ejemplo, puede permitir que los usuarios especifiquen la versión en texto del contenido de la primera página de la aplicación y aplicar esta opción a las siguientes páginas de la aplicación que el usuario visite.

```
<% strScreenResolution = Session("ScreenResolution")
If strScreenResolution = "Low" Then
%>
Ésta es la versión de texto de la página.
<% Else %>
Ésta es la versión multimedia de la página.
<%End If %>
```

Si hace referencia a una variable con alcance de sesión más de una vez en una secuencia de comandos, piense en asignarle una variable local, como en el ejemplo anterior, para mejorar el rendimiento.

Alcance de Aplicación

Para asignar alcance de aplicación a una variable, almacénela en el objeto **Application** de la misma forma que haría para un objeto Session. Por ejemplo, el comando siguiente almacena en el objeto **Application** un saludo específico de una aplicación:

```
<% Application("Bienvenida") = "¡Sea bienvenido a nuestro sitio!" %>
```

Para recuperar la información del objeto Application se pueden utilizar cualquiera de los métodos ya vistos. De nuevo, si hace referencia a una variable con alcance de aplicación en su secuencia de comandos repetidamente, debe asignarle una variable local para mejorar el rendimiento.

El comando Response.Write

Envía al explorador el texto que le sigue. Utilícelo desde una instrucción cuando desee generar de forma dinámica el texto devuelto al explorador. Por ejemplo, puede generar una cadena de texto que contenga los valores de varias variables. Observe que dispone de varias maneras de insertar comandos en las páginas HTML. Puede incluir procedimientos dentro de delimitadores ASP.

Espacio en blanco en las secuencias de comandos

ASP quita el espacio en blanco de los comandos si es el lenguaje principal de una página, y si hay otros lenguajes conserva el espacio en blanco para que los lenguajes que interpreten la posición o la sangría puedan interpretarlo correctamente. El espacio en blanco incluye espacios, tabuladores, retornos y saltos de línea. En VBScript y JScript se puede utilizar el espacio en blanco que sigue al delimitador de apertura y que precede al de cierre para facilitar la lectura de los comandos. Todas las instrucciones siguientes son válidas:

```
<% Color = "Negro" %>
<%Color="Negro"%>
<%
Color = "Negro"
%>
```

ASP quita el espacio en blanco que se encuentre entre los delimitadores de cierre de las instrucciones y los delimitadores de apertura de las instrucciones siguientes. Sin embargo, se recomienda utilizar espacios para mejorar la legibilidad.

Los programadores de Visual Basic pudieran imaginar que también existen los InputBox o los MsgBox en Vbscript como sucede en Visual Basic, y es cierto. Sin embargo hay que considerar que estos comandos, aún cuando se ejecutan del lado del servidor sólo el navegador de Microsoft interpreta el resultado correctamente. Si se desea utilizar este tipo de entrada y salida de datos o mensajes, es recomendable que utilice mejor herramientas de JavaScript, como son los llamados **alert**.

SENTENCIAS DE CONTROL

En VbScript como en cualquier otro lenguaje existen sentencias que nos permiten repetir varias veces ciertas acciones, o validar ciertas condiciones. A continuación se muestran brevemente estas sentencias.

Condicionales

Los bucles IF son comunes en cualquier lenguaje de programación. Sirven para comprobar si una variable cumple con un requisito o esta dentro de algún rango. Simplemente se fijan en el valor de la variable y si este concuerda con nuestra(s) condición(es) del IF, entonces continua, de lo contrario se salta el IF.

La sentencia SELECT CASE es otra sentencia condicional, y es especialmente útil para cuando se analizan diferentes valores que una variable puede tener.

Un ejemplo donde podemos combinar ambos es el siguiente:

```
<%dim mes, fecha
fecha = Date()              'obtenemos la fecha actual
mes = Month(fecha)          'obtenemos el número del mes corriente
select case mes
 case 1
  Response.Write("Es Enero" & "<br>")
 case 2
  response.write("Es Febrero." & "<br>")
 case 4
  response.write("<font color=BLUE>Es Abril." & "</font><br>")
end select
if (mes<=6) then
 Response.Write("<b>Falta mucho para navidad!." &  "</b><br>")
else
 Response.Write("Cada vez falta menos para Navidad." & "<br>")
end if
%>
```

El resultado es el de la figura 5.10.

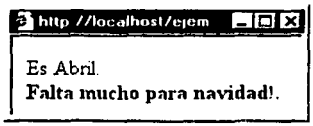

Figura 5.10 Ejemplo de Select...Case...

Los Bucles Do While... Loop sirven para ejecutar las instrucciones contenidas en el mientras se cumpla o no la condición, según se establezca.

```
<%
tam = 1
'escribe en el navegador hasta que tam sea menor que 8
Do while tam < 8
 %>
 <font size=<%=tam%>>
 <%
 response.write("Tamaño del texto:  " & tam )
 response.write("</FONT><BR>")
 tam = tam + 1
Loop
%>
```

Que escribirá en el navegador lo que se muestra en la figura 5.11.

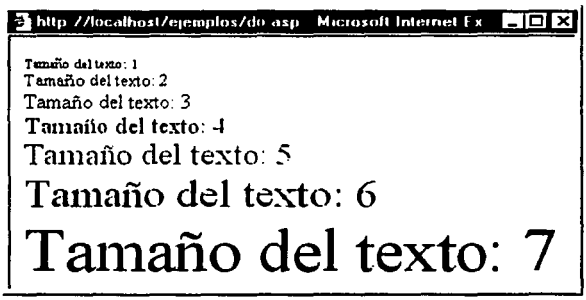

Figura 5.11 Ejemplo de bucles DO...WHILE...

BUCLES DO UNTIL ... LOOP

Este bucle funciona de manera muy similar al anterior, con la salvedad de que el BLOQUE DE INSTRUCCIONES se ejecuta hasta que (UNTIL) se cumple la condición. Cuando se cumple la condición deja de ejecutarse el bucle. La estructura general es la siguiente:

```
DO UNTIL (condición)
   BLOQUE DE INSTRUCCIONES
LOOP
```

Como siempre, la mejor manera de entender el funcionamiento es con un ejemplo. Supongamos el mismo caso de antes. Se tiene que pedir la clave mientras no se introduzca correctamente. Veamos como hacerlo.

Bucle Do...Until

```
<HTML>
<HEAD>
<TITLE>Prueba de DO WHILE ... LOOP</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE = "VBScript">
   OPTION EXPLICIT
   DIM clave
   clave = ""
   DO UNTIL (clave = "AUTORIZADO")
      ' Ejecuta hasta que la clave es igual a AUTORIZADO
      clave = INPUTBOX ("Teclea la clave:")
   LOOP
   MSGBOX ("La clave ya es correcta")
</SCRIPT>
</BODY>
</HTML>
```

Mientras la clave no es AUTORIZADO, aparece la ventana de la figura 5.12, como sería en Visual Basic.

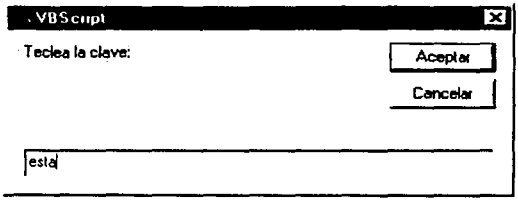

Figura 5.12 Mensaje del bucle DO...UNTIL

Y cuando la clave es correcta aparece la ventana de la figura 5.13.

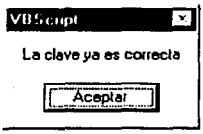

Figura 5.13 Mensaje de clave correcta

Hay que remarcar que, si bien este ejemplo muestra claramente que tanto el InputBox como el MsgBox funcionan de forma similar en VBScript como en Visual Basic en el navegador de Microsoft, el Internet Explorer, esto no funciona en el Netscape Navigator, debido a que son especificaciones de Microsoft. Por tanto, se recomienda que si se desea utilizar salidas (o mensajes al usuario) de este tipo, se utilice JavaScript, como pueden ser los llamados alert.

BUCLES WHILE ... WEND

Estos bucles datan de los primeros tiempos de BASIC. Han sido sustituidos por los bucles DO WHILE ... LOOP, cuyo funcionamiento es idéntico. Sin embargo, los intérpretes actuales todavía los soportan como apoyo a los programadores acostumbrados al modelo antiguo. Su estructura general es la siguiente:

```
WHILE (condición)
   BLOQUE DE SENTENCIAS
WEND
```

De todas formas, me permito aconsejar al lector que no se acostumbre a usar este formato de bucle, ya que es posible que futuras versiones del intérprete dejen de contemplarlo.

ROMPIENDO BUCLES

En ocasiones es necesario interrumpir de manera forzada la ejecución de un bucle, por ejemplo si se produce una situación que aconseje la continuación del programa ignorando el resto del proceso del bucle. Para ello se emplea la instrucción EXIT. Esta

instrucción presenta tres formatos distintos, según el tipo de bucle en que se implemente. Si queremos causar la ruptura de un bucle FOR... NEXT, utilizaremos EXIT FOR. Para romper un bucle DO WHILE ... LOOP o un bucle DO UNTIL ... LOOP utilizaremos la instrucción EXIT DO. Si lo que queremos es romper un bucle WHILE ... WEND usaremos EXIT WHILE. Veamos un ejemplo práctico. Vamos a suponer que estamos contando números y queremos que la cuenta se interrumpa cuando el cuadrado del número por el que vamos sea mayor que 99. Veamos el siguiente código:

Salir de un bucle

```
<HTML><HEAD><TITLE>Prueba de EXIT</TITLE></HEAD><BODY>
<%   DIM num
   DIM cuad
   num = 0
   cuad = 0
   DO WHILE (num < 1000)
      num = num + 1
      cuad = num * num
      IF (cuad > 99) THEN
      EXIT DO
      END IF
      response.write("<font color=blue>" & num & "</font> al cuadrado es <font
color=red>" & cuad & "</font><BR>")
   LOOP
   response.write("Se rompió el ciclo.")
%></BODY></HTML>
```

El resultado del código anterior es el de la figura 5.14

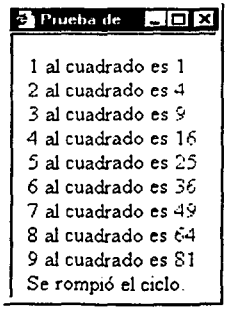

Figura 5.14 Romper un bucle DO...WHILE...

Este ejemplo ilustra el funcionamiento de la instrucción EXIT DO. Los otros dos formatos actúan de manera similar. La instrucción EXIT no es de uso muy frecuente, ya que casi siempre se pueden encontrar soluciones mas elegantes, pero conviene conocerla. Ya se empieza a intercalar más el código ASP entre el código HTML, y ya no se emplea la declaración del lenguaje, y se ve que el resultado es el mismo que si se hubiera hecho esto.

BUCLES FOR... NEXT

Esta estructura se emplea cuando es necesario repetir un bloque de operaciones un número determinado de veces. Un bucle FOR ... Next emplea para ello un variable de control que actúa como contador de las veces que se ha procesado el bucle. La variable que actúa como contador parte de un valor_inicial. Cada vez que se ejecuta el BLOQUE DE SENTENCIAS incluido en el bucle la variable de contador se incrementa en una unidad. Opcionalmente el incremento puede ser diferente de la unidad, incluyendo el parámetro STEP seguido del índice de incremento que deseemos. Cuando el contador alcanza el valor_final se deja de ejecutar el bucle y se sigue ejecutando el programa a partir de la instrucción que va detrás de NEXT. La estructura general es la siguiente:

```
FOR contador = valor_inicial TO valor_final STEP incremento
   BLOQUE DE SENTENCIAS
NEXT
```

Ejemplo del Bucle For...Next

```
<HTML><HEAD><TITLE>FOR...NEXT</TITLE></HEAD><BODY>
<%dim linea
response.write("Inicia la cuenta de líneas<br>")
FOR linea = 2 TO 5
 response.write("Línea " & linea & "<br>")
NEXT
%>
</BODY></HTML>
```

El resultado es el que se muestra en la figura 5.15.

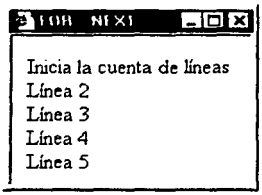

Figura 5.15 Bucle For...Next

5.1.2.1 LOS FORMULARIOS Y EL INTERCAMBIO DE DATOS

Una de las características importantes de la Web es la posibilidad de que los usuarios nos proporcionen información, lo que brinda interactividad. Utilizando formularios podemos pedir ciertos datos a nuestros usuarios, y en función de lo que nos indiquen efectuar unas acciones u otras. Se puede incluir un formulario en una página Web con las etiquetas <FORM> y </FORM>. Los parámetros más importantes son action, para indicar la dirección del archivo al que se mandarán los datos recogidos por el formulario, y method, que sirve para elegir la forma en qué se transmitirán éstos datos, el cual puede tener alguno de los dos valores posibles: GET y POST. El primero es adecuado para casos en los que los datos recogidos por el formulario no vayan a ser procesados para efectuar alguna operación, mientras que el segundo es mejor para el caso contrario.

Por ejemplo, si pedimos la introducción de una cadena de caracteres para luego realizar una búsqueda en una base de datos de acuerdo a dicha cadena, se sugiere utilizar GET. Si la queremos para modificarla y/o introducirla en la base de datos, se sugiere utilizar POST. Ambos métodos se pueden utilizar indistintamente para cualquier caso. Si no se indica nada en method se supone GET, y si es action el que no usamos, los datos recogidos por el formulario se mandarán a la misma página en la que se encuentra el formulario.

Básicamente se requieren de dos páginas: la que contendrá el formulario (que puede ser HTML o ASP) y la que procesará la petición (que debe ser una página ASP).

Por cuestiones de seguridad todas las páginas del sistema , incluyendo formularios, tendrán extensión asp, aún cuando se maneje contenido HTML.

Veamos un ejemplo de cada uno de estos métodos. Consideremos que tenemos un formulario para el envío de datos. Al pulsar el botón Enviar, el contenido de cada elemento del formulario es enviado a la página que indicamos en el atributo ACTION de la etiqueta FORM. Para recoger los valores que han sido pasados a través de un formulario utilizamos el objeto Request y, dependiendo de cómo fueron pasados, por el método GET o POST, usaremos QueryString o Form, como veremos a continuación.

Método GET

Si usamos el método GET, los datos son enviados mediante la URL y pueden ser vistos en esta. Para recogerlos deberemos usar Request.QueryString("variable"). Un ejemplo de esto lo podemos ver cuando utilizamos un servicio de búsquedas en Internet, por ejemplo, si en Google hacemos una búsqueda de las palabras "ingreso UNAM", obtendremos las ventanas de las figuras 5.16 Y 5.17. Al ver la URL del documento, nos damos cuenta que ha cambiado

de la página principal http://www.google.com a la página de resultados

http://www.google.com/search?q=**ingreso+UNAM**&hl=es&lr=lang_es%7Clang_en

Como se ve, los datos introducidos en el formulario de búsqueda se han incluido en la URL, como resultado de utilizar el método GET. Por tanto, si queremos conservar ocultos lo más posible estos datos, este método no es el adecuado.

Método POST

A diferencia del método GET, con el método POST los datos enviados no se ven en la barra del navegador. Para recogerlos deberemos usar Request.Form("variable"). Puede consultar el sub-capítulo 5.1.2.3 para más detalles. Los resultados se ven en las figuras 5.18 y 5.19.

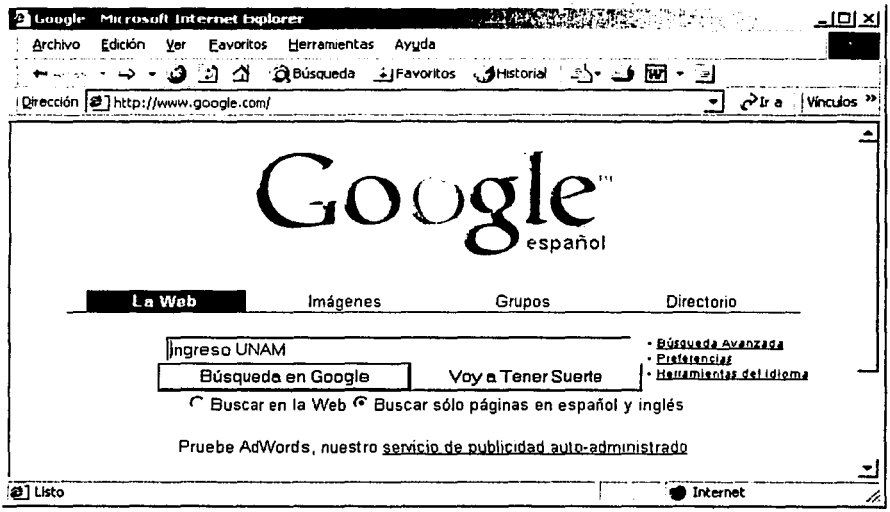

Figura 5.16 Formulario base para el método GET de ejemplo

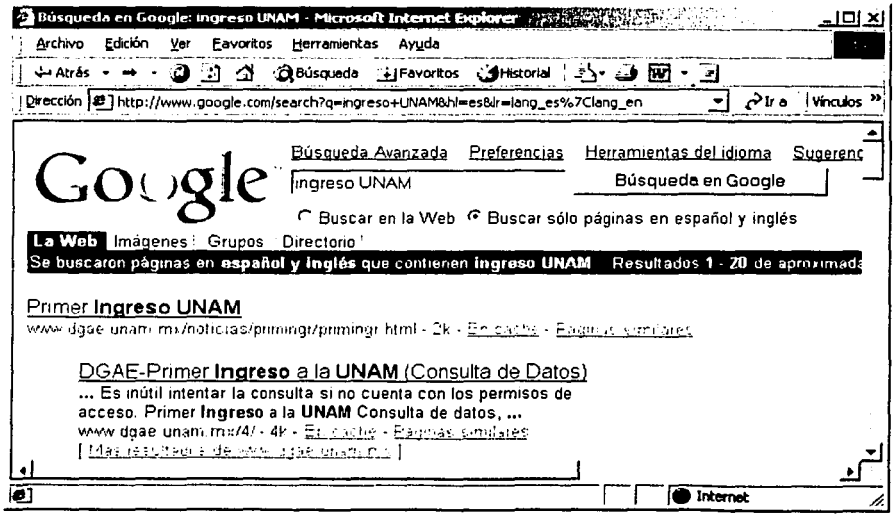

Figura 5.17 Formulario procesado

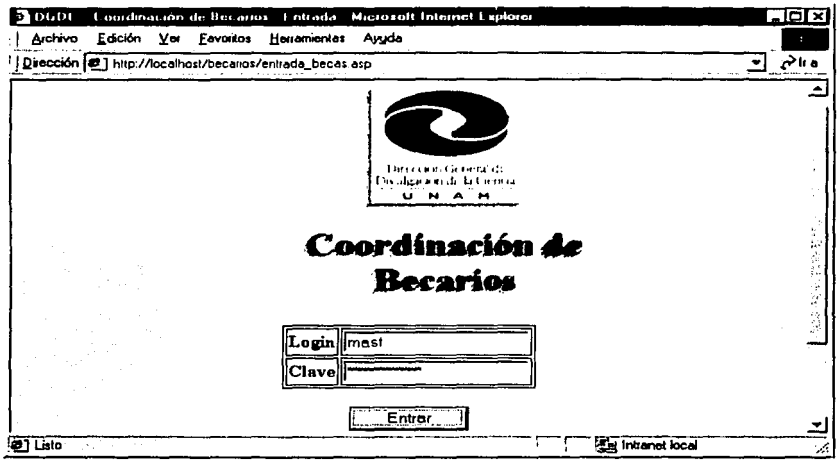

Figura 5.18 Formulario base para el método Post

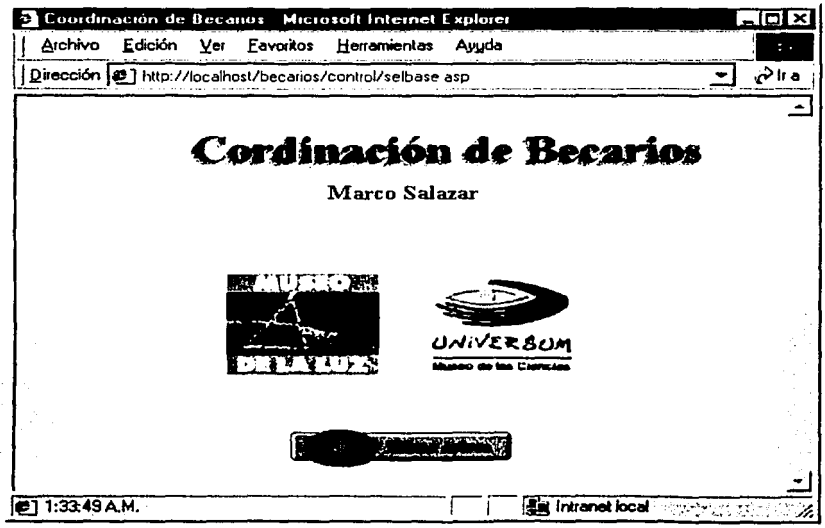

Figura 5.19 Formulario procesado

5.1.2.2 EL ACCESO A LOS DATOS

Existen diferentes formas de realizar una conexión con una Base de Datos (BD en adelante). La manera de hacer la conexión depende del motor de BD que utilicemos. A continuación mostraremos los tipos de conexiónes que existen.

1 - Conexión con DSN

Ya hemos visto cómo crear un origen de datos con DSN (capítulo 4). Ahora debemos conectar la BD en la pagina ASP con el DSN que creamos.

```
<%
'Definimos la variable para la conexión.
Dim Conex
Set Conex = Server.CreateObject ("ADODB.Connection")
'y ya estamos conectados a nuestra base de datos.
Conex.Open "nombre DSN de la BD"
'aqui abrimos la tabla. ...
%>
```

No es difícil, sólo hay que aprenderse la línea de conexión. Pero hay que tener en cuenta que este tipo de conexión no servirá si se sube, además de su sitio, su BD a Internet. Para esto se debe usar la siguiente conexión.

2 - Conexión sin DSN

Este tipo de conexión es más complicada, pero es lo que debemos utilizar si queremos olvidar el panel de control, ya que hacemos la conexión a la BD mediante comandos. Hay que tener cuidado con la sintaxis.

Para Access usando ODBC

```
<% Dim Conex
'Creamos el objeto de conexión ahora...
Set Conex = Server.CreateObject ("ADODB.Connection")
Conex.open "DRIVER={Microsoft Access Driver (*.mdb)};
DBQ=C:\misitio\db\mibase.mdb;" %>
```

Para Access 97 usando OLEDB:

```
<%Dim Conex
'Creamos el objeto de conexión ahora...
Set Conex = Server.CreateObject ("ADODB.Connection")
Conex.Open"Provider=Microsoft.Jet.OLEDB.3.51;
Data  Source=C:\misitio\db\mibase.mdb;" %>
```

Para Access 2000 usando OLEDB:

```
<%Dim Conex
'Creamos el objeto de conexión ahora...
Set Conex = Server.CreateObject ("ADODB.Connection")
Conex.open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\misitio\db\mibase.mdb;" %>
```

Para SQL Sever:

```
<% Dim Conex
'Creamos el objeto de conexión ahora...
Set Conex = Server.CreateObject ("ADODB.Connection")
Conex.Open "driver={SQL Server};server=TU_SERVIDOR; database=NOMBRE_BASE;
uid=nombre_usuario;pwd="clave_usuario" %>
```

En los 3 primeros ejemplos, referidos todos a conexiones con bases de datos Access, use un path (ruta) fijo. Se puede hacer uso de la función Server.MapPath(), la cual devuelve el path donde ejecutamos el script. Consideremos el siguiente ejemplo

```
<% Dim path
'guardamos en la variable path lo que devuelve la función
path = Server.MapPath("./")
Response.Write path
%>
```

El código ejecutado, muestra el contenido de la variable path

C:\InetPub\WWWRoot\directorio_actual

La variable path contiene el directorio donde se ejecuta el script. Entonces esto se podría usar para indicar el path a la base de datos (que en los ejemplos anteriores lo escribíamos a mano).

Author's Name
Author's Institution

Abstract

Publication Info

1. Se debe tener la BD con los usuarios autorizados
2. Se elabora una página-formulario donde el usuario proporcionará su login y su clave
3. Una página ASP realizará el proceso de autenticación del usuario, realizando una búsqueda en la BD de usuarios autorizados
4. Para lograr que todas las páginas del sitio queden protegidas, debemos utilizar un código ASP en cada una de las páginas, el cual comprobará si ese usuario fue reconocido en el proceso de autenticación

Algo que no se ha mencionado es que se creó un anexo al sistema, y es el que se encarga de dar los permisos al mismo. Personal autorizado maneja este sistema (Sistema de Administración) y desde el se accede a la BD de usuarios ya mencionada, además de otra BD de administradores. Los administradores son quienes realizan operaciones sobre nuevos usuarios del sistema: altas, bajas, cambio de datos como la clave, etc., y nada más. Los usuarios son quienes, una vez autorizados por los administradores, pueden trabajar con el sistema, con los permisos que les fueron asignados.

A continuación describiremos las acciones realizadas para la seguridad de las páginas. Como ya se ha dicho, los ejemplos presentados nada tienen que ver con el sistema original, por cuestiones de privacidad, y sólo se muestran las partes que sirvan a otros para asegurar su sitio.

La página de entrada es la que pide la autenticación del usuario. Al proporcionar un login y una clave, el sistema realiza una búsqueda en la BD de usuarios autorizados. Si no lo encuentra regresa a la misma página. Si el usuario es reconocido, el sistema le permite la entrada a todas las páginas del sistema, de acuerdo a los permisos que posea. Este punto es importante, ya que no se puede acceder a ninguna página del sistema sin haber pasado por la página de entrada. Esto brinda la seguridad de que aún cuando el sistema se encuentre en Internet y se pensara que cualquier persona pudiera saber la dirección de alguna de las páginas del mismo, no podrá verla, ya que al no haber sido reconocido por el sistema, es enviado a la página de entrada para que se identifique.

Otra forma de proteger las páginas es utilizando un script de java que bloquea las funciones del botón derecho del ratón, tanto para el Internet Explorer como para el Netscape Navigator. Una de las funciones del botón derecho es ver el código de la página, opción que también está disponible en las opciones del navegador. Entonces, si bien esta opción queda bloqueada en el botón derecho, queda disponible en el menú del navegador, a menos que la ventana donde opere la página la hayamos creado sin las opciones del menú, lo que también nos quitaría la opción de salvar la página. No obstante, hay que recordar que como ya se ha visto antes, el navegador sólo mostraría código HTML puro, ya que las directivas ASP se procesaron en el servidor, por lo que el cliente solo muestra el resultado de dicho procesamiento, por lo que realmente no hay problema. En el sistema se emplea este script, sólo como medida extra de seguridad.

El siguiente es el código de la página de entrada. Nótese que el script que bloquea el botón derecho debe ir antes que cualquier otra cosa en la página.

```
<Script language=JavaScript>
function right(e) {
if (navigator.appName == 'Netscape' && (e.which == 3 || e.which == 2)){
alert("Pulsación no permitida.");
return false;
}
else if (navigator.appName == 'Microsoft Internet Explorer' &&
(event.button == 2 || event.button == 3)) {
alert("Pulsación no permitida.");
return false;
}
return true;
}
document.onmousedown=right;
if (document.layers) window.captureEvents(Event.MOUSEDOWN);
window.onmousedown=right;
</script>
<% Session("Conocido")=False %>
<HTML>
<HEAD>
<TITLE>DGDC - Coordinación de Becarios - Entrada</TITLE>
</HEAD>
<BODY text=black background="/images/botones/Fazu12.jpg">
<center>
```

```
<div id="sombra" style="position:relative;width:250;height:50;top:10;left:30;text-
align:center;filter:shadow(color=BLUE)">
<IMG SRC="/images/dgdc_logo.gif" ALT=" DGDC ">
<p clear=all>
<font color=CYAN size=6>
<b>Coordinación de Becarios</b>
</font></div><p>
<FORM NAME="busca" METHOD="post" ACTION="pagina_busca.asp ">
<TABLE BORDER="1" bordercolor=blue>
<TR><TD><b>Login</b></TD><TD><INPUT TYPE="text" NAME="login" size=20
maxlength=15></TD>
<TR><TD><b>Clave</b></TD><TD><INPUT TYPE="password" NAME="clave" size=20
maxlength=15></TD>
<!--    <TD><INPUT TYPE="submit"></TD><TD><INPUT TYPE="reset"></TD>-->
</TABLE><br>
<INPUT TYPE="submit" value="    Entrar    ">
</FORM>
<p>
Página optimizada para una resolución de pantalla de 800x600 pixeles<p>
Compatible con <IMG SRC="/images/msie.gif"> y <IMG SRC="/images/Nets03.gif">
</center>
</BODY>
</HTML>
```

La pantalla que se muestra es la misma que la de la figura 5.18. el mensaje de alerta generado por el script es el de la figura 5.20.

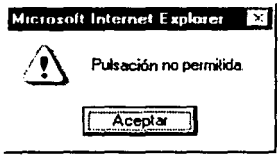

Figura 5.20 Mensaje que produce el bloqueo del botón derecho del ratón

Note que el objeto FORM tiene la opción post en METHOD, y hace referencia a pagina_busca.asp en el método ACTION. Como ya se vio, el método post hace que los datos enviados del formulario de entrada a la página donde se realiza la identificación del usuario permanezcan ocultos, lo que garantiza la seguridad de la página y de la clave introducida.

El campo de texto de la clave muestra siempre asteriscos en lugar de la palabra en sí, como ocurre en cualquier otro sitio profesional como los servicios de correo.

El código de la página referenciada por el método ACTION es como el que sigue

```
<% Dim consulta,usuario,clave,conocido,login
'variables que reciben los datos del formulario
login=Request.Form("login")
clave=Request.Form("clave")
'solo si se escribió un login y una clave hace la búsqueda
if len(login)>0 AND len(clave)>0 then
 busca
else 'te regresa si faltan datos
 Response.Redirect "/becarios/entrada_becas.asp"
end if
%>

<%
sub busca
dim esta
'sentencia SQL
consulta="SELECT login, clave FROM quienes WHERE login='" & login & "' AND
clave='" & clave & "'"
'conectar a la bd
SET conecta = SERVER.CREATEOBJECT("ADODB.CONNECTION")
conecta.open "los_usuarios"
'ejecutar consulta
SET rs = conecta.execute(consulta)
esta=False
'si es fin de archivo
if rs.EOF then
 Session("Conocido")=False
else
 Session("Conocido")=True
 Session("usuario")=rs.Fields("usuario")
 permisos=rs.Fields("sus_permisos")
 Session.TimeOut=60
 esta=True
end if
'cierra la BD
rs.Close
conecta.Close
set rs=nothing
set conecta=nothing
'si se encontró, muestra el nombre del usuario
if esta then
```

```
Session("usuario")= "<b>" & Session("visor") & "</b>"
'obtiene el tipo de página que muestra, según los permisos del usuario
select case permisos
 case "Puede modificar"
  entrar="modificar.asp"
 case "Es un Tutor"
  entrar="es_tutor.asp"
 case "Es un Directivo"
  entrar="es_directivo.asp"
case else
   entrar="/becarios/entrada_becas.asp"
end select
'manda a la página correspondiente al tipo de usuario
 Response.Redirect entrar
'si no se encontró, regresa a la página de entrada
else
 Response.Redirect "/becarios/entrada_becas.asp"
end if
end sub
%>
```

Al inicio del código de la página se declaran las variables que emplearemos. Ahora se ve la utilidad de Request.Form("variable"), que es el encargado de pedir los datos al formulario. Después nos aseguramos que se hallan escrito datos en los campos de texto para poder hacer la búsqueda, de lo contrario no tiene caso.
Aquí vemos una sentencia que no se había mencionado:

```
Response.Redirect "nombre_pagina"
```

Response.Redirect redirecciona el navegador a la página designada por nombre_pagina. Es importante que note que no hay código HTML en esta página, es puro lenguaje ASP. El Response.Redirect debe escribirse antes que el código HTML, si lo hubiera, de lo contrario produce un error. En la variable *consulta* almacenamos nuestra sentencia SQL, que nos servirá para hacer la búsqueda en la BD. Para una mayor referencia sobre el lenguaje SQL, consulte la bibliografía citada. Después hacemos el acceso a la BD, como ya se ha visto.
La sentencia conecta.execute(consulta) ejecuta el SQL. La variable *esta* nos servirá para saber si la búsqueda produjo resultados, y en base a ello decidiremos si seguimos con el resto del código o regresamos a la página de entrada. El EOF nos dice si estamos en el fin del archivo de la BD, lo que indica que la búsqueda no produjo

resultados. Si no es el fin de archivo, entonces el usuario fue reconocido y obtenemos su nombre y sus permisos.

La variable de sesión *Conocido* es nuestra bandera que emplearemos en todas las páginas del sistema para saber si alguien está autorizado a ver una página o no. El objeto Session.TimeOut=60 establece que el tiempo de vida de la página es de 60 segundos. El tiempo de vida de una página se entiende como el periodo en que el usuario trabaja en una página. El servidor detecta cuando una página ha dejado de ser activa y luego de un rato, la página caduca. Por defecto el tiempo de vida es de 20 segundos. Se establecen 60 segundos para que el usuario pueda trabajar sin presiones de tiempo, ya que cuando la página caduque el sistema regresa al usuario a la página de entrada.

Después cerramos la BD y si la variable *esta* es verdadera (True) redireccionamos al usuario al tipo de página que le corresponda de acuerdo a sus permisos. Como se ve, todo se hace dentro de una sub-rutina. Cuando el usuario es identificado, se encuentra en la página que le corresponde. En el caso de la figura 5.21, el usuario puede modificar los datos.

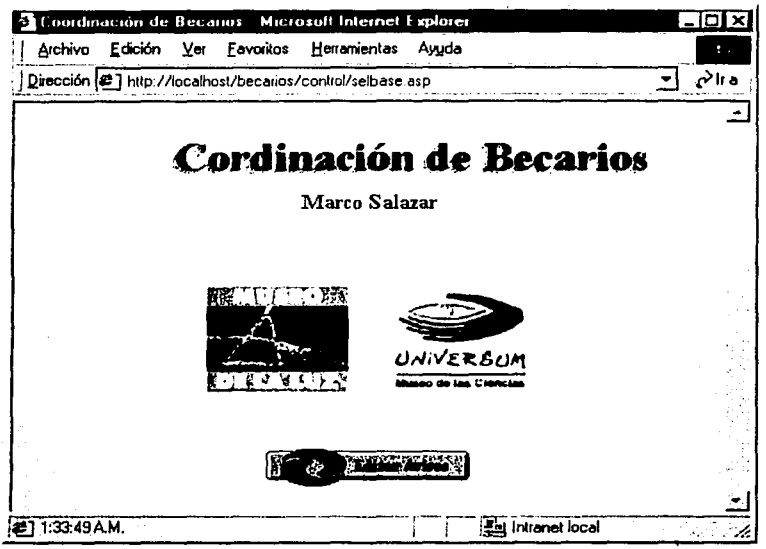

Figura 5.19 Usuario conocido

Queda el asunto de la seguridad en el resto de las páginas. Para eso nos sirve la variable de sesión Conocido. Basta con colocar el siguiente código ASP al inicio de cada página

```
<%Dim permiso
permiso=Session("Conocido")
if not permiso then
 Response.Redirect "/becarios/entrada_becas.asp"
else
 'resto del código de la página
%>
```

Este código verifica el valor de la variable de sesión *Conocido*, y si es falsa es que el usuario no ha sido reconocido o intento acceder a la página sin haber pasado por la página de entrada por lo que el sistema lo redirecciona para que se identifique en la página de entrada. Si el usuario es conocido, después de la sentencia *else* se escribe el resto del código de nuestra página.

Hay que mencionar que si el usuario cierra por error la ventana del navegador donde está visualizando alguna de las páginas del sistema, tiene que identificarse de nuevo para poder entrar, debido a que todas las variables se destruyen al cerrar el navegador.

Tocando el tema de los errores, el ASP no tiene un control específico sobre ellos. Sólo puede acceder a propiedades tales como número, descripción, etc., que son de lectura únicamente.

Los errores que se podrían producir son de dos tipos: errores en la programación o errores en tiempo de ejecución. Los errores de programación se producen si, por ejemplo, se escribe mal alguna instrucción de ASP, una sentencia SQL, etc. Ese tipo de errores ya fue solucionado, las páginas han sido revisadas y no tienen ese tipo de errores. Los errores en tiempo de ejecución pueden ocurrir cuando el usuario trabaja con el sistema, y básicamente se refieren al acceso a la BD.

Por ejemplo, la figura 5.22 muestra el mensaje que se produce cuando una sentencia SQL está incompleta (error de programación), y la figura 5.23 muestra el error que se produce cuando un usuario intenta acceder al sistema desde Internet, y al mismo tiempo un usuario en la máquina servidor abre la BD en modo Diseño.

Si se diera el caso de que más de un usuario intente acceder al sistema al mismo tiempo, lo cual es prácticamente imposible debido a que el tiempo de conexión a Internet es distinta para cada usuario aún estando en la misma sub-red, no hay problema, ya que la BD permite lecturas concurrentes (es decir, al mismo tiempo). Si, por otro lado, se intenta modificar la BD desde sitios distintos, la BD se bloquea en la primer petición de actualizar los datos hasta que termina, y continúa con la siguiente petición de actualización. El usuario prácticamente no se da cuenta de ello.

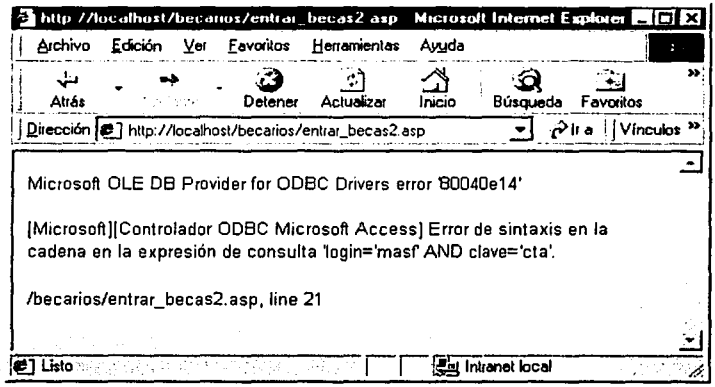

Figura 5.22 Error en la sentencia SQL

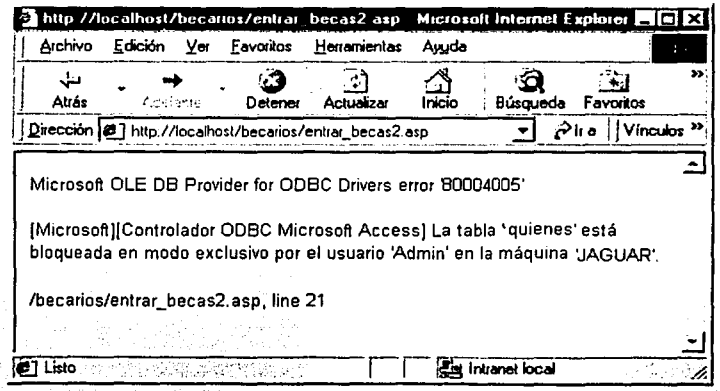

Figura 5.23 Error de acceso a la BD de los usuarios conocidos

5.2 EL SISTEMA EN INTERNET

El proceso ha sido largo, hemos pasado de un sistema para una sola PC hasta un sistema manejable desde Internet. Todos los aspectos tratados en todos los capítulos han aportado partes importantes para la elaboración del sistema.

Ahora solo resta establecer los criterios de acceso al sistema, y presentar el mismo ante la Coordinación de Becarios de la DGDC para su aprobación y utilización.

5.2.1 USUARIOS Y PERMISOS

Los tipos de usuarios se muestran en la tabla 5.

Tipo de Permiso	Descripción	Sistema
Coordinación	Puede modificar y consultar la BD de los becarios.	Becarios de ambos museos
Tutor	Puede consultar la BD de los becarios a su cargo solamente. No puede hacer modificaciones.	Becarios del museo correspondiente
Directivo	Puede consultar la BD de los becarios de cualquier tutor. No puede hacer modificaciones.	Becarios de ambos museos

Tabla 5 Permisos del Sistema en Internet

5.2.2 SISTEMA FINAL

El sistema para Internet fue instalado en una PC con Windows NT 4.0, con el Service Pack 6 y el Option Pack 6, obtenidos del sitio de Microsoft. Cuenta con 64 MB de memoria RAM, 2 GB de espacio en disco y un monitor con una resolución de pantalla de 800x600 pixeles, con una profundidad de color de 32 bits. La recomendación es que la memoria RAM sea de por lo menos 64 MB, y el desempeño del servidor será mejor con una memoria mayor, sobre todo si hay varios usuarios conectados al mismo tiempo al sistema, por la creación de las variables requeridas para cada uno.

En cuanto al espacio en disco, bastan 3 MB para el almacenamiento de las BD del sistema, por lo que 2 GB son mas que suficientes.

A continuación se muestran las pantallas que forman el Sistema de Control de Becarios para la DGDC. Las funciones del sistema son básicamente las mismas que para el sistema local creado con Visual Basic, solo que ahora son en Internet. Se muestran las pantallas correspondientes a un usuario con permisos de Coordinador. Los otros tipos de páginas son similares a estas, con menos opciones de acuerdo a los permisos. Siguiendo el esquema, se muestra la página correspondiente al Sistema del Museo de la Luz.

La pantalla principal es la misma que la de la figura 5.21. En ella podemos elegir entre los becarios del Museo de la Luz o los del UNIVERSUM. La figura 5.24 muestra las opciones del sistema del Museo de la Luz.

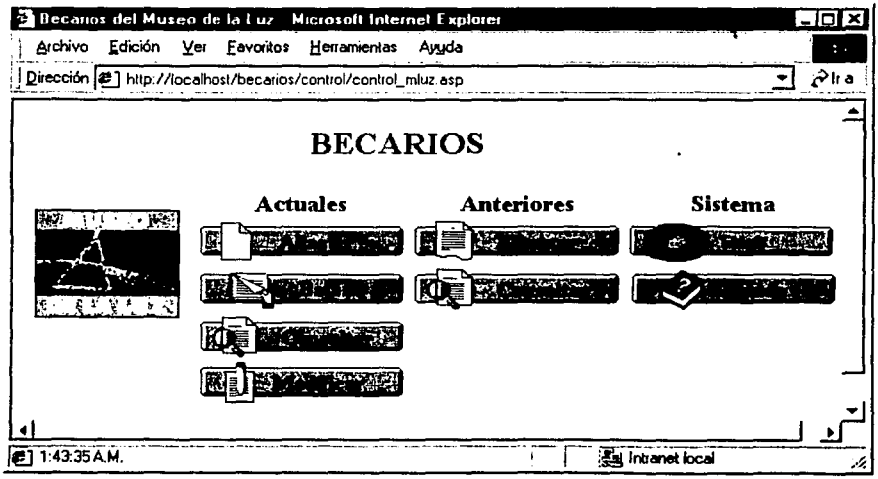

Figura 5.24 Opciones del Sistema del Museo de la Luz

Altas permite agregar nuevos becarios. Sigue el esquema del sistema local, por lo que pide un Número de Cuenta antes de agregar el registro como se ve en la figura 5.25.a. Si el becario ya se encuentra, el sistema lo informa. El sistema también identifica si ese becario no existe en la BD de becarios actuales pero se encuentra en la BD de becarios anteriores (5.25.b), lo que implica que esa persona ya fue becario del museo en otra ocasión. Esto es una mejora respecto al sistema local. Si no se encontró el becario nuevo en ningún sistema, permite que se agregue ese nuevo becario

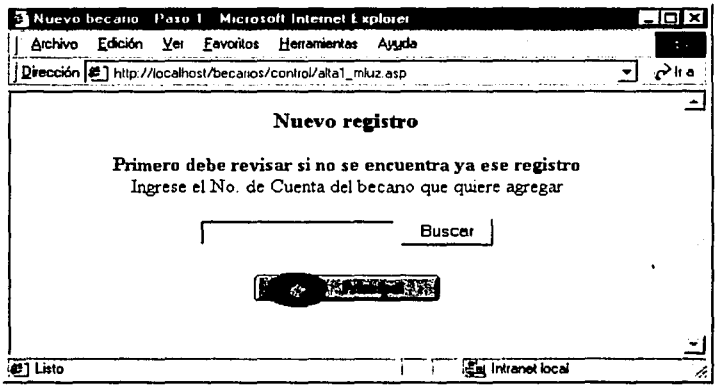

Figura 5.25.a Alta de un nuevo becario

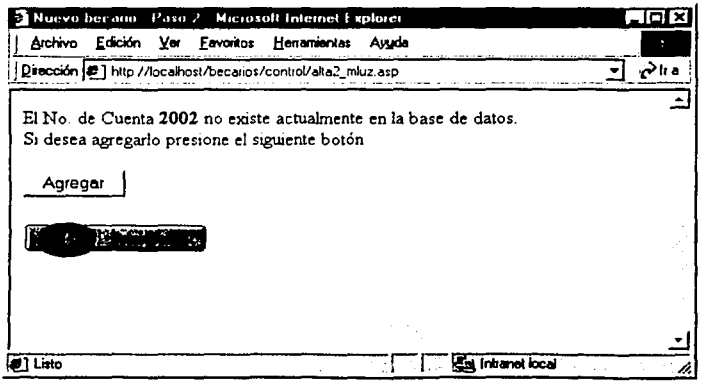

Figura 5.25.b El becario no existe actualmente

Bajas muestra una lista donde se encuentran los nombres de todos los becarios actuales. Para eliminar alguno basta con seleccionarlo y pulsar el botón Eliminar, como se ve en la figura 5.26. El becario es eliminado de la BD de los becarios actuales, y es ingresado a la DB de los becarios anteriores. Hacemos esta división para mantener un control de los becarios que tiene y que ha tenido cada museo, como era en el sistema local.

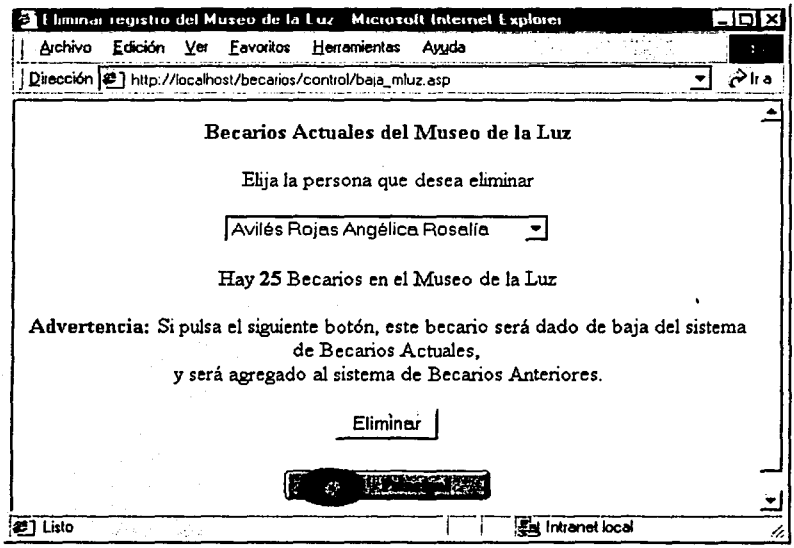

Figura 5.26 Eliminar un becario de la BD

Consultar muestra una serie de casillas de verificación, que representan los campos que se desea visualizar en la consulta. También se tienen los campos de texto necesarios para introducir las características de los datos buscados: por Nombre, Área, etc. La figura 5.27 muestra el área de consultas.

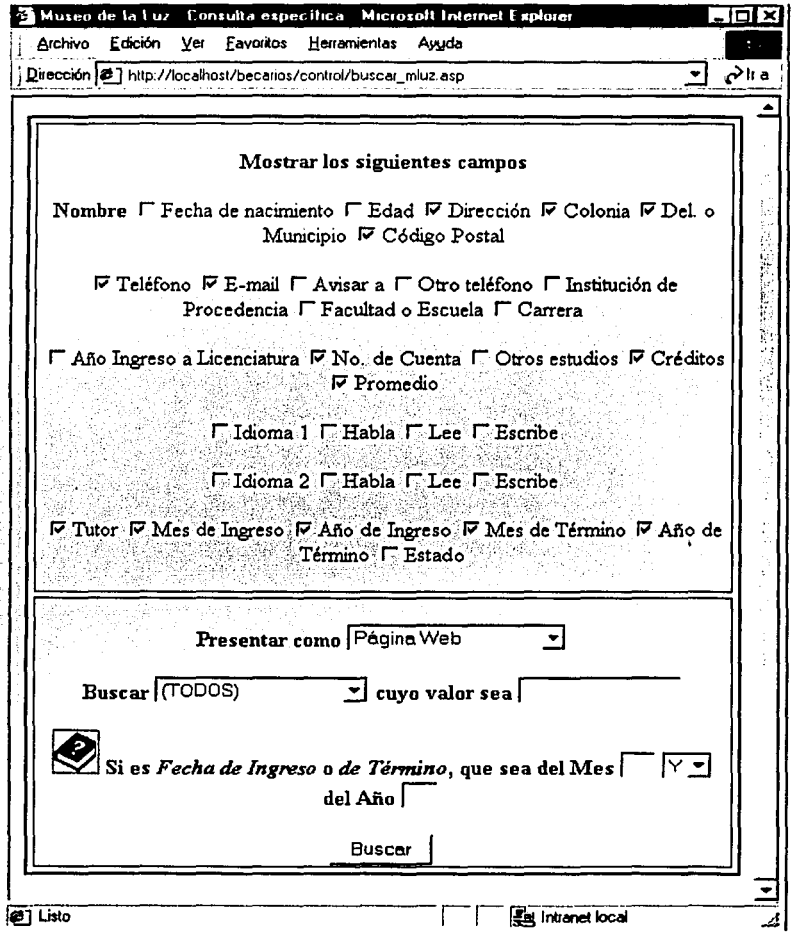

Figura 5.27 Consultar la BD

Aquí hay una parte importante y nueva. Los resultados de la búsqueda ahora se pueden presentar en formato HTML o como un documento de Excel, según se elija. Para presentar una tabla HTML como una tabla de Excel, se debe colocar la línea

Response.ContentType = "application/vnd.ms-excel"

al inicio de la página que contenga la tabla. Además, la página no debe tener ninguna etiqueta HTML que no sean las correspondientes a la tabla. Tampoco debe contener las etiquetas que marcan el inicio y fin de la página y del cuerpo de la misa, es decir, no debe llevar <HTML>,<BODY></BOBY> ni </HTML>. Al seleccionar que los resultados se presenten como documento de Excel, el navegador abre la aplicación que pueda abrir este tipo de archivos y muestra el documento. Las computadoras Mac tienen un programa que puede abrir documentos de tipo Excel, por lo que no hay ningún problema. De cualquier forma, siempre se pueden ver los resultados en el formato HTML tradicional.

Modificar muestra una ventana donde se listan todos los becarios del museo (figura 5.28) y al elegirlo y pulsar Seleccionar aparecen en esa misma página los datos del becario, y se muestran sus datos actuales y se pueden hacer cambios sobre los mismos.

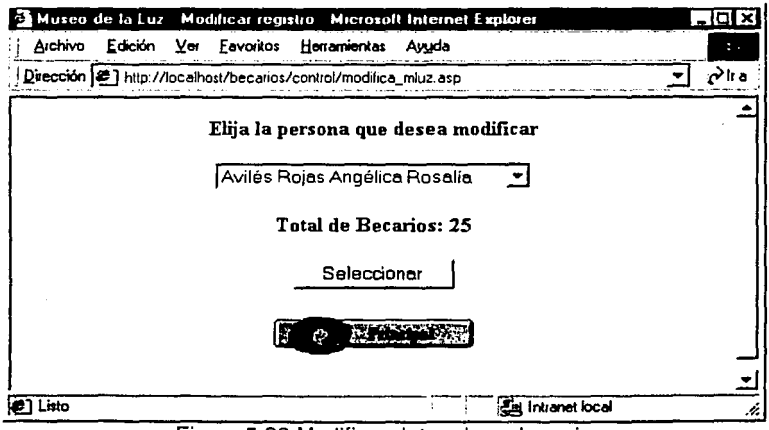

Figura 5.28 Modificar datos de un becario

Eliminar trabaja con la BD de los becarios que ha tenido el museo. Funciona como la opción de Bajas, salvo que los datos se pierden completamente de esta forma. La figura 5.29 muestra esta parte.

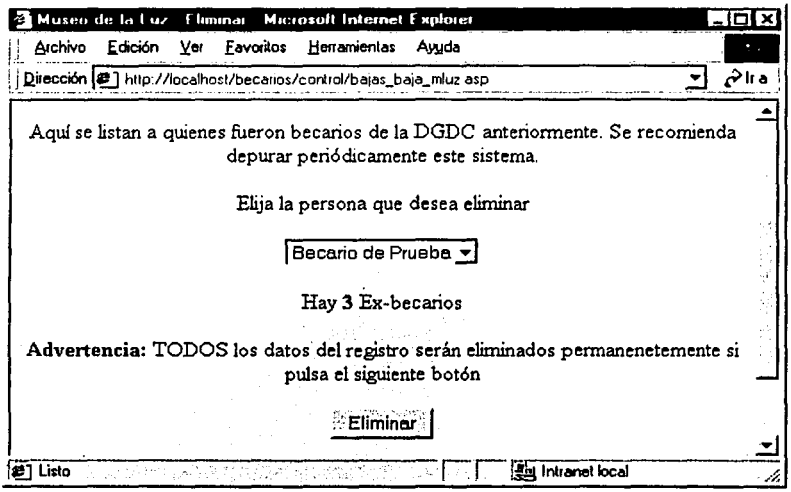

Figura 5.29 Eliminar un becario permanentemente de la BD

Elegir regresa a la página principal, por lo que permite seleccionar el sistema de trabajo, ya sea el del Museo de la Luz o el del UNIVERSUM. También permite elegir la opción de Editar los avisos de la DGDC. Por confidencialidad, esta parte de Avisos no se muestra.

Ayuda muestra una ventana de ayuda, con la descripción de los elementos del sistema, como se ve en la figura 5.30.

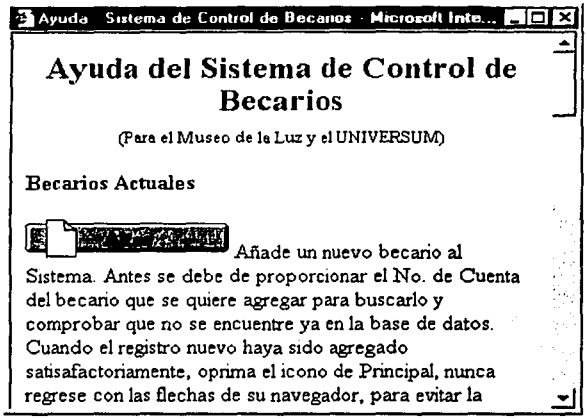

Figura 5.30 Ayuda del sistema

El mismo esquema de las páginas presentadas para el Museo de la Luz se aplica al sistema del UNIVERSUM, cambiando obviamente los datos.

Como ha de haber notado desde el inicio del trabajo con el sistema de Internet, las páginas muestran la dirección http://localhos/directorio . Esto es porque la dirección del sistema real se omite por cuestiones de seguridad, y se muestran los ejemplos del sistema establecido en la computadora donde fue creado el sistema. El funcionamiento es el mismo en ambos casos. Otro punto importante es el tiempo de acceso a los datos, lo cual depende de las características de la conexión a Internet por parte del cliente, ya que en pocos segundos el servidor accesa la información, al estar en la misma PC.

El sistema final fue puesto en marcha en Enero del 2002, inicialmente en un servidor temporal bajo Windows 98 en la ENEP Aragón y posteriormente fue migrado a un servidor en el UNIVERSUM a una PC con Windows NT 4 Server con 128 MB en RAM y 2.3 GB de espacio en disco, en Abril del mismo año, y continúa funcionando a la fecha. No se encontraron errores y a la fecha su funcionamiento ha sido satisfactorio.

CONCLUSIONES

El utilizar un modelo para el análisis permite separar las diferentes áreas que forman el desarrollo de un sistema de información. El estudio de viabilidad, análisis y especificación de requerimientos, especificación del sistema lógico y el diseño físico tratados por la metodología SSADM[1], aún con el carácter rígido de los Británicos, es una herramienta (no la única, pero si la que se utilizó) cuyas aportaciones forman y enriquecen el desarrollo del sistema requerido, permitiendo encontrar las posibles soluciones al problema plateado.

La labor que se realizó fue intensa, y el trabajo de campo fue muy importante. Las múltiples reuniones con el personal a cargo permitieron esclarecer todo el procedimiento que se realiza para que un estudiante pueda ser becario de la DGDC, y así poder determinar la mejor forma de procesar dicha información en la computadora. Se reconocieron datos que realmente no tenían una razón de ser en la forma en que se estaba llevando el control.

Es importante señalar que si bien es cierto que la cercanía que se tenga con quienes serán los usuarios del sistema es importante en la medida que puede ayudar a encontrar las partes principales del mismo, también puede traer desventajas.

Como se comentaba en el capítulo 2, en cada una de las reuniones se obtenían nuevas opiniones de lo que la aplicación debería tener, lo que en un momento nos alejó del objetivo real del desarrollo. La mayoría de las sugerencias realmente no tenían que ver con aspectos importantes, eran mas bien *añadidos* o *extras* que se deseaba el sistema tuviera y que realmente tenían que ver con el trabajo de otra área.

Por lo tanto es importarte determinar los objetivos a corto plazo que permitan llegar en un tiempo establecido al objetivo principal, ya que de lo contrario se cae en un círculo que difícilmente se rompe y que retrasa la entrega del producto final.

[1] Metodología SSADM, capítulo 1

Así pues, se consideraron sólo aquellas que fueran relevantes para que el sistema realizara su trabajo primordial: el control de becarios.

El empleo de las ASP[2] permitió crear un sistema que se puede utilizar en Internet, sin importar la plataforma que se utilice. Haber creado una versión preliminar de tipo local (el de Visual Basic) sentó las bases para desarrollar rápidamente la versión para Internet puesto que el funcionamiento principal ya se conocía, prácticamente sólo fue cuestión de migrarlo puesto que como se vió en el capítulo 5.2.2 se mantuvo en su mayoría una similitud entre las características de las funciones del sistema local y el de Internet.

Otro aspecto importante fue la elección del servidor para el sistema en Internet. Hay que mencionar que un servidor Apache donde pueden ejecutarse scripts en php o JavaScripts es una alternativa muy importante, sobre todo si se dispone de un sistema Linux y se considera la estabilidad que se obtiene de esta combinación. Considerando lo expuesto en el capítulo 1 el elegir el PWS[3] o el IIS[4] es la mejor solución si se quiere trabajar con el lenguaje ASP y se cuenta con Windows 9x/NT.

Como ya se vió, existen diferentes herramientas que permiten trabajar con bases de datos en Internet: cgi's, lenguajes como php, perl, etc. Sin embargo, el ASP nos brindó la ventaja de que permite crear páginas dinámicas con mayores capacidades, más allá de sólo trabajar con las bases de datos. Permite entre otras cosas, personalizar una página para cada usuario, condicionar el resultado de un proceso con base a los criterios que se establezcan, control de eventos, etc. Además de que el ASP tiene una forma fácil de pasar dinámicamente el contenido de una consulta a una base de datos a un documento de Microsoft Excel, parte importante a la hora de hacer los reportes en base a las consultas que se pueden hacer en el sistema, como se vió en el capítulo 5.2.2. Por otro lado, ASP permite trabajar con las mismas bases de datos con las que trabaja el sistema local, incluso usando los mismos DSN[5].

[2] ASP: Active Server Pages, capítulo 5.1.2
[3] PWS: Personal Web Server, capítulo 5.1.1
[4] IIS: Internet Information Server, apéndice A
[5] DSN: Domain Name Server: Nombres de Origen de Dominios, capítulo 4

Un manejo similar de las bases de datos se puede obtener también con JavaScript, pero su lenguaje es un poco más elaborado y estructurado al trabajar con objetos, además de que al manejar el lenguaje VBScript que ASP maneja por omisión, por tener las mismas raíces que el de Visual Basic, permite que el paso de uno al otro sea casi directo. Incluso las características del lenguaje que no se conocen se pueden intuir por la misma razón. Esto permitió que el desarrollo del sistema de Internet se diera de una forma muy rápida.

Es notorio que el crecimiento de Internet y el avance en las tecnologías empleadas para enriquecer el trabajo sobre el mismo atraerá cada vez más usuarios. La interactividad que el usuario tendrá permite una amplia gama de posibilidades, entre ellas para las bases de datos. Por ello el sistema desarrollado tiene un largo tiempo de vida, puesto que su funcionamiento será el mismo y la tecnología empleada, ASP, cubre perfectamente las necesidades que se requieren, sin olvidar lo importante que es que el sistema pueda utilizarse en plataformas distintas, con solo tener un navegador de Internet.

Por lo tanto, puedo decir que el objetivo de esta Tesis: "Implementación de un Sistema Informático para el Control de Becarios para la Dirección General de Divulgación de la Ciencia (DGDC), el cual pueda ser utilizado vía Internet", ha sido cubierto en su totalidad satisfactoriamente.

Bibliografía

[1] Procesamiento de Datos; Análisis de Sistemas
Jusseaume, Leo
México, Trillas, 1970

[2] Análisis y diseño de Sistemas de Información
Senn, James
México, McGraw-Hill, 1988

[3] Sistemas de Información
Lucas, Henry
Madrid, Paraninfo, 1987

[4] Sistemas de Información para la toma de Decisiones
Cohen Karen, Daniel
México, McGraww-Hill, 1996

[5] Elementos y Herramientas en el Desarrollo de Sistemas de Información
G. Prattini, Mario
Madrid, Ra-Ma, 1995

[6] Fundamentos y Modelos de Bases de Datos
Miguel Castaño, Adoración de
Madrid, Ra-Ma, 1997

[7] HTML Dinámico, ASP y JavaScript
Bobadilla Sancho, Jesús
Madrid, Ra-Ma, 1999

[8] VBScript by Example
Honeycutt, Jerry
E.U.A, QUE Corporation, 1996

[9] Windows NT Server, Survival Guide
Sant'Angelo, Rick
Indiana, SAMS Publishing, 1996

[10] Creación de Aplicaciones Web en Windows NT
Bobadilla, Jesús
España, Ra-Ma, 1999

Sitios visitados en Internet

Windows NT Server 4.0 Option Pack for Intel (x86)
http://www.microsoft.com/ntserver/nts/downloads/recommended/NT4OptPk/ntsx86.asp

Windows NT 4.0 Service Pack 6
http://www.microsoft.com/downloads/release.asp?ReleaseID=15103&area=search&ordinal=28

Tutorial IIS
http://www.htmlpoint.com/iis/

Introducción a la programación en ASP
http://www.desarrolloweb.com/articulos/244.php?manual=8

Artículos de ASPfácil
http://www.aspfacil.com/articulos/

Maestros del Web
http://www.maestrosdelweb.com/

SUN Chili!Soft ASP
http://www.chilisoft.com/chiliasp/default.asp

Manual de ASP. Tutorial de ASP. WebEstilo
http://www.webestilo.com/asp/

LaVariable.com. Artículos sobre ASP
http://www.lavariable.com/articulos.asp?tema=ASP

Manual de HTML y ASP
http://sestud.uv.es/manual.esp/indice.htm

Artículos de SoloASP
http://www.soloasp.com.ar/articulo.asp

ASPTutor.com. Tutorial de SQL
http://www.asptutor.com/sql/

Cybercursos.net Curso de SQL
http://www.cybercursos.net/sql/sql.html

Nota: Es posible que al momento de consultar estas direcciones algunas ya no se encuentren en el servidor de origen, por aspectos propios del sitio. Al momento de elaborar el presente trabajo, los sitios funcionaban en su totalidad.

APENDICE A: EL INTERNET INFORMATION SERVER (IIS)

Ya vimos cómo es que el PWS opera bajo Windows 9x. Sin embargo, si bien el desarrollo del sistema fue bajo Windows 98, el sistema quedó finalmente instalado en una PC con Windows NT 4, lo que implica el uso del IIS.

El presente apartado tiene como finalidad mostrar a grandes rasgos los aspectos relacionados a la instalación y configuración del IIS en Windows NT, ya que si bien como ya se mencionó (en el capítulo 5) el funcionamiento de las páginas ASP bajo el PWS y el IIS es el mismo, un aspecto importante bajo NT es la seguridad misma del sistema operativo, independientemente de la programación que realizamos para controlar el acceso a las páginas. Para mayor detalle puede consultar el libro "Creación de Aplicaciones Web en Windows NT", de Jesús Bobadilla[1].

Actualmente se encuentra ya la versión 5 del IIS, sin embargo nos centraremos en las versiones anteriores, 3 y 4, sobre las que hay más información y que sin duda forman parte de la versión 5. Los servicios que proporciona la versión 3 son Gopher, FTP y WWW (servidor de Web). La versión 4 ofrece opciones adicionales como el servicio de correo electrónico y de noticias. Es necesario que el protocolo TCP/IP se encuentre configurado para trabajar con el IIS. Por lo regular esto se realiza al momento de instalar el sistema operativo, así que nos enfocaremos en la instalación y configuración del IIS 3 y 4.

Internet Information Server 3

Consideraremos que se cuenta ya con el paquete de instalación que se puede obtener en el sitio web de Microsoft (capítulo 5). De entre las opciones que podemos seleccionar se encuentran:

- Administrador del servicio de Internet: Instala el programa de administración local de los servicios FTP, Gopher y WWW.

[1] Creación de Aplicaciones Web en Windows NT. Bobadilla, Jesús. España, Ra-Ma, 1999

- Servicios World Wide Web: Instala el servidor Web.
- Ejemplos del servicio WWW: Instala páginas de ejemplo con las que se puede probar el servicio de web.
- Administración de Servicios de Internet (HTML): Versión HTML del Administrador del servicio de Internet al cual se puede acceder de forma local o remota, teniendo los permisos necesarios.
- Servicio Gopher: Instala el servidor Gopher.
- Servicio FTP: Instala el servidor FTP.
- Administración y controladores ODBC: Instala o actualiza los controladores ODBC (Open Data Base Conectivity) vistos en el capítulo 4.

Aparece una ventana que nos permite elegir los directorios donde instalaremos los diferentes servicios. Es recomendable dejar los que se marcan por omisión. Después se instalan los controladores necesarios en C:\WINNT\System32\Inetsrv, así como los componentes ODBC seleccionados. Se reinicia la PC para que se actualice el sistema. También es necesario instalar los componentes de ASP, mismos que se encuentran en el paquete de instalación y que por lo general se instalan con el resto de los servicios. Si no fuera así, quizá necesite una versión más actualizada del programa, para lo cual puede referirse a las ligas revisadas al final del presente trabajo, luego de la bibliografía.

Ahora nos dirigimos al botón de Inicio -> Programas -> Microsoft Internet Server (Común) -> Administrador de Servicios de Internet. Nos situamos sobre el servicio de WWW y con el botón derecho del ratón elegimos la opción Propiedades.

La pestaña de Servicio contiene lo siguiente:

- Puerto TCP: Puerto asignado para el funcionamiento del servicio. El valor predeterminado para el servicio WWW es de 80.
- Tiempo de espera de la conexión: Indica el tiempo de espera que el servidor Web espera antes de desconectar a un usuario luego de que no ha detectado actividad del mismo, como ya se vio en el capítulo 5.

- Máximo número de conexiones: Limita el número máximo de conexiones simultáneas con el servidor.
- Inicio de sesión anónimo: Permite especificar la información de los usuarios que no son miembros del dominio del servidor, es decir, permite que se conecten a nuestro servidor usuarios que no necesariamente tengan una cuenta en la PC que funcione como el servidor WWW.
- Nombre del usuario: El usuario predeterminado es IUSR_nombre_pc, que se crea automáticamente en NT al instalar el IIS, con una palabra clave aleatoria. Si se cambia la contraseña, hay que asegurarse que sea la misma en NT yen el IIS. Por defecto, el usuario anónimo (anonymous) tiene derechos de usuario de tipo "Iniciar sesión local". Si se desean crear otros usuarios para que se conectan a nuestro sitio, se les deben de dar los mismos derechos que al usuario anónimo.
- Contraseña: Palabra clave del usuario anónimo.
- Autentificación de la contraseña: Por lo general se deja la opción de Permitir anónimos, de lo contrario se establece el tipo de autentificación. La autentificación básica envía la contraseña sin codificar, como texto. La autentificación Desafío/Respuesta de Windows NT cifra automáticamente los nombres de usuario y las contraseñas.

Dentro de la pestaña Directorio, una parte importante es la que se refiere a "Se permite examinar directorios". Si se habilita, los visitantes pueden ver y modificar la estructura y contenido del sitio Web, lo que no es conveniente. Otro punto es el referente al Acceso, que se refieren a los atributos del directorio, que deben coincidir con los asignados por Windows NT. Permisos de lectura es para páginas con HTML puro, mientras que los permisos de Ejecución son para nuestros documentos ASP.

La pestaña Avanzada permite una forma de restringir el acceso al sitio Web, filtrando las direcciones IP de los equipos que pueden conectarse al servidor. Sin embargo, esto no es conveniente si nuestros usuarios no utilicen siempre el mismo equipo para utilizar el sistema que reside en nuestro servidor. Por defecto el IIS permite el acceso a cualquier máquina.

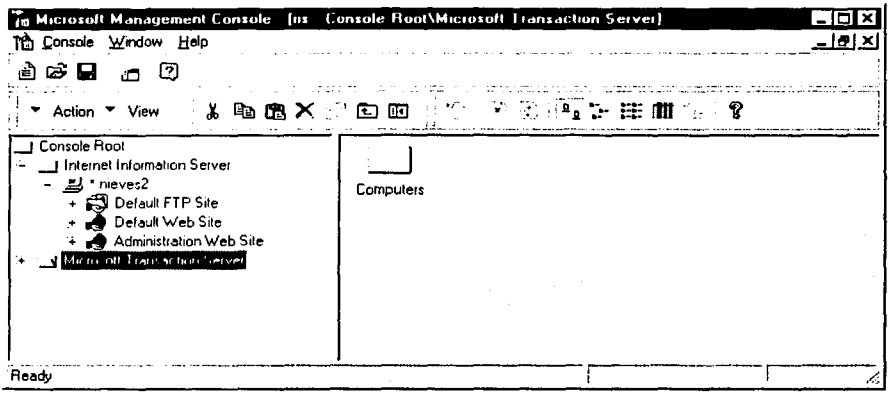

Figura A Microsoft Internet Service Manager

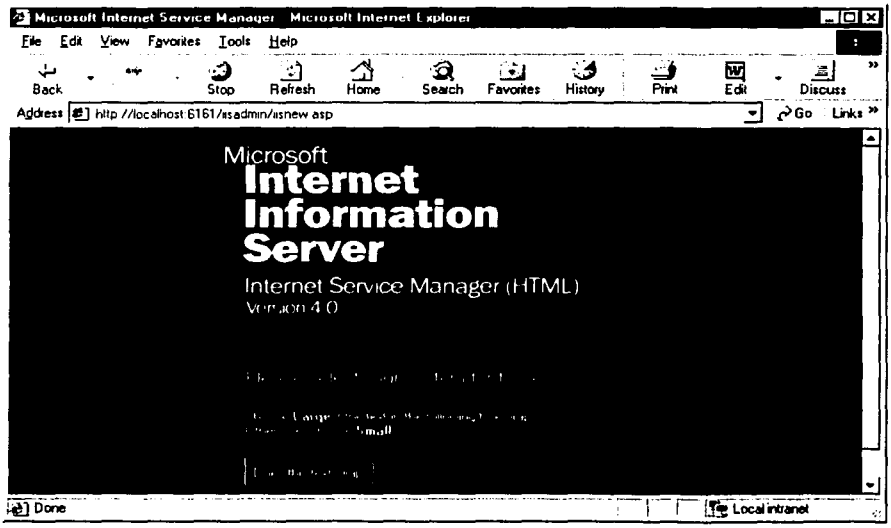

Figura B Microsoft Internet Service Manager en HTML

Aspectos de Seguridad

Es necesario realizar un paso más para garantizar la seguridad de nuestro sitio Web. La programación que realizamos en el capítulo 5 permite que sólo los usuarios autorizados utilicen el sistema de becarios; sin embargo, Windows NT tiene sus propios métodos de seguridad, los cuales hay que pasar antes de poder entrar siquiera a la página de acceso al sistema. Esto representa una ventaja frente al PWS y Windows 98, pero también puede ser un dolor de cabeza si no se tienen los conocimientos necesarios sobre NT.

Windows NT puede restringir la utilización de los recursos del sistema, tanto a nivel local como de forma remota. NT valida al usuario y su contraseña antes de que pueda ingresar al servicio, en este caso el servidor de Web.

Como ya se mencionó, IIS crea un usuario anónimo al instalarse, el cuál se administra en las propiedades del servicio WWW. También se puede acceder a él por las herramientas administrativas de NT. La clave inicial del usuario anónimo se genera de forma aleatoria y se actualiza en IIS y en NT, por lo que al hacer cambios en alguno es necesario hacerlos también en el otro. Este usuario debe tener permiso de Iniciar sesión en modo local, y que esté activada la casilla de Permitir anónimos de las propiedades del IIS. Hay que revisar y restringir los derechos de los grupos Todos e Invitados, a los que inicialmente pertenece el usuario anónimo.

Internet Information Server 4

Ofrece los mismos servicios de la versión 3, además de un servidor de correo y otro de noticias. También dispone de diversas tecnologías para el acceso a bases de datos por medio de MDAC (Microsoft Data Access Components), tales como ODBD (Open Data Base Conectivity), ADO (Microsoft ActiveX Data Objects), y MRDS (Microsoft Remote Data Service). Así mismo, dispone de una máquina virtual de Java en el servidor.

Requerimientos para la Instalación

En cuanto a software, es necesario el Windows NT 4, ya sea Server o Workstation. En este último también se puede instalar el PWS, pero se recomienda el IIS por cuestiones de seguridad colectiva con NT. También se requiere la versión 4.01 como mínimo del Internet Explorer para instalar y utilizar el IIS 4.

El IIS 4 se puede instalar de tres formas: mínima, típica y personalizada.

La instalación mínima incluye

- IIS 4
- Servicio FTP
- Servicio SMTP (Correo electrónico)
- Consola de Administración de Componentes MMC (Microsoft Management Console)
- Servicios de Administración por Internet
- Microsoft Transaction Server (MTS)
- Microsoft Data Access Components (MDAC)

La instalación típica añade a la mínima

- ASP
- Documentación
- Extensiones de FrontPage
- Microsoft Script Debugger
- Windows Script Host
- Index Server

Y la instalación personalizada incluye además de todo lo anterior

- Certificate Server
- Servicio NNTP (Servicio de Noticias)
- Site Server Express
- Servicios RAS

Si bien no es necesario instalar todos los componentes disponibles, existen algunos que son necesarios para la funcionalidad de otros.

En cuanto a hardware, los requerimientos son los siguientes:

Dispositivo	Mínimo requerido	Mínimo aconsejable	Mínimo recomendado
Procesador	486 a 50 Mhz	Pentium a 200 Mhz	Pentium a 90 Mhz
Memoria	16 MB	64-128 MB	32-64 MB
Disco	50 MB	200 MB	200 MB
Monitor	VGA	SVGA	SVGA