



# UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

---

## FACULTAD DE INGENIERÍA

### “SISTEMA DE SEGURIDAD DE ARCHIVOS DE INFORMACIÓN EMPRESARIAL PARA EL ÁREA DE FINANZAS DEL GRUPO ICA (SISEA)”

T E S I S

QUE PARA OBTENER EL TÍTULO  
DE INGENIERO EN COMPUTACIÓN  
PRESENTAN:

EDMUNDO CARMONA CASTILLO  
JAIME RAMÍREZ MORENO  
ALEJANDRA RETEGUÍN PIMENTEL

DIRECTOR DE TESIS: M. EN I. LAURO SANTIAGO CRUZ



CD. UNIVERSITARIA

MÉXICO D.F., ABRIL 2003



#### *AGRADECIMIENTOS COMUNES:*

*Agradecemos el apoyo del PAT (Programa de Apoyo a la Titulación) en la realización de este trabajo y reconocemos el gran esfuerzo que hacen para el fortalecimiento académico de los estudiantes de la Facultad de Ingeniería.*

*Así mismo, agradecemos su participación en este trabajo de tesis a nuestros compañeros del PAT: Raúl Guillermo Jonapá Megchún y a Rey Felipe Garrido Jiménez.*

#### *AGRADECIMIENTOS ALEJANDRA*

*A mis padres:*

*Gracias papá y mamá por el apoyo incondicional que me han brindado durante toda mi vida. Gracias por estar conmigo siempre. Lo que soy hoy es gracias a su esfuerzo y ejemplo. Los quiero mucho.*

*A mi hermano:*

*Gracias Jorge por apoyarme siempre, por enseñarme tantas cosas y por todos tus cuidados. Te quiero mucho.*

## *AGRADECIMIENTOS EDMUNDO*

*A mis padres:*

*Por su apoyo incondicional y esfuerzo diario que se refleja en esta etapa más de mi vida. Este trabajo es de ustedes y hecho por ustedes. Gracias.*

*A mis hermanos:*

*José, Rosa, Beatriz, Martha, Rodolfo, Daniel, Francisco, Alba Luz y Lilián, por su apoyo y ejemplo, gracias a todos y recuerden que “querer es poder”.*

*A Edith:*

*Sin su ayuda y ánimos, este trabajo no estaría completado. Gracias.*

*A todos aquellos que aportaron su grano de arena para que este trabajo fuera realizado. Gracias.*

## *AGRADECIMIENTOS JAIME*

*Este trabajo nunca habría sido posible sin la colaboración incansable de muchas personas. En el desarrollo de la tesis de fueron decisivos mis compañeros y el tutor, al seguir el proyecto desde sus comienzos y sus crecientes esfuerzos.*

*Me gustaría agradecer a mis padres (los dos) por todo aquello que ellos hacen, a él PAT por la oportunidad de brindarnos este ciclo final, y por último a la facultad de ingeniería por darnos la oportunidad de crecer en esta profesión.*

# ÍNDICE GENERAL

<b>Introducción</b> .....	1
<b>Capítulo I. Generalidades</b>	
1.1. Sistemas operativos .....	3
1.1.1. Sistemas operativos por su estructura .....	4
1.1.2. Sistemas operativos por servicios.....	4
1.1.3. Sistemas operativos por la forma de ofrecer sus servicios .....	5
1.2. Bases de datos .....	5
1.2.1. Definición de bases de datos .....	6
1.2.2. Diseño de una base de datos en un sistema de información.....	7
1.3. Conceptos fundamentales de la programación orientada a objetos .....	7
1.3.1. Metodología de diseño orientado a objetos .....	10
1.4. Infraestructura de comunicaciones .....	16
1.4.1. Tipos de Redes.....	16
1.4.2. Topologías de Red.....	19
1.4.3. El Modelo TCP/IP .....	23
1.4.4. Tipos de servicio .....	25
<b>Capítulo II. Planteamiento del problema y propuesta de solución</b>	
2.1. Aspectos importantes sobre la empresa ICA.....	29
2.1.1. Estructura Interna .....	31
2.1.2. Área de Finanzas.....	32
2.1.3. Operatividad .....	33
2.2. Identificación del problema .....	33
2.2.1. Requerimientos del usuario .....	35
2.3. Opciones de solución.....	35
2.3.1. Manejadores de Bases de Datos.....	36
2.3.2. Sistemas Operativos.....	37
2.3.3. Herramientas para la creación de Páginas Web.....	41
<b>Capítulo III. Diseño del sistema</b>	
3.1. Aplicación de la metodología .....	44

---

3.1.1. Diagrama de casos de uso .....	45
3.1.2. Diagrama de Clases .....	52
3.1.3. Diagrama de Objetos y de eventos.....	57
3.1.4. Diagrama de flujo de datos (DFD) .....	60
3.1.5. Diagrama de Secuencia.....	66
3.2. Aplicación de las Herramientas .....	70
3.2.1. Diccionario de Datos.....	70
3.2.2. Diagrama Entidad-Relación .....	76
<b>Capítulo IV. Desarrollo del sistema</b>	
4.1. Construcción del Sistema .....	82
4.1.1. Creación de la base de datos .....	82
4.1.2. Creación de las tablas en la base de datos .....	84
4.1.3. Conexión ActiveX Data Objects (ADO).....	95
4.2. Construcción de Pantallas .....	97
<b>Capítulo V. Pruebas y reportes del sistema</b>	
5.1. Pruebas del Sistema.....	101
5.1.1. Categoría de pruebas .....	102
5.1.2. Consideraciones importantes para la ejecución de las pruebas .....	103
5.1.3. Infraestructura de pruebas.....	104
5.1.4. Pruebas aplicadas .....	105
5.2. Generación de reportes del sistema .....	113
5.2.1. Reporte de registro de usuarios.....	113
5.2.2. Reporte de archivos dados de alta .....	119
5.2.3. Reporte de accesos por día.....	120
5.2.4. Reporte de accesos por mes y año .....	121
5.2.5. Reporte de consultas por día.....	122
5.2.6. Reporte de consultas por mes y año .....	123
<b>Capítulo VI. Instalación y liberación del sistema</b>	
6.1. Instalación del sistema.....	124
6.1.1. Normatividad para redes bajo Windows NT 2000.....	125
6.1.2. Puesta a punto de los servidores.....	127

6.1.3. Instalación de SQL Server 2000 y de SISEA.....	128
6.1.4. Administración del Sistema.....	135
6.1.5. Control de cambios .....	136
6.1.6. Documentación para el usuario .....	136
6.1.7. Capacitación .....	148
6.1.8. Tipos de Mantenimiento.....	148
<b>Capítulo VII. Resultados y conclusiones</b>	
Resultados y conclusiones .....	150
<b>Bibliografía</b> .....	154
<b>Apéndice</b> .....	A1

# ÍNDICE DE TABLAS

## Capítulo I. Generalidades

Tabla 1.1. Categorías UTP y sus características.....	18
Tabla 1.2. Redes Ethernet 10Base-X y sus características.....	21
Tabla 1.3. Redes Ethernet 100Base-X y sus características.....	21
Tabla 1.4. Resumen de Protocolos .....	23

## Capítulo II. Planteamiento del problema y propuesta de solución

Tabla 2.1. Tabla de comparación entre los diferentes manejadores de bases de datos.....	38
Tabla 2.2. Sistemas operativos en los diferentes manejadores de bases de datos ..	41

## Capítulo III. Diseño del sistema

Tabla 3.1. Objetos involucrados en el sistema (Continúa) .....	57
Tabla 3.1. Objetos involucrados en el sistema (Continúa) .....	58
Tabla 3.1. Objetos involucrados en el sistema .....	59
Tabla 3.2. Objetos de un diagrama de flujo de datos .....	61

## Capítulo IV. Desarrollo del sistema

Tabla 4.1. Cadena de conexión.....	96
Tabla 4.2. Cadena de conexión ODBC .....	96



---

# ÍNDICE DE FIGURAS

## Capítulo I. Generalidades

Figura 1.1. Método OOD por Grady Booch .....	12
Figura 1.2. Comparación entre los modelos OSI y TCP/IP .....	25

## Capítulo II. Planteamiento del problema y propuesta de solución

Figura 2.1. Organigrama Grupo ICA.....	31
Figura 2.2. Relación entre Finanzas y el resto de la empresa.....	33
Figura 2.3. Solicitud de consulta de información .....	34

## Capítulo III. Diseño del sistema

Figura 3.1. Diagrama de Casos de uso del actor Usuario .....	46
Figura 3.2. Diagrama de Casos de uso del actor Administrador .....	49
Figura 3.3. Diagrama de casos de uso del actor Superusuario .....	51
Figura 3.4. Representación de las clases.....	52
Figura 3.5. Clase Usuario .....	53
Figura 3.6. Clase Permisos .....	53
Figura 3.7. Clase Archivos.....	54
Figura 3.8. Clase Ticker .....	54
Figura 3.9. Clase tipo de usuario.....	54
Figura 3.10. Clase Departamento.....	55
Figura 3.11. Clase Proyectos .....	55
Figura 3.12. Clase Reportes.....	55
Figura 3.13. Clase Estadísticas .....	56
Figura 3.14. Clase Región .....	56
Figura 3.15. Diagrama de clases del sistema.....	57
Figura 3.16. Diagrama de objetos y eventos (Continúa) .....	59
Figura 3.16. Diagrama de objetos y eventos .....	60
Figura 3.17. Diagrama de flujo de datos, nivel 1 .....	61

---

Figura 3.18. Diagrama de flujo de datos, nivel 2 .....	62
Figura 3.19. DFD nivel 3, Proceso Alta Archivos .....	63
Figura 3.20. DFD nivel 3, Proceso Baja Archivos .....	64
Figura 3.21. DFD nivel 3, Proceso Alta Usuarios .....	65
Figura 3.22. DFD nivel 3, Proceso Acceso Usuarios .....	66
Figura 3.23. Diagrama de Secuencia (Continúa).....	67
Figura 3.23. Diagrama de Secuencia .....	68
Figura 3.24. Modelado de Entidades .....	77
Figura 3.25. Diagrama Entidad-Relación del Sistema .....	78

#### **Capítulo IV. Desarrollo del sistema**

Figura 4.1. Creación de la base de datos SISEA .....	83
Figura 4.2. Base de datos SISEA .....	83
Figura 4.3. Tablas de la base de datos SISEA .....	84
Figura 4.4. Creación de la tabla Documentos.....	85
Figura 4.5. Conexión de ERWin con SQL Server 2000 .....	86
Figura 4.6. Generación del script para la creación de las tablas .....	86
Figura 4.7. Definición de los Frames en la pantalla.....	98
Figura 4.8. Presentación del formato general de las pantallas .....	99
Figura 4.9. Formato de presentación de la pantalla principal .....	99

#### **Capítulo V. Pruebas y reportes del sistema**

Figura 5.1. Prueba de validación en la pantalla de administración.....	106
Figura 5.2. Pantalla que muestra el error al no insertar datos .....	106
Figura 5.3. Prueba de Integración .....	107
Figura 5.4. Pantalla de Navegación del Sitio .....	108
Figura 5.5. Lista de Documentos y Ligas .....	108
Figura 5.6. Pantalla de Consulta de Documentos .....	109
Figura 5.7. Prueba de Caja Negra.....	109
Figura 5.8. Alta de Usuarios .....	110
Figura 5.9. Pantalla de Alta de Documentos .....	111
Figura 5.10. Pantalla de Ligas a Sitios de Internet .....	111

---

Figura 5.11. Pantalla Consulta de documentos .....	112
Figura 5.12. Pantalla de Alta de Información del Ticker .....	112
Figura 5.13. Reporte de registro de usuarios de Tesorería .....	113
Figura 5.14. Reporte de registro de usuarios de Planeación Estratégica .....	114
Figura 5.15. Reporte de registro de usuarios de Relación con Inversionistas .....	115
Figura 5.16. Reporte de registro de usuarios de Planeación Financiera .....	116
Figura 5.17. Reporte de registro de usuarios de Desinversión .....	117
Figura 5.18. Reporte Inversión y Financiamiento .....	118
Figura 5.19. Reporte de archivos dados de alta .....	119
Figura 5.20. Total de accesos por día .....	120
Figura 5.21. Total de accesos por año .....	121
Figura 5.22. Gráfica Total de accesos por mes .....	121
Figura 5.23. Total de consultas por día .....	122
Figura 5.24. Total de consultas por año .....	123
Figura 5.25. Gráfica Total de consultas por mes .....	123

## **Capítulo VI. Instalación y liberación del sistema**

Figura 6.1. Parte principal de la instalación .....	128
Figura 6.2. Servidor de base de datos .....	129
Figura 6.3. Instancia de instalación .....	129
Figura 6.4. Asignación del Servidor .....	130
Figura 6.5. Instalación de Herramientas .....	130
Figura 6.6. Datos requeridos para la instalación .....	131
Figura 6.7. Selección de la opción de herramientas .....	131
Figura 6.8. Selección tipo de información .....	132
Figura 6.9. Culminación de la instalación .....	132
Figura 6.10. Asistente de instalación .....	133
Figura 6.11. Nombre sitio virtual .....	134
Figura 6.12. Asignación del directorio físico .....	134
Figura 6.13. Listo el directorio virtual .....	135
Figura 6.14. Acceso al Usuario .....	137
Figura 6.15. Consulta de información .....	138

---

Figura 6.16. Administración de usuarios .....	139
Figura 6.17. Pantalla de Alta de Usuario .....	140
Figura 6.18. Pantalla de Modificación de datos del Usuario .....	140
Figura 6.19. Pantalla de confirmación de eliminación de Usuario .....	141
Figura 6.20. Pantalla de Lista de Documentos .....	142
Figura 6.21. Pantalla de Alta de Archivo .....	143
Figura 6.22. Pantalla de Agregar Folder.....	144
Figura 6.23. Pantalla de Agregar Liga .....	145
Figura 6.24. Pantalla de Confirmación de Eliminación de Documentos .....	145
Figura 6.25. Pantalla de Actualización de Documentos .....	146
Figura 6.26. Pantalla de Cambio de Folder .....	147
Figura 6.27. Pantalla de Cambio de Link.....	147

# INTRODUCCIÓN

Actualmente las corporaciones que se encuentran en una constante expansión, tienden a tener problemas en el control de la información y comunicación. La Empresa ICA, y en particular el Área de Finanzas, notoriamente ha empezado a presentar tales características, que a falta de una solución acertada, se ha visto envuelta en procesos que limitan el desempeño de las actividades laborales y que repercuten entre otros aspectos en la imagen de la compañía.

Ante esta problemática, el trabajo de tesis presentado aquí, tiene como objetivo primordial proponer el desarrollo de un sistema que trabaje en ambiente *Web*, que cumpla con las características operativas, que permitan lograr la automatización de procesos dentro del sistema corporativo, de tal forma, que se pueda mejorar la comunicación entre los distintos departamentos, y con ello obtener, mayores beneficios de productividad y colaboración entre los empleados.

La propuesta de desarrollo de un sistema dentro del Área de Finanzas de Grupo ICA, parte de la premisa de que un sistema que trabaja en ambiente *Web* por sus características, juega un papel estratégico creciente en el proceso de gestión empresarial moderno. Contexto dentro del cual, la propuesta de solución brinda la integración de la empresa y sus departamentos a un mejor nivel de colaboración con la ayuda de las nuevas tecnologías *Web*, de manera que puedan obtenerse mayores beneficios en la ejecución de las actividades diarias de cada usuario.

Cabe aclarar que este trabajo de tesis no está enfocado al desarrollo y aplicación de esquemas de seguridad de *software*, sino al desarrollo de un sistema que dé orden y agilice la consulta a los archivos de cada uno de los usuarios. Por otro lado, el desarrollo del sistema no se presenta en su totalidad ya que al cubrir seis departamentos es muy repetitivo en muchas acciones, así que se hace el desarrollo para un departamento y los otros cinco se ven de forma análoga.

El presente trabajo está integrado de siete capítulos, que son: I. Generalidades, II. Planteamiento del problema y propuesta de solución, III. Diseño del sistema, IV. Desarrollo del sistema, V. Pruebas y reportes del sistema, VI. Instalación y Liberación del sistema y VII. Resultados y conclusiones. Además se incluyen la bibliografía consultada y un apéndice generado.

En el capítulo I se presentan conceptos generales relacionados al desarrollo de un sistema de información, como son: conceptos fundamentales de programación, metodologías de diseño, sistemas operativos y comunicaciones.

En el capítulo II se describe de forma general a la empresa ICA y su forma de operación, identificando las principales características de sus problemas en el Área de Finanzas. Esto se logra haciendo un análisis del problema para proponer posibles soluciones.

En el capítulo III se presentará el diseño del sistema modelando los diagramas: caso de uso, dinámico, objeto y de eventos, además presentamos el diccionario de datos, el diagrama entidad relación y la normalización.

El capítulo IV comprende el desarrollo del sistema, con la creación de la base de datos, la creación de las tablas, así como la construcción de las pantallas de interacción del sistema con el usuario.

El capítulo V comprende las pruebas realizadas y reportes que genera el sistema.

En el capítulo VI se presentará la implantación final del sistema, en donde se describe la forma en que se realizaron: la puesta a punto del servidor y la instalación del sistema; se describen también algunos puntos importantes para la operación eficiente del sistema, como son: la administración, el control de cambios, el mantenimiento y el soporte técnico.

En el capítulo VII se verán los resultados obtenidos del desarrollo del sistema, así como las conclusiones correspondientes.

Por último se presenta la bibliografía consultada y un apéndice, en este último se incluyen los listados parciales de los programas desarrollados.

# CAPÍTULO I

## GENERALIDADES

En este capítulo presentaremos conceptos generales relacionados al desarrollo de un sistema de información, como son: conceptos fundamentales de programación, metodologías de diseño, sistemas operativos y comunicaciones, así como características de ASP, SQL y Windows NT 2000.

### 1.1. Sistemas operativos

A mediados de los años 80's comienza el auge de las redes de computadoras y la necesidad de sistemas operativos en red y sistemas operativos distribuidos. La red mundial Internet se va haciendo accesible a toda clase de instituciones y se comienzan a dar muchas soluciones y problemas al querer hacer convivir recursos residentes en computadoras con sistemas operativos diferentes. Para los 90's la programación orientada a objetos cobra auge, así como el manejo de objetos desde los sistemas operativos. Las aplicaciones intentan crearse para ser ejecutadas en una plataforma específica y poder ver sus resultados en la pantalla o monitor de otra diferente. Por ejemplo, ejecutar una simulación en una máquina con *UNIX* y ver los resultados en otra con *DOS* (*Disk Operative System, Sistema Operativo de Disco*). Los niveles de interacción se van haciendo cada vez más profundos.

Existen diversas clasificaciones de los sistemas operativos, sin embargo, sólo mencionaremos algunas, las cuales estarán involucradas en el desarrollo de este trabajo, como son: por su estructura, por servicios y por la forma de ofrecer sus servicios.

### **1.1.1. Sistemas operativos por su estructura**

Los sistemas operativos, por su estructura, pueden clasificarse en Jerárquica y de Cliente-Servidor, éstos se presentan a continuación:

Estructura jerárquica. A medida que fueron creciendo las necesidades de los usuarios y se perfeccionaron los sistemas, se dividió el sistema operativo en pequeñas partes, de tal forma que cada una de ellas estuviera perfectamente definida y con una clara interfaz con el resto de los elementos.

Se puede pensar también en estos sistemas como si fueran multicapa. En esta estructura se basan prácticamente la mayoría de los sistemas operativos actuales.

Otra forma de ver este tipo de sistema es la denominada de anillos concéntricos o anillos, donde cada anillo tiene una apertura, conocida como puerta o trampa, por donde pueden entrar las llamadas de las capas inferiores. De esta forma, las zonas más internas del sistema operativo o núcleo del sistema estarán más protegidas de accesos indeseados desde las capas más externas. Las capas más internas serán, por tanto, más privilegiadas que las externas.

Estructura Cliente-Servidor. El tipo más reciente de sistemas operativos es el denominado Cliente-Servidor, que puede ser ejecutado en la mayoría de las computadoras, ya sean grandes o pequeñas. Este sistema sirve para toda clase de aplicaciones, por tanto, es de propósito general y cumple con las mismas actividades que los sistemas operativos convencionales.

El núcleo tiene como misión establecer la comunicación entre los clientes y los servidores. Los procesos pueden ser tanto servidores como clientes. El núcleo provee solamente funciones muy básicas de memoria, entrada/salida, archivos y procesos, dejando a los servidores proveer la mayoría de los requerimientos que el usuario final o programador puede usar. Estos servidores deben tener mecanismos de seguridad y protección que, a su vez, serán filtrados por el núcleo que controla el hardware.

### **1.1.2. Sistemas operativos por servicios**

Es la más comúnmente usada y conocida desde el punto de vista del usuario final, se clasifica por el número de usuarios (monousuario o multiusuario), por el número de tareas (monotarea o multitarea) y por el número de procesos (uniproceto o multiproceto). A continuación se muestran estos casos:

Multiusuario. Los sistemas operativos multiusuario son capaces de dar servicio a más de un usuario a la vez, ya sea por medio de varias terminales conectadas a la computadora o por medio de sesiones remotas en una red de comunicaciones. No importa el número de procesadores en la máquina ni el número de procesos que cada usuario puede ejecutar simultáneamente.



Multitarea. Un sistema operativo multitarea es aquél que le permite al usuario estar realizando varias labores al mismo tiempo. Por ejemplo, puede estar editando el código fuente de un programa durante su depuración mientras compila otro programa, a la vez que está recibiendo correo electrónico en un proceso en *background*. Es común encontrar en ellos interfaces gráficas orientadas al uso de menús y el ratón, lo cual permite un rápido intercambio entre las tareas para el usuario, mejorando su productividad.

Multiproceso. Un sistema operativo multiproceso se refiere al número de procesadores del sistema, que es más de uno y éste es capaz de usarlos todos para distribuir su carga de trabajo.

### **1.1.3. Sistemas operativos por la forma de ofrecer sus servicios**

Esta clasificación también se refiere a una visión externa, como es la del usuario al acceder a los servicios. Con base en ello se presentan los sistemas operativos de red y los distribuidos.

Sistemas operativos de red. Se definen como aquellos que tienen la capacidad de interactuar con sistemas operativos en otras computadoras a través de un medio de transmisión con el objeto de intercambiar información, transferir archivos, ejecutar comandos remotos y muchas otras actividades.

Sistemas operativos distribuidos. Éstos abarcan los servicios de los sistemas operativos de red, logrando integrar recursos (impresoras, unidades de respaldo, memoria, procesos, unidades centrales de proceso) en una sola máquina virtual que el usuario accede en forma transparente. Es decir, ahora el usuario ya no necesita saber la ubicación de los recursos, sino que los conoce por nombre y simplemente los usa como si todos ellos fuesen locales a su lugar de trabajo habitual.

Para planear el desarrollo de un sistema de información debemos considerar, entre otras cosas, el sistema operativo en el que funciona la aplicación, la base de datos que soportará el contenido y el lenguaje de programación. Es por ello que una vez que hemos revisado las características generales de los sistemas operativos, procederemos con los de las bases de datos.

## **1.2. Bases de datos**

Las bases de datos son componentes esenciales de los sistemas de información, usadas en forma rutinaria desde las computadoras menores hasta las supercomputadoras. El diseño de las bases de datos es utilizado para encontrar mejores mecanismos para resguardar la información.

Cerca del año 1965, las bases de datos penetraban por primera vez como una herramienta de respaldo de datos y, antes de que aparecieran las computadoras de tercera generación, el software ejecutaba las operaciones de entrada/salida de los

dispositivos de almacenamiento. La codificación incluida en los programas de aplicación se encargaba de la organización de los datos, y esto de manera muy elemental, por lo general sólo servía a modo de simples archivos secuenciales en cinta. La mayoría de los archivos servían sólo para una aplicación y esto involucraba mucho trabajo, ya que se tenía que volver a generar el archivo y no existía el mecanismo de actualizarlos, por tanto era tedioso trabajar con este tipo de bases de datos.

La segunda etapa reconoció la naturaleza cambiante de los archivos y de los dispositivos de almacenamiento. El efecto de los cambios que se introducían en el hardware y el software hizo posible modificar la distribución física de los datos sin que por ello se alterase su estructura lógica, siempre que no se introdujesen cambios en los contenidos de los registros ni en la estructura fundamental de los archivos.

Para la década de los 70's surge el modelo relacional, es decir, la definición de lo que conocemos como matemática del álgebra relacional. Después de la mitad de la década se establece ORACLE y realiza la primera implementación real del modelo relacional, el primer RDBMS (*Relational Database Management System*, Sistema de Gestión de Bases de Datos Relacionales) comercial de ORACLE, siendo la versión 2.0.

Ya en la década de los 80's y después de que se han realizado cambios importantes se funda INFORMIX, dando paso a la tercera generación de DBMS, siendo tan importante que se utiliza en la actualidad. Poco tiempo después se establece SQL, que es una de las herramientas principales de trabajo que se utilizan actualmente, destacando en paralelo el uso del modelo relacional.

Antes de que termine esta década se establece Dbase III y IV, se establecen las estructuras de consultas con un lenguaje de programación y con ello surgen las herramientas de SQL de ORACLE, se comenzó a utilizar como herramienta de base de datos a SQL Server por parte de las compañías Microsoft y Sybase.

Hoy en día se utilizan el diseño y estructura de base de datos con multimedia, vía Internet o Intranet, se envuelve a gran número de redes por las cuales a diario se distribuye información por sistemas operativos UNIX y herramientas de bases de datos como ORACLE.

### **1.2.1. Definición de bases de datos**

Puede definirse como una colección de datos interrelacionados en conjunto sin redundancias perjudiciales o innecesarias; su funcionalidad es la de servir a una aplicación de la mejor manera posible. Los datos se almacenan de modo que resulten independientes de los programas o sistemas desarrollados y sólo se utilizan las aplicaciones para insertar o extraer datos de ellas.

Los objetivos principales de una base de datos son: minimizar la redundancia, evitar la inconsistencia de datos, compartir datos, aplicar restricciones de seguridad, conservar la integridad de los datos, la abstracción de la información y la independencia de los datos.

### 1.2.2. Diseño de una base de datos en un sistema de información

Un sistema de información es el conjunto de actividades que regulan la distribución y el comportamiento de los datos relevantes para la administración de una empresa o sector gubernamental. Una base de datos es cualquier conjunto grande de datos que se encuentran estructurados y almacenados en una computadora. Los sistemas de gestión de bases de datos son los paquetes de software que sirven para almacenar, manipular y recuperar datos de una computadora.

El diseño de un sistema de información es una actividad que debe incluir la planificación, especificación y desarrollo de cada componente del sistema. Un desglose típico del diseño de un sistema de información es el siguiente:

Los sistemas para diseñar bases de datos operan sobre esquemas de datos y de funciones. Estos deben representarse y tratarse dentro del contexto de una metodología coherente para el diseño conceptual, lógico y físico de datos. Para el diseño de la base de datos se deben considerar las siguientes características:

- Interfaz amigable para el usuario.
- Amplia cobertura.
- Conjunto de herramientas robusto e integrado.
- Seguimiento de la metodología y diseño.
- Arquitectura abierta y extensible.

Después de haber descrito las partes generales de las bases de datos se abordan los conceptos de la programación orientada a objetos.

### 1.3. Conceptos fundamentales de la programación orientada a objetos

Actualmente una de las áreas más candentes de la programación de sistemas que operan en la industria y en el ámbito académico es la orientación a objetos. La orientación a objetos promete mejoras de amplio alcance en la forma de diseño, desarrollo y mantenimiento del software; ofreciendo una solución a largo plazo a los problemas y preocupaciones que han existido desde el comienzo en el desarrollo de software: la falta de portabilidad del código y reusabilidad, código que es difícil de modificar, ciclos de desarrollo largos y técnicas de codificación no intuitivas.

La programación orientada a objetos (OOP, *Object Oriented Programming*) tiene tres características básicas: debe estar basada en objetos, en clases y ser capaz de tener herencia de clases. Muchos lenguajes cumplen uno o dos de estos puntos, pero muy pocos cumplen los tres. La barrera más difícil de sortear es usualmente la herencia. El elemento fundamental de la OOP es, como su nombre lo indica, el objeto. Podemos definir un objeto como un conjunto complejo de datos y programas que poseen estructura y forman parte de una organización.

Esta definición especifica varias propiedades importantes de los objetos. En primer lugar, un objeto no es un dato simple, sino que contiene en su interior cierto número de componentes bien estructurados. En segundo lugar, cada objeto no es un ente aislado, sino que forma parte de una organización jerárquica o de otro tipo.

En la tecnología orientada a objetos la herramienta principal para soportar la abstracción es la clase. Podemos definir a una clase como una descripción genérica de un grupo de objetos que comparten características comunes, dichas características se especifican en sus atributos y comportamientos. En otras palabras, una clase es un molde o modelo en donde se especifican las características que definen a un objeto de manera general, a partir de una clase podemos definir objetos particulares. En programación orientada a objetos se dice que un objeto es una instancia de la clase, es decir, un objeto es un caso particular del conjunto de objetos que comparten características similares, definidas en la clase.

Los elementos más importantes que deben tener los objetos de software, para cumplir con el paradigma de orientación a objetos, son:

- Abstracción.
- Encapsulamiento.
- Herencia.
- Polimorfismo.

Siendo sus características más importantes las siguientes:

Abstracción: La abstracción es el proceso en el cual separamos las propiedades más importantes de un objeto de las que no lo son. Es decir, por medio de la abstracción definimos las características esenciales de un objeto del mundo real, los atributos y comportamientos que lo definen como tal, para después modelarlo en un objeto de software.

En el proceso de abstracción no debemos preocuparnos por la implementación de cada método o atributo, solamente debemos definirlo de forma general. Por ejemplo, supongamos que deseamos escribir un programa para representar el sistema solar, por medio de la abstracción podemos ver a este sistema como un conjunto de objetos, algunos de estos objetos son los planetas, que tienen un estado dado por sus características físicas, digamos tamaño o masa, entre otras. Estos objetos tendrán también comportamientos como la translación y la rotación que pueden mencionarse como los más evidentes. Visualizar las entidades que deseamos trasladar a nuestros programas, en términos abstractos, resulta de gran utilidad para un diseño óptimo de nuestro software, ya que nos permite comprender más fácilmente la programación requerida.

Encapsulamiento: También referido como ocultamiento de la información, es la propiedad de la orientación a objetos que nos permite asegurar que la información de un objeto le es desconocida a los demás objetos en la aplicación. Es muy frecuente

referirse a los objetos de software como cajas negras, esto se debe principalmente a que no necesitamos, dentro de la programación orientada a objetos, saber como está instrumentado un objeto para que éste interactúe con los demás objetos. Generalmente una clase se define en dos partes: una interfaz por medio de la cual los objetos que son solicitados y de la misma interactúan con los demás objetos en la aplicación, y la implementación de los miembros de dicha clase (métodos y atributos). Una aplicación orientada a objetos está constituida, como mencionamos anteriormente, por módulos. Estos módulos se implementan mediante clases, las cuales generalmente representan abstracciones de objetos del mundo real. Es por medio del encapsulamiento que podemos definir los atributos y los métodos de una clase para que los objetos que se solicitan de ésta trabajen como unidades independientes de los demás objetos con los que interactúan.

Herencia: La herencia es un concepto central del paradigma orientado a objetos, emergiendo de dos contextos básicos: abstracción y reutilización. Las clases hijas pueden especializar el comportamiento *heredado* de sus padres. La herencia separa la programación de los ADT (*Abstract Data Type*, Tipos de Datos Abstractos) de la OOP.

La herencia es el mecanismo que permite crear nuevas clases hijas (conocidas como subclases o clases derivadas) a partir de clases padre existentes. Por medio de la herencia, una subclase puede heredar los atributos y métodos de la superclase (la clase padre).

Las clases hijas heredan de sus padres los métodos y estructuras de datos. Pueden sumar nuevos métodos a las clases hijas, sustituir o modificar los métodos heredados para definir el comportamiento de las clases nuevas. Los métodos de las clases padre no se ven afectados por estas modificaciones. Se puede usar la herencia para extender el comportamiento de un objeto. Algunos modelos de objetos soportan herencia simple y son los que tienen una sola clase padre. Otros modelos soportan herencia múltiple y pueden tener más de una clase padre directa.

La herencia simple y múltiple es soportada por la tecnología orientada a objetos por medio de clases derivadas. Una clase puede también heredar las propiedades de más de una clase, heredando todas las propiedades contenidas en cada una de ellas, siendo conocido este concepto como herencia múltiple. Una clase derivada es declarada con un nombre seguida de los nombres de las clases base. Una clase derivada puede heredar cada una de las partes públicas de las clases base.

Para soportar la herencia múltiple la clase derivada puede tomar como una clase base a otra clase derivada permitiendo la construcción de jerarquías de clases. Una estructura de herencia es una manera de ofrecer reutilización y extensibilidad bajando el costo de mantenimiento para cumplir los objetivos de ingeniería de software.

Polimorfismo: Literalmente significa "muchas formas". Es una forma de decir que un mismo método puede hacer diferentes cosas, dependiendo de la clase en que esté implementada. Los objetos en diferentes clases reciben el mismo mensaje y reaccionan

en diferentes formas. El remitente no conoce la diferencia, el receptor interpreta el mensaje y proporciona el comportamiento apropiado.

El polimorfismo permite que se vean dos objetos similares por medio de una interfaz común, eliminando la necesidad de diferenciar entre los dos objetos. Además el principio fundamental que está detrás de las interfaces de usuario basadas en objetos es el mecanismo que permite a las subclases sustituir un método heredado y hacer un método propio sin afectar los métodos heredados originales.

Además de estudiar los conceptos fundamentales de la OOP, es importante conocer sobre la metodología de diseño orientada a objetos.

### **1.3.1. Metodología de diseño orientado a objetos**

Se empieza por definir qué es el análisis y diseño orientado a objetos, así como, qué es un método y qué es una metodología. Después se expondrán las características de cada uno de los métodos.

Definición de análisis y diseño orientado a objetos: La esencia del análisis y diseño orientado a objetos es enfatizar considerando el dominio de un problema y una solución lógica desde la perspectiva de objetos. Análisis orientado a objetos significa crear un modelo orientado a objetos de requerimientos externos (un modelo del problema). Diseño orientado a objetos significa pasar el modelo de análisis a un nivel más alto, a un modelo orientado a objetos de una solución.

Definición de método y metodología: Es importante saber la diferencia que hay entre método y metodología. Un método es un proceso disciplinado para generar un conjunto de modelos que describe varios aspectos de un sistema de software en desarrollo, usando alguna notación bien definida. Una metodología es una colección de métodos aplicados a través del ciclo de vida del desarrollo y unificado por algo en general.

Un diseño orientado a objetos no es un diseño *top down* ni *bottom up*, aunque envuelve elementos próximos a ambos. En el diseño *top down*, se trata de comprender el problema en su aspecto más general; una vez superado este paso, se trata de descomponer el problema general en subproblemas más pequeños y fáciles de resolver, procediendo con éstos de forma similar. El diseño opuesto a *top down* es el diseño *bottom up*, en el que se empieza resolviendo problemas básicos, para a continuación construir el problema en su totalidad.

Un diseño orientado a objetos, entre otras cosas, nos hace pensar sobre el aspecto fundamental del problema a resolver, y sobre la representación de diversos aspectos fundamentales, como los objetos (cosas, conceptos, o entidades), esto es la esencia del diseño *top down*. Pero el diseño orientado a objetos también sugiere que se mire hacia la meta, pensando en cómo pueden ser compuestos los objetos fundamentales para alcanzar dicha meta, esto es la esencia del diseño *bottom up*.

Donde el diseño orientado a objetos toma ventaja es en la compartición y reutilización del código. En cambio, las rutinas básicas de un diseño *top down*, que son creadas a propósito, probablemente no son reutilizables.

Los métodos para el diseño orientado a objetos comparten algunos pasos básicos del diseño y sólo varían un poco:

- Identificar los objetos del espacio de la solución y sus atributos así como el nombre de los métodos (servicios u operaciones).
- Establecer la visibilidad que tendrá el objeto en relación con los otros objetos.
- Determinar la interfaz de cada objeto y sus responsabilidades.
- La implementación y prueba de los objetos.

Durante la etapa del diseño es importante saber sobre la arquitectura del sistema:

- Las clases existentes y su relación.
- Los mecanismos que se utilizan para regular la forma en que los objetos colaboran.
- Donde es recomendable que cada clase y objeto se declare.

Existen varios métodos para el diseño orientado a objetos, como son:

- Método OOD (*Object-Oriented Design*, Diseño Orientado a Objetos) por Grady Booch
- Método OMT (*Object Modeling Technique*/Técnicas de Modelado Objetos) por Rumbaugh.
- Metodología Propia de la Empresa

Para elegir la metodología más adecuada para el desarrollo de un sistema de información es necesario conocer cómo se aplica cada una de ellas, por lo que las describiremos brevemente.

### **Método OOD por Grady Booch**

En este método el desarrollo es iterativo y se incrementa poco a poco, donde el proceso termina cuando ya no se tienen abstracciones primitivas o cuando las clases y los objetos pueden ser implementados a partir de componentes del software ya existentes.

El método OOD define modelos diferentes para describir un sistema. El modelo lógico (dominio del problema) es representado en la estructura clase y objeto. En el diagrama de clase se construye la arquitectura, el modelo estático. El diagrama de objeto se muestra como la interacción de clases, éstos capturan algunos momentos de la vida del

sistema y ayuda a describir el comportamiento dinámico. Una representación gráfica de este método se muestra en la Figura 1.1.

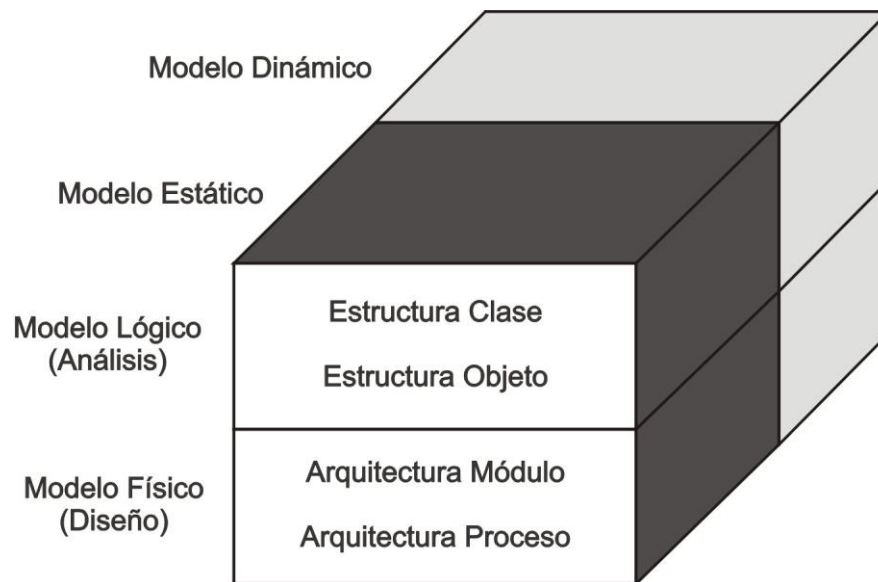


Figura 1.1. Método OOD por Grady Booch.

El proceso del método se divide en: Micro proceso y Macro proceso.

En el Micro proceso la explicación se dividirá en cuatro partes:

1. Identificación de las clases y objetos.
2. Identificación de la semántica de las clases y objetos.
3. Identificación de las relaciones entre clases y objetos.
4. Especificación de la interfaz entre clase y objeto e implementación.

Identificación de las clases y objetos: se identifican las clases y objetos para que se establezcan límites del problema. En el análisis se decide qué es importante y qué no lo es para el desarrollo del sistema. En el diseño se determinan aquellas abstracciones que son importantes y que forman parte del espacio de la solución.

Identificación de la semántica de las clases y objetos: en la etapa del análisis se establece el comportamiento (operaciones) y los atributos para cada una de las clases y objetos identificadas en el paso anterior. En la etapa del diseño se utiliza para llevar a cabo una separación de las partes relacionadas en la solución.

Identificación de las relaciones entre clases y objetos: en el análisis se identifican las asociaciones que existen entre las clases y los objetos considerando algunas relaciones de herencia y de agregación; se construyen diagramas de clase, donde se establecen las asociaciones entre las abstracciones y se incluyen las operaciones y los atributos.



También se construyen diagramas de objetos considerando nuevas interacciones de las clases y los objetos.

Especificación de la interfaz entre clase y objeto e implementación: en el análisis se busca identificar nuevas clases y objetos que van a servir para el siguiente nivel de abstracción y que van a ser consideradas en el siguiente paso de este proceso iterativo, así como perfeccionar aquellas abstracciones ya definidas con anterioridad. En el diseño se crean representaciones tangibles de las abstracciones existentes para que se pueda obtener un modelo de las abstracciones que pueda ser ejecutable.

### **Método OMT (Object Modeling Technique) por Rumbaugh**

En este método, Rumbaugh introduce conceptos referentes al modelado que se utiliza como una técnica de diseño. Los conceptos y las notaciones que involucra el modelado orientado a objetos se utilizan durante el análisis y el diseño.

Etapas del método OMT:

#### 1.- Análisis.

El objeto del análisis es desarrollar un modelo de lo que va a hacer el sistema. Este modelo se divide en tres: modelo del objeto, modelo dinámico y modelo funcional, mediante esta división se representan a los objetos y sus relaciones, flujo de control dinámico y la transformación funcional del sistema respectivamente. El proceso de captura de requisitos y de consultas con el solicitante debería de extenderse a lo largo de todo el análisis.

#### 2.- Diseño del sistema.

Durante la etapa del análisis se determina que es lo que la implementación debe hacer, la cual se representa con la ayuda del modelo dinámico, de objeto y funcional. Durante la etapa del diseño del sistema se determina la estrategia a seguir para resolver el problema y construir una solución. Si se trata de algún proyecto muy grande es aquí donde los subsistemas se identifican y particionan de manera ordenada para que el desarrollo sea más manejable y sencillo.

#### 3.- Diseño del objeto.

Se construye un modelo de diseño que se base en el modelo del análisis, pero en el que se incluyen algunos detalles de la implementación que deben de ir de acuerdo con la estrategia que se eligió en la etapa anterior. En esta etapa se determina la definición completa de las clases y las asociaciones que serán usadas durante la implementación, tales como las interfaces y los algoritmos utilizados para la implementación de las operaciones. También se incluyen nuevos objetos, atributos de los objetos y las operaciones. El diseñador debe elegir la manera de implementar los objetos para lograr obtener el mínimo de tiempo de ejecución y de memoria requerida.

## Metodología Propia de la Empresa

En el desarrollo de sus sistemas la empresa ICA emplea una metodología basada en OOD, OMT y utiliza como apoyo herramientas UML (Unified Modeling Language, Lenguaje Unificado de Modelado).

En el Análisis se profundiza en las necesidades del usuario para la comprensión del problema y se documentan los requerimientos para el desarrollo del sistema. Dentro de este punto existe la identificación del problema y opciones de solución.

En la etapa de Diseño se trabaja con los resultados del análisis, haciendo las revisiones pertinentes hasta que el diseño se haya completado y así definir cómo va a funcionar el sistema, para obtener un grupo de objetos que interactúen y puedan ejecutar las operaciones descritas en el modelo de operación.

El Diseño consta de los siguientes elementos:

- Diagrama de Casos de Uso:

El diagrama de casos de uso representa las formas en la que los actores (cada tipo de usuario que interactúa en el sistema) realizan sus funciones en el sistema.

- Diagrama de Clases:

En este punto se deben identificar las clases candidatas, e identificar sus responsabilidades y colaboraciones. Este diagrama muestra las clases (descripciones de objetos que comparten características comunes) que componen el sistema y cómo se relacionan entre sí.

- Diagrama de Objetos y Eventos:

Una vez identificadas las clases, sus responsabilidades y colaboraciones, se elabora el diagrama de objetos y eventos que consiste en identificar una serie de objetos (instancias de las clases) y sus relaciones.

- Diagrama de Flujo de Datos:

Este diagrama que representa el flujo de la información y de las transformaciones que se aplican a los datos al moverse desde la entrada hasta la salida; se observa a través de niveles, que son dados por los procesos, esto permite visualizar el sistema desde el punto de vista de los datos y no de quién trabaja con ellos.

- Diagrama de Secuencia:

Por medio del diagrama de secuencia observamos la interacción entre los objetos y los mensajes que intercambian entre sí, junto con el orden temporal de los mismos.

De estos elementos se obtienen los siguientes complementos:

- Diccionario de Datos

Los sistemas de información tienden a ser complejos y se caracterizan por la heterogeneidad. Algunos almacenan diferentes tipos de datos cada uno de ellos con su propio modelo, además los mismos pueden ser vistos por usuarios de diferentes niveles de abstracción, por lo que los usuarios deben entender el significado de todos los tipos de datos, para ello son ayudados por la disponibilidad de un Diccionario de Datos, que es un depósito integrado de todos los tipos de datos producidos, controlados, intercambiados y mantenidos. Esta información se debe presentar de manera uniforme.

- Diagrama entidad-relación (ER)

El diagrama ER es parte de las herramientas conceptuales que existen para describir datos y las relaciones que existen entre ellos. El diagrama ER está basado en una percepción del mundo, como si se tratara de una colección de objetos básicos (entidades), representados por una colección finita de tablas de dos dimensiones (columnas y renglones) y las relaciones que existen entre estos objetos, las cuales representan acciones o situaciones reales. El diagrama ER se caracteriza por su simplicidad, precisión y flexibilidad.

El diagrama ER se compone principalmente de: entidades, relaciones y atributos. Las entidades hacen referencia a un objeto que es distinguible a través de un rectángulo. Una entidad puede ser tangible o intangible.

Una relación se refiere a la relación de dos o más entidades, puede tener atributos, permite expresar la cantidad de entidades con las que puede asociarse otra entidad. Las relaciones de acuerdo al número de elementos que se involucran de ambos lados de las entidades que participan se clasifican en tres tipos: Uno a Uno (1-1), Uno a Muchos (1-M) y Muchos a Muchos (M-M).

Un atributo es una función con rango de valores que permiten identificar a un conjunto de entidades. Para colocar los atributos de manera correcta en una entidad se utiliza el método de normalización que permite verificar si nuestro modelo es correcto o bien si funcionará al ser implementado.

- Normalización

Es un proceso que consiste en comprobar que las tablas definidas cumplen unas determinadas condiciones. Se pretende garantizar la no existencia de redundancia y una cierta coherencia en la representación mediante un esquema relacional de las entidades y relaciones del diagrama ER. Mediante la normalización se pueden solucionar diversos errores en el diseño de la base de datos así como mejorarlo. También se facilita el trabajo posterior del

administrador de la base de datos y de los desarrolladores de aplicaciones. En este proceso, las relaciones deben cumplir con las Formas Normales, que describiremos más a detalle en el diseño del sistema en el capítulo 3.

La Etapa de Desarrollo es la construcción del sistema basado en los resultados del análisis y diseño; es decir la generación de código fuente, *scripts*, interacción con la base de datos y la generación de pantallas.

Dentro de la Etapa de Pruebas el objetivo es evaluar si el sistema funciona correctamente con respecto a las necesidades del usuario. El proceso se lleva a cabo con pruebas al sistema y éstas pueden ser fáciles o difíciles dependiendo qué tan cuidadosamente se hayan realizado las actividades iniciales de análisis y diseño, y representa una revisión final de las especificaciones del diseño y codificación.

La Etapa de Liberación incluye la instalación, documentación para usuarios finales y administradores, además de la puesta a punto del sistema, capacitación técnica, monitoreo y control de cambios.

## **1.4. Infraestructura de comunicaciones**

Las comunicaciones son elementos muy importantes para el buen desarrollo de un sistema, ya que sin éstas no existiría una plena explotación de los recursos y por lo tanto, no se vería reflejado el trabajo elaborado para un fin determinado, como lo es el de este proyecto. Es por ello que se mencionan algunos elementos esenciales, pensados para que este sistema tenga un funcionamiento adecuado, presentando también diversas opciones para integrar una buena comunicación.

### **1.4.1. Tipos de Redes**

Las redes consisten en dos o más dispositivos electrónicos unidos que comparten recursos (archivos, CD-ROM's o impresoras) y que son capaces de realizar transacciones electrónicas. Las redes pueden estar unidas por cable, líneas de teléfono, ondas de radio, satélites, etc.

Las redes se pueden clasificar básicamente en términos de cobertura en:

- Redes LAN (*Local Area Network*, Redes de Área Local)
- Redes MAN (*Metropolitan Area Network*, Redes de Área Metropolitana)
- Redes WAN (*Wide Area Network*, Redes de Área Extensa)

Redes LAN: Es una red de datos de alta velocidad, tolerante a fallas, que cubre un área geográfica relativamente pequeña. Por lo regular conecta estaciones de trabajo, computadoras personales, impresoras y otros dispositivos en un solo edificio u otra área geográfica limitada. Las LAN tienen muchas ventajas para los usuarios de

computadoras, entre otras, el acceso compartido a dispositivos y aplicaciones, el intercambio de archivos entre los usuarios conectados y la comunicación entre usuarios vía correo electrónico y otras aplicaciones.

Los estándares de las redes LAN especifican el cableado y la señalización en las capas física y de enlace de datos del modelo OSI. Las redes Ethernet, FDDI y Token Ring son tecnologías LAN ampliamente utilizadas.

Los usuarios que requieren ejecutar una aplicación frecuentemente pueden bajarla del servidor y luego correrla desde su disco duro local. Los usuarios pueden mandar a imprimir u otras funciones con otros dispositivos a través de aplicaciones que corren en el servidor de la red LAN. Un usuario puede compartir archivos con otros usuarios en el servidor de la red, pero los permisos de lectura y escritura los proporciona el administrador de la LAN.

Redes de Área Metropolitana: Las redes de área metropolitana (MAN) cubren extensiones mayores como puede ser una ciudad, mediante la interconexión de redes LAN por medio de una línea grande de transmisión llamada *backbone*. En general tiene los mismos usos que las redes LAN. Una MAN se extiende sobre un área geográfica mayor que una LAN, pero en un área geográfica menor que una WAN.

Redes de Área Extensa: Una WAN es una red de comunicación de datos que da servicio a usuarios localizados en un área geográfica amplia y generalmente utiliza los dispositivos de transmisión que ofrecen las compañías de telecomunicaciones. Las redes Frame Relay, SMDS y X.25 son ejemplos de WANs.

Las tecnologías WAN operan en las tres capas inferiores del modelo de referencia OSI: la capa física, la capa de enlace de datos y la capa de red. Una red de área extensa (WAN) cubre grandes regiones geográficas como un país, un continente o incluso el mundo. El cable transoceánico o los satélites se utilizan para enlazar estos puntos tan distantes entre sí, en general se utilizan medios públicos para interconectarlas como la línea telefónica. El mejor ejemplo de una WAN es internet.

Para que una red pueda transmitir la información es necesario un medio de transmisión.

Medios de Transmisión: Existen diferentes tipos de medios por los cuales transmitir información:

- Cable
- Microondas
- Satélites
- Infrarrojo

El cableado de red: El cable es uno de los medios a través del cual fluye la información a través de la red. Hay distintos tipos de cable de uso común en redes LAN. Una red puede utilizar uno o más tipos de cable, aunque el tipo de cable utilizado siempre está sujeto a la topología de la red que se instala y el tamaño de ésta.

Los tipos de cable más utilizados en las redes LAN son:

- Cable UTP (*Unshielded Twisted Pair*, par trenzado sin apantallar)
- Cable STP (*Shielded Twisted Pair*, par trenzado apantallado)
- Cable coaxial
- Cable de fibra óptica

El cable UTP es un tipo de cable muy popular, su calidad y, consecuentemente, la cantidad de información que es capaz de transmitir varían en función de la categoría del cable. Las categorías van desde el cable de teléfono, que sólo transmite la voz, hasta la categoría 5, capaz de transferir 100 Mbps (megabits por segundo) de información. La tabla 1.1 muestra las características principales por cada categoría de cable UTP.

Tipo	Máxima tasa de transferencia	Uso
CAT 1	Menor a 1 Mbps	Voz (Cable telefónico)
CAT 2	4 Mbps	Utilizado principalmente para cableado de redes con topología anillo
CAT 3	16 Mbps	Voz y datos en redes ethernet 10BASE-T
CAT 4	20 Mbps	Redes con topología anillo a 16 Mbps
CAT 5	100 Mbps 1000 Mbps (4 pares)	155 Mbps en redes ATM o redes Gigabit Ethernet

*Tabla 1.1. Categorías UTP y sus características.*

Un medio de transmisión puede operar de diferentes modos, acorde con los equipos con los que trabaja, siendo estos: *simplex*, *half duplex* y *full duplex*.

**Método *Simplex*:** Los datos pueden transmitirse en un solo sentido pero no es muy común en empresas que procesan datos, ya que en la mayor parte del equipo de cómputo la comunicación es de dos vías, aún con los dispositivos de sólo recepción, como las impresoras de alta velocidad comunican un mensaje de reconocimiento al dispositivo.

**Método *Half Duplex*:** Los mensajes pueden ser transmitidos en cualquier sentido, pero sólo uno a la vez. A menudo la línea de comunicación entre una impresora y el CPU es *half duplex*.

Método *Full Duplex*: El flujo se desplaza en dos sentidos al mismo tiempo.

Se ha mencionado la importancia de las redes y su transmisión en comunicaciones, pero en el buen funcionamiento de una red también está su topología, de la cual existen varios tipos.

### 1.4.2. Topologías de Red

Se le conoce como topología de red a la forma o conectividad física de la misma, es decir, la forma como sus nodos se conectan y comunican entre sí. Existen cinco principales topologías de redes LAN:

- Topología de bus
- Topología en árbol
- Topología en anillo
- Topología en estrella
- Topología de malla

Topología en Bus: Este tipo de arreglo es muy popular en redes de área local. El control de flujo de tráfico es relativamente sencillo por el hecho de que todas las estaciones están conectadas a un medio de transmisión común, lo cual permite que todas ellas reciban o envíen todas y cada una de las transmisiones. Sin embargo, su principal desventaja consiste precisamente en que sólo existe un canal de transmisión para todos los dispositivos que se encuentran conectados a la red. En consecuencia, si existe una falla en dicho canal, se pierde toda la red. Esto se puede evitar mediante el uso de canales redundantes.

Topología en árbol: Combina las características de un bus lineal y la topología en estrella. Consiste en grupos de estrellas conectadas a un *backbone* lineal. En la mayoría de los casos el nodo de mayor jerarquía es el que controla la red. Algunas implementaciones incorporan un poco de distribución del sistema al permitir que los nodos de un orden de jerarquía controlen a sus subordinados de orden inferior, lo que reduce la carga del servidor de mayor jerarquía. Aunque esta topología es muy atractiva desde el punto de vista de simplicidad de control, es muy sensible a problemas que ocasionen cuellos de botella. En algunos casos, el nodo de mayor jerarquía controla todo el tráfico entre los nodos. Esto no únicamente baja el rendimiento real del servidor para otros procesos, sino que también presenta problemas de confiabilidad. Si el servidor central falla, la red completa deja de operar a menos que se cuente con un servidor de respaldo. El mantenimiento y crecimiento de este tipo de redes son muy sencillos, pues no cuesta demasiado trabajo agregar nodos subordinados.

Topología en anillo: Esta topología recibe este nombre por el aspecto circular del flujo de datos. En la mayoría de los casos la información fluye exclusivamente en un sentido

con una estación recibiendo la señal y transmitiéndola a la siguiente estación en el anillo.

Esta configuración es muy popular puesto que no tiene problemas de cuellos de botella que presentan las estructuras de árbol. Además, la lógica necesaria para implementar una red en anillo es relativamente sencilla. Cada componente tiene la tarea de aceptar los datos, enviarlos al siguiente nodo, o enviarlos nuevamente al anillo al siguiente componente intermedio. Sin embargo, como todas las redes, esta configuración también tiene sus propios problemas. El principal es que, como en la topología de bus, existe un sólo canal que conecta a todos los componentes de red. Si el canal entre dos nodos adyacentes falla, se pierde toda la red. Una solución a esta situación puede ser tener canales de respaldo. También se pueden instalar *switches* que enruten automáticamente los datos para evitar nodos congestionados o inoperantes.

Topología en estrella: Todos los dispositivos están conectados a un *hub* central. Los datos en este tipo de redes fluyen del emisor hasta el concentrador, mismo que realiza todas las funciones de red. En vista de que el hub es el responsable de rutear el tráfico de los demás componentes, también es responsable de aislar las fallas. Esto es relativamente sencillo en una red tipo estrella porque las líneas pueden ser aisladas individualmente para identificar el problema. Sin embargo, al igual que en la topología de árbol, esta red es susceptible de sufrir cuellos de botella y fallas en el nodo central. Del mismo modo, es posible establecer enlaces redundantes y un respaldo del nodo central, para aumentar su confiabilidad.

Topología en malla: En este tipo de topologías se hacen conexiones punto a punto entre cada una de las unidades de red. Como se requiere que cada unidad tenga una interfaz con cada uno de los dispositivos de red, esta topología no se considera muy práctica. A menos que cada estación requiera enviar frecuentemente mensajes a todas las demás estaciones porque se requiere un ancho de banda muy grande. Esta topología ofrece multiplicidad de caminos entre los nodos, ya que es posible rutear el tráfico evitando componentes que fallen o nodos ocupados. Además de que presenta una relativa inmunidad a fallas y cuellos de botella. Otra de las ventajas que presenta esta topología es que si ocurre una falla, puede aislarse el nodo para que no afecte a otras unidades, debido a que cada enlace es independiente de los demás. Aunque esta configuración ofrece el máximo de confiabilidad, el control de flujo de información es sumamente costoso y complejo, tanto en consumo de recursos como en precio. El inconveniente que presenta este tipo de topologías es la dificultad de instalación y reconfiguración debido a que requiere mucho cableado, sobretodo al aumentar el número de nodos o mover los nodos de lugar.

Con base en la información dada, se podrán comprender de una mejor manera los estándares de transmisión en redes que forman el conjunto de conceptos y características dadas.

Estándares de transmisión en redes LAN: Se le conocen como protocolos de transmisión al conjunto de normas que rigen la comunicación entre las computadoras de una red. Estas normas especifican qué tipo de cables se utilizarán, qué topología



tendrá la red, qué velocidad tendrán las comunicaciones y de qué forma se accederá al canal de transmisión.

Los estándares de transmisión en redes LAN más comunes son:

- Ethernet
- Fast Ethernet
- Local Talk
- Token Ring
- FDDI (*Fiber Distributed Data Interface*)

**Ethernet:** Es el estándar de redes LAN más ampliamente utilizado. Fue desarrollado por Xerox, DEC e Intel. Permite topologías bus, estrella o árbol y soporta tasas de transferencia de datos de 10Mbps, también conocidas como 10BASE-T. Una LAN Ethernet típica utiliza cable coaxial o un tipo especial de par trenzado. Los dispositivos están conectados a un cable y compiten por el acceso a la red utilizando el protocolo de acceso múltiple con detección de colisiones, CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*); es un método de acceso que maneja demandas simultáneas, es decir, cuando una estación quiere acceder a la red escucha si hay alguna transmisión en curso y si no es así transmite. En el caso de que dos redes detecten probabilidad de emitir y emitan al mismo tiempo, se producirá una colisión pero esto queda resuelto con los sensores de colisión que obligan a la retransmisión de la información.

La siguiente tabla muestra las características de las redes Ethernet 10 Base-T:

Tipo de Ethernet	Velocidad [Mbps]	Distancia [m]	Medio
10Base-5	10	500	Coaxial grueso
10Base-2	10	185	Coaxial fino
10Base-T	10	100	UTP
10Base-F	10	2000	Fibra óptica

*Tabla 1.2. Redes Ethernet 10Base-X y sus características.*

**Fast Ethernet:** Algunas nuevas versiones de redes Ethernet son las llamadas 100Base-T o Fast Ethernet, que soportan tasas de transferencia de 100Mbps. La nueva versión Gigabit Ethernet soporta tasas de transferencia de 1 Gbps.

La tabla 1.3 muestra las características de las redes Ethernet 100 Base-X:

Tipo de Ethernet	Velocidad [Mbps]	Medio
100Base-TX	100	UTP CAT 5
100Base-FX	100	Fibra óptica
100Base-T4	100	UTP CAT 3 modificado (2 líneas más al CAT 3)

*Tabla 1.3. Redes Ethernet 100Base-X y sus características.*

Para aumentar la velocidad de la red de 10Mbps a 100Mbps se han definido nuevos estándares de Ethernet denominados en conjunto FastEthernet, éstas son tres tipos de redes Ethernet que quedan reducidas a la topología en estrella.

Local Talk: Fue desarrollado por Apple Computer, para computadoras *Macintosh*. El método de acceso al medio es el CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance). Este método, similar al de Ethernet (CSMA/CD), se diferencia en que la computadora anuncia su transmisión antes de realizarla. Mediante el uso de adaptadores Local Talk y cables UTP especiales se puede crear una red de computadoras Mac a través del puerto serie. El sistema operativo de estos establece relaciones punto a punto sin necesidad de software adicional, aunque se puede crear una red cliente servidor con el software AppleShare. Con el protocolo Local Talk se pueden utilizar topologías bus, estrella o árbol usando cable UTP, aunque la velocidad de transmisión es muy inferior a la ethernet.

Token Ring: Token Ring fue desarrollado por IBM a mediados de los 80's. Una red token ring es una red en la que todas las computadoras están conectadas en una topología de anillo o estrella y un dígito binario o *token* es usado para prevenir colisiones de datos entre dos computadoras que quieran enviar mensajes al mismo tiempo. Token Ring es el segundo estándar más ampliamente usado en las redes de área local, después de Ethernet. La tecnología Token Ring provee tasas de transferencia de 4 ó 16 Mbps. A continuación se muestra el procedimiento básico:

- a. Paquetes vacíos de información circulan continuamente en el anillo.
- b. Cuando una computadora tiene que enviar un mensaje, inserta un "*token*" en el paquete vacío (esto puede consistir simplemente en cambiar un cero por un uno en el *bit* "*token*" del paquete) e inserta un mensaje y un identificador del destinatario.
- c. El paquete es examinado después por cada una de los siguientes nodos. Si el nodo ve que es el destinatario del mensaje, copia el mensaje del *frame* y cambia de nuevo el "*token*" a vacío.
- d. Cuando el paquete regresa a su origen, verifica que el "*token*" ha sido cambiado a cero y ese mensaje ha sido copiado y recibido, por lo que lo remueve del paquete.
- e. El paquete continúa circulando en un paquete vacío, listo para ser tomado por un nodo cuando tenga un mensaje que enviar.

FDDI: Esta red se utiliza principalmente para interconectar dos o más redes locales que con frecuencia están a grandes distancias. En este tipo de redes, la topología es anillo dual. La transmisión se da en uno de los anillos, pero si tiene lugar un error en la transmisión el sistema es capaz de utilizar una parte del segundo anillo para cerrar el anillo de transmisión. Se monta sobre cables de fibra óptica y pueden alcanzar velocidades de 100 Mbps. Si el anillo secundario no es necesitado como backup, también puede llevar tráfico extendiendo su capacidad hasta 200 Mbps. Un anillo dual puede extenderse hasta 100 km.

Para comprender de manera más general los estándares de transmisión de redes, se muestra una tabla comparativa (ver tabla 1.4) donde podemos observar los tipos de medio, la velocidad a la que pueden operar y las diferentes topologías posibles de configurar a través de los diferentes protocolos de redes.

Estándar	Cable	Velocidad	Topología
Ethernet	Par trenzado, coaxial, fibra óptica	10 Mbps	Bus, Estrella, Árbol
Fast Ethernet	Par trenzado, fibra óptica	100 Mbps	Estrella
Local Talk	Par trenzado	0.23 Mbps	Bus o Estrella
Token Ring	Par trenzado	4 – 16 Mbps	Estrella, Anillo
FDDI	Fibra óptica	100 Mbps	Anillo

*Tabla 1.4. Resumen de Protocolos.*

### 1.4.3. El Modelo TCP/IP

El Departamento de Defensa de los Estados Unidos definió un conjunto de reglas que establecieron cómo conectar computadoras entre sí para lograr el intercambio de información, soportando incluso desastres mayores en la red. Fue así como se definió el conjunto de protocolos de TCP/IP (*Transfer Control Protocol/Internet Protocol*). Para los años 80 una gran cantidad de instituciones estaban interesadas en conectarse a esta red que se expandió por todo Estados Unidos.

El protocolo TCP/IP es la pieza fundamental que constituye la red Internet, ya que define una red de conmutación de paquetes, es decir, toda la información que se va a transmitir es fragmentada en trozos o paquetes. Cada uno de estos paquetes es enviado a la dirección de la computadora donde debe llegar la información, salta de red en red hasta llegar a su destino. Cada computadora tiene una dirección IP, estas direcciones son identificadores unívocos de cada computadora. Estos paquetes viajan a través de unos “encaminadores” o “routers” que eligen las rutas más viables que seguirán dichos paquetes hasta llegar a sus correspondientes destinos. Para controlar tanto el soporte físico como lógico de ese tráfico de datos se utiliza el protocolo TCP/IP, que es el corazón de esta red y hace que funcione, este protocolo se divide en dos funciones: TCP y IP.

TCP se encarga fundamentalmente de particionar la información, empaquetarla, numerarla y añadir la información necesaria para que cuando lleguen a su punto de destino estos paquetes puedan ser desempaquetados y ensamblados en el orden correcto. La información extra que se añade sirve para determinar cuando llegue a su punto de destino, si está toda la información o se ha perdido algún paquete durante la transmisión.

IP se encarga de encaminar dichos paquetes de datos hasta su destino gracias a sus direcciones electrónicas o direcciones "IP". La IP de una computadora o su número Internet (*Internet number*) es un número binario de 32 bits dividido en 4 campos de 8 bits (u octetos) separados por puntos y representados por sus correspondientes valores decimales (de 0 a 255). Por ejemplo, el número 121.212.334.34 podría ser la dirección de una máquina o *host*, los tres primeros números indican la red a la que pertenece nuestra computadora, y el último sirve para diferenciar nuestro dispositivo de los otros que cuelguen de la misma red. Esta distribución jerárquica de la Red Internet permite enviar y recibir rápidamente paquetes de información entre dos dispositivos conectados en cualquier parte del Mundo a Internet, y desde cualquier subred a la que pertenezcan. Si quisiéramos establecer una conexión con otra computadora en la red los programas y aplicaciones necesitan saber cuál es el número de la máquina con la que nos queremos comunicar.

El funcionamiento de IP es similar al de un número telefónico, cada usuario tiene asignado una dirección IP de cuatro números, cada número es un *byte* que su valor puede variar entre 0 y 255. Si nos hemos equivocado al introducir el nombre del dominio remoto de la computadora éste no será encontrado y recibiremos un mensaje en nuestra pantalla de error del tipo "*Unrecognized host name*" o similar.

El modelo TCP/IP consta de 4 capas:

Capa Host a Red. La capa inferior del modelo TCP/IP se relaciona con la capa física respecto del modelo OSI, y contiene varios estándares del IEEE, como el 802.3 llamado Ethernet, que establece las reglas para enviar datos por cable coaxial delgado (10Base2), cable coaxial grueso (10Base5), par trenzado (10Base-T), fibra óptica (10Base-F) y su propio método de acceso al medio físico. El 802.4 llamado Token Bus, que puede usar estos mismos medios pero con un método de acceso diferente y otros estándares denominados genéricamente como 802.X

Capa de Red. Esta capa maneja la comunicación de una máquina a otra. Ésta acepta una solicitud para enviar un paquete desde la capa de transporte, junto con una identificación de la máquina hacia la que se debe enviar el paquete. Encapsula el paquete en un datagrama IP, llena el encabezado del datagrama, utiliza un algoritmo de ruteo para determinar si se puede entregar el datagrama directamente o si debe enviarlo la interfaz de red apropiada para su transmisión. La capa de red también maneja la entrada de datagramas, verifica su validez y utiliza un algoritmo de ruteo para determinar si el datagrama debe procesarse de manera local. La capa de red borra el encabezado del datagrama y selecciona, de entre varios protocolos de transporte, un protocolo con el que manejará el paquete. Por último, la capa de red envía los mensajes ICMP (*Internet Control Message Protocol, Protocolo de Internet de Control de Mensajes*) de error y de control necesarios y maneja todos los mensajes ICMP entrantes.

Capa de transporte. Su principal tarea es proporcionar la comunicación entre un programa de aplicación y otro. Este tipo de comunicación se conoce frecuentemente como comunicación punto a punto. La capa de transporte regula el flujo de información.

Puede también proporcionar un transporte contable, asegurando que los datos lleguen sin errores y en secuencia. Para hacer esto, el protocolo de transporte tiene el lado de recepción enviando acuses de recibo de retorno y la parte de envío retransmitiendo los paquetes perdidos. Está formada por dos protocolos: TCP y UDP. El primero es un protocolo confiable y orientado a conexión, lo que significa que ofrece un medio libre de errores para enviar paquetes. El segundo es un protocolo no orientado a conexión y no es confiable.

Capa de Aplicación. En esta capa, los usuarios llaman a una aplicación que accede servicios disponibles a través de Internet. Una aplicación interactúa con uno de los protocolos de nivel de transporte para enviar o recibir datos. Cada programa de aplicación selecciona el tipo de transporte necesario, el cual puede ser una secuencia de mensajes individuales o un flujo continuo de octetos. El programa de aplicación pasa los datos en la forma requerida por el nivel de transporte para su entrega. En la última capa se encuentran decenas de aplicaciones ampliamente conocidas actualmente, las más populares son los protocolos WWW, FTP, Telnet, DNS, el servicio de correo electrónico (SMTP), etc. En la Figura 1.2. presentamos una tabla comparativa de modelos OSI y TCP/IP.

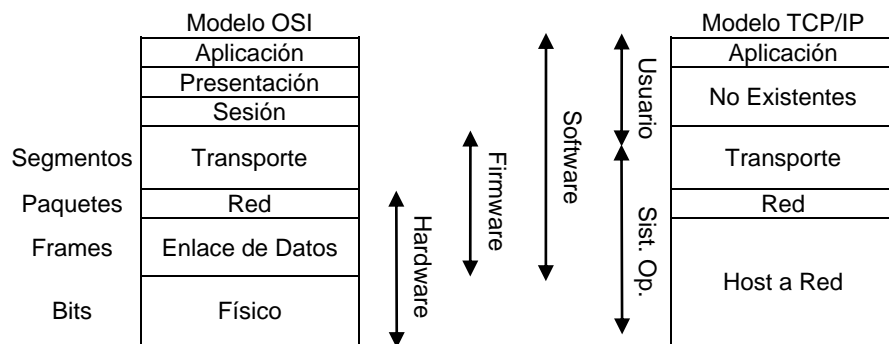


Figura 1.2. Comparación entre los modelos OSI y TCP/IP.

Dado que los protocolos son un pilar de Internet, han surgido una gran cantidad de servicios que proporcionan diversas facilidades al usuario.

#### 1.4.4. Tipos de servicio

Los servicios más conocidos que se proporcionan a través de Internet son el correo electrónico (e-mail), FTP (*File Transfer Protocol*, Protocolo de Transferencia de Archivos) y la conexión remota (Telnet), así como también nuevas aplicaciones, como foros de discusión (News y Usenet) y el más conocido y llamativo, el World Wide Web (WWW).

Telnet. Una de las utilidades básicas es la conexión remota a otra computadora (sesiones de trabajo en computadoras remotas). Se realiza mediante el protocolo Telnet, a través del cual un usuario puede contactar con cualquier computadora remota de Internet, siempre que se le permita el acceso a esa máquina. La computadora desde la que se lleva a cabo la conexión, ya sea una computadora personal, una minicomputadora, etc., se convierte en un terminal remoto de cualquiera de los *hosts* de Internet que le permitan el acceso, y desde él se podrá acceder a servicios públicos, como, por ejemplo, bases de datos. Una computadora se puede conectar a otra del mismo edificio o a una tercera que esté en el extremo más alejado del mundo. Una vez que se logra la conexión, el teclado funciona como si estuviese conectado directamente a la computadora, pudiéndose acceder a los servicios que la máquina remota proporciona a sus terminales locales. Para poder acceder a una computadora remota, se debe especificar su nombre en Internet o dirección IP. Una vez conectado, se solicita el login de acceso y el password.

Uno de los sistemas operativos que habitualmente utilizan los *hosts* conectados a Internet es UNIX. Al acceder a una máquina UNIX mediante Telnet se trabaja en el directorio HOME del usuario como directorio de trabajo, y se pueden ejecutar los comandos típicos de UNIX. Se pueden iniciar sesiones de Telnet desde una computadora *host* o desde una computadora personal que tenga la aplicación Telnet.

Correo Electrónico (E-Mail). El correo electrónico (*e-mail*) permite a los usuarios de Internet enviar mensajes que incluyan textos, imágenes, sonidos, etc., Mediante el correo electrónico se pueden mandar mensajes de unas líneas o de cientos de páginas en escasos segundos de un lugar del mundo a otro.

El correo electrónico tiene una serie de características que lo hacen muy útil:

- Es muy rápido; un mensaje puede tardar unos minutos en llegar de un extremo a otro del planeta.
- Cabe la posibilidad de contestarse o reenviarse automáticamente.
- Se pueden mandar copias a un grupo de personas.
- Es posible enviar mensajes con “acuse de recibo” para cerciorarse de que realmente han llegado a su destino.

Cada usuario del correo electrónico posee una dirección electrónica y un buzón situado en una computadora conectada a Internet. La dirección electrónica está formada por dos partes, separadas por el símbolo arroba @. A la izquierda del símbolo @ se indica el nombre del usuario (el que se teclaea como *login* para acceder al *host*) y en su derecha se pone el nombre en Internet del *host* donde se recibe el correo (por ejemplo, emartinez@avantel.net).

En el momento en que se envía un mensaje a un usuario, no es necesario que la persona a la que va a enviar su correo esté conectada a Internet en ese mismo instante. Cuando el mensaje alcanza el destino, queda depositado en el buzón del usuario, a la espera de que éste se conecte con el *host*, lo recoja y lo lea.

Para mandar y recoger el correo electrónico se debe utilizar un programa específico, que puede estar en el propio *host* donde se recibe el correo o en nuestra computadora personal. En la actualidad, lo normal es que el programa de correo se encuentre en la computadora personal. En el primer caso, hay que conectarse con el *host* mediante la aplicación Telnet. En el caso de utilizar un programa de correo desde una computadora personal, a este programa se le suele denominar cliente o agente e-mail, ya que necesita que se esté ejecutando un programa servidor de e-mail en el *host* que recibe el correo. El servidor de e-mail se puede comunicar con el programa cliente para enviarle los mensajes recibidos en el buzón y para recoger aquellos que se deseen mandar a su destino. Existen numerosos clientes de correo electrónico para las computadoras personales y todos ellos facilitan enormemente la gestión del correo.

FTP. FTP permite a un usuario que tenga autorización (es decir, que posea cuenta o entre como usuario anónimo) conectarse a una computadora remota de Internet y enviar o copiar archivos de cualquier clase.

FTP ha posibilitado la aparición de los llamados servidores FTP anónimos, los cuales son máquinas de acceso libre que contienen documentación de todo tipo, archivos con imágenes, video, sonido, etc. Pueden ser utilizados como cualquier FTP, pero el nombre de acceso será *anonymous*. Muchos sistemas necesitan como clave de acceso su dirección de correo electrónico (e-mail). Si esto falla, la palabra de acceso más genérica es *guest* (invitado).

La aplicación FTP se puede iniciar desde la línea de comando de una máquina UNIX o desde una computadora personal con otro entorno en el que se encuentre instalada la aplicación de transferencia de archivos (FTP). Para hacer FTP es preciso especificar la dirección IP o el nombre de la máquina con la que queremos intercambiar archivos. En la actualidad, lo normal para comenzar una sesión de FTP es hacerlo desde una computadora personal, que debe tener la aplicación FTP.

NEWS. Es un servicio de información utilizado por algunos servidores de Internet. Las News están constituidas por más de 3000 grupos de interés, donde personas de todo el mundo hacen preguntas, dan respuestas, opinan, anuncian o ven lo que otros hacen. Si un equipo electrónico presenta un error periódico, una pregunta en el grupo adecuado de News puede ayudar a encontrar respuestas de quienes han tenido ese mismo problema, o de técnicos que trabajan en la empresa que diseñó el equipo. Hay grupos de conversación sobre computadoras, política, aficiones, ciencias sociales, cocina, etc.

World Wide Web (WWW). World Wide Web es una de las más recientes y espectaculares herramientas de Internet, y una de las que más ha contribuido a su popularización. Este es un servicio que permite a los usuarios de Internet, en un ambiente multimedia (integración de texto, imagen y sonido), tener acceso a la información de los servidores WWW. El W3 se basa en el concepto de hipertexto, cuya característica principal es que los documentos contienen enlaces (links), es decir, palabras que hacen referencia a otros documentos relacionados con el tema, ya sean

texto, archivos, imágenes, sonido, etc. Por tanto, Web es una estructura de enlaces entre páginas de hipertexto.

Con un programa visualizador (*Browser*) se puede navegar en un entorno gráfico muy fácilmente de un documento a otro; basta con pulsar en las palabras enlace (haciendo click). Estos recursos suelen estar en diferentes partes del mundo, pero la conexión es transparente para el usuario.

Después de mencionar los conceptos básicos procederemos al análisis de la problemática del presente trabajo.



# CAPÍTULO II

## PLANTEAMIENTO DEL PROBLEMA Y PROPUESTA DE SOLUCIÓN

En este capítulo se describe de forma general a la empresa ICA y su forma de operación, identificando las principales características de sus problemas en el Área de Finanzas. Esto se logra haciendo un análisis del problema para proponer posibles soluciones.

### **2.1 . Aspectos importantes sobre la empresa ICA**

ICA es la empresa de ingeniería y construcción más grande de México. Desde su fundación en 1947 ha proporcionado sus servicios a clientes de los sectores público y privado, en México y en otros países.

ICA ha trabajado en diversos países de América Latina desde los años sesenta; en 1988 comenzó en el mercado de Estados Unidos y en 1998 inició actividades en Europa y Asia. ICA se ha asociado con empresas líderes en el mundo, para emprender y desarrollar nuevos proyectos.

Como resultado de estas asociaciones están las empresas ICA Fluor, ICA Reichmann e ICA CPC Argentina.

La actividad fundamental de ICA ha sido la construcción de obras de infraestructura, obras urbanas e instalaciones industriales. En el campo de la construcción de obras de infraestructura, ICA ha participado en la ejecución de destacadas obras carreteras, incluyendo puentes y túneles, como la Autopista del Sol, en los Estados de Morelos y Guerrero; obras marítimas, como muelles y escolleras en numerosos puertos mexicanos; algunos puentes sobre el mar, como los 4 kilómetros del tramo marítimo del Corredor Sur, en Panamá; y aeroportuarias, construyendo, ampliando y modernizando

aeropuertos, tanto en México como en otros países. También ha construido presas de almacenamiento de agua para riego y control de avenidas, como Chingaza en Colombia; numerosos distritos de riego en México y en el extranjero; y proyectos hidroeléctricos como Infiernillo y Aguamilpa en México, o Anchicayá en Colombia.

Con relación a la construcción de obras urbanas, ICA ha realizado importantes obras de edificación, desarrollos habitacionales, complejos comerciales, vialidades, puentes, pasos a desnivel y sistemas de transporte colectivo. Se pueden mencionar el Hotel Nikko, el paso a desnivel en Reforma y Arquímedes, el Estadio Azteca y la Unidad Independencia, en la Ciudad de México. El Centro Magno en Guadalajara, puentes y pasos a desnivel del Paseo Tollocan en Toluca, Estado de México, y el 90% de las obras del Metro de la Ciudad de México.

En el área de la construcción de instalaciones industriales, ICA ha construido todo tipo de plantas manufactureras, automotrices y de proceso, como la planta IEM en Aguascalientes, la Nissan en Cuernavaca o las plantas cementeras en Hermosillo y Zapotiltic en México, o El Salvador, en Centroamérica. Para la Comisión Federal de Electricidad ha construido termoeléctricas como Carbón II en Coahuila, López Mateos en Tuxpan Veracruz, o Petacalco en Guerrero. Ha tendido líneas de transmisión en casi todo el territorio nacional. Para PEMEX ha construido plantas petroquímicas en cinco complejos, instalaciones para producción de nitrógeno, como la planta de Cantarell, o para la eliminación de azufre, como la dehidrosulfuradora de Tula.

A principios de los 90's ICA incursionó en el campo de la operación de infraestructura. Además de la operación y mantenimiento de carreteras y autopistas de cuota, esta área incluye la administración y manejo de sistemas de distribución de agua potable para ciudades y poblados, así como la captación y tratamiento de aguas residuales, la recolección y disposición integral de desechos urbanos, y la administración de estacionamientos subterráneos.

Entre los principales proyectos que opera el área se pueden mencionar: la Autopista Maravatío - Zapotlanejo y el Túnel de Acapulco, en México; en Centro y Sudamérica la vialidad Corredor Sur, en la Capital de Panamá y la Autopista Caracas - La Guaira, en Venezuela. El Sistema de Agua potable del Municipio de Aguascalientes, en la capital de dicho Estado, y la cuarta parte del Sistema de Agua Potable del Distrito Federal, en la Ciudad de México. También se cuentan las plantas de tratamiento de Ciudad Acuña y Ciudad Obregón, en Estados del norte de la República Mexicana, así como los Servicios de Aseo en Guadalupe, Nuevo León y en Mérida, capital del Estado de Yucatán

Aprovechando las experiencias acumuladas en la construcción de más de 40,000 viviendas, en localidades como la Ciudad de México; Ciudad Juárez, en Chihuahua; Los Mochis, en Sinaloa; León, en Guanajuato; Puebla, capital de dicho Estado; Jiutepec, en Morelos; y Tijuana y Mexicali, en Baja California, ICA incursiona en el mercado actual de la vivienda, comercializando nuevos productos a través de atractivos esquemas de financiamiento. Con este enfoque, ICA ofrece productos inmobiliarios de la más alta

calidad en conjuntos de vivienda, centros comerciales y edificios de oficinas en diferentes ciudades

### 2.1.1. Estructura Interna

De forma interna, ICA cuenta con una estructura organizacional muy amplia (Figura 2.1), integrada por una Asamblea de Accionistas, un Consejo de Administración y un Comité Ejecutivo que se encarga de asesorar al Presidente de la empresa.

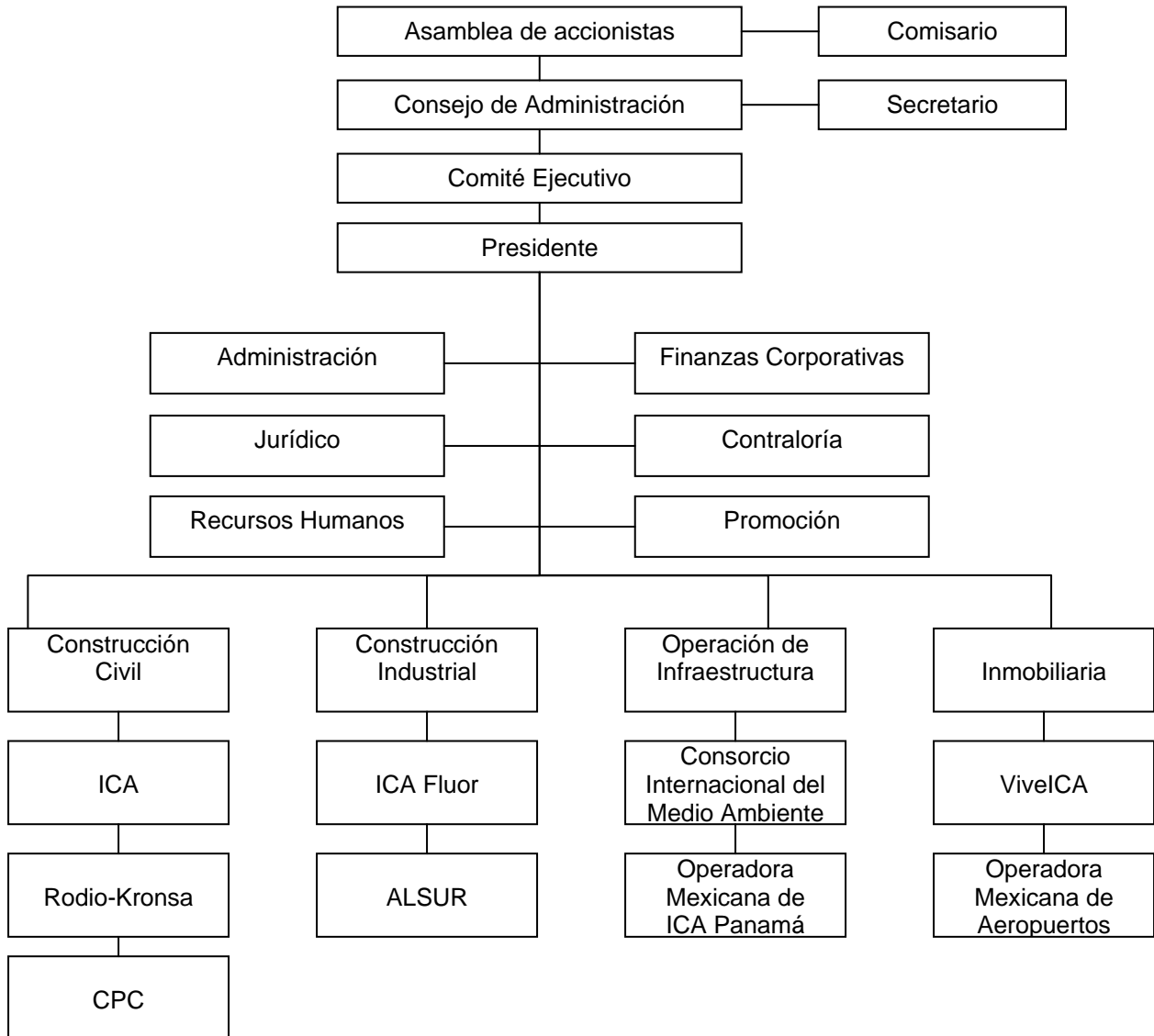


Figura 2.1. Organigrama Grupo ICA.

La empresa se divide en seis principales áreas, como son: Administración, Jurídico, Recursos Humanos, Finanzas Corporativas, Contraloría y Promoción.

El área de Finanzas es el motivo del presente trabajo, por lo que a continuación revisaremos con detenimiento su operación específica.

### **2.1.2. Área de Finanzas**

Las funciones financieras son muy importantes dentro de una empresa, ya que comprenden la provisión de la administración del dinero y de los activos. Es el Área de Finanzas quien recaba toda la información contable o fiscal de la empresa, es la encargada de lograr ingresos suficientes, así como el cálculo de los egresos y el manejo financiero de cualquier faltante o sobrante. Los departamentos que integran el Área de Finanzas son: Tesorería, Planeación Estratégica, Planeación Financiera, Relación con Inversionistas, Desinversión e Inversión y Financiamiento.

Tesorería se encarga de asegurar el máximo rendimiento de los recursos financieros de la empresa y lograr su continuidad operativa, teniendo una operación centralizada que permita obtener las mejores condiciones de negociación ante las instituciones financieras, en cuanto a inversiones, divisas, créditos y cualquier otro tipo de operación de esta naturaleza, sin permitir que esto afecte la operación diaria de la empresa.

Planeación Estratégica es la herramienta de ICA en el diseño de la estrategia corporativa y la definición de líneas de acción a corto, mediano y largo plazo, a fin de generar permanentemente valor a sus accionistas, de acuerdo con las directrices establecidas por la Presidencia y el Comité Ejecutivo. Su objetivo es tener la capacidad técnica siempre actualizada, ética profesional y calidad invariable.

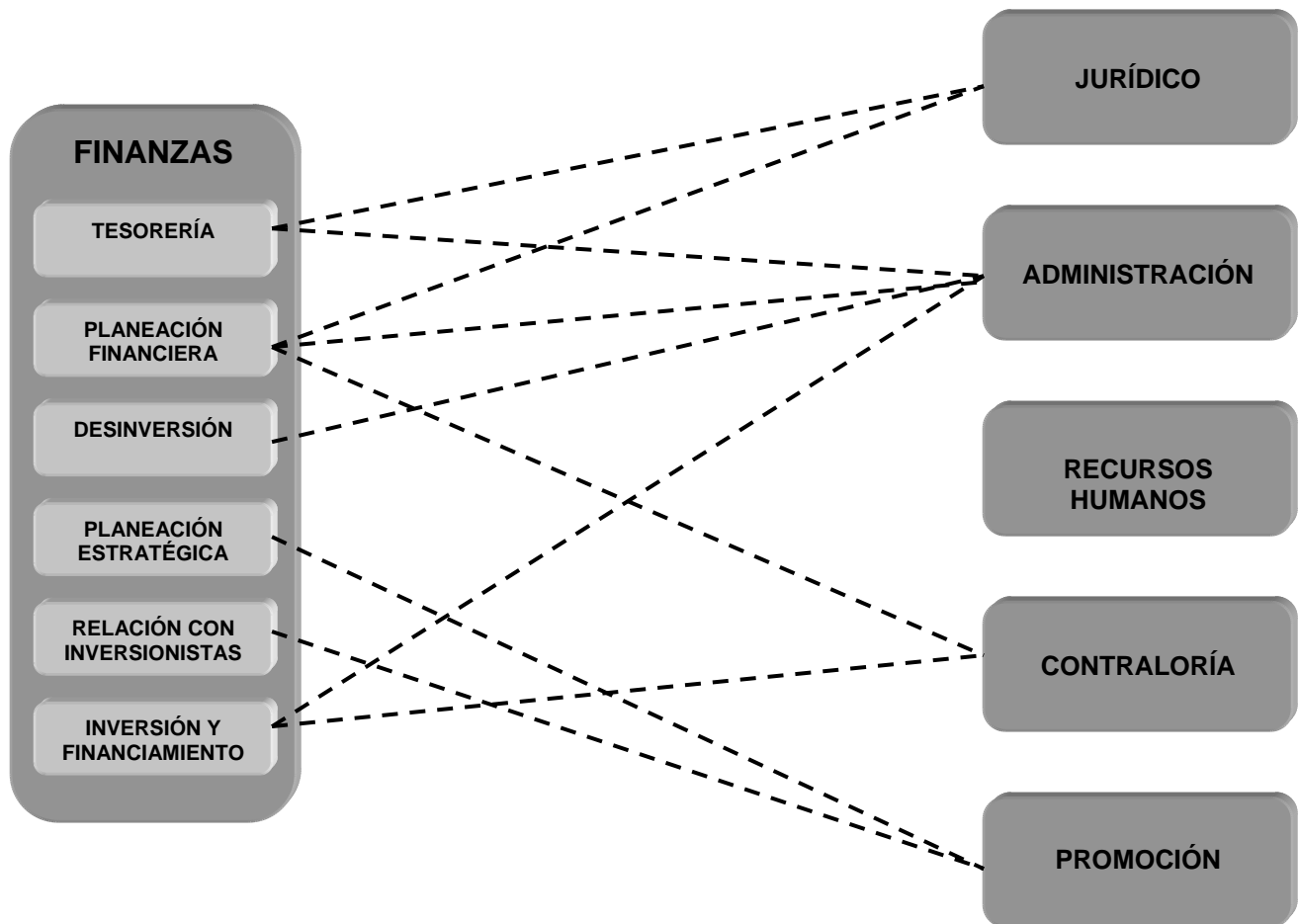
Planeación Financiera se encarga de proporcionar productos y servicios financieros para cubrir las necesidades de los clientes, permitiendo una operación eficiente y maximizando la rentabilidad de negocios de ICA, todo esto a través de un equipo de trabajo comprometido con la mejora continua.

Relación con Inversionistas se encarga de proporcionar información financiera clara, precisa y oportuna de ICA a sus actuales y potenciales inversionistas, analistas e instituciones interesadas en el desempeño de la empresa.

Desinversión se encarga de estructurar y proporcionar recursos mediante la desinversión de activos, que aumenten el valor de la acción de ICA.

Inversión y Financiamiento se encarga de estructurar, proporcionar y controlar los recursos financieros para los proyectos que aumenten el valor de la acción de ICA.

La importancia del Área de Finanzas, y su relación con otras áreas de la empresa, es que mantiene un esencial intercambio de información que permite alcanzar un mayor rendimiento y mejores resultados para la empresa misma. Desde el punto de vista financiero, valga la redundancia, la Figura 2.2 muestra la relación entre los departamentos de Finanzas y el resto de la empresa.



*Figura 2.2. Relación entre Finanzas y el resto de la empresa.*

Una vez mencionados los departamentos de Finanzas, se describirá el procedimiento de intercambio de información dentro del área.

### **2.1.3. Operatividad**

Dentro del Área de Finanzas existe un procedimiento para la consulta de información, esta información se encuentra manejada por el encargado del archivo de cada departamento y es controlada a través de oficios o requerimientos escritos, así como de documentos de entrega del material consultado. El procedimiento se muestra y describe en la Figura 2.3.

## **2.2. Identificación del problema**

Después de describir el proceso de operación del Área de Finanzas, se muestran los principales problemas detectados:

- No se garantiza la confidencialidad de documentos.
- El manejo administrativo de los documentos no es el óptimo y tampoco es confiable.
- No existe una organización de la información contenida en los documentos, lo cual provoca una lentitud en el proceso de solución de problemas a los usuarios.
- Se refleja un elevado costo por procedimientos de operación no automatizados, tales como papeleo.


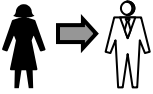





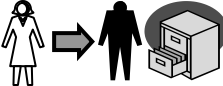
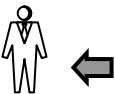


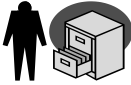
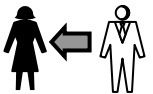




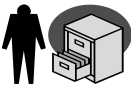
Depto. A		Depto. B	
			1. El usuario del Departamento A requiere consultar información de archivo del Departamento B.
			2. El usuario del Departamento A solicita la información a su jefe mediante un requerimiento escrito.
			3. El jefe del Departamento A firma la solicitud y la entrega al jefe del Departamento B.
			4. El jefe del departamento B autoriza la solicitud mediante su firma y la entrega al encargado de archivo de su departamento.
	 		5. El encargado de archivo del departamento B localiza la información y la entrega al jefe del departamento A mediante oficio de recepción de información para consulta.
			6. El jefe del departamento A entrega la información a su subordinado.
	 		7. Una vez consultada la información, el empleado del departamento A entrega la información al archivo del departamento B y recaba firma y sello para asegurar la entrega.

Figura 2.3. Solicitud de consulta de información.

A partir de la problemática definida, se muestra la necesidad de optimizar tiempos en la consulta de información, manteniendo una amplia seguridad y confidencialidad en dicha información. En la actualidad no se ha logrado concentrar dicha información en un sistema automatizado, que mantenga un estricto control de los accesos a la información financiera las 24 horas del día, los 365 días del año.

### 2.2.1. Requerimientos del usuario

Una vez identificado el problema, mencionaremos los requerimientos del usuario con relación al sistema:

- Se debe asegurar la confidencialidad de los documentos entre los diferentes departamentos, verificando que los documentos confidenciales sean vistos solamente por la persona a quien van dirigidos.
- Obtener una disminución en los tiempos y procesamiento de la información al automatizarlos.
- Debe existir un control de seguridad en los accesos a la información.
- El sistema debe ser amigable y contener una ayuda en línea durante la navegación en el mismo.
- Que se minimicen los costos de realización utilizando *hardware* y *software* que posee la empresa.
- Que se pueda acceder al sistema desde cualquier equipo que cumpla requerimientos mínimos, que permitan ejecutar y transferir archivos sin problemas de horarios y zona geográfica.

### 2.3. Opciones de solución

Una vez revisados los requerimientos del usuario, haciendo un análisis del flujo y tamaño de la información, analizaremos las diferentes opciones de *software* para poder llevar a cabo la implantación del sistema. Empezaremos con los manejadores de bases de datos, y así veremos en qué sistemas operativos y plataformas pueden ser ejecutados; también las herramientas que se pueden utilizar para manipular los datos y para el *front-end* del sistema.

Las opciones de manejadores de bases de datos presentadas son:

- SQL Server 2000
- Informix Online 7.2
- Oracle 9i

A continuación se mencionan algunas de las características de estos manejadores de bases de datos.

### 2.3.1. Manejadores de Bases de Datos

#### SQL Server 2000

Las principales características de este manejador son:

- Escalabilidad: se adapta a las necesidades de la empresa, soportando desde unos pocos usuarios a varios miles.
- Gestión: cuenta con una interfaz gráfica que reduce la complejidad de las tareas de administración y gestión de la base de datos.
- Orientado al desarrollo: Visual Basic, Visual C++, Visual J++, Visual Interdev y muchas otras herramientas son compatibles con Microsoft SQL Server.
- Plataforma de desarrollo fácil y abierta: integrada con tecnologías de Internet como *ActiveX*, Microsoft Transaction Server y herramientas de gestión y desarrollo para Internet como FrontPage, Microsoft Office y Visual Interdev.
- Diseñado para Internet: es un gestor de bases de datos que contiene de forma integrada la posibilidad de generar contenido HTML de forma automática.
- Maneja datos distribuidos, puede hacer llamadas a procedimientos remotos servidor a servidor (procedimientos almacenados remotos).
- Maneja un sistema de dispositivos espejo con recuperación automático para tolerancia a fallos de dispositivos.
- En cuanto a programación, cuenta con *triggers*, procedimientos almacenados, disparador de eventos antes y después de conexiones.

#### Informix Online 7.2

Entre las características principales de este manejador están:

- Opera bajo el sistema operativo UNIX, proporciona alto desempeño y alta disponibilidad de información. Incluye características como discos espejo y mecanismos ágiles de recuperación de información. Es por ello que con frecuencia se utiliza en aplicaciones de misión crítica.
- Puede trabajar bajo la plataforma cliente/servidor mediante el producto de conectividad Informix Net. Éste soporta los protocolos TCP/IP y *StartGroup*, y una variedad de computadoras y *hardware* de comunicación de datos como *Ethernet*, *Token Ring* y *StarLan*.
- Soporta bases de datos distribuidas mediante el producto de conectividad Informix Star y el manejador de base de datos *OnLine*. Mediante estos productos se puede consultar y modificar varias bases de datos Informix, en diferentes máquinas, como si fuera una sola base de datos centralizada.
- Cuenta con un conjunto de tablas para registrar el diccionario de datos. En el caso particular del manejador de bases de datos *OnLine* se tiene la posibilidad de guardar en forma histórica las modificaciones hechas al diccionario de datos.



## Oracle 9i

Algunas características de este manejador son:

- Se ejecuta sobre *hardware*, *software* y redes de cualquier otra base de datos relacional, lo que lo hace que sea un producto muy portable y que sea soportado por más del 70% de los *back-ends* y *front-ends* que hay en el mercado.
- Soporta el manejador de *triggers* y *store procedures*.
- Cuenta con opciones de seguridad para el manejo de la base de datos. Maneja los esquemas de discos espejos para poder continuar con los procesos en caso de que un disco se dañe.
- Cuenta con un *log* (registro) de transacciones, en donde se almacenan las modificaciones que se hagan a la base de datos.
- Oracle está diseñado para proveer facilidades de almacenamiento y recuperación de información de diferentes formatos.
- Para aumentar la velocidad de procesamiento y la consistencia de los datos, implementa el uso de grandes áreas de memoria para almacenar datos, además de que utiliza transacciones, lo que hace que garantice la integridad de los datos, llevando un control de los registros que han sido modificados dentro de una transacción a fin de poder deshacer todos los cambios en caso de que por algún motivo el proceso no se termine completamente.
- Oracle es soportado por un gran número de lenguajes, con lo cual se pueden desarrollar de forma rápida aplicaciones que impliquen un gran volumen de información.
- Existen en el mercado diversas herramientas para el manejo de Oracle, que facilitan las tareas de desarrollo de aplicaciones; por ejemplo, existen herramientas de modelado de base de datos que generan completamente el código para la construcción de la base de datos.

La comparación entre los diferentes manejadores de bases de datos se puede ver en la tabla 2.1.

### 2.3.2. Sistemas Operativos

Algunos de los sistemas operativos en que son ejecutados los manejadores de bases de datos descritos son:

- Windows NT
- UNIX
- OS/2

Las características principales de estos sistemas son:

#### Windows NT

Existe el Windows NT Server y el Windows NT Workstation.

## Windows NT Server

Entre las características principales de este sistema están:

- 256 sesiones RAS (*Remote Access Service*, Servicio de Acceso Remoto).
- Desempeño del servidor: ajuste del desempeño para aplicaciones, archivos e impresión. La versión liberada de Windows NT Server soporta hasta cuatro multiprocesadores en un ambiente de multiprocesamiento simétrico.
- Tolerancia a fallos: soporta tecnología RAID (*Redundant Array of Inexpensive Disks*, Arreglo Redundante de Discos Baratos) para protección de datos.
- Microsoft IIS( *Internet Information Server*, Servidor de Información en Internet): integración de IIS con Windows NT Server significa que la instalación y administración de un *Web Server* es simplemente otra parte del sistema operativo. Con IIS 2.0 o superior, es posible administrar en forma remota.

	<b>Microsoft SQL Server 2000</b>	<b>Informix Online7.2</b>	<b>Oracle 9i</b>
Opciones violación de integridad referencial	Sólo restrictiva	Restrictiva excepto el borrado en cascada	Restrictiva excepto el borrado en cascada
Vistas actualizables (con opción de verificación)	Sí	Sí	Sí
Tipos de datos definidos por el usuario	Sí	No	Sí
Estructuras de índices	<i>Clustered</i>	<i>B + tree y clustered</i>	<i>B-tree, bitmap y hash</i>
Estructuras de tablas	Sin selección	Sin selección	<i>Heap y clustered</i>
Facilidades de optimización "tunning"	<i>Fill factors</i>	<i>Extents fragmentación de tablas o round robin</i>	Distribución de índices de tablas
Nivel de <i>triggers</i>	Basado en conjuntos de registros	Basado en registro y basado en conjuntos de registros	Basado en registro y basado en conjuntos de registros
Ejecución del <i>trigger</i>	Después de la operación que disparó el <i>trigger</i>	Antes y después de la operación que disparó el <i>trigger</i>	Antes y después de la operación que disparó el <i>trigger</i>
Anidamiento del <i>trigger</i>	Sí	Sí	Sí
Lenguaje empleado en <i>stored procedure. Transact-sql</i>	<i>Transact-sql</i>	<i>Sql</i>	<i>Pl/sql</i>

*Tabla 2.1. Tabla de comparación entre los diferentes manejadores de bases de datos.*

- Ayudantes administrativos: incluye ayudantes que favorecen el desempeño de tareas comunes.

- Servicios adicionales de red: proporciona servicios adicionales de red, incluye ruteo multiprotocolo, servidor DNS (*Domain Name System*, Sistema de Nombres de Dominio) y WINS (*Windows Internet Name Service*, Servicio de Nombres de Internet en Windows).

### Windows NT Workstation

Entre las características principales de este sistema están:

- Desempeño de escritorio: soporta multitarea *preemptive* para todas las aplicaciones. Soporta múltiples microprocesadores para un desempeño real en multitareas.
- Perfiles de *hardware*: crea y mantiene una lista de configuraciones de *hardware* para conocer las necesidades de computadoras específicas.
- Microsoft Internet Explorer: provee un navegador que es rápido y simple de usar y compatible con los estándares existentes.
- Windows Messaging: recibe y mantiene correo electrónico, incluyendo archivos y objetos creados en otras aplicaciones.
- Servicios *Peer Web*: provee un servidor personal de *Web*, optimizado para ejecutarse sobre Windows NT Workstation 2000.
- Seguridad: provee seguridad local por archivos, carpetas, impresoras y otros recursos. Los usuarios pueden ser autenticados tanto por la computadora local o un controlador de dominio en lugar de acceder cualquier recurso en la computadora o red.
- Estabilidad del sistema operativo: soporta cada aplicación en su propio espacio de direcciones de memoria. Las aplicaciones con fallas de funcionamiento no afectarán otras aplicaciones o al sistema operativo.

### **Sistema operativo UNIX**

Algunas de las características principales de este sistema son:

- Su componente principal es el *kernel* que se encarga de asignar tareas y manejar el almacenamiento de datos. El usuario rara vez opera directamente con el *kernel*, que es la parte residente en memoria del sistema operativo.
- Otro componente importante es el *shell*, ésta es la utilidad que procesa las peticiones de los usuarios. Cuando alguien teclea un comando en la terminal, el *shell* interpreta el comando y llama el programa deseado. También es un lenguaje de programación de alto nivel que puede utilizarse en la combinación de programas de utilidad para crear aplicaciones completas.
- El *shell* puede soportar múltiples usuarios, múltiples tareas, y múltiples interfaces para sí mismo. Los dos *shells* más populares son el BourneShell (System V) y el Cshell (BSD Unix), debido a que usuarios diferentes pueden usar diferentes *shells* al mismo tiempo, entonces el sistema puede aparecer diferente para usuarios diferentes. Existe otro *shell* conocido como KornShell, así llamado en honor de su diseñador, que es muy popular entre los programadores.

- UNIX incluye una gran variedad de programas de utilidad que pueden ser fácilmente adaptados para realizar tareas específicas. Estas utilerías son flexibles, adaptables, portables y modulares, y pueden ser usadas junto con filtros y redireccionamientos para hacerlos más poderosos.
- Es un sistema multiusuario, ejecutando en cada uno de ellos un conjunto diferente de programas.
- Es un sistema multitareas, permite la realización de más de una tarea a la vez. Pueden ejecutarse varias tareas en su interior, mientras se presta toda la atención al programa desplegado en la terminal.
- Además, UNIX permite proteger los archivos del usuario contra el acceso por parte de otros usuarios.
- Los dispositivos (como una impresora o una terminal) y los archivos en disco son considerados como archivos por UNIX. Cuando se da una instrucción al UNIX puede indicársele que envíe el resultado a cualquiera de los diversos dispositivos o archivos. Esta desviación recibe el nombre de redireccionamiento de la salida.
- En forma similar, la entrada de un programa puede redireccionarse para que venga de un archivo en disco. En UNIX, la entrada y la salida son independientes del dispositivo, pueden redireccionarse hacia o desde cualquier dispositivo apropiado.
- Comunicación entre procesos: UNIX permite el uso de conductos y filtros en la línea de comandos. Un conducto (*pipe*) redirige la salida de un programa para que se convierta en entrada de otro. Un filtro es un programa elaborado para procesar un flujo de datos de entrada y producir otro de datos de salida. Los conductos y filtros suelen usarse para unir utilerías y realizar alguna tarea específica.

## Sistema operativo OS/2

Algunas de las características principales de este sistema son:

- OS/2 fue diseñado para ejecutar programas DOS al tiempo que proporciona capacidades extendidas, incluyendo multitarea y redes.
- Cuando en OS/2 se ejecuta el software de gestión LAN, se pueden aprovechar las características de seguridad similares a las proporcionadas por UNIX.
- Cuenta con un *kit* completo de multimedia y el *Bonus Pak*, un *kit* de aplicaciones que permite ponerse a trabajar con el ordenador, contiene elementos como un *kit* de conexión a Internet completo, el paquete integrado IBM Works basada en *OpenDoc* (formado por un procesador de textos, hoja de cálculo, base de datos y gráficos de empresa, junto con el PIM, que añade más funcionalidades aprovechando las capacidades *drag&drop* del WPSHELL), *soft* de terminal, *soft* de captura y tratamiento de video, etc.
- Soporta casi cualquier dispositivo existente en el mercado: *Cd-roms*, impresoras, tarjetas de sonido, soporte PCMCIA, tarjetas de video, tarjetas de captura de video, tarjetas SCSI, etc.

- Puede ejecutar programas DOS, OS/2 16bits, OS/2 32 bits, Windows 3.x (incluida además el API Win32s, con lo que se podrían ejecutar incluso programas Windows de 32bits).
- Cuenta con librerías *OpenDoc* (compatibles con OLE 2.0, pero más potentes).
- Librerías *OpenGL*, que permiten aprovechar las capacidades 3D de las tarjetas que soporten este estándar.
- API de desarrollo Open32, que permiten recompilar con suma facilidad las aplicaciones escritas para Windows 95 y Windows NT, de forma que aprovechen al máximo los recursos de OS/2.
- Cuenta con un extenso soporte de conectividad, lo que le convierte en el cliente de red universal, pudiendo conectarse a casi cualquier servidor (no sólo OS/2 Warp Server, sino Windows NT Server, Novell, etc.).
- La característica estrella de cara al marketing: el *VoiceType*, se trata de un software reconocedor de voz, capaz de funcionar con cualquier tarjeta de sonido, y que permite al usuario trabajar exclusivamente mediante el dictado de comandos. Este sistema, al contrario de otros disponibles hasta el momento, reconoce el habla de forma continua, de modo que no sólo se puede usar para navegar por el escritorio y controlar programas, sino que sirve para dictar cualquier tipo de texto, como artículos, cartas, etc., sin tocar una sola tecla. Se trata, por tanto, de un avance de los que serán, sin duda, los sistemas operativos del futuro.

Los diferentes sistemas operativos que trabajan con los manejadores de bases de datos se pueden ver en la tabla 2.2.

	<b>Microsoft SQL Server</b>	<b>Informix Online 7.2</b>	<b>Oracle 9i</b>
Windows NT	Sí	No	Sí
UNIX	No	Sí	Sí
OS/2	Sí	No	Sí

Tabla 2.2. Sistemas operativos en los diferentes manejadores de bases de datos.

### 2.3.3. Herramientas para la creación de Páginas Web

Algunas opciones de herramientas para la creación de páginas Web interactivas que se utilizarán en el *front-end* para la implantación de los sistemas son los siguientes:

- PHP (*Hypertext Preprocessor*, *Preprocesador de hipertexto*)
- ASP (*Active Server Pages*, *Servidor de Páginas Activas*).
- JSP (*Java Server Pages*, *Servidor de Páginas Java*).

### **PHP (*Hypertext Preprocessor*, *Procesador de Hipertexto*)**

Algunas de las características de éste son:

- Se puede procesar la información de formularios, generar páginas con contenidos dinámicos, o mandar y recibir *cookies*.
- Puede ser utilizado en cualquiera de los principales sistemas operativos del mercado, incluyendo Linux, muchas variantes Unix (incluido HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS y probablemente alguno más.
- Soporta la mayoría de servidores *Web* de hoy en día, incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape, iPlanet, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd y muchos otros. PHP tiene módulos disponibles para la mayoría de los servidores, para aquellos otros que soporten el estándar CGI, PHP puede usarse como procesador CGI.
- Se tiene la libertad de escoger el sistema operativo y el servidor al gusto del usuario.
- Se tiene la posibilidad de usar programación de procedimientos ó programación orientada a objetos. Aunque no todas las características estándares de la programación orientada a objetos están implementadas en la versión actual de PHP.
- Con PHP no se está limitado a resultados en HTML. Entre las habilidades de PHP se incluyen, creación de imágenes, archivos PDF y películas Flash (usando libswf y Ming) sobre la marcha. También se pueden presentar otros resultados, como XHTML y archivos XML. PHP tiene la opción de auto generar estos archivos y grabarlos en el sistema de archivos en vez de presentarlos en la pantalla.
- Quizás la característica más potente y destacable de PHP es su soporte para una gran cantidad de bases de datos. Escribir una interfaz vía *Web* para una base de datos es una tarea simple con PHP. Algunas bases de datos que son soportadas actualmente por PHP son: IBM DB2, Informix, FrontBase, mSQL, MySQL, Oracle (OCI7 and OCI8), Sybase, Unix dbm.

### **ASP (*Active Server Pages*, *Servidor de Páginas Activas*)**

Algunas de las características de ASP son:

- También conocido como *Web Forms*, permite crear páginas *Web* programables como parte de una aplicación *Web* global. *Web Forms* simplifica el desarrollo de las aplicaciones *Web*.
- Proporcionan un modelo de programación basado en eventos en el servidor, parecido al desarrollo basado en formularios que se encuentra en herramientas

de desarrollo basadas en Microsoft Win32, tales como el sistema de desarrollo Microsoft Visual Basic.

- Permiten una completa separación entre el formato HTML y la lógica de la aplicación. La lógica, o código asociado a la página, se compila y proporciona un rendimiento mucho mejor. Además, este código se puede escribir en cualquier lenguaje de Microsoft, por lo que permite al usuario aprovechar mejor sus habilidades.
- Ofrecen una experiencia enriquecida de tiempo de diseño, proporciona una experiencia de desarrollo rápido de aplicaciones (RAD, *Rapid Application Development*, Desarrollador de Aplicaciones Rápidas) para crear y administrar formularios *Web*.
- Es compatible con un enriquecido y útil conjunto de controles y componentes que ofrecen un modelo coherente de objeto de tipo seguro. Además, el marco de trabajo se presta de manera natural a la extensibilidad mediante los componentes personalizados y de otros fabricantes.

### **JSP (Java Server Pages, Servidor de Páginas Java)**

Algunas de las características de JSP son:

- Efectúa algunas simplificaciones al lenguaje *Java*.
- Es un lenguaje interpretado completamente, intercalado directamente dentro de las páginas HTML, no compilado en pseudocódigo, como *Java*.
- Un programa en JSP se basa en el modelo orientado a objetos, pero en objetos extensibles incorporados en el navegador, sin clases ni herencia.
- Soporta pocos tipos de datos, como los valores numéricos, booleanos y de cadena.
- Las sentencias de JSP pueden reconocer y responder a eventos de usuarios como las pulsaciones de ratón, la entrada de formularios y la navegación por páginas.

Una vez presentados el planteamiento del problema y la propuesta de solución, continuaremos con el diseño del mismo, el cual contempla varios diagramas a realizar, aplicando la metodología elegida.

# CAPÍTULO III

## DISEÑO DEL SISTEMA

En este capítulo se presentará el diseño del sistema, para ello haremos uso de los diagramas caso de uso, dinámico, objeto y de eventos. Además presentaremos el diccionario de datos, el diagrama entidad relación y la normalización.

Con base en la metodología presentada en el capítulo I procederemos a diseñar el sistema.

### 3.1. Aplicación de la metodología

Hasta el momento se ha explicado la problemática del sistema pero no se ha enfocado la metodología a emplear para el desarrollo del mismo. Sabemos que no existe una metodología que nos garantice en un 100% el correcto funcionamiento del mismo, pero existen formas para generar un sistema que cumpla con los requerimientos.

Se analizaron métodos en el capítulo I en los que destaca OOD, OMT, Metodología propia de la empresa y cada uno tiene diferentes vistas o mecanismos para el desarrollo de un sistema.

Con base en lo anterior, no existe algún método que nos certifique o garantice que sus pasos son los más adecuados para el desarrollo de cada sistema, por ello, para este sistema en particular se hizo una combinación de métodos, misma que se explicará conforme avanza el capítulo.

Para llegar a un acuerdo sobre los requisitos, condiciones y posibilidades que debe cumplir el sistema es necesario hacer un modelo de casos de uso. Este modelo



describe lo que hace el sistema para cada tipo de usuario; contiene actores, casos de uso y sus relaciones.

El modelo de casos de uso sirve como acuerdo entre clientes y desarrolladores, y proporciona la entrada fundamental para el diseño y las pruebas.

Cada tipo de usuario se representa mediante uno o más actores, por lo que es muy importante identificarlos para delimitar un sistema de su entorno, y así esbozar quién y qué actores interactuarán con el sistema y qué funcionalidad (casos de uso) se espera del sistema. Para esto, requerimos de entradas del cliente (ICA) y de los usuarios que utilizarán el sistema.

Al analizar con ICA los requerimientos del sistema se definió una clasificación de tres niveles de usuario: el Usuario, el Administrador y el Superusuario y sus funciones respectivas en el sistema; y de dos tipos de archivos: privados y públicos.

El Usuario podrá acceder a aquellos archivos privados a los cuales se le haya otorgado permiso en la administración de archivos y a todos los archivos públicos, ligas y podrá ver el *ticker*, también podrá solicitar alta, baja y actualización de archivos y ligas. El Administrador puede acceder a la parte de administración de archivos exclusivamente del departamento al que pertenece para crear, borrar o modificar carpetas, archivos y ligas, también puede ver el *ticker*. Y el Superusuario tiene permisos de acceso a todas las pantallas de administración. Puede administrar los archivos de cualquiera de los seis departamentos, para crear, borrar o modificar carpetas, archivos y ligas. Es el responsable de administrar el catálogo de los Usuarios y el texto contenido en la marquesina o *ticker*.

Los archivos privados son aquellos a los que se les asigna un usuario para que pueda ser consultado por él y sólo él; los archivos públicos son los que pueden ser consultados por cualquier usuario sin necesidad de permisos.

Con base en los tipos de usuario definidos como los actores del sistema, procedemos a definir los casos de uso de cada uno de ellos, es decir, la descripción de lo que cada actor puede hacer en el sistema.

### **3.1.1. Diagrama de casos de uso**

#### *Caso de uso Usuario*

El nombre de un caso de uso debe definir cuál es el objetivo de la interacción entre el actor y el sistema. En el diagrama de la Figura 3.1 tenemos los casos de uso: consultar archivos públicos, consultar archivos privados, consultar ligas, ver *ticker*, solicitar alta de archivos, solicitar alta de ligas, solicitar baja de archivos, solicitar baja de ligas, solicitar cambios de archivos y solicitar cambios de ligas.

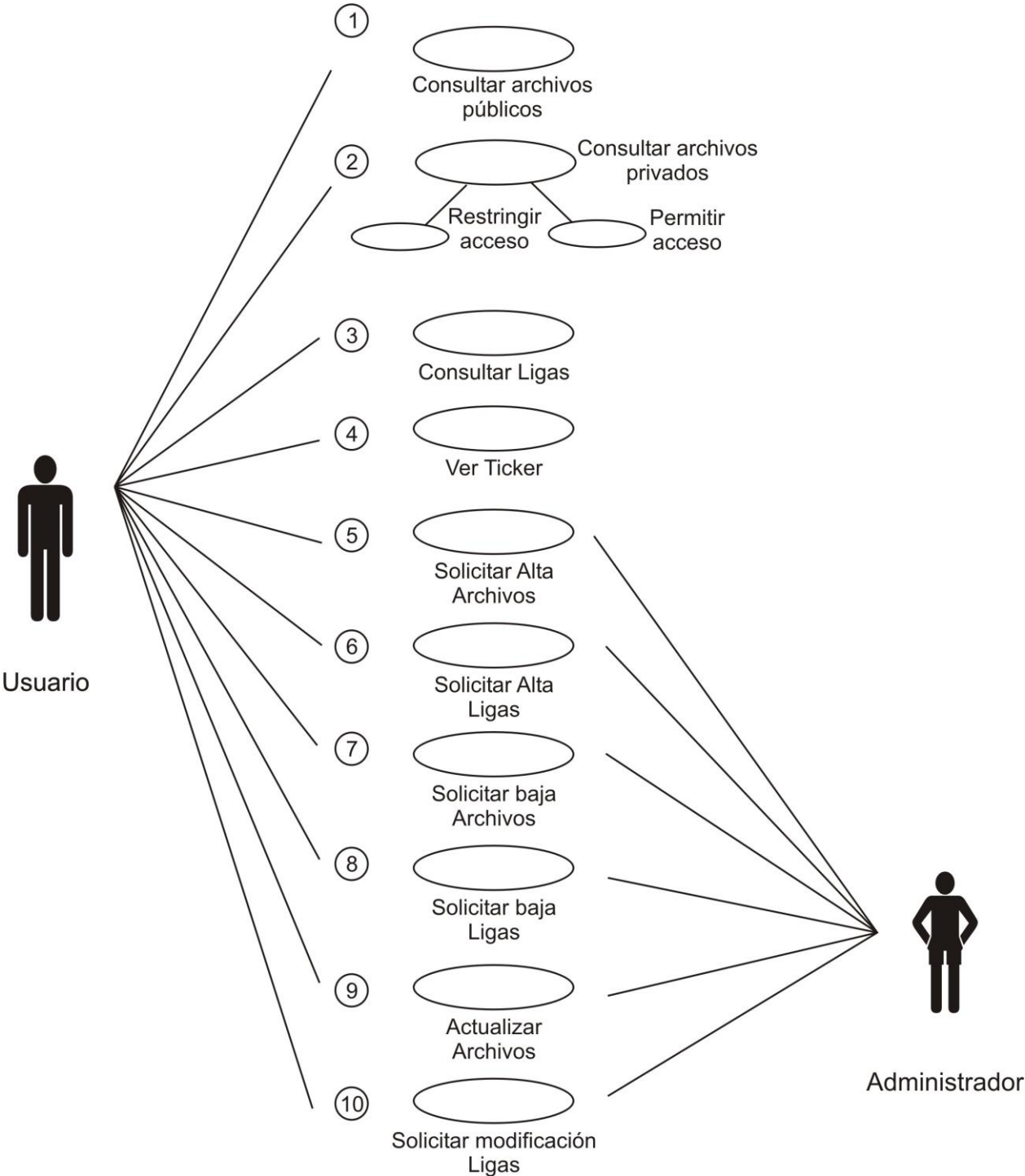


Figura 3.1. Diagrama de Casos de uso del actor Usuario.

Una vez que identificamos los casos de uso del actor usuario describimos paso a paso con algunas frases que resumen las acciones que el sistema necesita hacer cuando interactúa con el actor.

1. Caso de uso Consultar archivos públicos:

- El Usuario consulta la lista de archivos públicos.
- El Usuario accede a los archivos públicos elegidos.
- El sistema permite el acceso a los archivos públicos solicitados.

2. Caso de uso Consultar archivos privados:

- El Usuario consulta la lista de archivos privados.
- El Usuario solicita el acceso a los archivos privados.
- El sistema permite o restringe el acceso a los archivos privados.

3. Caso de uso Consultar ligas:

- El Usuario consulta la lista de ligas.
- El Usuario accede a la liga elegida.
- El sistema permite el acceso a las ligas.

4. Caso de uso Ver *ticker*:

- El sistema presenta información de la empresa de forma aleatoria y permite su acceso permanente a todos los usuarios.

5. Caso de uso Solicitar alta de archivos:

- El Usuario identifica el archivo que desea dar de alta en el sistema.
- El Usuario solicita al Administrador el alta del archivo proporcionando los permisos correspondientes de acceso al archivo.
- El administrador da de alta en el sistema el archivo solicitado incluyendo los permisos correspondientes.

6. Caso de estudio Solicitar alta de ligas:

- El Usuario identifica la liga que desea dar de alta en el sistema.
- El Usuario solicita al Administrador de alta la liga.
- El Administrador da de alta en el sistema la liga.

7. Caso de uso Solicitar baja de archivos:

- El Usuario identifica el archivo que desea dar de baja en el sistema.
- El Usuario solicita al Administrador la baja del archivo.
- El Administrador da de baja en el sistema el archivo solicitado.
- El sistema borra el registro del archivo.

8. Caso de uso Solicitar baja de ligas:

- El Usuario identifica la liga que desea dar de baja en el sistema.
- El Usuario solicita al Administrador de baja la liga.
- El Administrador da de baja en el sistema la liga.

- El sistema borra la liga de los registros.

9. Caso de uso Solicitar cambios de archivos:

- El Usuario identifica el archivo que desea modificar en el sistema.
- El Usuario solicita al Administrador modificar el archivo.
- El Administrador modifica en el sistema el archivo solicitado.

10. Caso de uso Solicitar modificar ligas:

- El Usuario identifica la liga que desea modificar en el sistema.
- El Usuario solicita al Administrador modificar la liga.
- El Administrador modifica la liga en el sistema.

*Caso de Uso Administrador*

Otro actor que encontramos en el sistema es el Administrador, sus casos de uso se observan en la Figura 3.2.

Los casos de uso 1, 3 y 4 de la Figura 3.2. trabajan de la misma forma para el Usuario y el Administrador. En el caso de uso 2, el sistema no restringe el acceso al Administrador, porque cuenta con mayores privilegios.

5. Caso de uso Dar de alta archivos:

- El Administrador atiende la solicitud de alta del archivo por parte del Usuario.
- El Administrador da de alta en el sistema el archivo asignando los permisos de acuerdo a la solicitud.
- El sistema le permite al Usuario el acceso al archivo dado de alta.

6. Caso de uso Dar de alta ligas:

- El Administrador atiende la solicitud de alta de la liga por parte del usuario.
- El Administrador da de alta en el sistema la liga.
- El sistema le permite al Usuario el acceso a la liga dada de alta.

7. Caso de uso Dar de baja archivos:

- El Administrador atiende la solicitud de baja del archivo por parte del usuario.
- El Administrador da de baja en el sistema el archivo.
- El sistema borra el archivo.

8. Caso de uso Dar de baja ligas:

- El Administrador atiende la solicitud de baja de la liga por parte del usuario.
- El Administrador da de baja en el sistema la liga.
- El sistema le no permite al Usuario el acceso a la liga.

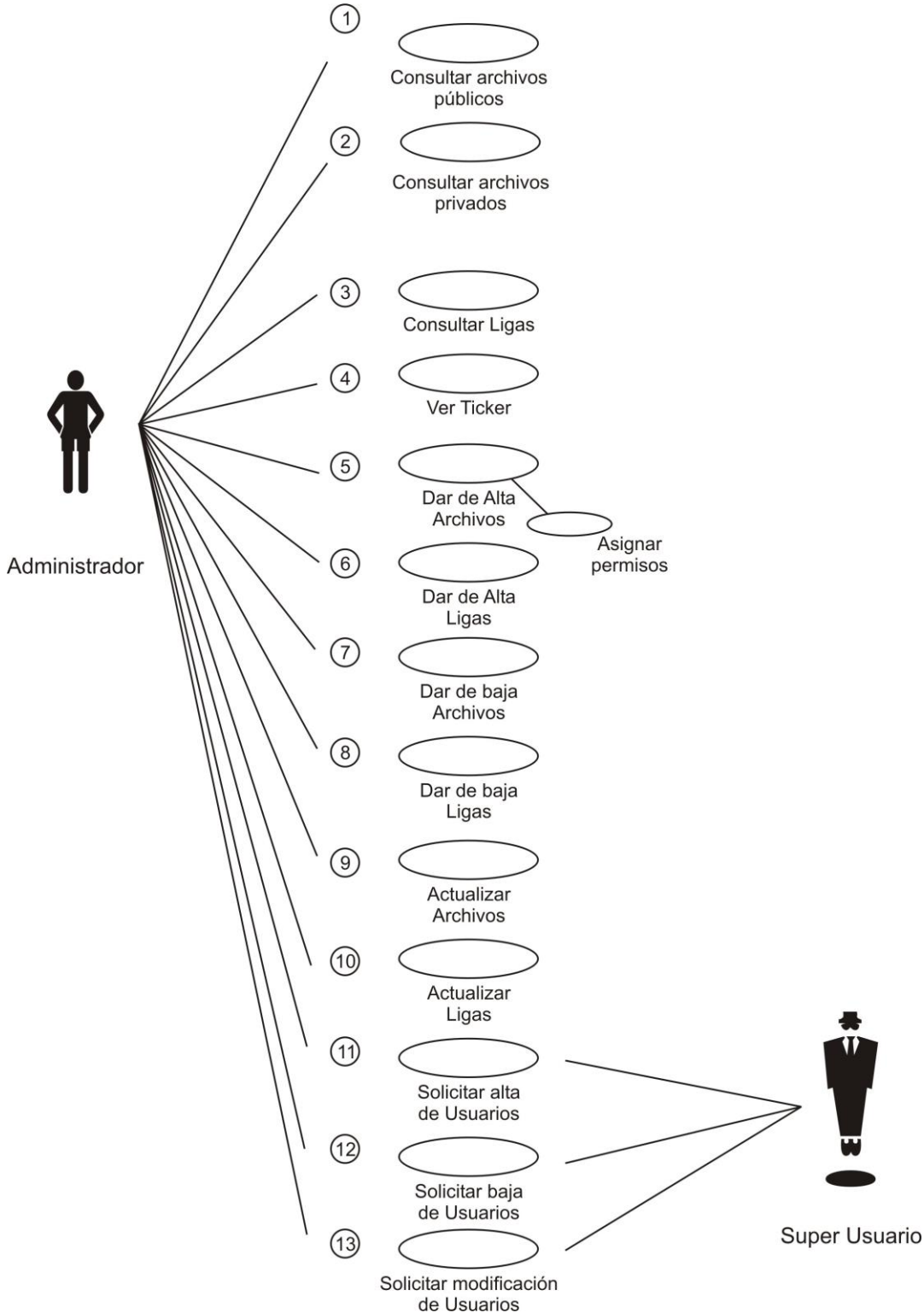


Figura 3.2. Diagrama de Casos de uso del actor Administrador.

9. Caso de uso Actualizar archivos:

- El Administrador atiende la solicitud de modificación del archivo por parte del usuario.
- El Administrador modifica en el sistema el archivo asignando los permisos de acuerdo a la solicitud.
- El sistema le permite al Usuario el acceso al archivo modificado.

10. Caso de uso Actualizar ligas:

- El Administrador atiende la solicitud de modificación de la liga por parte del usuario.
- El Administrador modifica en el sistema la liga.
- El sistema le permite al Usuario el acceso a la liga modificada.

11. Caso de uso Solicitar alta de usuario:

- El Administrador solicita al Superusuario el alta de archivos proporcionando los datos correspondientes.

12. Caso de uso Solicitar actualización de usuario:

- El Administrador solicita al Superusuario la actualización de archivos proporcionando los datos correspondientes.

### *Caso de Uso Superusuario*

El último actor que encontramos en el sistema es el Superusuario y sus casos de uso se presentan en la Figura 3.3.

Los casos de uso 1 al 10 trabajan de la misma forma para el Administrador que para el Superusuario, con excepción de que el Superusuario no requiere validación de acceso a los archivos.

11. Caso de uso Dar de alta usuarios:

- El Superusuario atiende la solicitud de alta de un Usuario por parte del Administrador.
- El Superusuario da de alta el usuario asignando la categoría de usuario y el área correspondientes, como datos principales.
- El sistema da de alta al nuevo usuario en la base de datos.

12. Caso de uso Dar de baja usuarios:

- El Superusuario atiende la solicitud de baja de un usuario por parte del Administrador.
- El Superusuario da de baja el usuario en el sistema.
- El sistema da de baja al usuario en la base de datos.

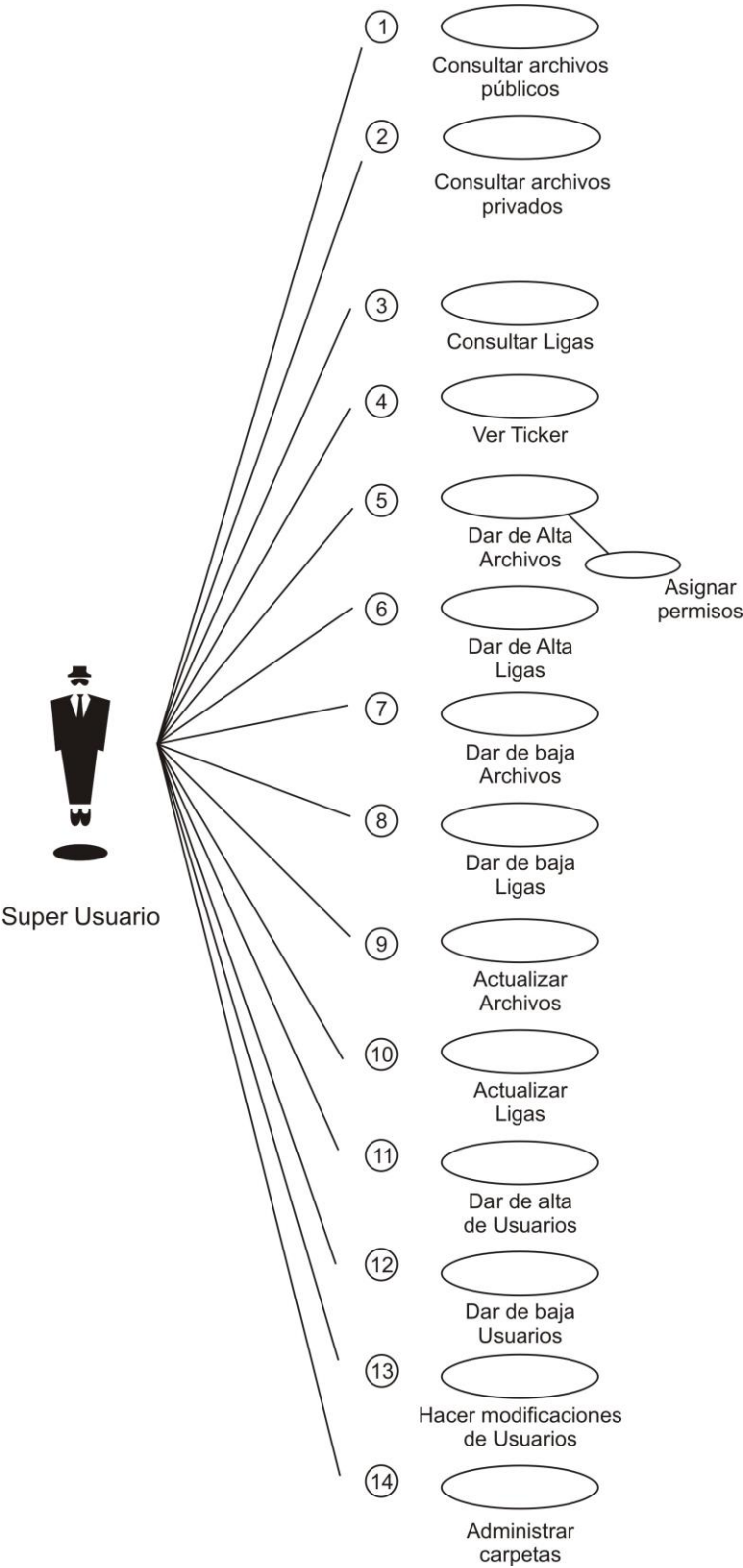


Figura 3.3. Diagrama de casos de uso del actor Superusuario.

13. Caso de uso Modificar usuarios:

- El Superusuario atiende la solicitud de modificación de un usuario por parte del Administrador.
- El Superusuario modifica el usuario asignando la categoría de usuario y los departamentos correspondientes, como datos principales.
- El sistema modifica las propiedades del usuario en la base de datos.

14. Caso de uso Administrar carpetas:

- El superusuario administra las carpetas que contienen los archivos públicos y privados.

Con lo anterior hemos visto cada una de las actividades que pueden desempeñar los actores en el sistema, en los diferentes casos de uso, el siguiente paso es representarlos en un diagrama de clases con sus atributos, funciones y características.

### 3.1.2. Diagrama de Clases

Una clase es la descripción de un concepto del dominio de la aplicación o de la solución de la aplicación, las clases son el centro alrededor del cual se organiza la vista de clases; otros elementos como los atributos y operaciones, pertenecen o se unen a las clases.

Las clases se dibujan como rectángulos. Las listas de atributos y de operaciones se muestran en compartimentos separados. Los compartimentos pueden ser suprimidos cuando no es necesario un detalle completo. Una clase puede aparecer en varios diagramas.

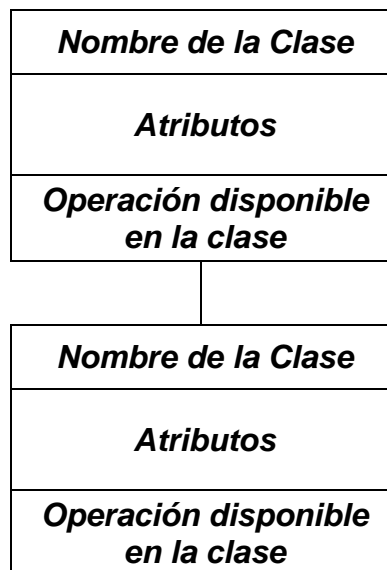


Figura 3.4. Representación de las clases.



Las relaciones entre clases se dibujan como las líneas que conectan rectángulos de clases. Una muestra de la descripción del diagrama de clases es la Figura 3.4.

Ahora veremos las clases más importantes del sistema basado en la descripción anterior y también en los diagramas de casos de uso.

La primera clase identificada fue la de Usuario (Figura 3.5), esta clase abarca los tres tipos de usuarios definidos en los diagramas de casos de uso y que consideran al Usuario, Administrador y Superusuario. Cumpliendo con los atributos fundamentales para que los usuarios puedan existir en el sistema y las funciones que utilizan.

<b>Usuario</b>
<b>Nombre: cadena</b> <b>Apellido paterno: cadena</b> <b>Apellido materno: cadena</b> <b>Password: cadena</b> <b>email: cadena</b> <b>Area: cadena</b> <b>Extensión: cadena</b> <b>Fecha de alta: fecha</b> <b>Puesto: cadena</b>
<b>Agregar()</b> <b>Eliminar()</b> <b>Modificar()</b> <b>Búsqueda()</b>

Figura 3.5. Clase Usuario.

Contemplando la estrategia de control y permisos de acceso a los archivos se pensó en la entidad de Permisos (Figura 3.6).

<b>Permisos</b>
<b>Usuario: cadena</b> <b>Documento: cadena</b>
<b>Agregar()</b> <b>Eliminar()</b> <b>Modificar()</b>

Figura 3.6. Clase Permisos.

Obviamente se debe contar con un módulo en el que debemos controlar los archivos a consultar, este módulo se encuentra definido por la clase Archivos (Figura 3.7).

<b>Archivos</b>
<b>Nombre: cadena</b> <b>Ruta: cadena</b> <b>Identificador: cadena</b>
<b>Agregar()</b> <b>Eliminar()</b> <b>Modificar()</b> <b>Búsqueda()</b>

Figura 3.7. Clase Archivos.

La clase *Ticker* (Figura 3.8) surge de la idea de tener texto deslizante para proveer de noticias de última hora al usuario.

<b>Ticker</b>
<b>Texto: cadena</b> <b>Activo: bit</b> <b>Fecha: fecha</b>
<b>Agregar()</b> <b>Eliminar()</b> <b>Modificar()</b> <b>Búsqueda()</b> <b>Activar()</b>

Figura 3.8. Clase Ticker.

Se requiere de una clasificación de las personas que contarán con acceso y control administrativo de la información, para ello se crea la clase Tipo de usuario (Figura 3.9).

<b>Tipo de Usuario</b>
<b>Nombre de tipo: cadena</b> <b>Area: cadena</b>
<b>Agregar()</b> <b>Eliminar()</b> <b>Modificar()</b>

Figura 3.9. Clase tipo de usuario.

Para la clase Departamento (Figura 3.10) se considera importante la identificación clara del personal a cargo de cada uno de los departamentos del área.

<b>Departamento</b>
<b>Nombre: cadena</b> <b>Nombre gerente: cadena</b> <b>Número telefónico: cadena</b>
<b>Agregar()</b> <b>Eliminar()</b> <b>Modificar()</b>

Figura 3.10. Clase Departamento.

Se contempla también una clase Proyectos (Figura 3.11) que contendrá la información de apertura, donde se tendrá una mejor organización de los documentos.

<b>Proyectos</b>
<b>Nombre: cadena</b> <b>Fecha inicio: fecha</b> <b>Fecha final: fecha</b> <b>Costo: entero</b> <b>Presupuesto: entero</b>
<b>Agregar()</b> <b>Eliminar()</b> <b>Modificar()</b> <b>Búsqueda()</b>

Figura 3.11. Clase Proyectos.

Los movimientos periódicos se registrarán y serán dirigidos a la clase Reportes (Figura 3.12).

<b>Reportes</b>
<b>Nombre: cadena</b> <b>Fecha creación: fecha</b> <b>Fecha modificación: fecha</b> <b>Descripción: cadena</b>
<b>Agregar()</b> <b>Eliminar()</b> <b>Modificar()</b> <b>Búsqueda()</b> <b>Imprimir()</b>

Figura 3.12. Clase Reportes.

También se debe tener un módulo de registro de los usuarios que consultaron el sitio para tener un control de tiempo y límite de usuarios conectados, por lo que se consideró la clase Estadísticas (Figura 3.13).

<b><i>Estadísticas</i></b>
<b><i>Nombre: cadena</i></b> <b><i>Fecha creación: fecha</i></b> <b><i>Fecha modificación: fecha</i></b> <b><i>Accesos: entero</i></b> <b><i>Tiempo visita: fecha</i></b>
<b><i>Agregar()</i></b> <b><i>Eliminar()</i></b> <b><i>Modificar()</i></b> <b><i>Búsqueda()</i></b> <b><i>Imprimir()</i></b>

*Figura 3.13. Clase Estadísticas.*

La clase Región (Figura 3.14) contempla el lugar donde se realiza la apertura del proyecto.

<b><i>Región</i></b>
<b><i>País: cadena</i></b> <b><i>Estado: cadena</i></b> <b><i>Ciudad: cadena</i></b>
<b><i>Agregar()</i></b> <b><i>Eliminar()</i></b> <b><i>Modificar()</i></b>

*Figura 3.14. Clase Región.*

Una vez mostradas las principales clases, las englobaremos en un diagrama de clases indicando solamente los nombres de éstas, su jerarquía (Figura 3.15).

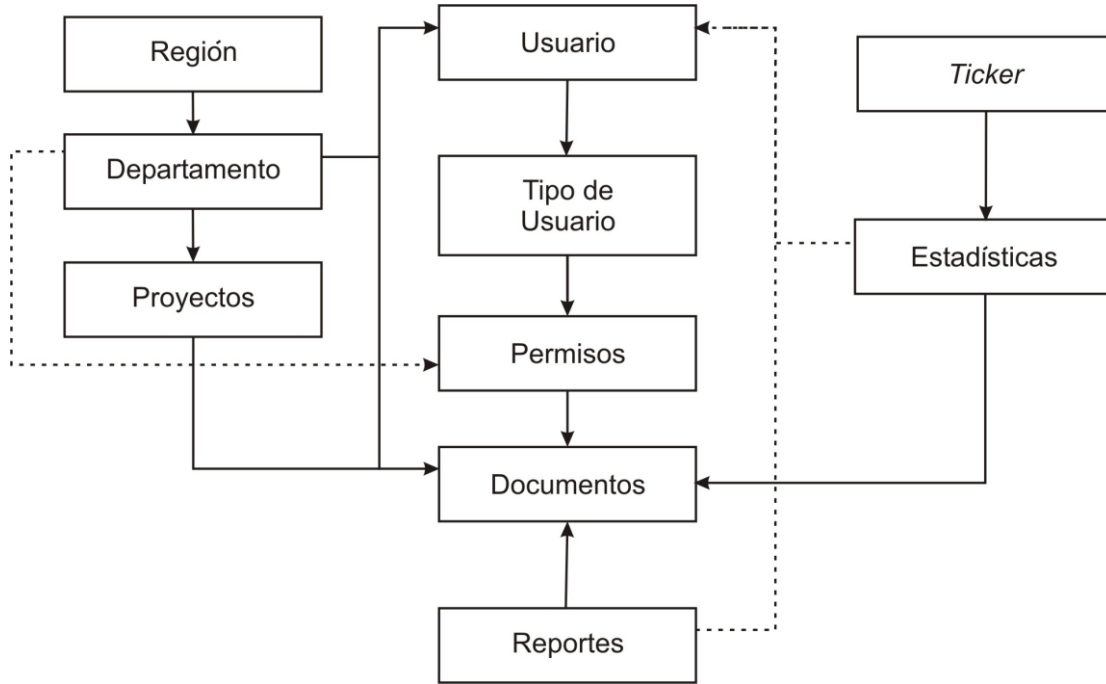


Figura 3.15. Diagrama de clases del sistema.

### 3.1.3. Diagrama de Objetos y de eventos

En este punto se deben identificar las clases candidatas, e identificar sus responsabilidades y colaboraciones. El modelado de clases-responsabilidades-colaboraciones aporta un medio sencillo de identificar y organizar las clases que resulten relevantes al sistema.

En la Tabla 3.1 hacemos una descripción de algunos objetos involucrados en el sistema que fueron el resultado del análisis hecho y así cumplir con los requerimientos del usuario.

Clase	Nombre	Tipo	Características	Responsabilidad	Colaboradores
1	Internet	Interacción	Persistencia	Abre y actualiza la base de datos, así como inicializa los objetos visitas y consultas. También forma la página principal del sistema ya que llama a las clases menú, funciones, consultas y presenta que conforman ésta.	Colabora con las clases menú, funciones, consultas y presenta para alcanzar su objetivo.

Tabla 3.1. Objetos involucrados en el sistema. (Continúa)

2	Menú	Interacción	Persistencia	Abre e inicializa la base de datos, obtiene el valor de los objetos visitas y consultas. Llama al objeto fecha, despliega la opción a la llamada de la clase Internet, despliega el total de los valores de los objetos visitas y consultas.	Colabora con la clase Internet, área para alcanzar su objetivo.
3	Funciones	Interacción	Persistencia	Despliega el objeto Imagen superior y el objeto icono de Inicio.	Tiene la característica de ser una clase atómica, es decir, no incluye otras clases.
4	Consultas	Interacción	Tangibilidad	Abre e inicializa la base de datos, se encarga de realizar la consulta a la base de datos según los requerimientos de los objetos datos de la clase.	Tiene la característica de ser una clase atómica, es decir, no incluye otras clases.
5	Área	Interacción	Persistencia	Abre e inicializa la base de datos, se encarga de realizar la consulta a la base de datos según los requerimientos de los objetos datos de la clase Subárea.	Colabora con las clases Menú, Subárea para poder cumplir con la consulta y objetivo.
6	Estadísticas	Interacción	Tangibilidad	Abre e inicializa la base de datos, se encarga de realizar la consulta a la base de datos según los requerimientos de los objetos datos de la clase.	Con colaboraciones de la clase Consultas y Área, almacenando todos las interacciones del sistema.
7	Submenú	Interacción	Persistencia	Abre e inicializa la base de datos, obtiene el valor de los objetos visitas y consultas. Llama al objeto fecha, despliega la opción a la llamada de la clase Internet, despliega el total de los valores de los objetos visitas y consultas.	Colabora con las clases Menú, área para poder cumplir con la consulta y objetivo.
8	Archivos	Interacción	Persistencia	Abre e inicializa la base de datos, se encarga de realizar la consulta a la base de datos según los requerimientos de los objetos datos clase Permisos	Colabora con las clases Submenú, permisos para poder cumplir con la consulta y objetivo.

Tabla 3.1. Objetos involucrados en el sistema. (Continúa)

9	Permisos	Interacción	Persistencia	Compara los datos que suministra las clases área y despliega un resultado de los eventos	
10	Presenta	Interacción	Persistencia	Despliega los objetos de presentación de resultado y despliega los objetos datos generales de la distribución de archivos.	Tiene la característica de ser una clase atómica, es decir, no incluye otras clases.

Tabla 3.1. Objetos involucrados en el sistema.

El diagrama de Objetos y Eventos de las clases involucradas en el sistema se presenta en la Figura 3.16.

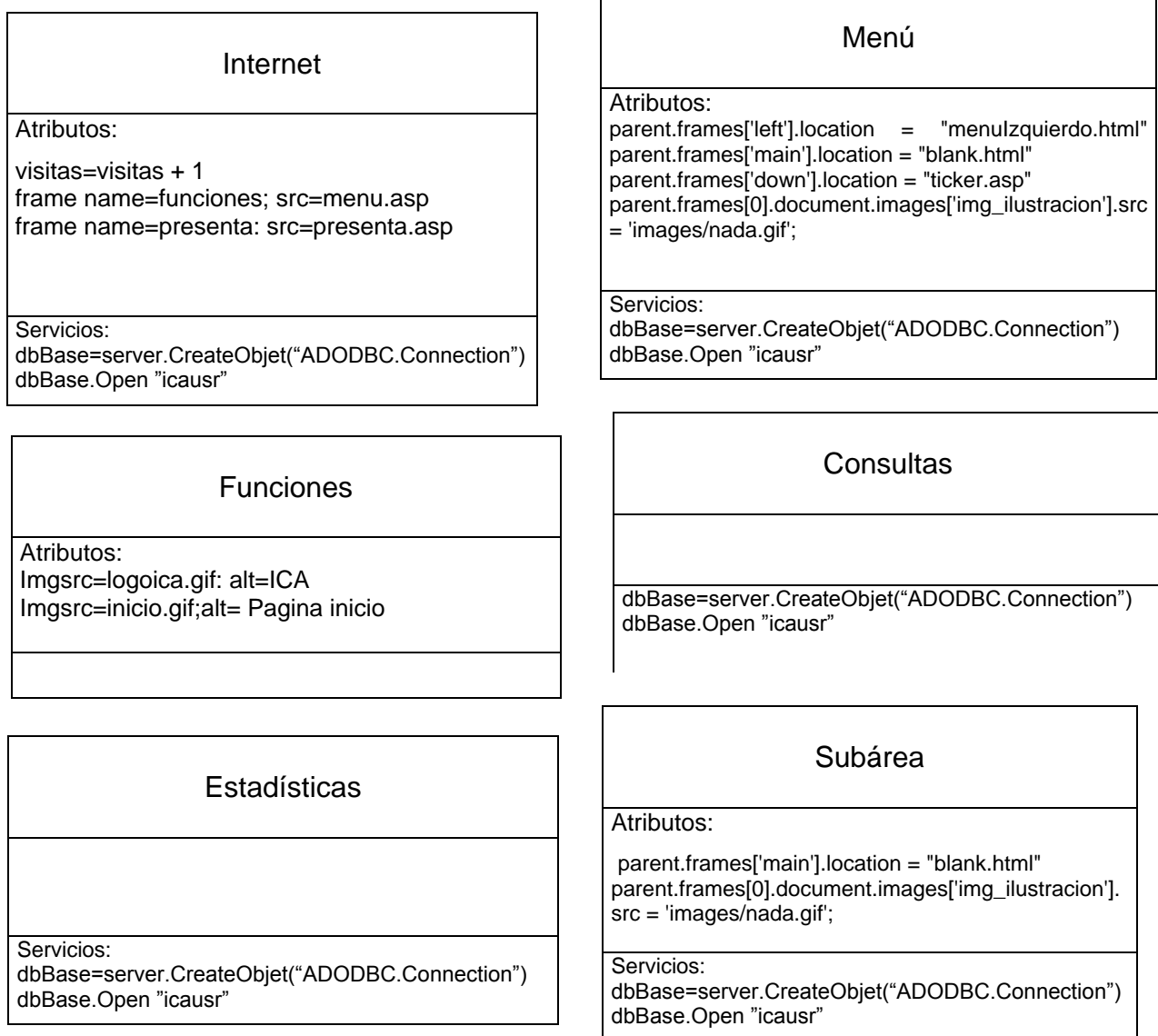


Figura 3.16. Diagrama de objetos y eventos. (Continúa)

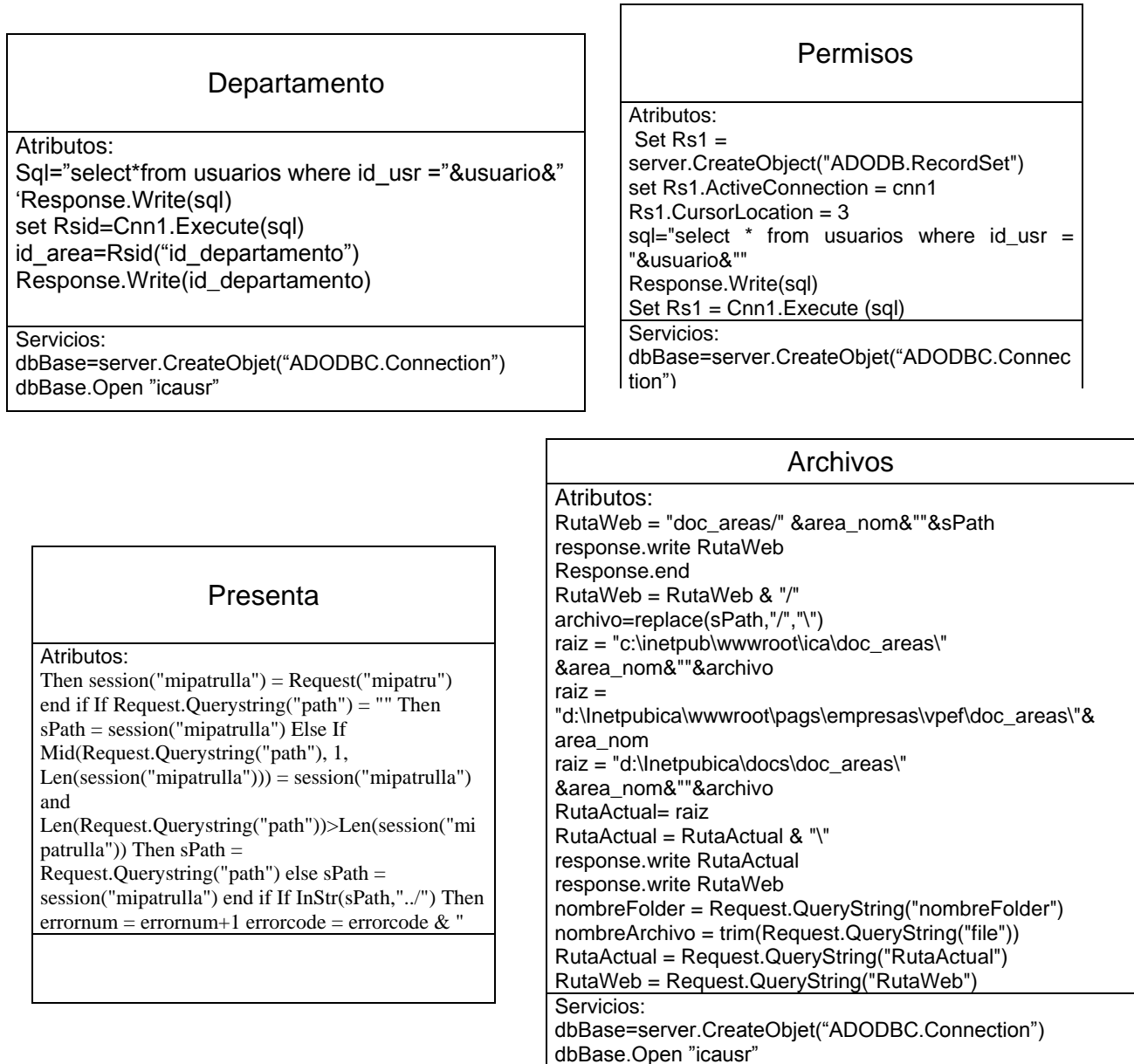


Figura 3.16. Diagrama de objetos y eventos.

Después del diseño de objetos y eventos necesitamos ver como los datos estarán distribuidos en el sistema, para eso diseñaremos un diagrama de flujo basándonos en los diagramas anteriores.




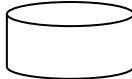
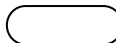
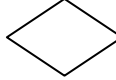
### 3.1.4. Diagrama de flujo de datos (DFD)

Una de las técnicas o herramientas más frecuentes utilizadas para el análisis estructurado y el manejo de los datos de una forma eficiente son los diagramas de flujo de datos.



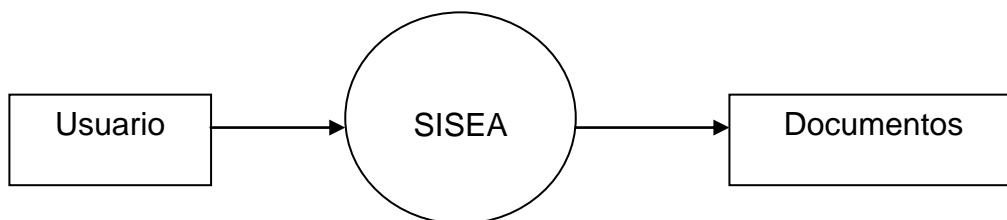
El diagrama de flujo de datos es una técnica gráfica que representa el flujo de la información y de las transformaciones que se aplican a los datos al moverse desde la entrada hasta la salida. Esta herramienta se observa a través de niveles, que son dados por los procesos, esto permite visualizar el sistema desde el punto de vista de los datos y no de quien trabaja con ellos.

La simbología utilizada en los diagramas de flujo de datos se ilustra en la Tabla 3.2.

Objeto	Representa
	Flujo de datos.
	Procesos.
	Entidades.
	Fuentes o depósitos de datos.
	Inicio o Fin del diagrama.
	Decisión.

*Tabla 3.2. Objetos de un diagrama de flujo de datos.*

El nivel 1 también llamado modelo fundamental del sistema, representa los elementos de software como un simple proceso, es decir, una caja negra, con una entrada y una salida. Así en la Figura 3.17, se puede ver que la entrada es el usuario y la salida los archivos que consulta.



*Figura 3.17. Diagrama de flujo de datos, nivel 1.*

El diagrama que se muestra en la Figura 3.18 es el referente al nivel 2 de DFD, ya que identifica de manera general los procesos del sistema. Cabe señalar que algunos de estos procesos como Alta, Baja y Actualiza de Archivos y Ligas son los mismos para

cada Departamento del Área de Finanzas, por lo cual no se muestran todos, además de que como mencionamos, el diagrama muestra al sistema de una forma general.

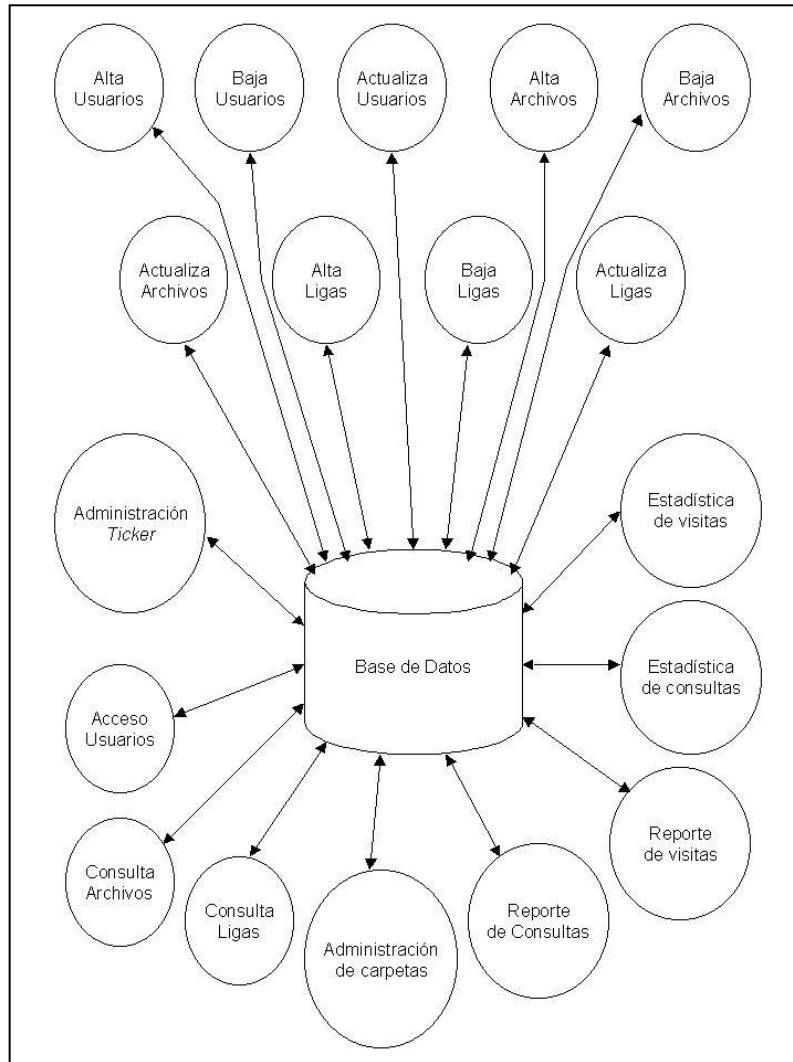


Figura 3.18. Diagrama de flujo de datos, nivel 2.

Como se puede observar en el diagrama, todos y cada uno de los procesos que realiza el sistema afectan directamente a la base de datos, siendo esta base de datos la integradora de todos los movimientos realizados durante la operación diaria.

Esta base de datos no solo recibirá información, sino también envía información a los usuarios de los diferentes Departamentos; en la base de datos se centralizan los archivos, los accesos a usuarios, los *tickers*, los reportes y las estadísticas.

Ahora mostramos cuatro procesos que por su importancia merecen ser mencionados, dichos procesos son Alta Archivos, Baja Archivos, Alta Usuarios, Acceso Usuarios. Los procesos de Alta y Baja Archivos son utilizados de la misma manera para cada uno de

los Departamentos. Los siguientes diagramas presentan el nivel 3 de un DFD ya que describen los procesos identificados en el nivel anterior.

La Figura 3.19 muestra el diagrama de flujo de Alta Archivos. Éste abarca desde la solicitud por parte del Usuario al Administrador de su departamento de dar de alta un archivo, hasta la consulta del archivo en el sistema por parte del Usuario.

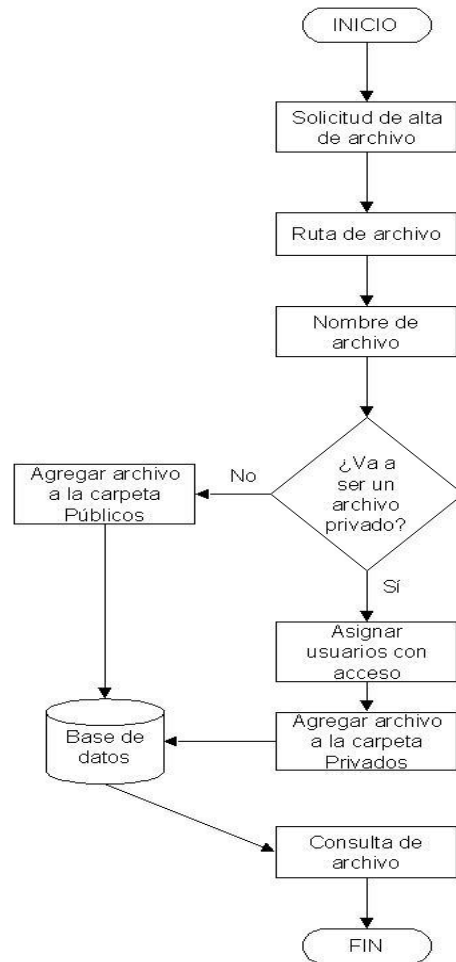
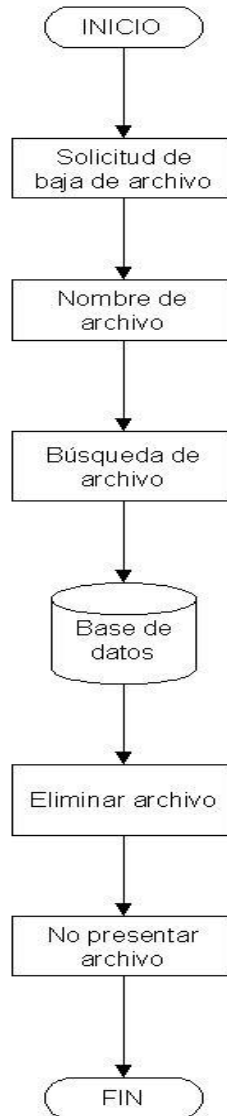


Figura 3.19. DFD nivel 3, Proceso Alta Archivos.

Como se puede observar en el diagrama, al solicitar el alta de un archivo en el sistema, se proporciona la ruta del archivo, es decir, la ubicación de éste, se le proporciona un nombre, se decide si va a ser un archivo público o privado, si es un archivo público se agrega a la carpeta Públicos, dándose de alta en la base de datos; si va a ser un archivo privado, se le asigna a los usuarios que podrán consultar el archivo y se agrega a la carpeta Privados, dándose de alta en la base de datos. Una vez que el archivo ha sido agregado a la base de datos podrá ser consultado.

La Figura 3.20 muestra el diagrama de flujo de Baja Archivos. Éste abarca desde la solicitud por parte del Usuario al Administrador de su departamento de dar de baja un archivo, hasta la ya no presentación del archivo en el sistema.



*Figura 3.20. DFD nivel 3, Proceso Baja Archivos.*

Como se puede observar en el diagrama, al solicitar la baja de un archivo en el sistema, se proporciona el nombre del archivo que se quiere dar de baja, el sistema hace la búsqueda del archivo en la base de datos, al encontrarlo lo borra de la base de datos y ya no lo presenta en la lista de archivos.

La Figura 3.21 muestra el diagrama de flujo de Alta Usuarios. Éste abarca desde la solicitud por parte del Administrador al Superusuario de dar de alta un usuario, hasta la confirmación de que el usuario está en el sistema.



Figura 3.21. DFD nivel 3, Proceso Alta Usuarios.

Como se puede observar en el diagrama, al solicitar el alta de un usuario en el sistema, se proporcionan los datos del usuario, éstos son: nombre, apellidos paterno y materno, fecha de alta, departamento al que pertenece, puesto, extensión, correo electrónico y se le asigna una contraseña y se confirma ésta, así también se le asigna un Id usuario con

el que será identificado en la base de datos; se determina el tipo de usuario al que pertenecerá, hay que recordar que son tres el Superusuario, que no tiene ningún tipo de restricciones para consultar los archivos de todos los departamentos; el Administrador que puede consultar todos los archivos de su departamento y el Usuario que sólo puede consultar los archivos que le hayan sido asignados. Una vez proporcionados estos datos se agrega al usuario a la base de datos y se confirma que haya sido dado de alta.

La Figura 3.22 muestra el diagrama de flujo de Acceso Usuarios. Éste abarca desde el ingreso de nombre de usuario y contraseña hasta la consulta de archivos en el sistema.

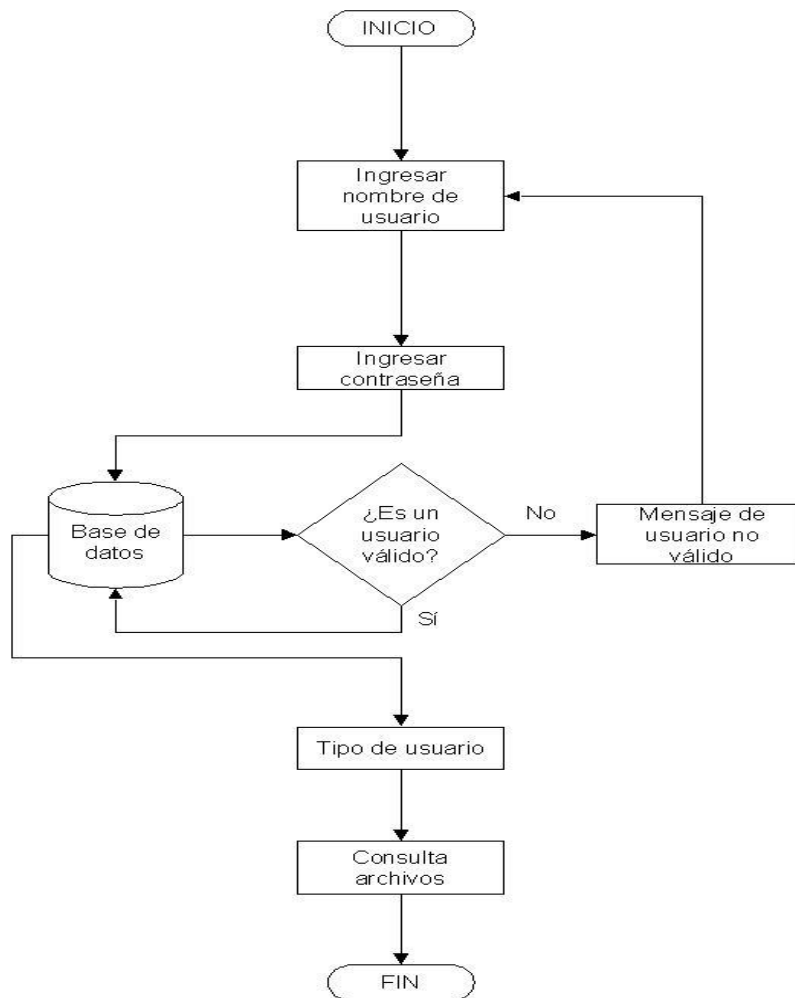


Figura 3.22. DFD nivel 3, Proceso Acceso Usuarios.

Como se puede observar en el diagrama, el usuario al ingresar su nombre de usuario y contraseña, el sistema verifica que sea un usuario válido, es decir, que se encuentre dado de alta en la base de datos; si es un usuario no válido, el sistema envía un mensaje de error haciéndole saber al usuario que no se encuentra en la base de datos; si se trata de un usuario válido, dependiendo de sus permisos, es decir, del tipo de usuario al que pertenece podrá consultar los archivos a los que tenga acceso.

### 3.1.5. Diagrama de Secuencia

El Diagrama de secuencia nos permite tener una mayor comprensión de la actividad que realiza el sistema, cuando éste tiene peticiones de las actividades del usuario.

A continuación se presenta un bloque de este proceso, ya que es muy extenso para poder presentarlo, pero además describiremos cada una de sus partes para su comprensión. Figura 3.23.

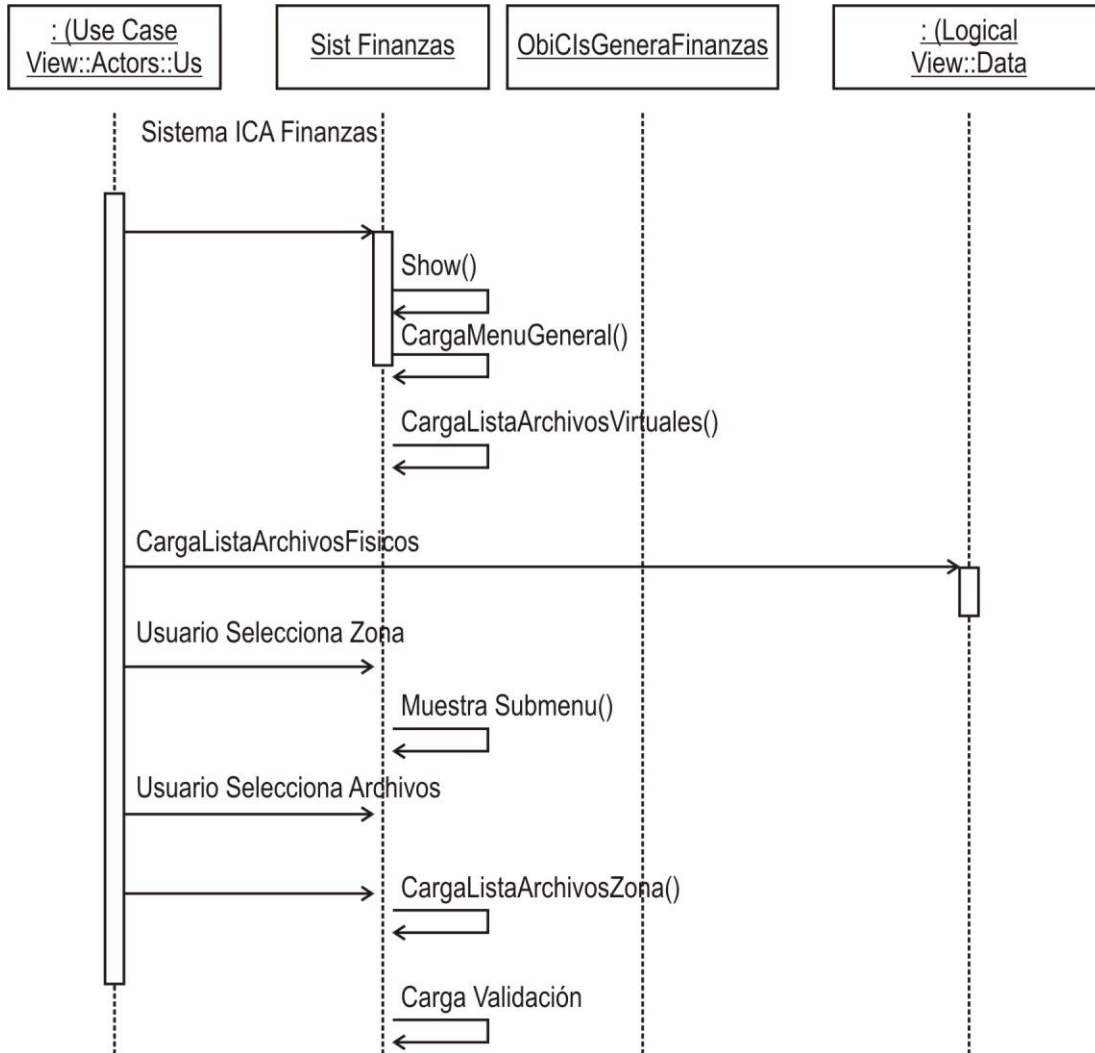


Figura 3.23. Diagrama de Secuencia. (Continúa)

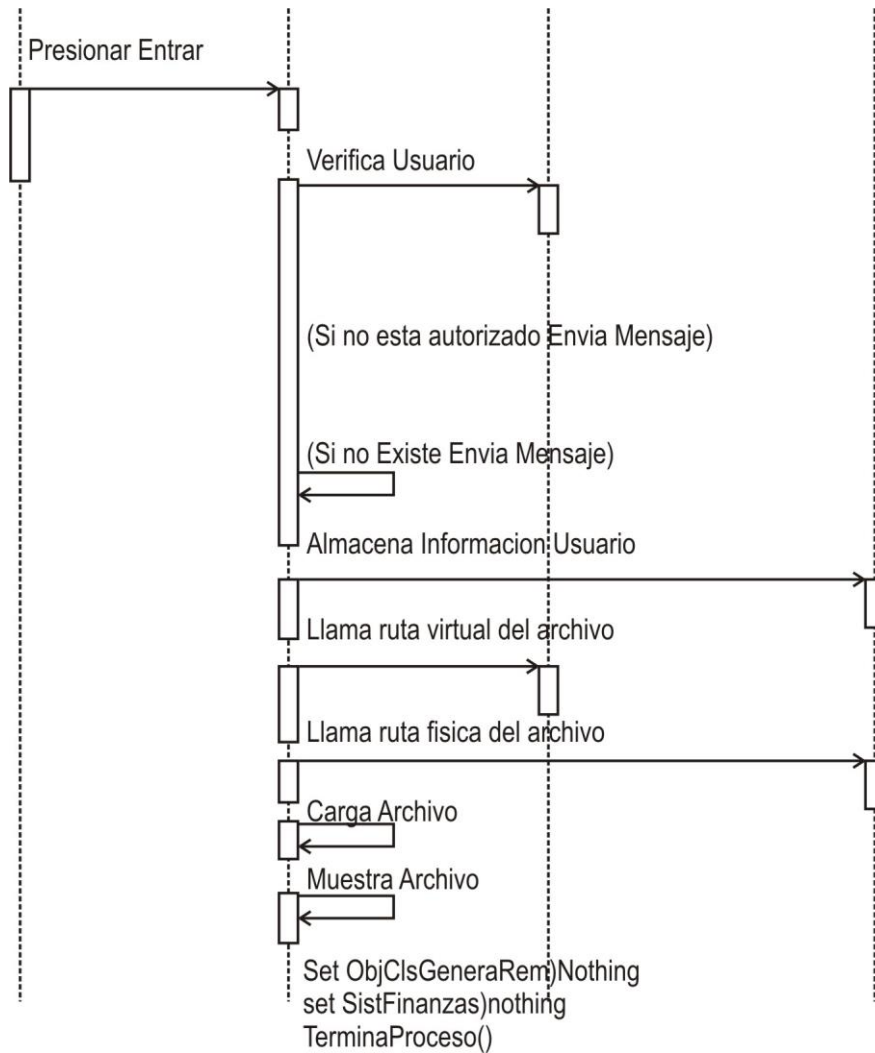
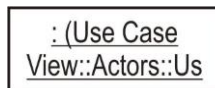


Figura 3.23. Diagrama de Secuencia.

En el diagrama de secuencia presentado se muestran cuatro bloques principales, el primero que presentaremos es el bloque de actores, éste comprende básicamente a la representación de los usuarios dentro del sistema, la representación de éste es la siguiente.

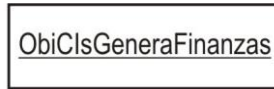


Adjunto a éste encontramos al sistema de Finanzas en general, éste se refiere básicamente al sitio en sí, en donde se llevarán a cabo todas las consultas, verificación de usuario, menú de navegación, etc., y su representación es la siguiente.

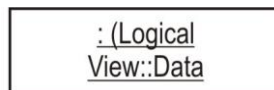




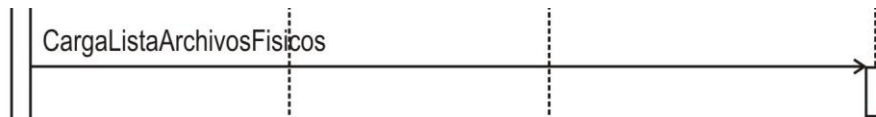
Después se muestra el bloque de objetos y de clases, éste representa a todas las clases y objetos generados para poder realizar la validación, carga de archivos, y procesos internos dentro del sistema su representación es la siguiente:



Por último tenemos la parte del bloque del Estado Lógico, éste representa a las llamadas a los estados físicos utilizados, por ejemplo, los archivos físicos, llamadas a la base de datos, este punto es importante ya que parte del sistema se concentra en el registro e integridad de archivos.



Ahora toca detallar la parte interna del diagrama, como primer punto tomamos a la línea de seguimiento, ésta se caracteriza por determinar el proceso que se va a realizar, por ejemplo, una consulta, una carga de documento, una verificación de registro, o simplemente una inserción a la base de datos.

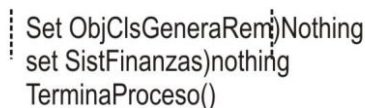


También en el diagrama existen líneas punteadas, éstas se utilizan para dar la separación entre cada bloque, y además para tener una mejor comprensión del flujo que en ese momento se está presentando.

En gran parte del diagrama de secuencia se presenta una flecha regresiva, ésta representa el momento en que se muestran las pantallas, los menús y validaciones en general, además de las cargas y muestras de archivos en este sistema en particular.



Finalizando esta descripción, se menciona al final del documento la liberación de los objetos ya que son necesarios para culminar la liberación del proceso, y finalmente el cierre del proceso.



Una vez finalizado el diseño conceptual describiremos la aplicación de las herramientas que se utilizaron para el desarrollo del sistema.

### 3.2. Aplicación de las Herramientas

Se utilizaron las herramientas del modelado de datos, el manejador de base de datos, y de tres niveles de normalización para tener un óptimo modelo.

En primer lugar detallaremos el diccionario de datos, éste describe las tablas utilizadas, atributos, tipo de datos, longitudes y descripción del campo.

A continuación se presenta el diagrama entidad-relación, generado con el modelador de datos Erwin.

Por último se muestra un ejemplo de un solo bloque del diagrama entidad-relación, desarrollando en conjunto con las tres primeras formas normales.

Cabe mencionar que los datos que se presentan en esta sección son sólo una parte de todo el sistema, ya que para los seis Departamentos del Área de Finanzas son las mismas entidades para cada uno de ellos.

#### 3.2.1 Diccionario de Datos

El diccionario de datos hace referencia a las tablas que van a integrar la base de datos de nuestro sistema y nos indica los campos que tendrán cada una de ellas, así como el tipo de dato que aceptará, la longitud del dato y una breve descripción del campo, como se muestra a continuación:

**Tabla:** Tipo\_region

<b><i>Nombre Columna</i></b>	<b><i>Tipo de Dato</i></b>	<b><i>Longitud</i></b>	<b><i>Descripción</i></b>
id_tipo	Int	6	Llave primaria
tip_nom	varchar	50	Descripción de la región

**Tabla:** Area

<b><i>Nombre Columna</i></b>	<b><i>Tipo de Dato</i></b>	<b><i>Longitud</i></b>	<b><i>Descripción</i></b>
id_area	Int	6	Llave primaria
area_nom	Varchar	50	Nombre del Area

area_gerente	Varchar	50	Nombre del Gerente
area_telefono	Varchar	50	Numero telefónico

**Tabla: Pais**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_pais	Int	6	Llave primaria
pais_nombre	Varchar	50	Nombre del país

**Tabla: Estado**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_estado	Int	6	Llave primaria
est_nom	Varchar	50	Nombre del estado

**Tabla: Ciudad**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_ciudad	Int	6	Llave primaria
ciu_nom	Varchar	50	Nombre del ciudad

**Tabla: Obra\_tipo**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_obra	Int	6	Llave primaria
Obra_nom	Varchar	50	Nombre de la obra

**Tabla: Proyecto**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_proyecto	Int	6	Llave primaria
proy_nom	varchar	50	Nombre del Proyecto
proy_fechaini	datetime	50	Fecha inicial del Proyecto

proy_fechafin	datetime		Fecha Final del Proyecto
proy_costo	int	6	Costo del proyecto
proy_metroscons	Int	6	Metros calculados
proy_levantamiento	varchar	50	Descripción del levantamiento
proy_fechaent_lev	datetime		Fecha de levantamiento
conc_presup	varchar	50	Concepto del Presupuesto

**Tabla: Presupuesto**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_presup	Int	6	Llave primaria
pres_fecha	datetime		Fecha de Apertura del Presupuesto

**Tabla: Tipo\_proyecto**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_tip	Int	6	Llave primaria
tip_nom	varchar		Descripción del tipo de proyecto

**Tabla: Control\_Proyecto**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_control	Int	6	Llave primaria
con_anteproyecto_sti	varchar	50	Descripción del anteproyecto
con_fechaent_ant	datetime		Fecha del anteproyecto
proy_fechafin	datetime		Fecha de la entrega real
con_fechaentreal_ant	datetime		Fecha de automática
con_fechaauto_ant	Datetime		Fecha automática real
con_fechaautoreal_ant	Datetime		Descripción del levantamiento
con_fechacontra_ant	Datetime		Fecha del contrato
con_fechacontrareal_ant	datetime		Fecha del contrato

			real
con_ejecutivo	varchar	50	Descripción del ejecutivo
con_fechaent_eje	datetime		Fecha de entrega
con_fechaentreal_pre	datetime		Fecha de entrega real
Con_presupuesto	decimal		Monto del presupuesto
con_fechaautoreal_pre	varchar	50	Fecha de entrega presolicitada real
con_fecha_inicio	datetime		Fecha de inicio
con_fecha_fin	datetime		Fecha Final
con_anticipo	decimal		Monto anticipo
con_indirecto	decimal		Monto indirecto
con_sobrepeso	decimal		Monto del sobre precio

**Tabla: Región**

<b>Nombre Columna</b>	<b>Tipo de Dato</b>	<b>Longitud</b>	<b>Descripción</b>
id_region	Int	6	Llave primaria
reg_nom	varchar	50	Nombre de la región
reg_gerente	varchar	50	Nombredel Gerente
reg_telefono	varchar	50	Numero telefónico

**Tabla: Documentos**

<b>Nombre Columna</b>	<b>Tipo de Dato</b>	<b>Longitud</b>	<b>Descripción</b>
id_doc	Int	6	Llave primaria
doc_nom	varchar	100	Nombre del documento
doc_ruta	varchar	200	Dirección de la ruta
doc_priv	char	2	Indentificador si es privado

**Tabla: Tipo\_usuario**

<b>Nombre Columna</b>	<b>Tipo de Dato</b>	<b>Longitud</b>	<b>Descripción</b>
-----------------------	---------------------	-----------------	--------------------

id_tipo	Int	6	Llave primaria
tipo_nombre	varchar	100	Nombre del tipo de usuario

**Tabla: Usuarios**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_usr	Int	6	Llave primaria
usu_password	varchar	100	Contraseña del usuario
usu_email	varchar	100	Dirección electrónica
usu_nombre	varchar	100	Nombre del usuario
usu_apaterno	varchar	50	Apellido paterno
usu_amaterno	varchar	50	Apellido materno
usu_ext	int	6	Extensión telefónica del usuario
usu_fecha	Date		Fecha de alta
usu_usr	Varchar	100	Clave del usuario
usu_puesto	varchar	100	Puesto del usuario

**Tabla: Permisos**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_usr	Int	6	Llave primaria
id_doc	varchar	100	Llave primaria del documento

**Tabla: Banner\_noticia**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_banner	Int	6	Llave primaria
bann_texto	varchar	500	Descripción de la noticia
baner_activo	varchar	100	Descripción activa de la noticia
banner_fecha	datetime		Fecha de apertura

**Tabla: Reportes**

<i>Nombre Columna</i>	<i>Tipo de Dato</i>	<i>Longitud</i>	<i>Descripción</i>
id_reporte	Int	6	Llave primaria

nombre_reporte	varchar	50	Descripción del reporte
fecha_creacion	datetime		Fecha de la creación
fecha_modificacion	datetime		Fecha de la modificación
descripcion	varchar	50	Descripción del reporte
comentario	varchar	50	Comentario particular

**Tabla: Estadística**

<b><i>Nombre Columna</i></b>	<b><i>Tipo de Dato</i></b>	<b><i>Longitud</i></b>	<b><i>Descripción</i></b>
id_estadistica	Int	6	Llave primaria
est_nombre	varchar	50	Descripción
est_fecha_creacion	datetime		Fecha de la creación
est_fecha_modificacion	datetime		Fecha de la modificación
est_numero_acceso_sitio_ano	int	6	Numero de accesos por año
est_numero_acceso_sitio_mes	int	6	Numero de accesos por mes
est_numero_acceso_sitio_dia	int	6	Numero de accesos por día
est_tiempo_duracion_visita	int	6	Tiempo de duración

**Tabla: Estadisticas\_archivo**

<b><i>Nombre Columna</i></b>	<b><i>Tipo de Dato</i></b>	<b><i>Longitud</i></b>	<b><i>Descripción</i></b>
id_est_arch	Int	6	Llave primaria
arc_nombre	varchar	50	Nombre del archivo
arc_ubicacion	varchar	50	Ubicación del archivo
arc_num_acceso_dia	int	6	Numero de accesos por día
arc_num_acceso_mes	int	6	Numero de accesos por mes
arc_num_acceso_ano	int	6	Numero de accesos por año
arc_total	int	6	Total de accesos

### 3.2.2. Diagrama Entidad-Relación

Para obtener un prototipo de bases de datos se hará uso de una técnica de modelado conocida como Entidad-Relación (ER).

Los modelos ER son parte de las herramientas conceptuales que existen para describir datos y las relaciones que existen entre ellos. El modelo ER está basado en una percepción del mundo como si consistiese de una colección de objetos básicos (entidades), representados por una colección finita de tablas de dos dimensiones (columnas y renglones) y las relaciones que existen entre estos objetos, las cuales representan acciones o situaciones reales.

Las características principales del modelo relacional son: simplicidad, precisión y flexibilidad.

**Simplicidad:** Las tablas son una forma familiar y explican por sí mismas la forma de representar datos. La mayoría de la gente que ha utilizado datos en forma de tabla, no requiere de un entrenamiento especial para entender o utilizar los datos que se representan en las tablas. En pocas palabras podemos decir que son amigables al usuario.

**Precisión:** Las tablas correctamente diseñadas mantienen un rigor matemático, dicen lo que significan y significan lo que dicen. Pueden ser implantadas y procesadas por una gran variedad de configuraciones de hardware y software.

**Flexibilidad.** Las tablas no solamente muestran la estructura de los datos sino pueden mostrar los datos también. Esto nos permite manejar el modelo antes de implementarlo. En otras palabras, las tablas son apropiadas no sólo para modelar datos sino para procesarlos también.

El modelado ER nos dice que podemos presentar en 3 fases:

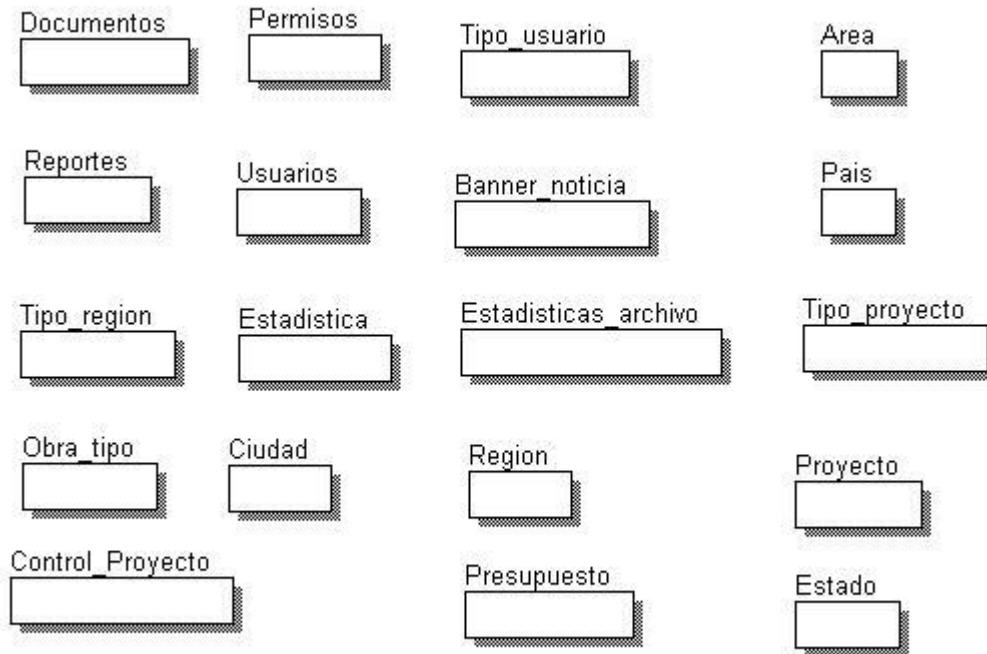
Entidades  
Relaciones  
Atributos

En la Definición de las Entidades y su modelado correspondiente debemos seguir los siguientes pasos:

- 1.- Descubrir entidades
- 2.- Definir el alcance de la entidad
- 3.- Definir una llave primaria
- 4.- Documentar



En la Figura 3.24 se muestran las entidades que fueron definidas para la creación del sistema.



*Figura 3.24. Modelado de Entidades.*

Los pasos explicados en la definición de sus relaciones son:

- 1.- Descubrir relaciones.
- 2.- Definir el alcance de la relación.
- 3.- Definir el tipo de relación.
- 4.- Documentar en el diagrama ER.
- 5.- Documentar en tablas.

Las relaciones, de acuerdo al número de elementos que se involucran de ambos lados de las entidades que participan, se clasifican entre tipos: Uno a Uno (1-1), Uno a Muchos (1-M) y Muchos a Muchos (M-M).

En la Figura 3.25 se muestra el diagrama Entidad-Relación del sistema en cuestión.

En la siguiente figura las entidades se presentan en forma de rectángulos, mientras que las relaciones obviamente están ligadas unas con las otras entidades.

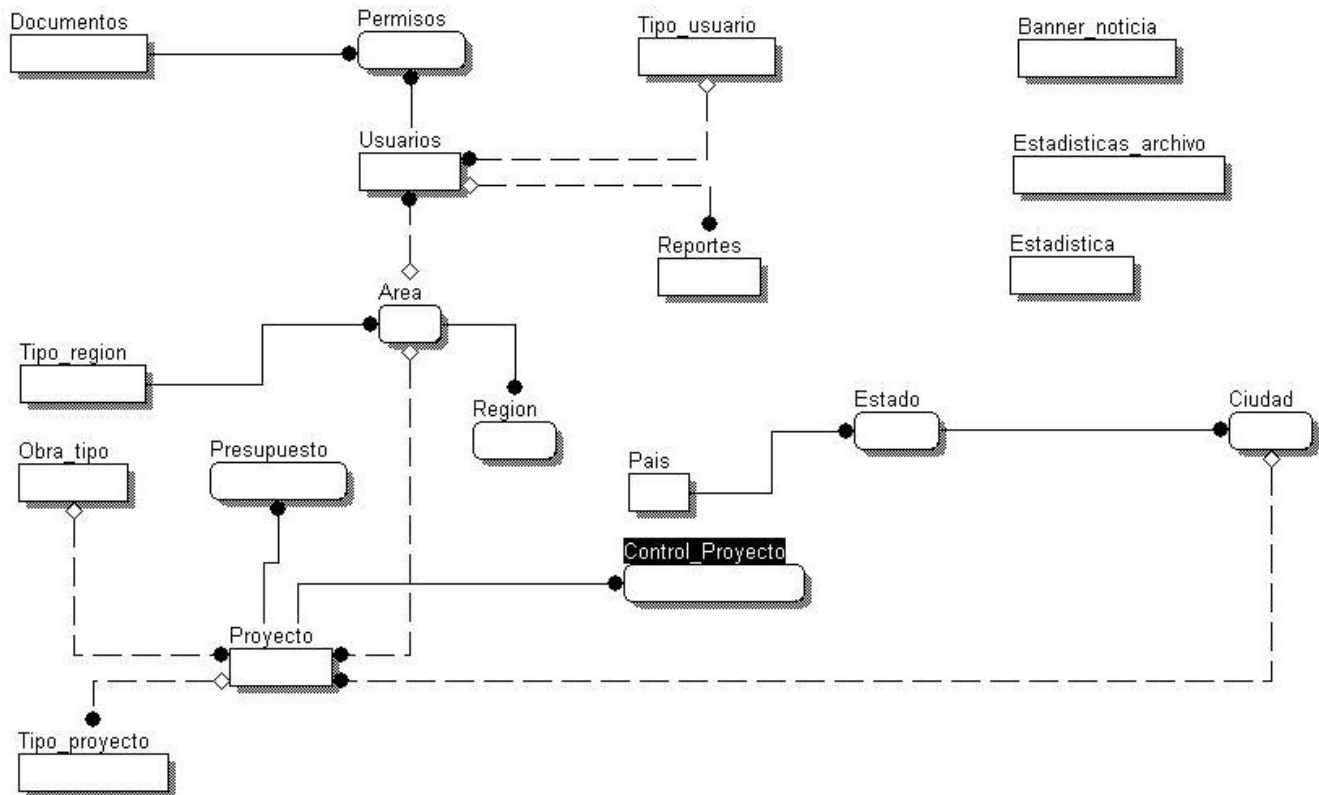


Figura 3.25. Diagrama Entidad-Relación del Sistema.

La definición de atributos se obtiene con los siguientes pasos:

- 1.- Descubrir atributos.
- 2.- Definir el alcance del atributo.
- 3.- Documentar el atributo en la entidad.

Para colocar los atributos de manera correcta en una entidad se utiliza el método de normalización, que permite verificar si nuestro modelo es correcto o bien si funcionará al ser implementado.

- **Normalización**

La normalización es un proceso que consiste en comprobar que las tablas (también denominadas relaciones en terminología propia del modelo relacional de datos) definidas cumplen unas determinadas condiciones. Se pretende garantizar la no existencia de redundancia y una cierta coherencia en la representación mediante un esquema relacional de las entidades y relaciones del modelo conceptual (diagrama E-R). Mediante la normalización se pueden solucionar diversos errores en el diseño de la base de datos así como mejorarlo. También se facilita el trabajo posterior del

administrador de la base de datos y de los desarrolladores de aplicaciones. En el proceso de normalización se utilizan principalmente tres niveles: primera, segunda y tercera normal.

*Primera forma normal*

Una relación R se encuentra en primera forma normal (1FN) sí y sólo sí por cada renglón columna contiene valores atómicos.

Una relación se encuentra en la primera forma normal cuando cumple lo siguiente:

Las celdas de las tablas poseen valores simples y no se permiten grupos ni arreglos repetidos como valores, es decir, contienen un solo valor por cada celda.

Todos los ingresos en cualquier columna (atributo) deben ser del mismo tipo.

Cada columna debe tener un nombre único, el orden de las columnas en la tabla no es importante.

Dos filas o renglones de una misma tabla no deben ser idénticas, aunque el orden de las filas no es importante.

Por lo general la mayoría de las relaciones del sistema cumplen con estas características, así que podemos decir que la mayoría de las relaciones se encuentran en la primera forma normal.

Por ejemplo en la relación Usuarios, tenemos la llave Id\_Estado y otro atributo llamado Est\_nom, podemos ver que para cada valor de la llave hay un único valor en Est\_nom; que no hay nombres de columnas repetidos, ni renglones repetidos. Por lo tanto Usuarios cumple con la 1FN.

Estado	
Id_estado	Est_nom
1	Querétaro
2	Jalisco
3	Puebla
4	San Luis Potosí
5	Zacatecas

Como esta relación maneja valores atómicos, es decir un solo valor por cada uno de los campos que conforman a los atributos de las entidades, ya se encuentra en primera forma normal.

Donde Id\_Usuario es la llave y Puesto\_Usuario es otro atributo no llave, de la tabla Usuarios.

*Segunda forma normal*

Una relación (R) está en segunda forma normal (2FN) sí y sólo sí está en 1FN y los atributos dependen funcionalmente de la llave primaria.

Una relación se encuentra en segunda forma normal, cuando cumple con las reglas de la primera forma normal y todos sus atributos que no son claves (llaves) dependen por completo de la clave. De acuerdo con esta definición, cada tabla que tiene un atributo único como clave está en segunda forma normal. Es decir, los atributos deben depender de toda la clave y no de una parte ella.

Tenemos la siguiente relación, llamada Ciudad:

Ciudad	
Id_Ciudad	Ciu_nom
35	Querétaro
55	Guadalajara
62	Puebla

En la tabla estado se muestra su normalización que es la siguiente:

Estado	
Id_Estado	Est_nom
1	Querétaro
2	Jalisco
3	Puebla

Las dos tablas presentadas no tienen relación alguna en este momento, pero como se necesita tener una, ésta queda de la siguiente forma:

La tabla: Ciudad

Ciudad			
Id_Ciudad	Ciu_nom	Id_Estado	Estado_nom
35	Querétaro	1	Querétaro
55	Guadalajara	2	Jalisco
62	Puebla	3	Puebla

*Tercera forma normal*

Una relación (R) está en tercera forma normal (3FN) sí y sólo sí está en 2FN y todos sus atributos no primos dependen no transitivamente de la llave primaria.

La dependencia transitiva se refiere a: en una afinidad (tabla bidimensional) que tiene por lo menos 3 atributos (A,B,C) en donde A determina a B, B determina a C pero no determina a A.

La 3FN consiste en eliminar la dependencia transitiva que queda en una segunda forma normal; en pocas palabras una relación está en tercera forma normal si está en segunda forma normal y no existen dependencias transitivas entre los atributos, nos referimos a dependencias transitivas cuando existe más de una forma de llegar a referencias a un atributo de una relación.

Por ejemplo, consideremos el siguiente caso, en donde existe dependencia transitiva:

Ciudad			
Id_Ciudad	Ciu_nom	Id_Estado	Estado_nom
35	Querétaro	1	Querétaro
55	Guadalajara	2	Jalisco
62	Puebla	3	Puebla

Se aprecia que Estado\_nom depende del valor de la Ciudad, además de que el Estado depende de Ciudad, también; pero el Estado\_nom no depende básicamente de la Ciudad, esta depende del valor de id\_estado, por lo tanto podemos separar esta tabla de la siguiente forma:

Ciudad		
Id_Ciudad	Ciu_nom	Id_Estado
22	Querétaro	1
30	Guadalajara	2
31	Puebla	3

Estado	
Id_Estado	Est_nom
1	Querétaro
2	Jalisco
3	Puebla

Una vez diseñado el sistema presentamos la construcción del mismo.

# CAPÍTULO IV

## DESARROLLO DEL SISTEMA

Este capítulo comprende lo que es el desarrollo del sistema, con la creación de la base de datos, la creación de las tablas, así como la construcción de las pantallas de interacción del sistema con el usuario.

### 4.1. Construcción del Sistema

El Sistema de Seguridad de Archivos de Información Empresarial (SISEA), consta de una base de datos en SQL Server 2000 a la cual se le introducen datos a través de varias paginas HTML y ASP.

#### 4.1.1. Creación de la base de datos

En la base de datos es en donde se concentra toda la información a cerca de los usuarios que pueden hacer consultas y documentos que se pueden consultar en el Sistema de Seguridad de Archivos de Información Empresarial. Dicha base de datos recibe el nombre de SISEA para poder identificarla fácilmente.

En las Figuras 4.1 y 4.2 se puede ver cómo se crea la base de datos en SQL Server 2000, el cual es un proceso bastante sencillo, donde se le da nombre a la base de datos y se le definen propiedades, como el tamaño.

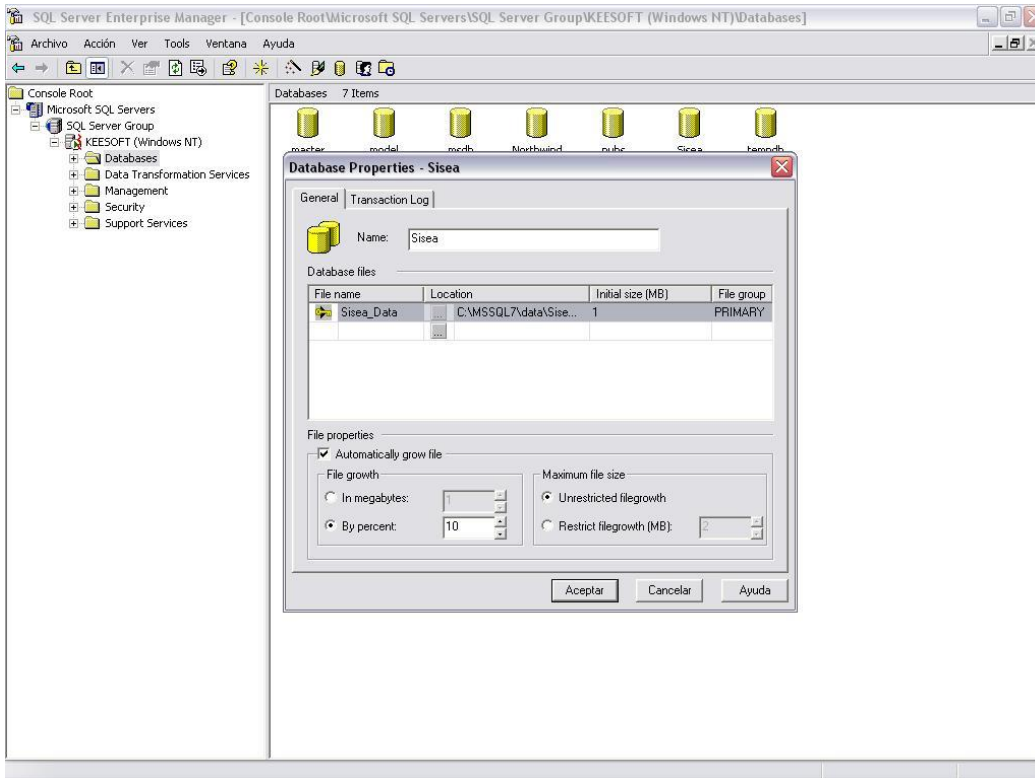


Figura 4.1. Creación de la base de datos SISEA.

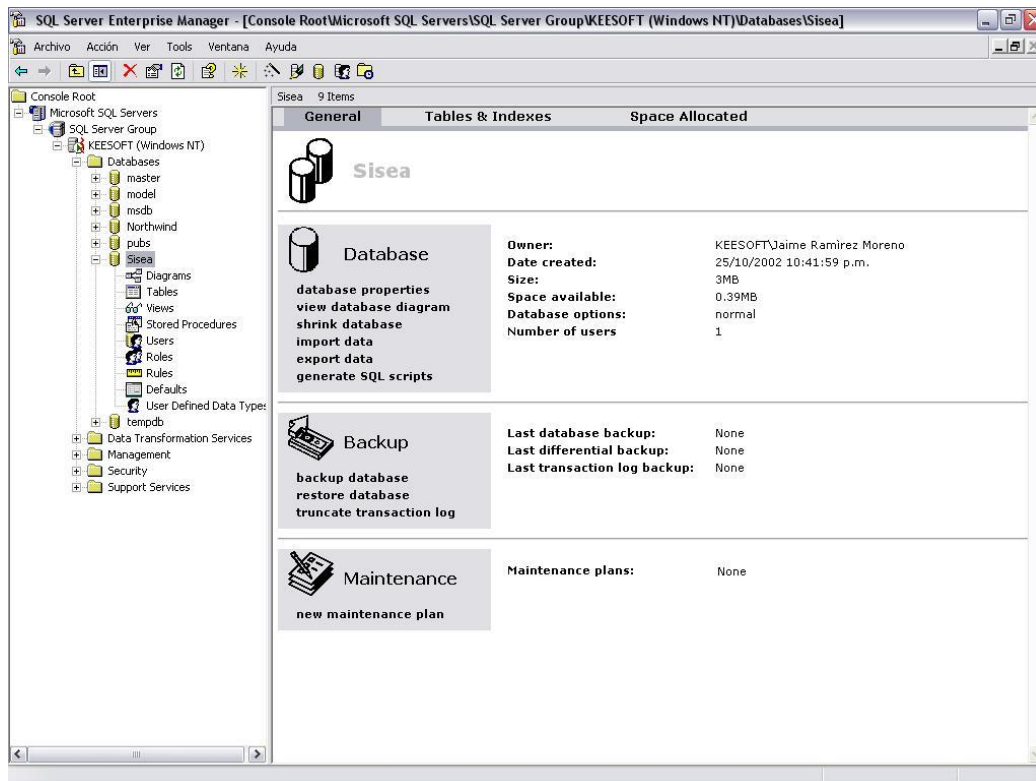


Figura 4.2. Base de datos SISEA.

Una vez creada la base datos se procede a crear las tablas que contendrá dicha base. El número de tablas son 7, Tipo\_usuario, Usuarios, Documentos, Estadística, Estadísticas\_archivo, Permisos y Banner\_noticia. Vea la Figura 4.3 donde se muestran dichas tablas.

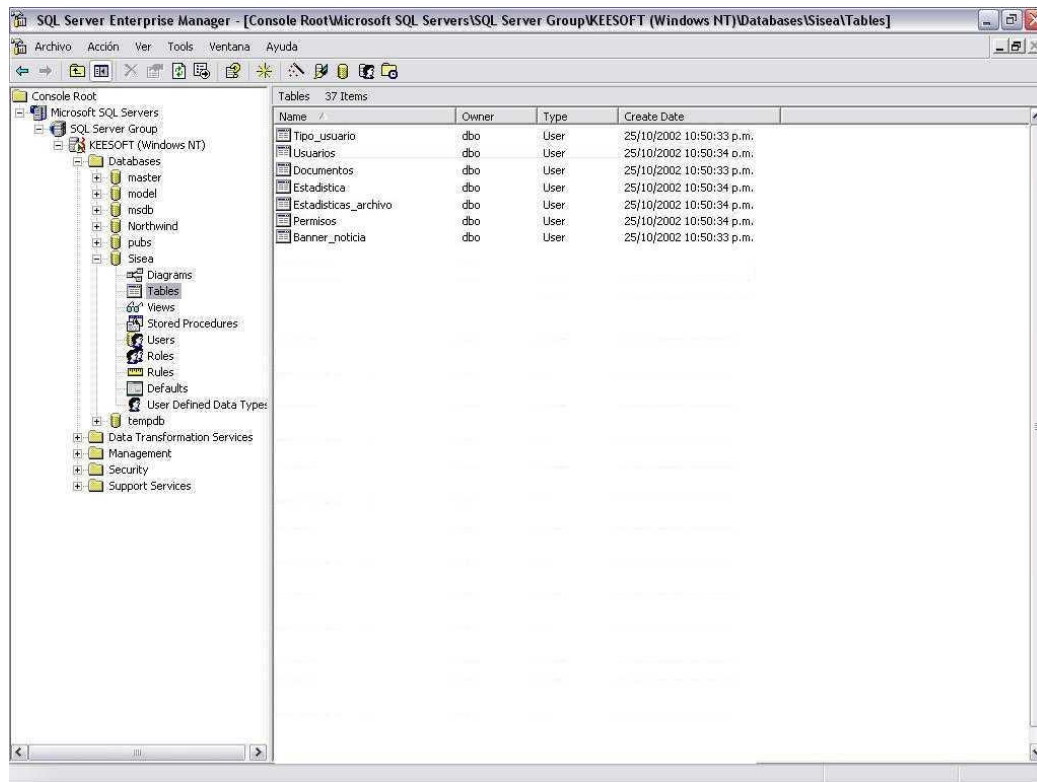


Figura 4.3. Tablas de la base de datos SISEA.

#### 4.1.2. Creación de las tablas en la base de datos

Para la creación de cada tabla se utilizó el ayudante de creación de tablas propio de SQL Server 2000, en el cual se definen cada uno de los campos que contendrá la tabla, así como cual será la llave primaria, el número de caracteres y el tipo de variable que aceptará cada uno de los campos y si aceptará valores nulos o no. Un ejemplo de la creación de una tabla lo podemos ver en la Figura 4.4 donde se crea la tabla Documentos, para ello se definen cada uno de los campos de esta tabla, así como sus propiedades (Name, Data Type, Size, Nulls, Default).

La tabla Documentos comprende lo que son todos los documentos que podrán ser consultados y se compone de los campos id\_doc, doc\_nom, doc\_ruta y doc\_priv.

La tabla Usuarios es una de las más importantes y grandes ya que en ella se encuentra toda la información de los usuarios, así como el tipo de usuario que será, que se relacionará con los permisos para consultar documentos; los campos que componen a



esta tabla son: id\_usr, id\_area, id\_tipo, id\_tip, usu\_password, usu\_email, usu\_nombre, usu\_apaterno, usu\_amaterno, usu\_ext, usu\_fecha, usu\_usr y usu\_puesto.

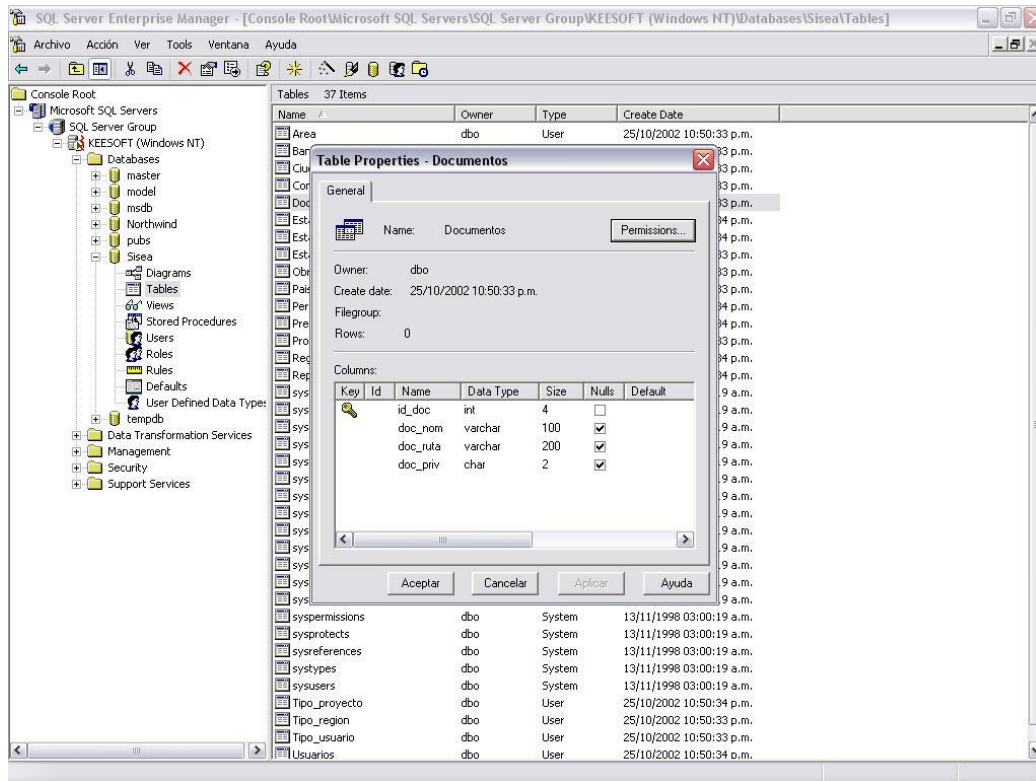


Figura 4.4. Creación de la tabla Documentos.

La tabla Tipo\_usuario como su nombre lo dice especifica el tipo de usuario y sus campos son: id\_tipo, tipo\_nombre.

La tabla Permisos relaciona los documentos con el tipo de usuario, sus campos son: id\_usr y id\_doc.

La tabla Estadística nos da datos con respecto al sitio, el número de visitas a éste, así como el tiempo de duración; se compone de los siguientes campos: id\_estadistica, est\_nombre, est\_fecha\_creacion, est\_fecha\_modificacion, est\_numero\_acceso\_sitio\_ano, est\_numero\_acceso\_sitio\_mes, est\_numero\_acceso\_sitio\_dia y est\_tiempo\_duracion\_visita.

La tabla Estadísticas\_archivo nos da información acerca del número de veces que ha sido consultado un documento, sus campos son: id\_est\_arch, arc\_nombre, arc\_ubicacion, arc\_num\_acceso\_dia, arc\_num\_acceso\_mes, arc\_num\_acceso\_ano y arc\_total.

En la tabla Banner\_noticia se pone información de interés general para los usuarios, sus campos son: id\_banner, bann\_texto, baner\_activo y banner\_fecha.

Es preciso aclarar que las tablas también pueden crearse por medio de un *script* generado de una herramienta llamada ERWin, en la que se diseña el diagrama entidad relación con sus propiedades y atributos. Una vez completo este diagrama, ERWin crea las tablas en SQL por medio de dicho *script*, como podemos ver en las Figuras 4.5 y 4.6; es así como creamos las tablas.

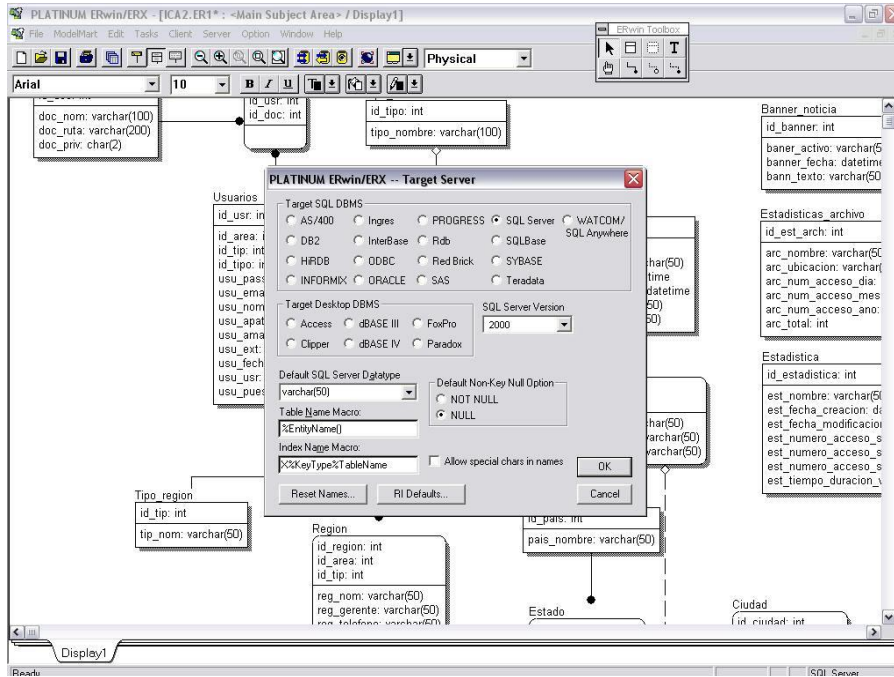


Figura 4.5. Conexión de ERWin con SQL Server 2000.

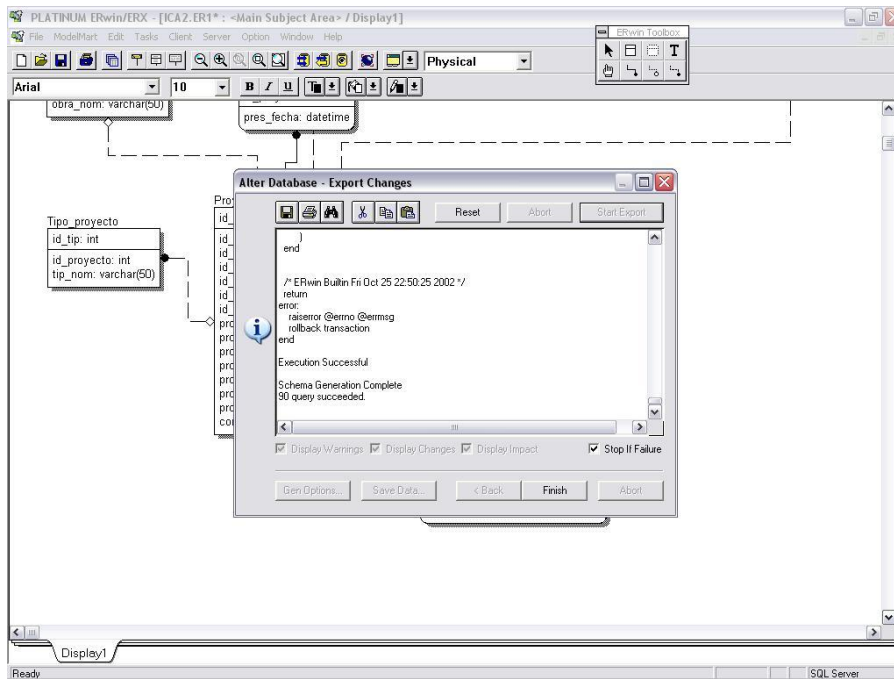


Figura 4.6. Generación del script para la creación de las tablas.

El script obtenido de ERWin para la creación de las tablas es el siguiente:

A ERWin le damos la instrucción:

```
/*
ACTION is CREATE Table Tipo_region
*/
```

Obtenemos el siguiente código de SQL, con el que creamos la tabla Tipo\_region, con su llave primaria id\_tip y el campo tip\_nom, con sus respectivos atributos.

```
CREATE TABLE Tipo_region (
    id_tip          int NOT NULL,
    tip_nom         varchar(50) NULL
)
go
```

Para cambiar un atributo de los campos de una tabla previamente generada, ERWin genera la siguiente instrucción en SQL, que nos permite identificar la llave primaria en la tabla Tipo\_region:

```
ALTER TABLE Tipo_region
    ADD PRIMARY KEY (id_tip)
go
```

El procedimiento para crear en ERWin las otras tablas de la base de datos es muy similar; la diferencia es el nombre de la tabla, sus campos y atributos.

```
/*
ACTION is CREATE Table Area
*/

CREATE TABLE Area (
    id_area          int NOT NULL,
    id_tip           int NOT NULL,
    area_nom         varchar(50) NULL,
    area_gerente     varchar(50) NULL,
    area_telefono    varchar(50) NULL
)
go

ALTER TABLE Area
    ADD PRIMARY KEY (id_area, id_tip)
go

/*
ACTION is CREATE Table Pais
*/

CREATE TABLE Pais (
    id_pais          int NOT NULL,
    pais_nombre     varchar(50) NULL
)
go
```

```

go
ALTER TABLE Pais
    ADD PRIMARY KEY (id_pais)
go

/*
ACTION is CREATE Table Estado
*/

CREATE TABLE Estado (
    id_estado          int NOT NULL,
    id_pais             int NOT NULL,
    est_nom            varchar(50) NOT NULL
)
go

ALTER TABLE Estado
    ADD PRIMARY KEY (id_estado, id_pais)
go

/*
ACTION is CREATE Table Ciudad
*/

CREATE TABLE Ciudad (
    id_ciudad          int NOT NULL,
    id_estado         int NOT NULL,
    id_pais           int NOT NULL,
    ciu_nom          varchar(50) NULL
)
go

ALTER TABLE Ciudad
    ADD PRIMARY KEY (id_ciudad, id_estado, id_pais)
go

/*
ACTION is CREATE Table Obra_tipo
*/

CREATE TABLE Obra_tipo (
    id_obra           int NOT NULL,
    obra_nom         varchar(50) NULL
)
go

ALTER TABLE Obra_tipo
    ADD PRIMARY KEY (id_obra)
go

/*
ACTION is CREATE Table Proyecto

```

```

*/
CREATE TABLE Proyecto (
    id_proyecto          int NOT NULL,
    id_area              int NULL,
    id_tip               int NULL,
    id_ciudad            int NULL,
    id_estado            int NULL,
    id_pais              int NULL,
    id_obra              int NULL,
    proy_nom             varchar(50) NULL,
    proy_fechaini       datetime NULL,
    proy_fechafin       datetime NULL,
    proy_costo           int NULL,
    proy_metroscons     int NULL,
    proy_levantamiento  varchar(50) NULL,
    proy_fechaent_lev   datetime NULL,
    conc_presup         varchar(50) NULL
)
go

ALTER TABLE Proyecto
    ADD PRIMARY KEY (id_proyecto)
go

/*
ACTION is CREATE Table Control_Proyecto
*/

CREATE TABLE Control_Proyecto (
    id_control          int NOT NULL,
    id_proyecto         int NOT NULL,
    con_anteprojectoesti varchar(50) NULL,
    con_fechaent_ant    datetime NULL,
    con_fechaentreal_ant datetime NULL,
    con_fechaauto_ant   datetime NULL,
    con_fechaautoreal_ant datetime NULL,
    con_fechacontra_ant datetime NULL,
    con_fechacontrareal_ant datetime NULL,
    con_ejecutivo       varchar(50) NULL,
    con_fechaent_eje    datetime NULL,
    con_fechaentreal_eje datetime NULL,
    con_presupuesto     decimal NULL,
    con_fechaent_pre    datetime NULL,
    con_fechaentreal_pre datetime NULL,
    con_fechaautoreal_pre varchar(50) NULL,
    con_fecha_inicio    datetime NULL,
    con_fecha_fin       datetime NULL,
    con_anticipo        varchar(50) NULL,
    con_indirecto       decimal NULL,
    con_sobrepresicio   decimal NULL
)
go

```

```
ALTER TABLE Control_Proyecto
    ADD PRIMARY KEY (id_control, id_proyecto)
go
```

```
/*
ACTION is CREATE Table Banner_noticia
*/
```

```
CREATE TABLE Banner_noticia (
    id_banner          int NOT NULL,
    bann_texto         varchar(500) NULL,
    baner_activo       varchar(50) NULL,
    banner_fecha       datetime NULL
)
go
```

```
ALTER TABLE Banner_noticia
    ADD PRIMARY KEY (id_banner)
go
```

```
/*
ACTION is CREATE Table Documentos
*/
```

```
CREATE TABLE Documentos (
    id_doc             int NOT NULL,
    doc_nom            varchar(100) NULL,
    doc_ruta           varchar(200) NULL,
    doc_priv           char(2) NULL
)
go
```

```
ALTER TABLE Documentos
    ADD PRIMARY KEY (id_doc)
go
```

```
/*
ACTION is CREATE Table Tipo_usuario
*/
```

```
CREATE TABLE Tipo_usuario (
    id_tipo           int NOT NULL,
    tipo_nombre       varchar(100) NULL
)
go
```

```
ALTER TABLE Tipo_usuario
    ADD PRIMARY KEY (id_tipo)
go
```

```

/*
ACTION is CREATE Table Usuarios
*/

CREATE TABLE Usuarios (
    id_usr          int NOT NULL,
    id_area         int NULL,
    id_tipo         int NULL,
    id_tip          int NULL,
    usu_password    varchar(100) NULL,
    usu_email       varchar(100) NULL,
    usu_nombre     varchar(100) NULL,
    usu_apaterno   varchar(50) NULL,
    usu_amaterno   varchar(50) NULL,
    usu_ext        varchar(50) NULL,
    usu_fecha      datetime NULL,
    usu_usr        varchar(100) NULL,
    usu_puesto     varchar(100) NULL
)
go

ALTER TABLE Usuarios
    ADD PRIMARY KEY (id_usr)
go

/*
ACTION is CREATE Table Permisos
*/

CREATE TABLE Permisos (
    id_usr          int NOT NULL,
    id_doc          int NOT NULL
)
go

ALTER TABLE Permisos
    ADD PRIMARY KEY (id_usr, id_doc)
go

/*
ACTION is CREATE Table Estadistica
*/

CREATE TABLE Estadistica (
    id_estadistica int NOT NULL,
    est_nombre     varchar(50) NULL,
    est_fecha_creacion datetime NULL,
    est_fecha_modificacion int NULL,
    est_numero_acceso_sitio_ano int NULL,
    est_numero_acceso_sitio_mes int NULL,
    est_numero_acceso_sitio_dia int NULL,
    est_tiempo_duracion_visita int NULL
)

```

```

go
ALTER TABLE Estadistica
    ADD PRIMARY KEY (id_estadistica)
go

/*
ACTION is CREATE Table Estadisticas_archivo
*/

CREATE TABLE Estadisticas_archivo (
    id_est_arch          int NOT NULL,
    arc_nombre           varchar(50) NULL,
    arc_ubicacion        varchar(50) NULL,
    arc_num_acceso_dia   int NULL,
    arc_num_acceso_mes   int NULL,
    arc_num_acceso_ano   int NULL,
    arc_total            int NULL
)
go

ALTER TABLE Estadisticas_archivo
    ADD PRIMARY KEY (id_est_arch)
go

/*
ACTION is CREATE Table Presupuesto
*/

CREATE TABLE Presupuesto (
    id_presup           int NOT NULL,
    id_proyecto         int NOT NULL,
    pres_fecha          datetime NULL
)
go

ALTER TABLE Presupuesto
    ADD PRIMARY KEY (id_presup, id_proyecto)
go

/*
ACTION is CREATE Table Region
*/

CREATE TABLE Region (
    id_region           int NOT NULL,
    id_area             int NOT NULL,
    id_tip              int NOT NULL,
    reg_nom             varchar(50) NULL,
    reg_gerente         varchar(50) NULL,
    reg_telefono        varchar(50) NULL
)
go

```



```
ALTER TABLE Region
    ADD PRIMARY KEY (id_region, id_area, id_tip)
go
```

```
/*
ACTION is CREATE Table Reportes
*/
```

```
CREATE TABLE Reportes (
    id_reporte          int NOT NULL,
    id_usr              int NULL,
    nombre_reporte     varchar(50) NULL,
    fecha_creacion     datetime NULL,
    fecha_modificacion datetime NULL,
    descripcion        varchar(50) NULL,
    comentario         varchar(50) NULL
)
go
```

```
ALTER TABLE Reportes
    ADD PRIMARY KEY (id_reporte)
go
```

```
/*
ACTION is CREATE Table Tipo_proyecto
*/
```

```
CREATE TABLE Tipo_proyecto (
    id_tip              int NOT NULL,
    id_proyecto        int NULL,
    tip_nom             varchar(50) NULL
)
go
```

```
ALTER TABLE Tipo_proyecto
    ADD PRIMARY KEY (id_tip)
go
```

**Para la creación de llaves foráneas ERWin genera el siguiente código en SQL:**

```
ALTER TABLE Area
    ADD FOREIGN KEY (id_tip)
    REFERENCES Tipo_region
go
```

```
ALTER TABLE Estado
    ADD FOREIGN KEY (id_pais)
    REFERENCES Pais
go
```

```
ALTER TABLE Ciudad
```

```

        ADD FOREIGN KEY (id_estado, id_pais)
                        REFERENCES Estado
go

ALTER TABLE Proyecto
        ADD FOREIGN KEY (id_area, id_tip)
                        REFERENCES Area
go

ALTER TABLE Proyecto
        ADD FOREIGN KEY (id_ciudad, id_estado, id_pais)
                        REFERENCES Ciudad
go

ALTER TABLE Proyecto
        ADD FOREIGN KEY (id_obra)
                        REFERENCES Obra_tipo
go

ALTER TABLE Control_Proyecto
        ADD FOREIGN KEY (id_proyecto)
                        REFERENCES Proyecto
go

ALTER TABLE Usuarios
        ADD FOREIGN KEY (id_area, id_tip)
                        REFERENCES Area
go

ALTER TABLE Usuarios
        ADD FOREIGN KEY (id_tipo)
                        REFERENCES Tipo_usuario
go

ALTER TABLE Permisos
        ADD FOREIGN KEY (id_doc)
                        REFERENCES Documentos
go

ALTER TABLE Permisos
        ADD FOREIGN KEY (id_usr)
                        REFERENCES Usuarios
go

ALTER TABLE Presupuesto
        ADD FOREIGN KEY (id_proyecto)
                        REFERENCES Proyecto
go

```

```
ALTER TABLE Region
    ADD FOREIGN KEY (id_area, id_tip)
        REFERENCES Area
go

ALTER TABLE Reportes
    ADD FOREIGN KEY (id_usr)
        REFERENCES Usuarios
go

ALTER TABLE Tipo_proyecto
    ADD FOREIGN KEY (id_proyecto)
        REFERENCES Proyecto
go
```

Una vez realizada la creación de la base de datos se describe la conectividad de estos elementos.

### 4.1.3. Conexión ActiveX Data Objects (ADO)

#### Acceso al origen de datos

*ActiveX Data Objects* es una tecnología para agregar a las páginas *Web* acceso a bases de datos. Podemos utilizar ADO para escribir secuencias de comandos compatibles con OLE DB, que permitan la conexión con bases de datos, hojas de cálculo, archivos de datos secuenciales o directorios de correo electrónico.

OLE DB es una interfaz de programación que permite exponer las funciones del sistema de administración de bases de datos. Con el modelo de objetos ADO tenemos acceso a estas interfaces (mediante lenguajes de secuencias de comandos, como *VBScript* o *JScript*) para agregar funciones de bases de datos a las aplicaciones *Web*. Además se puede utilizar ADO para tener acceso a bases de datos compatibles con la Conectividad abierta de bases de datos *Open Database Connectivity* (ODBC).

#### Crear una cadena de conexión

El primer paso en la creación de una aplicación de datos en *Web* consiste en proporcionar un método para que ADO encuentre e identifique el origen de datos. Para ello se utiliza una *cadena de conexión*, una serie de argumentos separados mediante un punto y coma que definen parámetros como el proveedor del origen de datos y la ubicación del mismo. ADO utiliza la cadena de conexión para identificar el *proveedor* OLE DB y para dirigir al proveedor al origen de datos.

El proveedor es un componente que representa el origen de datos y que expone la información en la aplicación en forma de conjuntos de filas. En la tabla 4.1. siguiente se enumeran las cadenas de conexión de OLE DB para varios orígenes de datos habituales:

Origen de datos	Cadena de conexión OLE DB
Microsoft(r) Access	Provider=Microsoft.Jet.OLEDB.4.0;Data Source=ruta física de acceso al archivo .mdb
Microsoft SQL Server	Provider=SQLOLEDB.1;Data Source=ruta de acceso a la base de datos del servidor
Oracle	Provider=MSDAORA.1;Data Source=ruta de acceso a la base de datos del servidor
Microsoft Indexing Service	Provider=MSIDXS.1;Data Source=ruta de acceso al archivo

*Tabla 4.1. Cadena de conexión.*

Para proporcionar compatibilidad con versiones anteriores, el proveedor OLE DB para ODBC admite la sintaxis de las cadenas de conexión ODBC. En la tabla 4.2. siguiente se enumeran las cadenas de conexión ODBC que se utilizan habitualmente.

Controlador del origen de datos	Cadena de conexión ODBC
Microsoft Access	Driver={Microsoft Access Driver (*.mdb)};DBQ=ruta física de acceso al archivo .mdb
SQL Server	DRIVER={SQL Server};SERVER=ruta de acceso al servidor
Oracle	DRIVER={Microsoft ODBC for Oracle};SERVER=ruta de acceso al servidor
Microsoft Excel	Driver={Microsoft Excel Driver (*.xls)};DBQ=ruta física de acceso al archivo .xls; DriverID=278
Microsoft Excel 97	Driver={Microsoft Excel Driver (*.xls)};DBQ=ruta física de acceso al archivo .xls;DriverID=790
Paradox	Driver={Microsoft Paradox Driver (*.db)};DBQ=ruta física de acceso al archivo .db;DriverID=26
Texto	Driver={Microsoft Text Driver (*.txt;*.csv)};DefaultDir=ruta física de acceso al archivo .txt
Microsoft Visual FoxPro(r) (con un contenedor de bases de datos)	Driver={Microsoft Visual FoxPro Driver};SourceType=DBC;SourceDb=ruta física de acceso al archivo .dbc
Microsoft Visual FoxPro (sin un contenedor de bases de datos)	Driver={Microsoft Visual FoxPro Driver};SourceType=DBF;SourceDb=ruta física de acceso al archivo .dbf

*Tabla 4.2. Cadena de conexión ODBC.*

## Conectarse al origen de datos

ADO proporciona el objeto **Connection** para establecer y administrar las conexiones entre las aplicaciones y los orígenes de datos compatibles con OLE DB o las bases de datos compatibles con ODBC. El objeto **Connection** incorpora propiedades y métodos que se pueden utilizar para abrir y cerrar conexiones con bases de datos, y para enviar consultas de actualización de la información.

Para establecer una conexión con una base de datos, se crea una instancia del objeto **Connection**. Por ejemplo, la secuencia de comandos siguiente crea una instancia del objeto **Connection** y procede a abrir una conexión:

```
<%
' Crea un objeto Connection.
Set cn = Server.CreateObject("ADODB.Connection")
' Abre una conexión mediante la cadena de conexión OLE DB.
cnn.Open "Provider= SQLOLEDB.1; Data Source=ica; User id = icausr; Password =
icausr "
%>
```

**Nota:** La cadena de conexión no contiene espacios en blanco ni antes ni después del signo igual (=).

En este caso, el método **Open** del objeto **Connection** se refiere a la cadena de conexión.

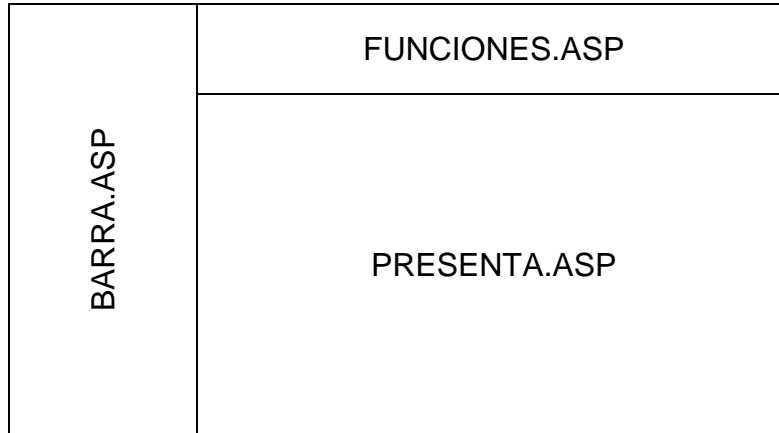
Ahora se describirán las pantallas que interactúan con el usuario.

## 4.2. Construcción de Pantallas

La creación de páginas en HTML (*HyperText Markup Language, Lenguaje de Marcación de Hipertexto*) comienza con la definición del esquema a emplear desde la página inicial. El diseño de esta página está encaminado a dar un panorama general de lo que una persona puede encontrar. Cada gráfico tiene su propio concepto, esto es, por medio de iconos y colores donde el usuario puede reconocer las opciones que tiene disponibles, sin llegar a ser molesto para la vista. En todo momento es necesario un contador de visitantes para llevar estadísticas del interés que muestra la gente sobre esta página.

ICA cuenta con un formato estándar en el diseño de su logo y tipografía, mismos que son ya utilizados en todos sus diseños de sistemas y páginas *Web*. Dicho estándar se compone de un logotipo en color blanco, con componentes RGB (246,254,249), fondo azul, con componentes RGB (0,76,186), ubicado en la parte superior izquierda de la pantalla sobre un fondo de combinaciones de azul y negro. El menú principal se localiza en la parte superior en color blanco con un fondo azul.

Basados en dicho estándar se construyeron las pantallas en HTML (Figura 4.7), dividiendo la ventana del navegador en dos columnas; la primera para el logo y menú izquierdo y la segunda, de mayor tamaño, se subdividió en dos secciones horizontales, la primera en donde sólo se ubicará el menú principal y en la segunda se presentarán las llamadas de función de cada menú.



*Figura 4.7. Definición de los Frames en la pantalla.*

En la figura se señalan los nombres de los archivos fuentes que son llamados para colocar su resultado dentro de los límites señalados por las etiquetas de columnas y líneas.

El área que se define como “FUNCIONES.ASP” contiene el encabezado formal que se utilizará durante toda la permanencia del usuario en el sitio, mostrando el nombre del lugar y el icono con la opción de retorno a ésta, la página principal.

El área que se define como “BARRA.ASP” tiene la finalidad de listar todas las opciones disponibles para el usuario en cada opción del menú. En la parte inferior de la columna se muestran algunos datos de interés al usuario como son el número de visitantes que ha recibido la página y el número de consultas que han sido realizadas dentro del sitio.

La parte definida como PRESENTA.ASP es el resultado de la selección de cada menú o búsqueda, en esta área se presentarán las pantallas principales o de resultados. En la Figura 4.8. se muestra el resultado final del desarrollo de una de las pantallas.

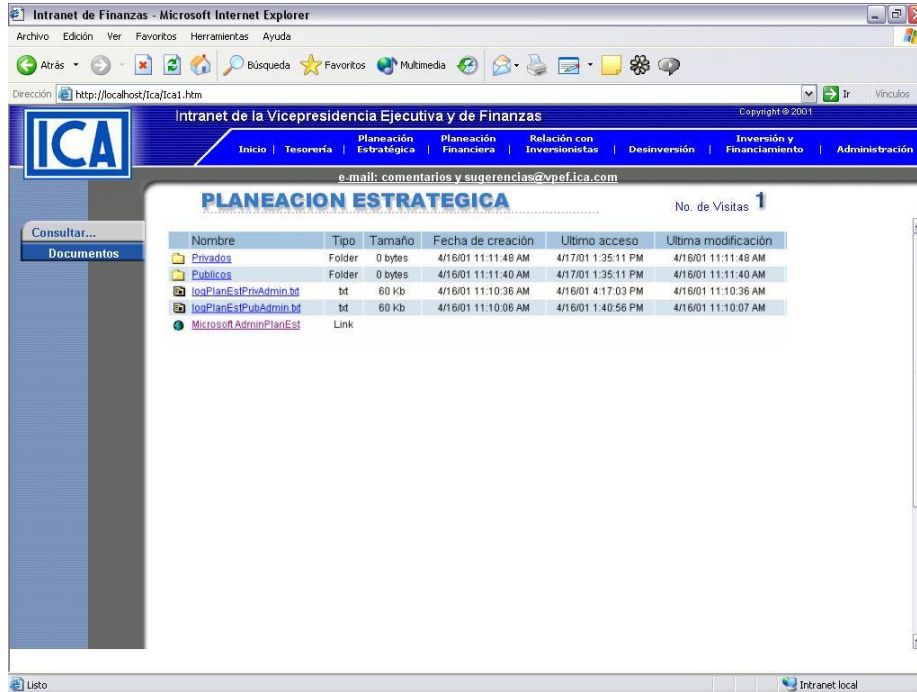


Figura 4.8. Presentación del formato general de las pantallas.

La siguiente parte del desarrollo consiste en la selección de la consulta que se desea realizar a la base de datos contenida en el sitio. Para conservar un mismo estilo durante toda la presentación del sitio se decidió conservar las dos áreas horizontales de la página principal, tomando la línea de funciones tal y como está pero en un frame definido como “ENCABEZADO”, eliminando la columna marcada como “BARRA.ASP” y asignando como el frame por omisión el área de mayor tamaño que será definida con el nombre de “PRINCIPAL”. La Figura 4.9. muestra la división final de la pantalla que será utilizada durante el resto del desarrollo.

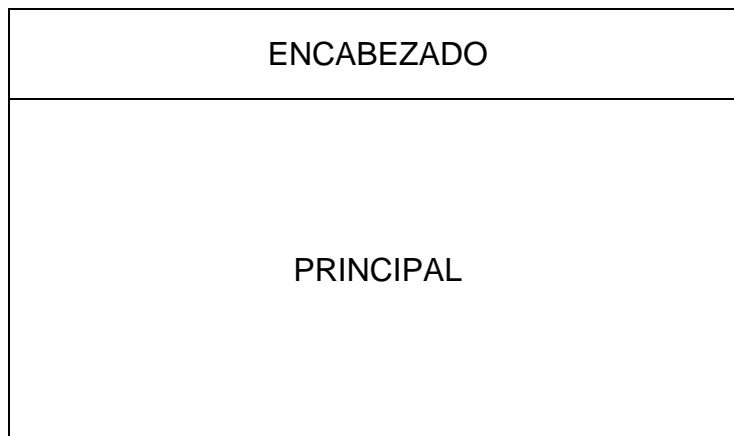


Figura 4.9. Formato de presentación de la pantalla principal

Como parte del desarrollo del sistema está necesariamente involucrado el código de creación de las pantallas, dado su volumen, parte de éste se incluye en el apéndice.

En este capítulo se construyó el sistema, enseguida se aplicarán los criterios de pruebas al mismo.



# CAPÍTULO V

## PRUEBAS Y REPORTES DEL SISTEMA

El presente capítulo comprende las pruebas realizadas y reportes que genera el sistema.

### 5.1. Pruebas del Sistema

Las pruebas son parte integral y vital del ciclo de vida del desarrollo de sistemas. Se realizan con el propósito de encontrar fallas y se establecen para mejorar la calidad del sistema. Las pruebas requieren que se descarten las ideas acerca de lo correcto que es el software desarrollado y que, al descubrir los errores, se logre superar cualquier conflicto en el sistema.

Las pruebas permiten:

- Asegurar la obtención y formalización de los requerimientos del usuario y verificar que son adquiridos de una manera completa, correcta y consistente.
- Verificar los requerimientos funcionales y estructurales para establecerlos como fundamento para la realización de las pruebas.
- Buscar y registrar fallas o defectos asociados a los requerimientos establecidos.
- Documentar los reportes para las pruebas realizadas.

Las categorías que conforman estas pruebas se presentan como aspectos importantes a definir.

### **5.1.1. Categoría de pruebas**

Las categorías que conforman las pruebas pueden ser de integración, regresión, volumen, aceptación del usuario, caja blanca, caja negra, unitarias, estáticas, funcionales y estructurales; las cuales se explican a continuación.

#### **Integración**

Son las pruebas realizadas a un grupo de programas para asegurar que los datos y controles sean pasados adecuadamente entre controles. La prueba de integración es una técnica sistemática para construir la estructura del programa mientras que, al mismo tiempo, se llevan a cabo pruebas para detectar errores asociados con la interacción. El objetivo es tomar los módulos probados en unidad y construir una estructura de programas que esté de acuerdo con lo que dicta el diseño.

#### **Regresión**

Son pruebas selectivas para detectar fallas que se hayan introducido durante las modificaciones a un sistema o componente, que permita verificar que estas modificaciones no impacten en forma negativa y que se siga cumpliendo con los requerimientos planteados.

#### **Volumen**

Son pruebas realizadas para verificar el comportamiento adecuado y eficiente de una aplicación bajo condiciones de volumen (número de operaciones), competencia de recursos (conurrencia) y carga máxima (velocidad de petición de ejecución de una operación), así como el comportamiento eficiente bajo las condiciones de volumen máximo (cantidad de datos) en las aplicaciones.

#### **Aceptación del usuario**

Son las pruebas finales ejecutadas por el usuario, para asegurar que el sistema satisfaga las necesidades de la organización o usuario final (validan que el sistema construido es el correcto).

#### **Caja blanca**

Son pruebas basadas en el conocimiento sobre la lógica y estructura internas. Usualmente dirigidas a la lógica.

#### **Caja negra**

Son pruebas funcionales basadas en los requerimientos sin conocimiento sobre cómo fue construido el sistema y usualmente dirigidas a los datos.

## **Unitarias**

Son las pruebas realizadas sobre un programa o módulo con la finalidad de encontrar problemas funcionales en la lógica y problemas técnicos en el código. La prueba unitaria centra el proceso de verificación en la menor unidad del diseño del software – el módulo. Usando la descripción del diseño detallado como guía, se prueban los caminos de control importantes, con el fin de descubrir errores dentro del ámbito del módulo. La complejidad relativa de las pruebas y de los errores descubiertos está limitada por el alcance estricto establecido por la prueba de unidad. La prueba de unidad siempre está orientada a la caja blanca y este paso se puede llevar a cabo en paralelo para múltiples módulos.

## **Estáticas**

Consiste en la revisión y validación de los documentos generados en las distintas fases de la vida de un proyecto. Verificación realizada sin ejecutar el código del sistema.

## **Funcionales**

Validan los requerimientos de la organización (lo que se supone que el sistema debe hacer), pretenden descubrir errores cometidos en la implantación de dichos requerimientos.

## **Estructurales**

Validan la arquitectura del sistema confirmando que todas sus partes funcionen sincronizadamente y que la tecnología está siendo usada apropiadamente. Se refieren a las características técnicas, como su comportamiento con grandes volúmenes de información, tiempos de respuesta, etc.

Una vez que se conoce que tipo de pruebas hay, veremos qué pasos deben seguirse para realizarlas.

### **5.1.2. Consideraciones importantes para la ejecución de las pruebas**

Es muy importante tomar en cuenta las siguientes consideraciones al realizar las pruebas, para así obtener mejores resultados en la detección de errores cometidos durante el desarrollo del sistema.

#### **Riesgos y suposiciones**

Los riesgos son aquellos factores que pueden afectar negativamente la ejecución de las pruebas. Las suposiciones son las premisas que pueden afectar positiva o negativamente la ejecución de las pruebas complicando o facilitando las actividades de las pruebas.

## **Condiciones y restricciones**

Generalmente son limitaciones o problemas de naturaleza técnica y están relacionadas con el desarrollo del proyecto en sí: la tecnología de pruebas, el estado de los ambientes de pruebas, etc.

## **Cobertura funcional de las pruebas**

Dentro de la cobertura funcional de las pruebas se deben describir y listar de manera clara y concisa las funciones a probar, así como aquellas funciones a no ser probadas aún siendo parte del proyecto, ya que son necesarias especialmente cuando se requiere explicar el por qué de su exclusión, definiendo el alcance de las pruebas y delimitando responsabilidades. Además, se debe documentar el ciclo del sistema a ser simulado con el objeto de ejecutar cada una de las funciones objeto de las pruebas. Esta simulación suele ser realizada con muestras de datos fuera de especificaciones.

## **Descripción de la arquitectura del sistema**

Para ello se consideran las especificaciones del software base sobre el cual está construido el sistema tal como la plataforma, el software de base de datos, el sistema operativo, el lenguaje de programación, etc.

Ahora veremos los elementos que son considerados para la aplicación de las pruebas.

### **5.1.3. Infraestructura de pruebas**

La infraestructura de pruebas se refiere a los elementos que se requieren para que se lleven a cabo éstas. Para entenderlos mejor explicamos cada uno de ellos.

#### **Ambiente de pruebas**

Se identifican los ambientes donde se ejecutan las pruebas, así como las características generales de los datos de prueba (qué datos necesitan y cómo se obtendrán), tomando como base el modelo de datos del proyecto. Es importante saber cuántos y cuáles datos serán seleccionados, para la estimación de la carga de trabajo necesaria para generarlos.

#### **Herramientas de pruebas**

Se identifican los productos a utilizar y el uso específico de ellos. Se debe determinar si es preciso vigilar todos los componentes o solamente algunos; el interés de vigilar a determinados componentes es justificado por la necesidad de verificar cuál es el comportamiento interno de dicho componente, esto es, cómo se realiza el procesamiento de la información. En el caso de estar solamente interesados en las entradas y salidas de los procesos, es suficiente muchas veces el verificar estas entradas y salidas sin tener en cuenta exactamente cómo se leen y cómo se generan.

## **Puntos de control y aprobaciones**

Se deben especificar los puntos de control en el transcurso de la construcción y ejecución de las pruebas, tal como el determinar los puestos de las personas que tendrán que autorizar la continuación de las pruebas acorde con el plan original o asumiendo las variaciones incorporadas del mismo.

## **Criterios de suspensión y conclusión de las pruebas**

Estos criterios se refieren a la suspensión o terminación de la ejecución de los casos de prueba cuando son necesarios otros componentes que no tienen listos, o cuando el número de los defectos encontrados sobrepasa el límite de los esperados, para lo cual es necesario regresar a la etapa de desarrollo y verificar las especificaciones.

Una vez descritas las diferentes pruebas que existen, algunas consideraciones para la aplicación de pruebas y los elementos que se utilizan en ellas, comentaremos las pruebas que se hicieron a nuestro sistema.

### **5.1.4. Pruebas aplicadas**

Tomando en cuenta la infraestructura de pruebas, se eligió que el ambiente de pruebas fuera el equipo donde se va a encontrar toda la información y el Sistema de Seguridad de Archivos de Información Empresarial (SISEA).

Se determinaron puntos de control y aprobaciones, entre cada página de HTML; se probaron todas las opciones del sistema con datos válidos y no válidos, si la página tiene más de una opción se regresaba para seguir probando todas las opciones incluidas en ella; en este caso se llevaba a cabo solamente un punto de control, que se daba cuando todas las opciones (ligas y celdas para recibir o introducir la información) de la página eran probadas, así se realizaba la aprobación o la no aprobación de la página.

Sobre la cobertura funcional de las pruebas, se probaron principalmente los accesos por medio de ligas — ya sea en texto, botones o íconos — a las páginas Web, las casillas para llenar la información requerida, el tipo de datos aceptados, también se probó que los accesos a la base de datos fueran correctos y que no hubiese problemas para desplegar la información.

Las pruebas que se aplicaron al SISEA fueron las siguientes:

- Unitarias
- De Integración
- De Aceptación del Usuario
- De Caja Negra
- Funcionales

Se eligieron éstas ya son las que nos ayudan a observar con mayor precisión los errores tanto de diseño como de programación que pueda tener nuestro sistema, así como su funcionalidad y aceptación del mismo.

## Pruebas Unitarias

Se hicieron pruebas en cada una de las páginas; las pruebas consistieron en validar los datos tanto de entrada como de salida, es decir se validó que al introducir datos correctos en las celdas y al oprimir el botón para hacer la consulta, los documentos desplegados fueran los correctos; también se validó que al introducir datos erróneos o de formato diferente el sistema desplegara un mensaje de error. En la Figura 5.1. Observamos la validación del *login* y *password* al entrar a la pantalla de administración del sistema. En caso de insertar datos correctos, el sistema debe pasar a la siguiente pantalla. De no validar el *login* y el *password* el sistema solicita nuevamente la información; y en caso de no insertar ningún dato, la pantalla muestra un mensaje solicitando la inserción de los mismos, como se muestra en la Figura 5.2.

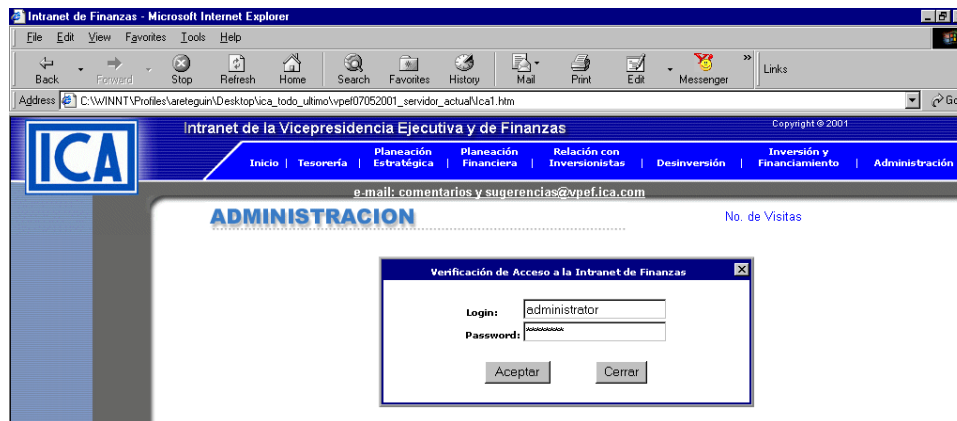


Figura 5.1. Prueba de validación en la pantalla de administración.

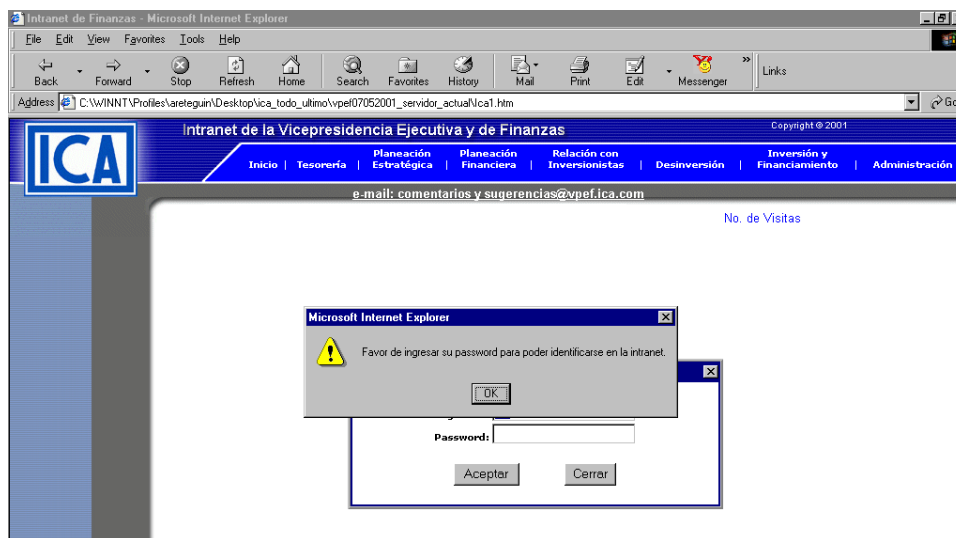


Figura 5.2. Pantalla que muestra el error al no insertar datos.

## Pruebas de Integración

Al tener todo el sistema integrado, se realizaron las pruebas para verificar que las ligas estuvieran bien y que no hubiera ningún problema al navegar en el sistema, al introducir datos ni al desplegarlos. El sistema se comportó muy bien en este aspecto, no se detectó ningún problema al tener el sistema integrado. Se comprobó la navegación en la página con base en el mapa del sitio que se muestra en la Figura 5.3.

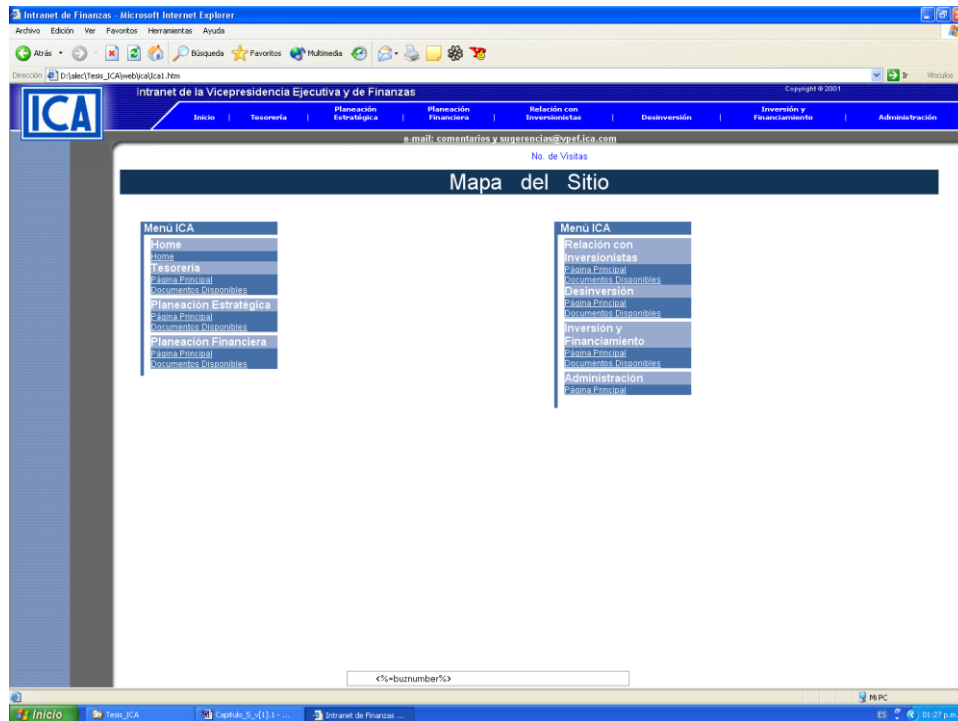


Figura 5.3. Prueba de Integración.

## Pruebas de Aceptación del Usuario

Se realizaron las pruebas de aceptación del usuario, en donde el usuario navegó por el sistema, ver Figura 5.4, y realizó consultas, tanto de documentos como de ligas, como se muestra en las figuras 5.5 y 5.6. Al usuario le agradó el sistema, la forma rápida y fácil en que se puede llevar a cabo una consulta de información de cualquier región del país.

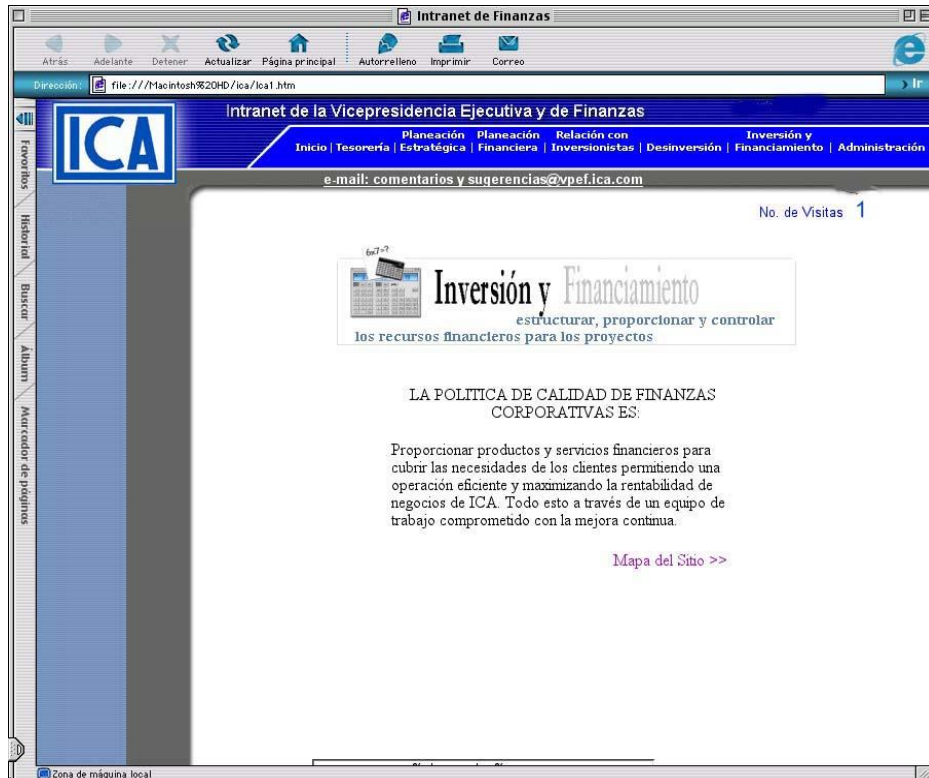


Figura 5.4. Pantalla de Navegación del Sitio.

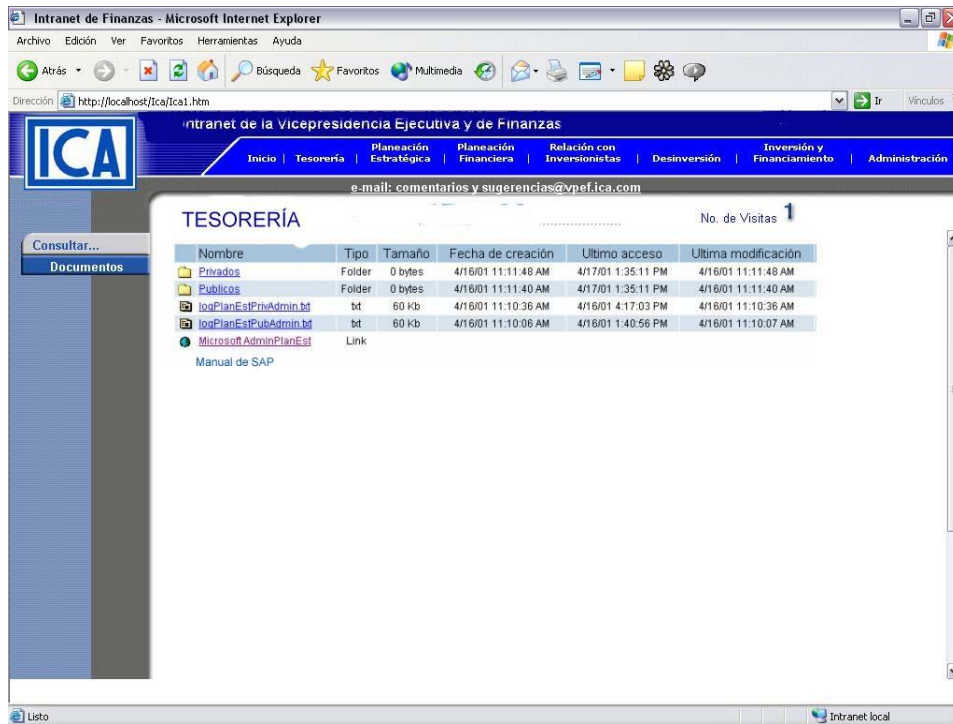


Figura 5.5. Lista de Documentos y Ligas.



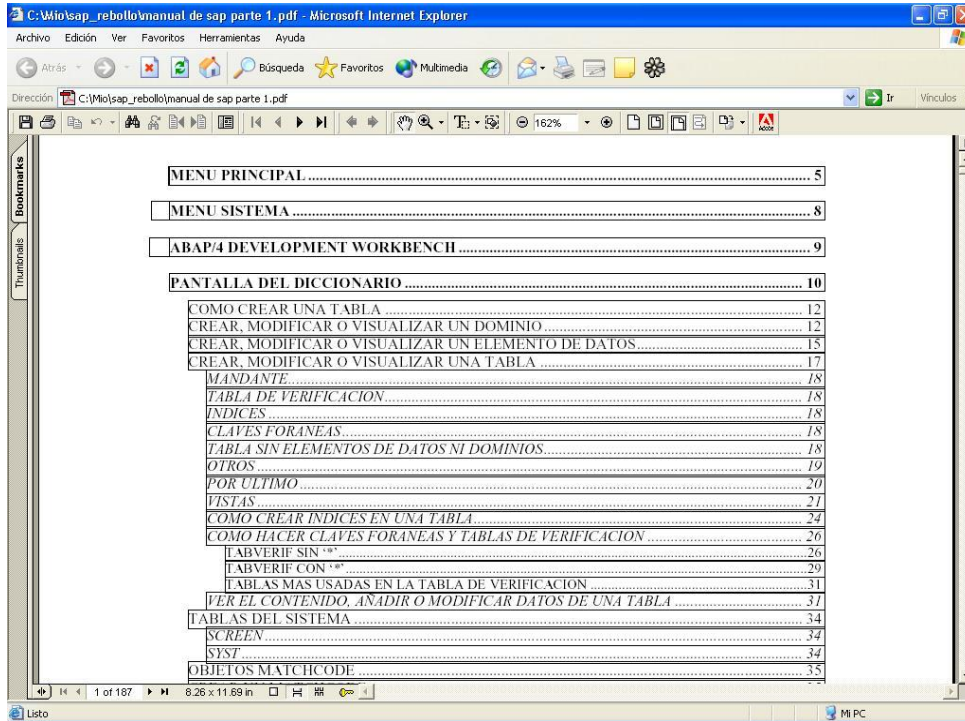


Figura 5.6. Pantalla de Consulta de Documentos.

## Pruebas de Caja Negra

Estas pruebas consistieron en entrar a la página e ir viendo las listas de los documentos de los diferentes departamentos de Finanzas y consultar aquellos a los que se tiene acceso, como se muestra en la Figura 5.7.

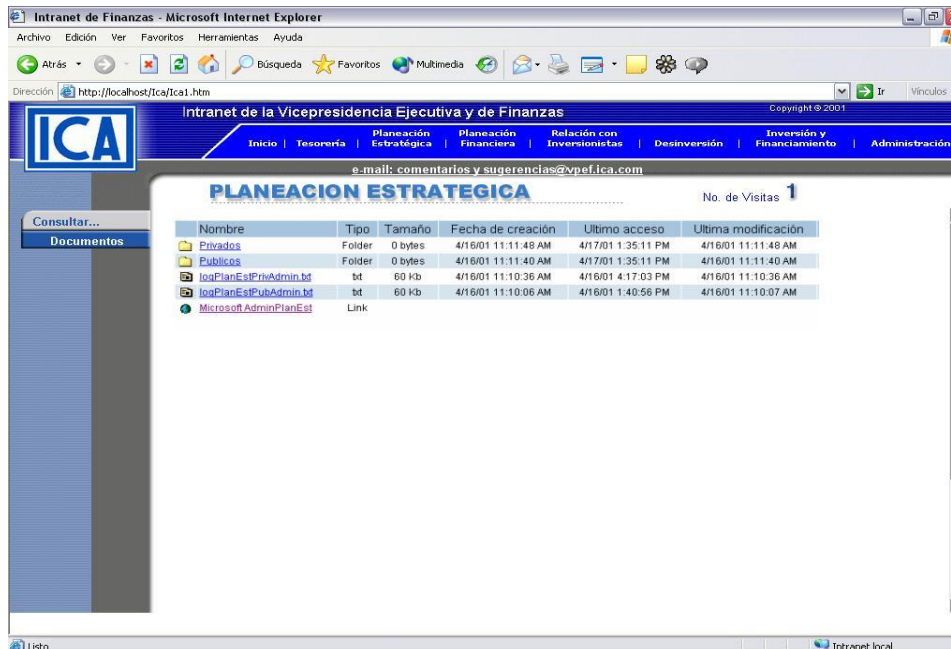


Figura 5.7. Prueba de Caja Negra.

## Pruebas Funcionales

En estas pruebas se validó que el sistema cumpliera con los requerimientos del usuario; para ello se dieron de alta diferentes tipos de usuarios, Figura 5.8, documentos a los cuales se les asignaron los usuarios que pueden verlos, Figura 5.9, ligas a sitios, Figura 5.10; se hicieron las consultas a documentos, se validó que los documentos fueran consultados sólo por los usuarios que tenían derecho a verlos, ver Figura 5.11 y se verificó que el *ticker* mostrara la información introducida, como muestra la Figura 5.12.



Figura 5.8. Alta de Usuarios.

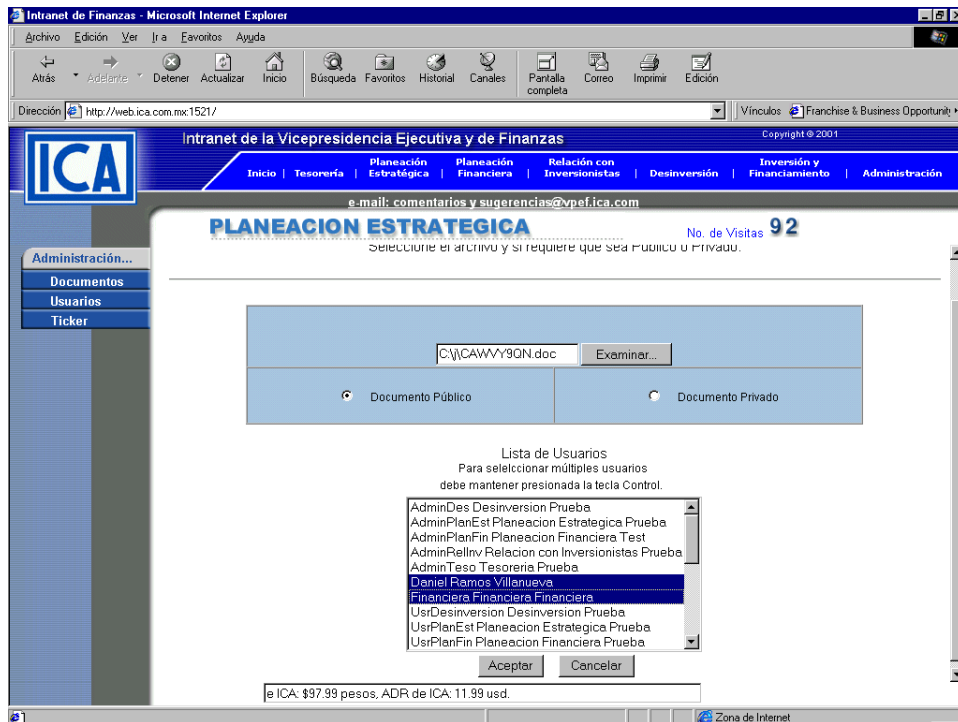


Figura 5.9. Pantalla de Alta de Documentos.

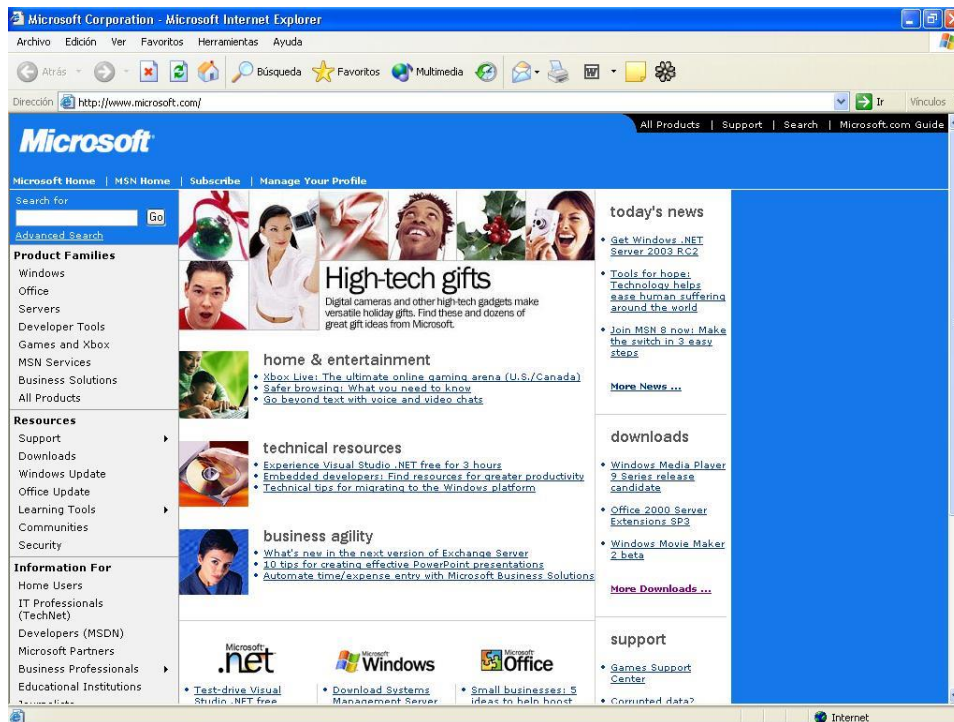


Figura 5.10. Pantalla de Ligas a Sitios de Internet.

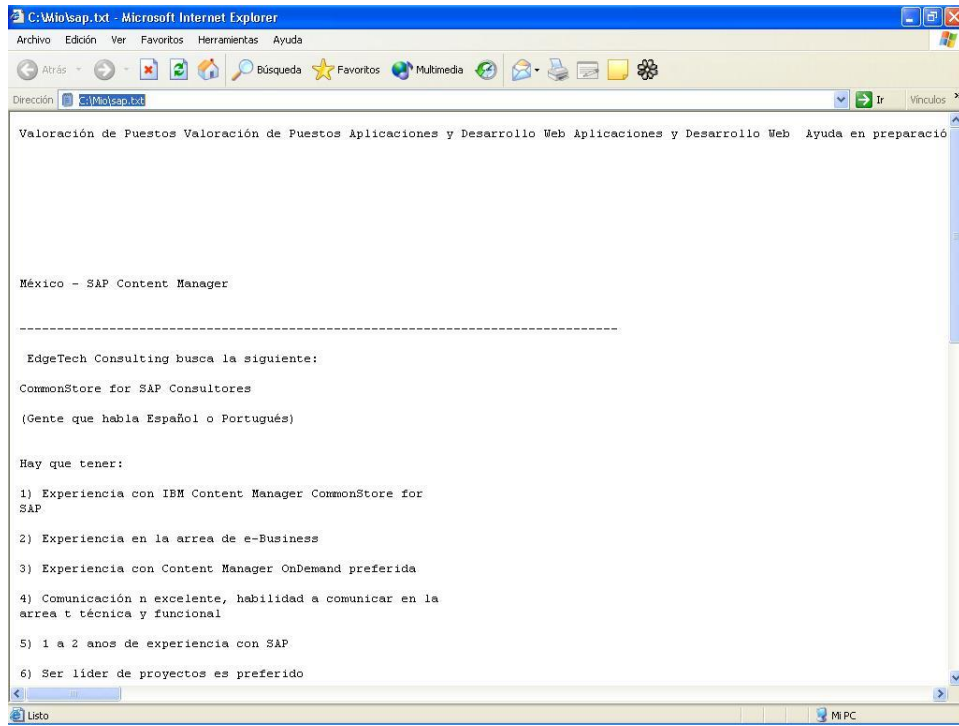


Figura 5.11. Pantalla Consulta de documentos.

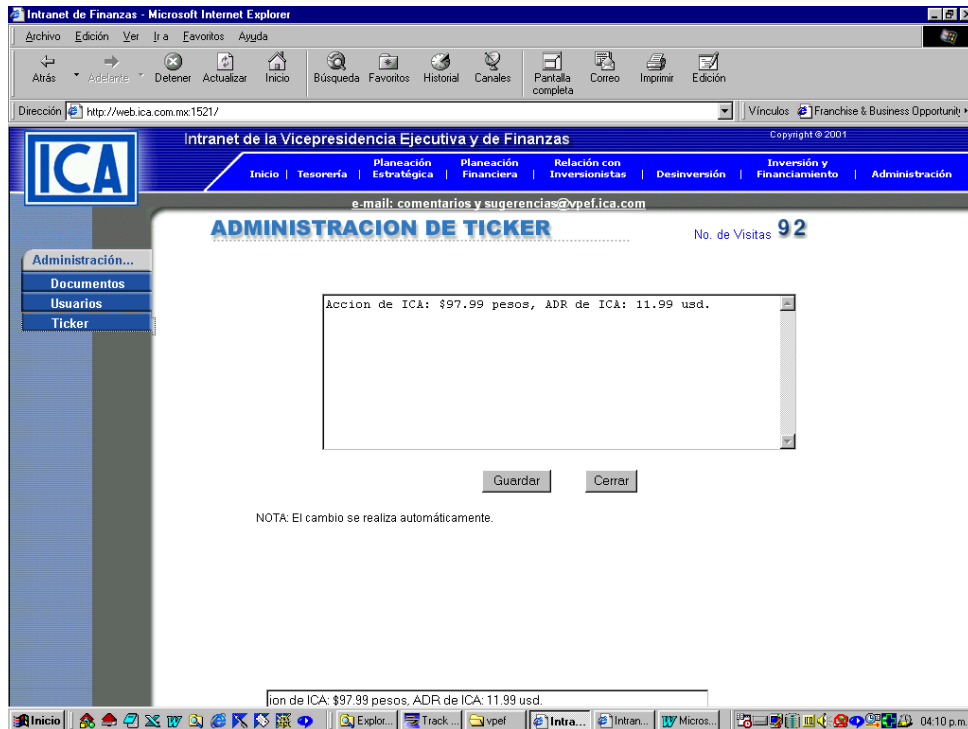


Figura 5.12. Pantalla de Alta de Información del Ticker.

## 5.2. Generación de reportes del sistema

Los reportes se generan mensualmente y son presentados a los directivos de la empresa para que éstos valoren el funcionamiento y avance del proyecto, con ello se mide los cambios pertinentes para el progreso y mejoramiento del sistema.

Los tipos de reportes son los siguientes:

- Registro de usuarios por mes (para cada Departamento).
- Archivos dados de alta por mes (para cada Departamento).
- Total de accesos por día, mes y año.
- Consultas por día, mes y año.

### 5.2.1. Reporte de registro de usuarios

El sistema está dividido en 6 departamentos, así que los reportes se generarán para cada uno de ellos, siendo entregados para su revisión. Además de obtener otro reporte de general de todos los usuarios.

El reporte que se presenta en la Figura 5.13 corresponde a los usuarios registrados el mes de octubre en el departamento de Tesorería, dividido en cuatro campos que muestran el id\_Usuario, Nombre Usuario, Fecha Alta y Tipo de Usuario.



**REPORTE DE REGISTRO DE USUARIOS PARA DEPARTAMENTO DE TESORERIA.**  
Mes: Octubre

<i>Id_Usuario</i>	<i>Nombre Usuario</i>	<i>Fecha Alta</i>	<i>Tipo de Usuario</i>
13	Agustín Ramírez Chávez	10/10/02	Usuario
01	Alberto Molina Pérez	10/10/02	Administrador
09	Armando Terioco Nardo	10/10/02	Usuario
05	Carlos Ruíz Díaz	10/10/02	Usuario
03	Cruz Loreto Pérez	10/10/02	Usuario
16	Emilio Godínez Sánchez	10/10/02	Usuario
19	Félix Salgado Patrón	10/10/02	Usuario
10	Flor Martínez Pérez	10/10/02	Usuario
08	Francisco Ruíz Gil	10/10/02	Usuario
11	Gerardo Martínez Moreno	10/10/02	Usuario

Figura 5.13. Reporte de registro de usuarios de Tesorería.

El reporte de la Figura 5.14. muestra los usuarios registrados en el mes de octubre del departamento de Planeación Estratégica.



**REPORTE DE REGISTRO DE USUARIOS PARA DEPARTAMENTO DE PLANEACIÓN ESTRATEGICA.**

Mes: Octubre

<i>Id_Usuario</i>	<i>Nombre Usuario</i>	<i>Fecha Alta</i>	<i>Tipo de Usuario</i>
13	Alfredo Chávez Ramírez	10/10/02	Usuario
09	Armando Terioco Nardo	10/10/02	Usuario
18	Fernando Andaluz Juárez	10/10/02	Usuario
08	Francisco Ruiz Gil	10/10/02	Usuario
02	Gustavo Olivos Cavaría	10/10/02	Usuario
05	Jesús Ruiz Díaz	10/10/02	Usuario
20	Jorge González García	10/10/02	Usuario
12	José Haláis Moreno	10/10/02	Usuario
01	Juan Pérez Molina	10/10/02	Administrador
06	Juan Valdezpino Sosa	10/10/02	Usuario
19	Leopoldo Salgado Patrón	10/10/02	Usuario
16	Luis Jiménez Sánchez	10/10/02	Usuario
15	Marcela Flores García	10/10/02	Usuario
04	Maria López Guzmán	10/10/02	Usuario
07	Marisol Peza Ruiz	10/10/02	Usuario
14	Miguel Ramírez Ramírez	10/10/02	Usuario
03	Mónica Pérez Loreto	10/10/02	Usuario
10	Rogelio Pérez Martines	10/10/02	Usuario
11	Rulo Martínez Moreno	10/10/02	Usuario
17	Sara Flores Tina	10/10/02	Usuario

*Figura 5.14. Reporte de registro de usuarios de Planeación Estratégica.*

El reporte de la Figura 5.15. muestra el registro de usuarios del mes de octubre del departamento de Relación con Inversionistas.



**Relación con Inversionistas**

**REPORTE DE REGISTRO DE USUARIOS PARA DEPARTAMENTO DE RELACIÓN CON INVERSIONISTAS.**

Mes: Octubre

<i>Id_Usuario</i>	<i>Nombre Usuario</i>	<i>Fecha Alta</i>	<i>Tipo de Usuario</i>
14	Adrián Peza Tapia	10/10/02	Usuario
08	Alejandro Fernández Martínez	10/10/02	Usuario
11	Arturo Valdez Noriega	10/10/02	Usuario
09	Carolina Alvarado Esparza	10/10/02	Usuario
16	Cecilia Gutiérrez Santiago	10/10/02	Usuario
02	Daisy Fernández López	10/10/02	Usuario
18	Diegos Castro Castillo	10/10/02	Usuario
03	Eduardo González Vázquez	10/10/02	Usuario
10	Eduardo Reyes Cortés	10/10/02	Usuario
01	Elizabeth Estrada Pérez	10/10/02	Administrador
17	Ericka Bocanegra Romero	10/10/02	Usuario
20	Lucia Lara Carvajal	10/10/02	Usuario
19	Magdalena De la Rosa Díaz	10/10/02	Usuario
04	Maite Jiménez Pérez	10/10/02	Usuario
05	Montserrat Gómez Martínez	10/10/02	Usuario
06	Reina Espinosa Martínez	10/10/02	Usuario
07	Tania Maria Garrido Jiménez	10/10/02	Usuario
15	Verónica Pérez Medina	10/10/02	Usuario
12	Yadira Ocampo Escobar	10/10/02	Usuario
13	Yanin Acosta Cuellar	10/10/02	Usuario

Figura 5.15. Reporte de registro de usuarios de Relación con Inversionistas.

En la Figura 5.16. mostramos el reporte de usuarios registrados en el mes de octubre del departamento de Planeación Financiera.



Figura 5.16. Reporte de registro de usuarios de Planeación Financiera.



El reporte de la Figura 5.17. muestra el registro de usuarios del mes de octubre del departamento de Desinversión.



**REPORTE DE REGISTRO DE USUARIOS PARA DEPARTAMENTO DESINVERSIÓN.**  
Mes: Octubre

<i>Id_Usuario</i>	<i>Nombre Usuario</i>	<i>Fecha Alta</i>	<i>Tipo de Usuario</i>
18	Alicia Pérez Pérez	10/10/02	Usuario
11	Anahi Valdez Bocanegra	10/10/02	Usuario
01	Ashianty Pastelin Gutiérrez	10/10/02	Administrador
20	Calipso Rodríguez Alberti	10/10/02	Usuario
05	Camilo Bautista Contreras	10/10/02	Usuario
06	Homero Montiel Ontiveros	10/10/02	Usuario
04	Javier González Jiménez	10/10/02	Usuario
17	Jordi Martínez Murray	10/10/02	Usuario
02	Josué Montoya López	10/10/02	Usuario
08	Lourdes Monroy Jiménez	10/10/02	Usuario
07	Nelly Del Valle	10/10/02	Usuario
13	Oscar Álvarez Guzmán	10/10/02	Usuario
15	Rebeca Sosa Moctezuma	10/10/02	Usuario
12	Rosario Paredes Páez	10/10/02	Usuario
03	Silvia Ortiz Santiago	10/10/02	Usuario
19	Susana Portilla Suárez	10/10/02	Usuario
10	Ulises González Vázquez	10/10/02	Usuario
16	Víctor Pedrosa Dávalos	10/10/02	Usuario
09	Viridiana Gutiérrez Torres	10/10/02	Usuario

*Figura 5.17. Reporte de registro de usuarios de Desinversión.*

El reporte de la Figura 5.18. muestra el registro de usuarios del mes de octubre del departamento de Inversión y Financiamiento.



**REPORTE DE REGISTRO DE USUARIOS PARA DEPARTAMENTO DE INVERSIÓN Y FINANCIAMIENTO.**

Mes: Octubre

<i>Id_Usuario</i>	<i>Nombre Usuario</i>	<i>Fecha Alta</i>	<i>Tipo de Usuario</i>
08	Agustín Iturriaga Pérez	10/10/02	Usuario
05	Cristian López Sánchez	10/10/02	Usuario
06	Eliu García Noriega	10/10/02	Usuario
19	Enrique Merino García	10/10/02	Usuario
20	Estephanie Martínez Ocampo	10/10/02	Usuario
15	Fátima Pada Dante	10/10/02	Usuario
10	Isabel Padilla Gómez	10/10/02	Usuario
04	Leticia Zúñiga	10/10/02	Usuario
02	Luis Cervantes Mendoza	10/10/02	Usuario
11	María Méndez Batiros	10/10/02	Usuario
09	Martha Palacios Barrera	10/10/02	Usuario
03	Melina Paulett Jardón	10/10/02	Usuario
07	Oliva Rosas Galicia	10/10/02	Usuario
12	Paris Olivos Rojas	10/10/02	Usuario
16	Patricia Tapia Cruz	10/10/02	Usuario
14	Roberto Vázquez González	10/10/02	Usuario
17	Teresa Dávila Fuente	10/10/02	Usuario
01	Vanely Romero Ruiz	10/10/02	Administrador
18	Vernarda Díaz Frias	10/10/02	Usuario
13	Yaneth Espinosa Gales	10/10/02	Usuario

*Figura 5.18. Reporte Inversión y Financiamiento.*

Otro reporte generado por el sistema es el de archivos dados de alta cada mes.

### 5.2.2. Reporte de archivos dados de alta

Se compone de 4 campos: id\_Documento, Nombre Documento, Fecha Alta y Usuario que Registro. Existe un reporte por departamento, en la Figura 5.19. se muestra el del departamento Desinversión.



**REPORTE DE ARCHIVOS DADOS DE ALTA.**

Mes: Octubre

<i>Id_Documento</i>	<i>Nombre Documento</i>	<i>Fecha Alta</i>	<i>Usuario que Registro</i>
01	Contador.xls	10/10/02	Ashianty Mendoza Gutiérrez
02	Desinversion.xls	10/10/02	Josué Montoya López
03	No_involucramiento.ppt	21/10/02	Silvia Ortiz Santiago
04	No_Involucramiento2.ppt	10/10/02	Javier González Jiménez
05	Plan1.doc	10/10/02	Camilo Bautista Contreras
06	Resultado_1trimestre.doc	10/10/02	Homero Montiel Ontiveros
07	Desinversion1.xls	18/10/02	Nelly Del Valle
08	Desinversion1.5.xls	10/10/02	Lourdes Monroy Jiménez
09	Plan2.doc	10/10/02	Viridiana Gutiérrez Torres
10	Resultado_2trimestre.doc	10/10/02	Ulises González Vázquez
11	Desinversion4.xls	10/10/02	Ana Leticia Montaña
12	Desinversion5.xls	12/10/02	Rosario Paredes Páez
13	Desinversion2.xls	10/10/02	Oscar Álvarez Guzmán
14	Plan3.doc	13/10/02	Galo Mendez Rodríguez
15	Resultado_3trimestre.doc	10/10/02	Rebeca Sosa Moctezuma
16	No_Involucramiento 5.ppt	11/10/02	Víctor Pedrosa Dávalos
17	No_Involucramiento 6.ppt	16/10/02	Jordi Martínez Murria

*Figura 5.19. Reporte de archivos dados de alta.*

Otra información que nos proporciona el sistema es el reporte de accesos por día, mes y año.

### 5.2.3. Reporte de accesos por día

El número de accesos de usuarios al sistema queda registrado de manera diaria y se presenta en forma de lista. Es importante resaltar que este número de accesos es hacia la página del sistema, de tal manera que no necesariamente el usuario debe pertenecer al área de Finanzas. Este reporte (Figura 5.20.) permitirá mantener una estadística periódica, y considerar en un momento determinado si existe mayor o menor demanda de los servicios del sistema y del equipo, para que en su momento, pueda ser prevenido de alguna posible disminución en el rendimiento.



 <b>SISEA</b> <b>ÁREA DE FINANZAS</b> <b>PERÍODO: 08/07/02 A 14/07/02</b>	
Día	Total de Accesos
Lunes	684
Martes	851
Miércoles	956
Jueves	860
Viernes	245
Sábado	66
Domingo	11

*Figura 5.20. Total de accesos por día.*

### 5.2.4. Reporte de accesos por mes y año

De la misma manera en que el sistema genera la información de los accesos por día, también es posible mostrarla por año y por mes, como se muestra en las Figuras 5.21 y 5.22.



**SISEA**  
**ÁREA DE FINANZAS**  
**PERÍODO: 2001 A 2002**

<b>Año</b>	<b>Total de Accesos</b>
2001	191,521
2002	229,873

*Figura 5.21. Total de accesos por año.*



**SISEA**  
**ÁREA DE FINANZAS**  
**PERÍODO: 01/02 A 07/02**

<b>Mes</b>	<b>Total de Accesos</b>
Enero	15,741
Febrero	17,524
Marzo	19,245
Abril	46,350
Mayo	24,750
Junio	10,100
Julio	17,291

*Figura 5.22. Gráfica Total de accesos por mes.*

Los reportes anteriores mostraron los totales de accesos por día, mes y año registrados por el sistema, pero también existen reportes de consultas, que son referidos a aquellos usuarios que tuvieron el acceso a la información confidencial, por lo que debieron acceder a través de una contraseña.

### 5.2.5. Reporte de consultas por día

Es precisamente a este tipo de accesos a los que el reporte se refiere al nombrarlo de consultas. La Figura 5.23 muestra el registro del Total de las consultas por día.



**SISEA**  
**ÁREA DE FINANZAS**  
**PERÍODO: 08/07/02 A 14/07/02**

Día	Total de Consultas
Lunes	484
Martes	510
Miércoles	656
Jueves	680
Viernes	252
Sábado	36
Domingo	11

*Figura 5.23. Total de consultas por día.*

### 5.2.6. Reporte de consultas por mes y año

El período por mes y año también está contemplado en los reportes de consultas, por lo que se muestran en las Figuras 5.24 y 5.25.



**SISEA**  
**ÁREA DE FINANZAS**  
**PERÍODO: 2001 A 2002**

Año	Total de Accesos
2001	137,083
2002	156,144

Figura 5.24. Total de consultas por año.



**SISEA**  
**ÁREA DE FINANZAS**  
**PERÍODO: 01/02 A 07/02**

Mes	Total de Accesos
Enero	18,024
Febrero	15,524
Marzo	15,924
Abril	11,246
Mayo	14,246
Junio	10,247
Julio	11,247

Figura 5.25. Gráfica Total de consultas por mes.

Una vez mostradas las pruebas del sistema y los reportes que éste genera, seguiremos con la fase de liberación en el equipo y lugar en que será instalado y puesto en funcionamiento.

# CAPÍTULO VI

## INSTALACIÓN Y LIBERACIÓN DEL SISTEMA

En este capítulo se presentará la implementación final del sistema, en donde se describe la forma en que se realizaron la puesta a punto del servidor y la instalación del programa, se describen también algunos puntos importantes para la eficiente operación del sistema, como son la administración, el control de cambios, el mantenimiento y el soporte técnico.

### 6.1. Instalación del sistema

Para hablar de una instalación del sistema, es necesario contar con una serie de información tal que nos permita llegar al objetivo del sistema mismo, es decir, contemplar todos los elementos necesarios para que el sistema desarrollado tenga un funcionamiento adecuado. Dentro de los elementos que se mencionan se encuentran los de hardware y software, que son con los que la empresa cuenta y que serán utilizados para la instalación del sistema, así como para mantener una constante vigilancia de su comportamiento. Desde luego, todo sistema debe contar con una documentación que permita al usuario conocer todos los puntos concernientes a su instalación, por ello, se presentará la parte del producto terminado, así como sus procesos que lo integran.

Para proceder a la implantación del sistema se requiere desarrollar las siguientes actividades:

- Instalación y puesta en operación del servidor Windows NT 2000 Server.
- Instalación y puesta en operación del servidor SQL Server 2000.
- Instalación y puesta en operación del sistema en IIS (*Internet Information Server*).



La infraestructura de la red dentro de la compañía, y dentro de la red local del área de sistemas en donde se instalará el servidor del sistema, ya está implementada y cumple con la normatividad establecida por la empresa para las redes que utilicen Windows NT 2000 como sistema operativo de red.

Dada esta normatividad, la implementación del servidor a utilizar para el Sistema de seguridad de archivos de información empresarial (SISEA) debe cumplir en su totalidad con dichas normas. Es por ello que a continuación se presenta la Normatividad para redes bajo Windows NT 2000.

### **6.1.1. Normatividad para redes bajo Windows NT 2000**

Esta normatividad deberá seguirse por todas aquellas personas encargadas directamente de la instalación y puesta a punto de servidores Windows NT 2000 Server. El cumplimiento de esta normatividad garantizará la operación óptima del servidor y evitará cualquier problema con la interconexión, además de permitir la creación de una plataforma eficiente y estandarizada para proporcionar los servicios de red corporativos.

La presente normatividad está enfocada a lograr reducir al máximo las labores relacionadas con la administración de redes.

- **Requerimientos de hardware para los servidores**

La parte más importante y crítica de una red la constituye el servidor. La función del servidor es la de compartir sus recursos a la red, ya sean servicios de disco, impresión o aplicaciones cliente/servidor. También debe proporcionar una plataforma robusta para la ejecución de aplicaciones de misión crítica.

Por todo lo anterior, se hace necesario que la computadora destinada a ser servidor cumpla con requisitos específicos para realizar de manera eficiente todas las funciones anteriormente mencionadas. Éstos incluyen los concernientes a sus requerimientos mínimos de hardware y software, sus requerimientos eléctricos y su ubicación física.

Con el fin de ajustarse a diferentes requerimientos de uso y aplicación se han definido dos tipos de servidores: el servidor departamental y el servidor *Web* corporativo.

El servidor departamental está enfocado a la ejecución de los servicios básicos de red (archivos, impresión, *software* institucional) y está proyectado para dar servicio al interior de la empresa.

Por otra parte, el servidor *Web* corporativo está enfocado solamente a la ejecución de la aplicación desarrollada, en conjunto con todo lo que se requiere para su buen funcionamiento.

Los requerimientos mínimos de hardware para el usuario son:

Servidor Pentium II (250 MHz)  
Memoria RAM DIMM de 64 MB  
Disco duro de 100 MB libres en DD,(IDE)  
Tarjeta de red PCI (Operación Dual 10 BASET/100 BASET X )  
Monitor SVGA color 14 “

Los requerimientos mínimos de hardware para el servidor departamental son:

Servidor Pentium III (550 MHz)  
Memoria RAM DIMM de 128 MB  
Unidad de CD-ROM 24X (IDE) –interna – (opcional)  
Disco duro de 20 GB (IDE)  
Drive interno de 3.5”  
Tarjeta de red PCI (Operación Dual 10 BASET/100 BASET X )  
2 puertos seriales y 1 puerto paralelo  
Mouse (bus)  
Monitor SVGA color 14 “

Los requerimientos mínimos de hardware para el servidor *Web* corporativo son:

Servidor Pentium IV (1.4 MHz)  
Memoria RAM DRIMM de 512 MB  
Unidad de CD-ROM 24X (SCSI) –interna – reiniciable  
Unidad de Respaldo 24 GB ( SCSI) – interna –  
2 Discos duros de 80 GB (SCSI)  
Drive interno de 3.5”  
3 bahías disponibles para crecimiento futuro  
3 ranuras de expansión libres PCI  
3 ranuras de USB para aditivos opcionales (escáners, video digital, fotografía, etc.)  
Tarjeta de red PCI (Operación Dual 10 BASET/100 BASET X)  
2 puertos seriales y 1 puerto paralelo  
Mouse (bus)  
Monitor SVGA color 17”  
UPS (*Uninterruptible Power Supplies*, Sistema Ininterrumpido de Potencia) con interfaz al servidor (1200 Watts)

Es importante señalar que las marcas y modelos a adquirir (CPU, UPS, tarjeta de red) estén registrados en el HCL (*Hardware Compatibility List*) de Windows NT 2000 Server.

- **Requerimientos de software para los servidores**

Windows NT Server 2000

SQL Server 2000 (sólo en el servidor departamental)

IIS (sólo en el servidor *Web* corporativo)

- **Requerimientos eléctricos de los servidores**

Los servidores deben estar alimentados por un UPS, y a su vez contar con una interfaz a éste para protegerlo de forma automática en caso de falla en el suministro eléctrico. Gracias a esta interfaz, es posible apagar de forma segura al servidor, evitando así la pérdida de información y posibles daños al equipo.

- **Ubicación física de los servidores**

Los servidores deben ser colocados en un área exclusiva y libre del paso (de preferencia en un lugar cerrado y con aire acondicionado), con el fin de que puedan ser operados sin dificultad cuando sea necesario.

El cumplimiento de esta normatividad permitirá:

- Una operación óptima del servidor.
- Evitar conflictos de operación.
- La integración e interconexión en la red.
- Facilitar actualizaciones futuras.
- La estandarización de los parámetros de operación.
- Facilitar el soporte técnico.

Una vez establecida la normatividad para los servidores, se debe realizar la puesta a punto, la cual comprende la instalación y configuración inicial de los servidores.

### **6.1.2. Puesta a punto de los servidores**

La ubicación de ambos servidores será dentro del Área de Sistemas ya que cumple con las condiciones eléctricas y de clima establecidas en la normatividad.

Para el caso de la puesta a punto del servidor departamental debe considerarse la instalación y configuración de SQL mostrada más adelante en este capítulo.

La puesta a punto del servidor *Web* corporativo en donde se instalará el SISEA contempla todas las normas presentadas anteriormente, por ello se le registró el nombre y el rol del servidor *Web* en el Área de Sistemas.

En el IIS los nombres para los directorios de los servicios son los directorios por omisión, es decir:

\\inetpub\ftproot	(Home-Directory para los servicios de FTP )
\\inetpub\gopherroot	(Home-Directory para los servicios de Gopher)
\\inetpub\wwwroot	(Home-Directory para los servicios de Web )

El nombre asignado al servidor y debido a que se firmará dentro del dominio empresarial será el siguiente:

SISEA

Se pidió la asignación de una dirección IP al área respectiva, otorgando la siguiente:

Dirección IP: 207.248.171.5

Una vez vistos los requerimientos pertenecientes a la normatividad para redes bajo Windows NT 2000, pasaremos a la instalación de SQL Server 2000 y del SISEA.

### 6.1.3. Instalación de SQL Server 2000 y del Sistema SISEA

A continuación se presentarán los pasos de instalación para el manejador de base de datos SQL Server 2000.

- Instalación de SQL Server 2000

La pantalla que se muestra en la Figura 6.1 es la principal de instalación en la cual se selecciona la opción de “Componentes de SQL Server 2000”.



Figura 6.1. Parte Principal de la instalación.

La siguiente pantalla es para seleccionar qué componentes se desean instalar, en el cual elegimos Instalar Servidor de Base de Datos, Figura 6.2.

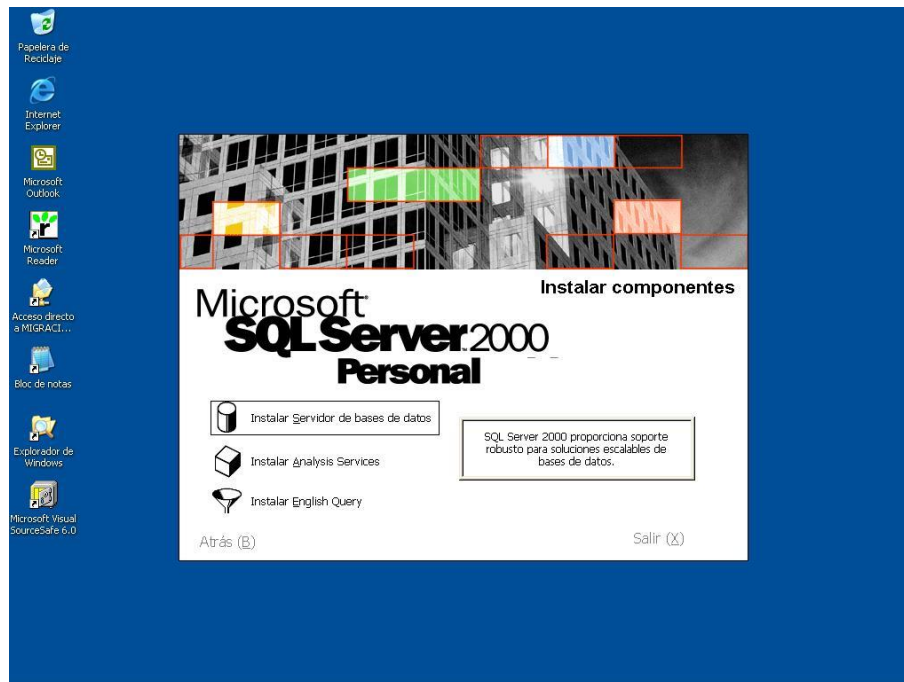


Figura 6.2. Servidor de Base de Datos.

Se crea una nueva instancia para la instalación del manejador de base de datos (Figura 6.3).

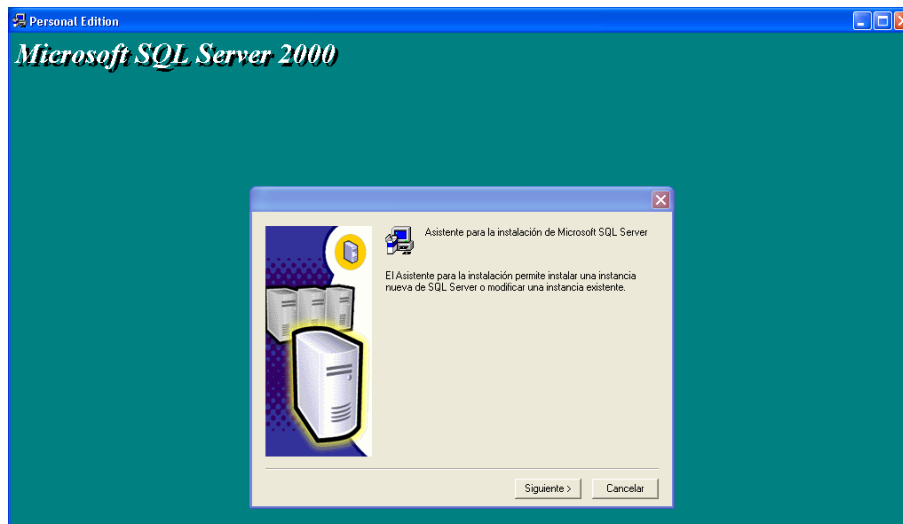


Figura 6.3. Instancia de Instalación.

Se proporciona el nombre del servidor, que en este caso va a ser remoto (Figura 6.4).

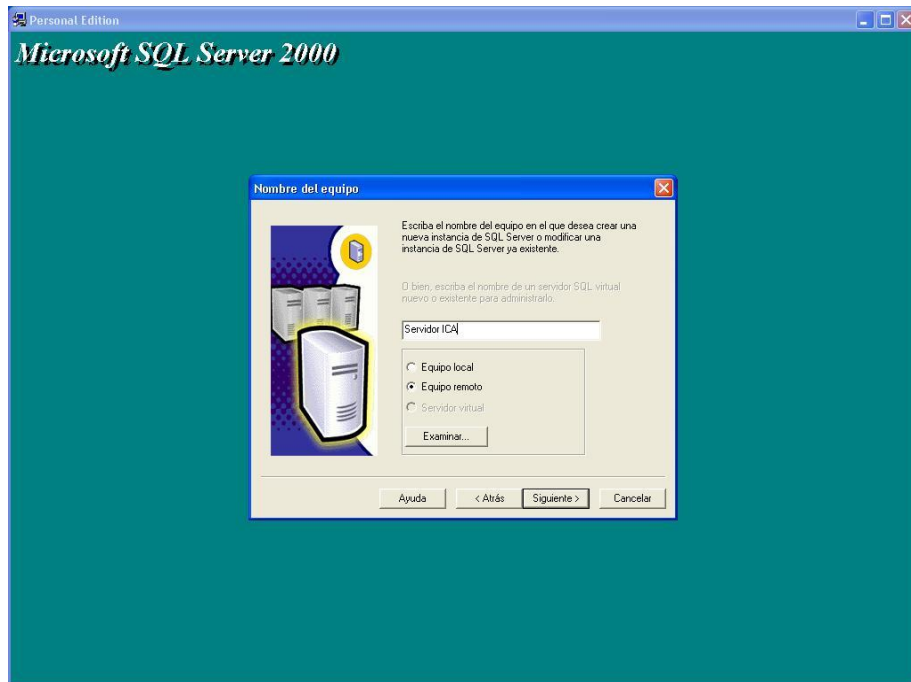


Figura 6.4. Asignación del Servidor.

Se confirma la nueva instancia y se instalan las herramientas de desarrollo (Figura 6.5).

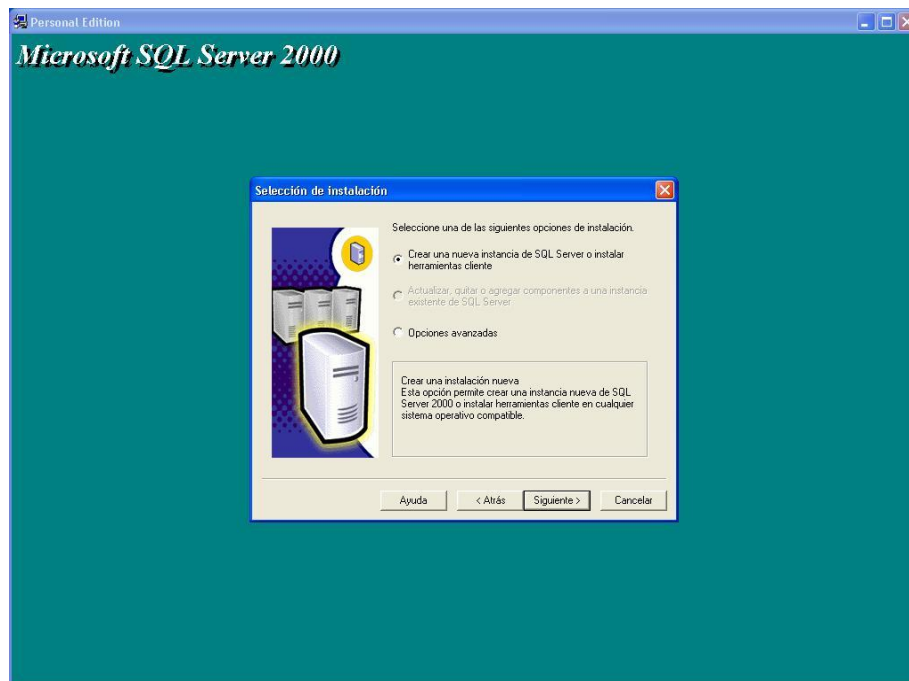


Figura 6.5. Instalación de Herramientas.

Se proporciona un usuario y la compañía que adquirió el producto (Figura 6.6).

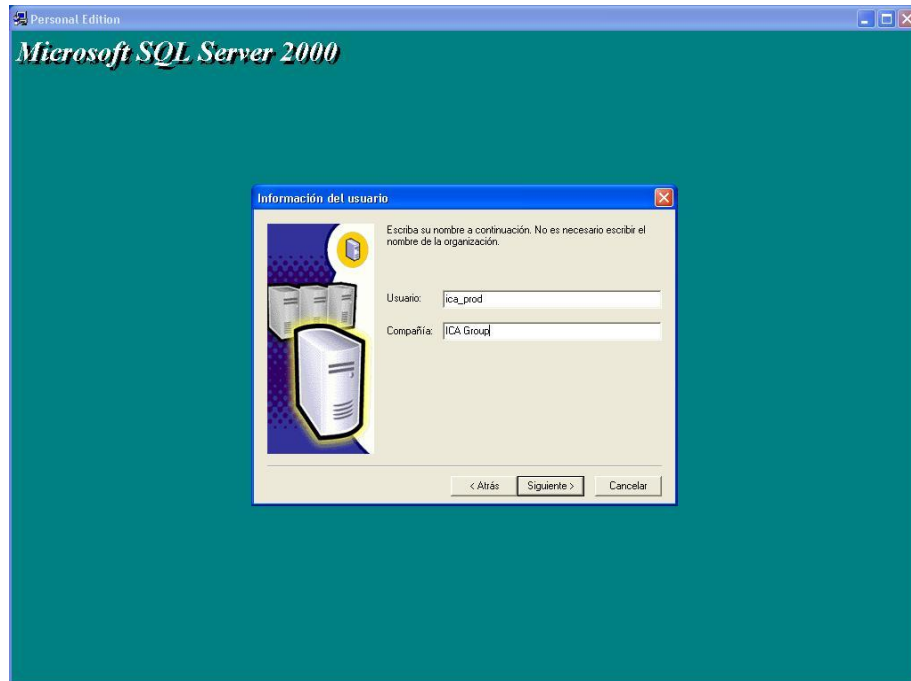


Figura 6.6. Datos Requeridos para la instalación.

Selección de herramientas, se seleccionan las herramientas cliente y servidor ya que se utilizarán las dos para la consulta y desarrollo de ésta (Figura 6.7).

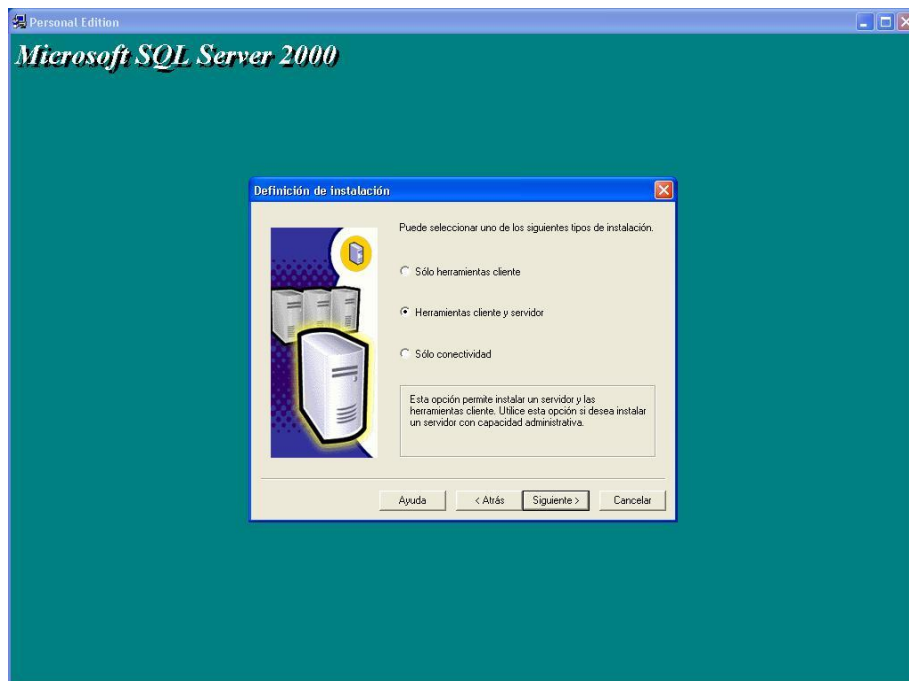


Figura 6.7. Selección de la opción de herramientas.

A continuación se presenta la opción del tipo de instalación, en el cual optamos por la típica, ya que ésta instala los componentes comunes del manejador (Figura 6.8).

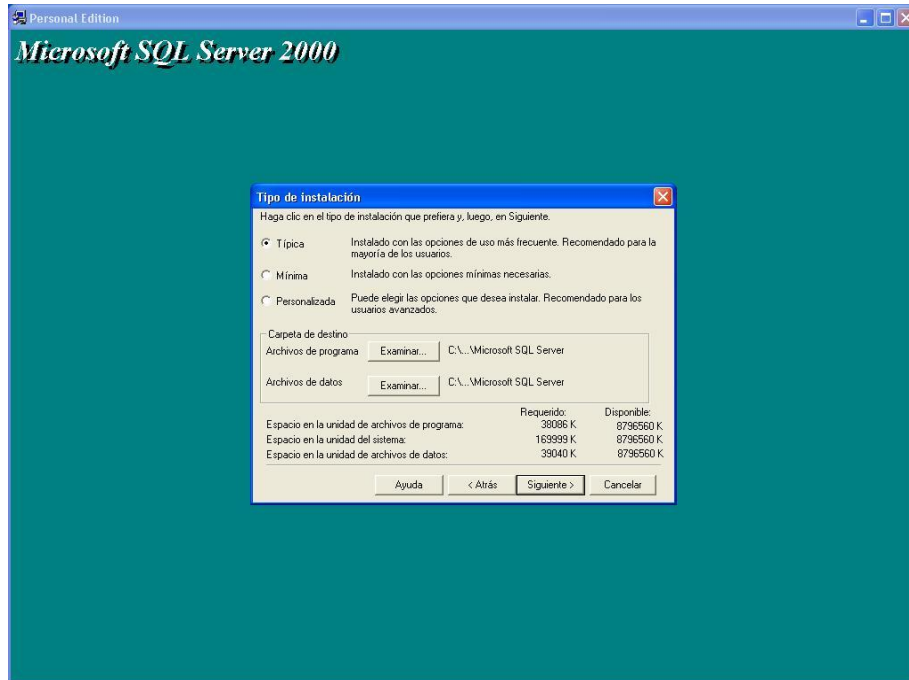


Figura 6.8. Selección tipo de información.

Por último, en la culminación de la instalación se opta por la reinicialización del sistema ya que éste debe cargar los nuevos controladores para el manejador (Figura 6.9).

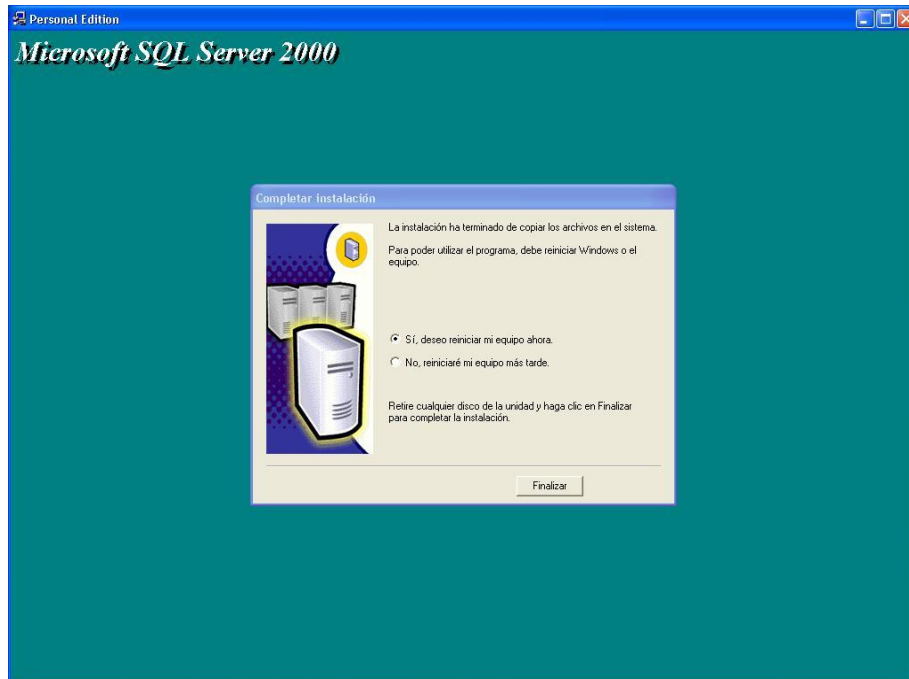


Figura 6.9. Culminación de la instalación.



- Instalación del sistema SISEA

Para poder instalar el sistema SISEA se requiere crear una instancia virtual en el servidor de Web IIS. A continuación se mencionan los pasos a seguir para la instalación de la instancia virtual:

En primer lugar se realiza una llamada al asistente para crear el directorio virtual (Figura 6.10).

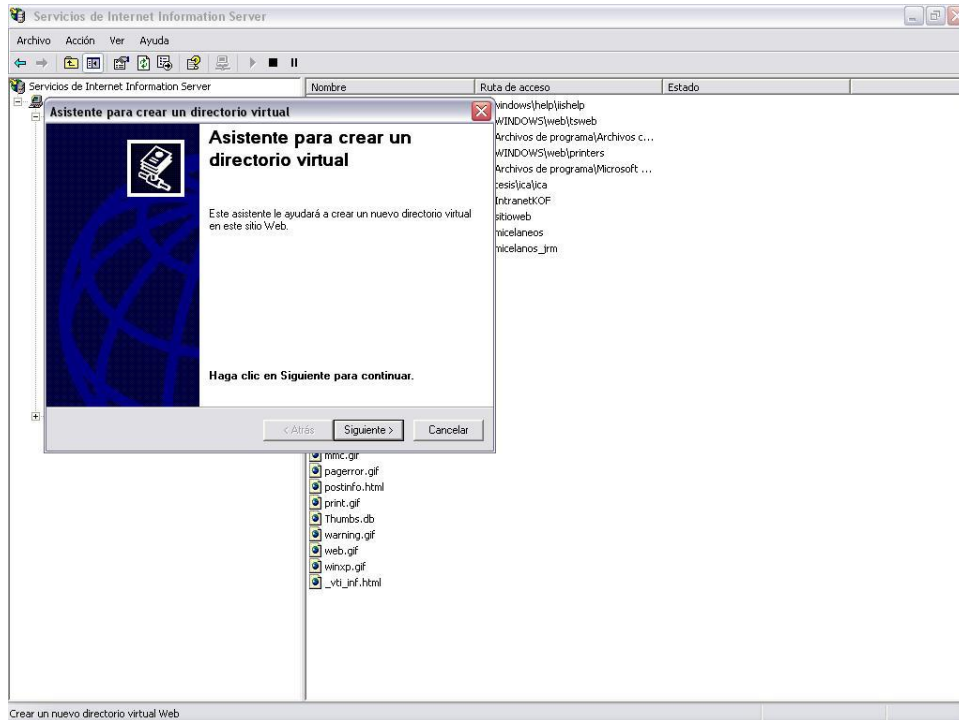


Figura 6.10. Asistente de instalación.

Asignación del nombre virtual del sitio a crear, en este caso se le proporciona el nombre de Icaweb por políticas de la propia empresa, Figura 6.11.

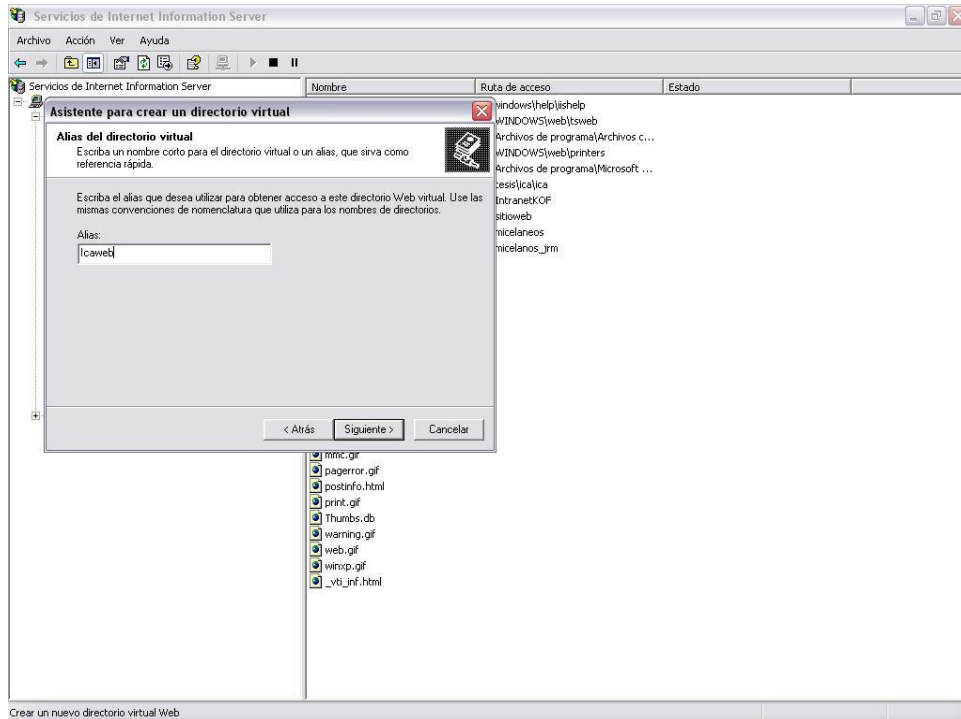


Figura 6.11. Nombre sitio virtual.

En la siguiente pantalla se selecciona el directorio físico, en donde se colocará el sistema para que pueda interactuar con el usuario y la base de datos (Figura 6.12).

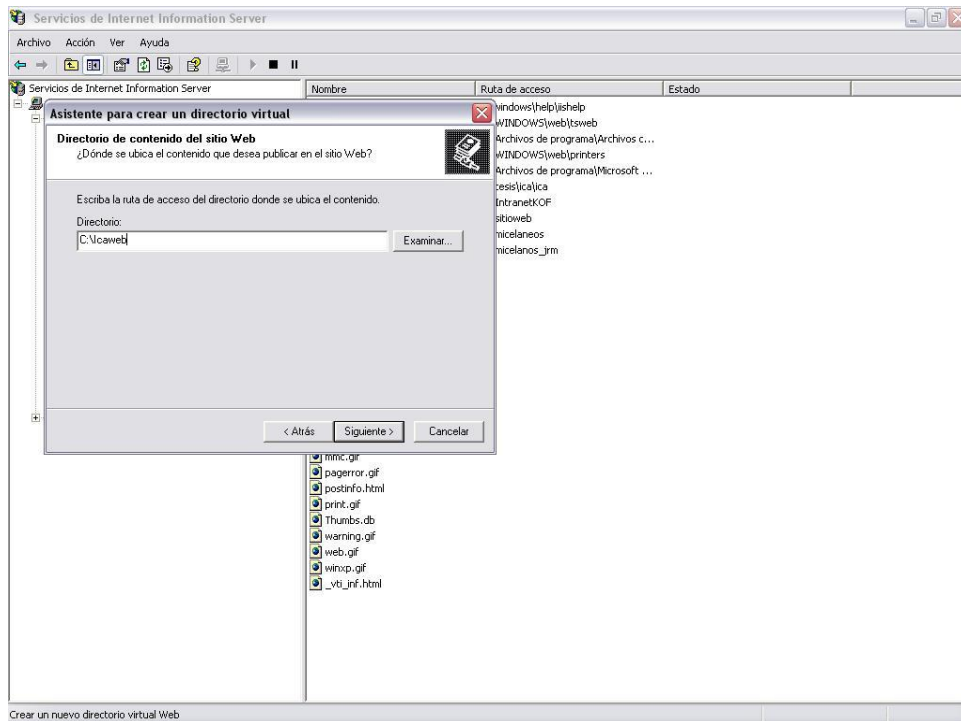


Figura 6.12. Asignación del directorio físico.

Por último se copian las carpetas y archivos correspondientes al sistema SISEA y que se encuentran en el directorio Icaweb, necesarios para que el sistema esté íntegro y funcionando para su utilización, Figura 6.13.

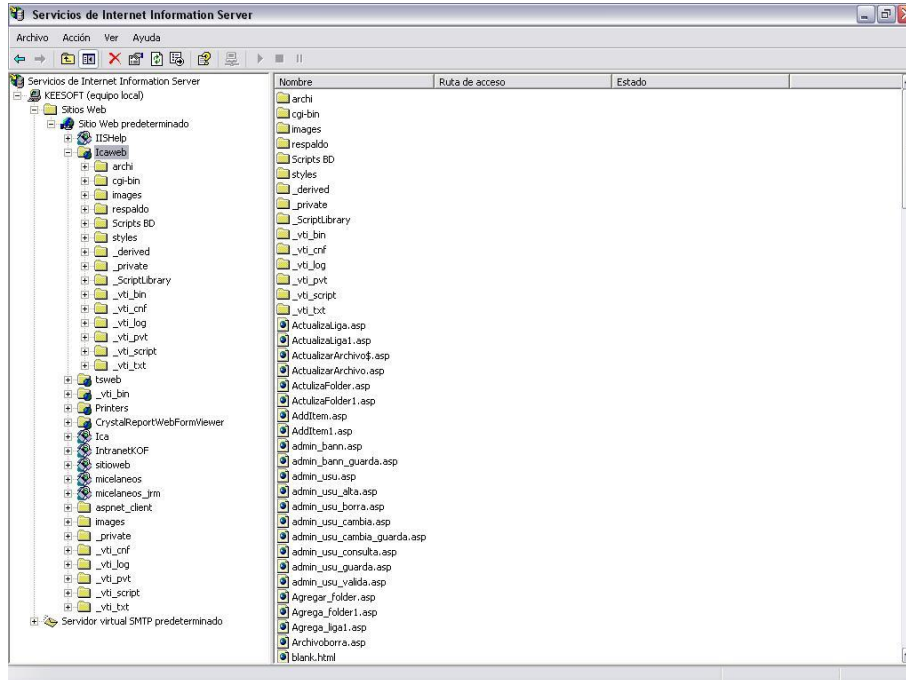


Figura 6.13. Listo el directorio virtual.

Ya una vez estando en funcionamiento el servidor explicaremos como se administrará éste.

#### 6.1.4. Administración del Sistema

Para la administración del Sistema de Seguridad de Archivos de Información Empresarial, se ha sugerido que el encargado de estas labores sea el propio administrador del servidor del Área de Sistemas de la empresa, como primera opción mientras se capacita a otro empleado con los conocimientos suficientes en informática del Área de Finanzas.

Esta determinación se tomó dadas las características y perfil (capacidad técnica y conocimientos de informática) que se requieren para realizar las labores de administración tanto del servidor como del sistema.

El administrador deberá asumir además de las responsabilidades descritas en la normatividad de la empresa para servidores bajo sistema operativo Windows NT, las siguientes responsabilidades para el SISEA:

- Administración de archivos, usuarios, ligas (altas/bajas/cambios) y del *ticker*.
- Administrar la seguridad del sistema, mediante la asignación de privilegios y contraseñas.
- Realizar los respaldos de la información (se recomienda cada 15 días).
- Aclaraciones, atención de dudas y comentarios de los usuarios sobre el funcionamiento del Sistema.
- Brindar soporte técnico a los usuarios del sistema.
- Llevar un control de los cambios realizados al sistema, así como altas, bajas y cambios en la información de las bases de datos del sistema.

Acerca de este último punto, ahora se muestra el control de cambios que se realizarán para el sistema.

### **6.1.5. Control de cambios**

Aun cuando el sistema está creado bajo los requerimientos del usuario, no están descartadas posibles modificaciones al sistema, con las que puedan aumentar su capacidad o agregar algún requerimiento más.

Es por esta razón que deberá llevarse un control de cambios realizados al sistema, indicando datos importantes como son: la fecha, motivo de la modificación, tipo de modificación, solución propuesta, persona que la autorizó y persona que realizó el cambio.

Este control servirá para poder llevar un histórico del sistema y de cada una de sus modificaciones por fallas, limitaciones del diseño y/o cualquier aumento de necesidades.

Durante los primeros 3 meses de la puesta en marcha del sistema, los desarrolladores de éste brindaremos el soporte técnico necesario. Éste se realizará a través del administrador del sistema, es decir, si se encuentran errores de funcionalidad, los usuarios deberán informar al administrador del sistema de éstos, con el fin de que los resuelva, en caso de que éste no pueda resolverlos consultará con nosotros para que se le brinde la asesoría necesaria y/o realizar el mantenimiento correctivo respectivo y solucionar de inmediato el o los errores presentados.

Para que el usuario pueda tener la capacidad de manejo del sistema, se requiere de una documentación para éste, la cual se muestra a continuación.

### **6.1.6. Documentación para el usuario**

Para mostrar la documentación del usuario se presentarán las pantallas que utilizará éste y la explicación del uso de las mismas. Las pantallas que se explicarán son las siguientes:

- Pantalla de Acceso al Usuario.
- Pantalla de Consulta de Información Confidencial.
- Pantalla de Administración de Usuarios.
- Pantalla de Administración de los documentos.

Cabe mencionar que estas pantallas simplifican la labor del usuario ya que cada una de ellas es semejante para los diferentes departamentos.

### Pantalla de Acceso al Usuario

Este módulo sirve para validar el acceso de cada usuario al sistema por medio de un *login* o nombre de usuario y una contraseña. En la Figura 6.14 se muestra la pantalla donde el usuario debe insertar estos dos campos, hacer *click* en Aceptar y el sistema valida el acceso y sus privilegios.

El sistema guarda estos datos en un *cookie* con el objeto de que el usuario no tenga que validarse en cada pantalla. Con una sola ocasión que haga este proceso será suficiente para validar sus privilegios en las pantallas. Dependiendo el usuario y la contraseña que insertemos, serán los privilegios de acceso a pantallas y documentos que tendrá cada usuario. Con los permisos validados, el usuario puede acceder a las pantallas de consulta.

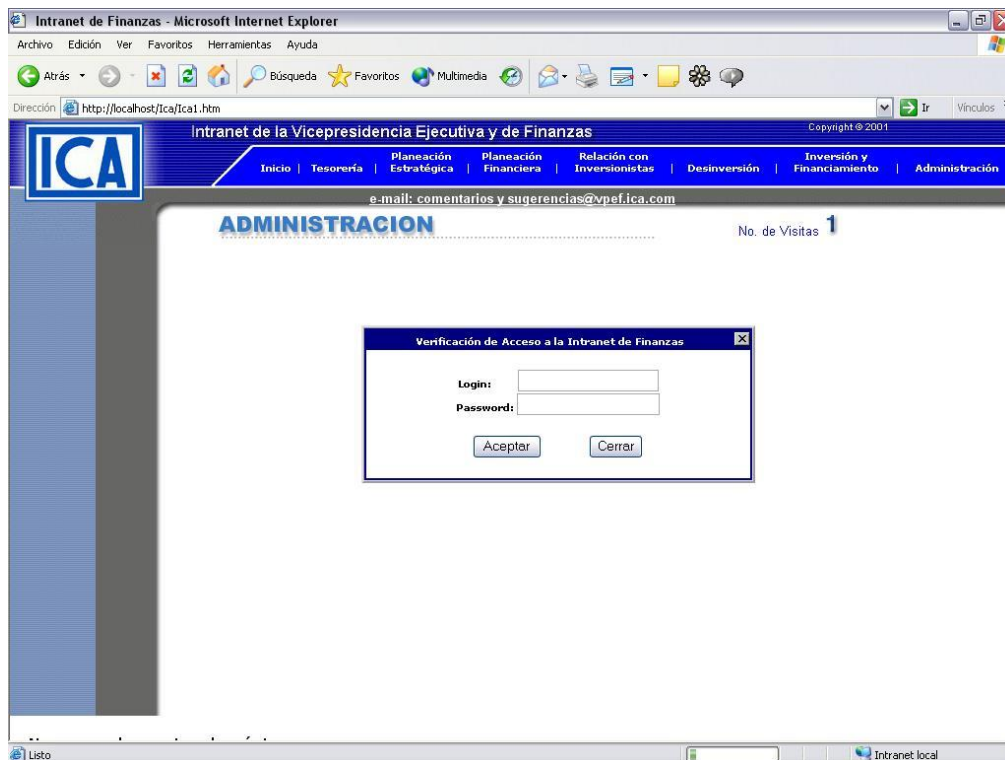


Figura 6.14. Acceso al Usuario.

## Pantalla de Consulta de Información Confidencial

Mediante la Pantalla de consulta de información confidencial podemos consultar los documentos contenidos tanto en *Folders* públicos como en *Folders* privados, dependiendo de los privilegios de cada usuario.

El acceso a esta funcionalidad es por medio de la pantalla de validación de usuarios mostrada con anterioridad. En caso de introducir una cuenta de Superusuario, se mostrará primero una pantalla de selección del área a la que se quiere acceder y posteriormente la lista de los documentos, *folders* y *links* disponibles dentro del área seleccionada, como se muestra en la Figura 6.15. En caso contrario, si es una cuenta de Administrador, directamente se presentan los elementos disponibles dentro del área a la que pertenezca el usuario.

Si es un usuario común, aparecen sus propios archivos y los que otros administradores le hayan permitido el acceso. Para establecer todos estos permisos, fue necesario crear una pantalla en la que sólo el Superusuario lleva a cabo la administración de usuarios.

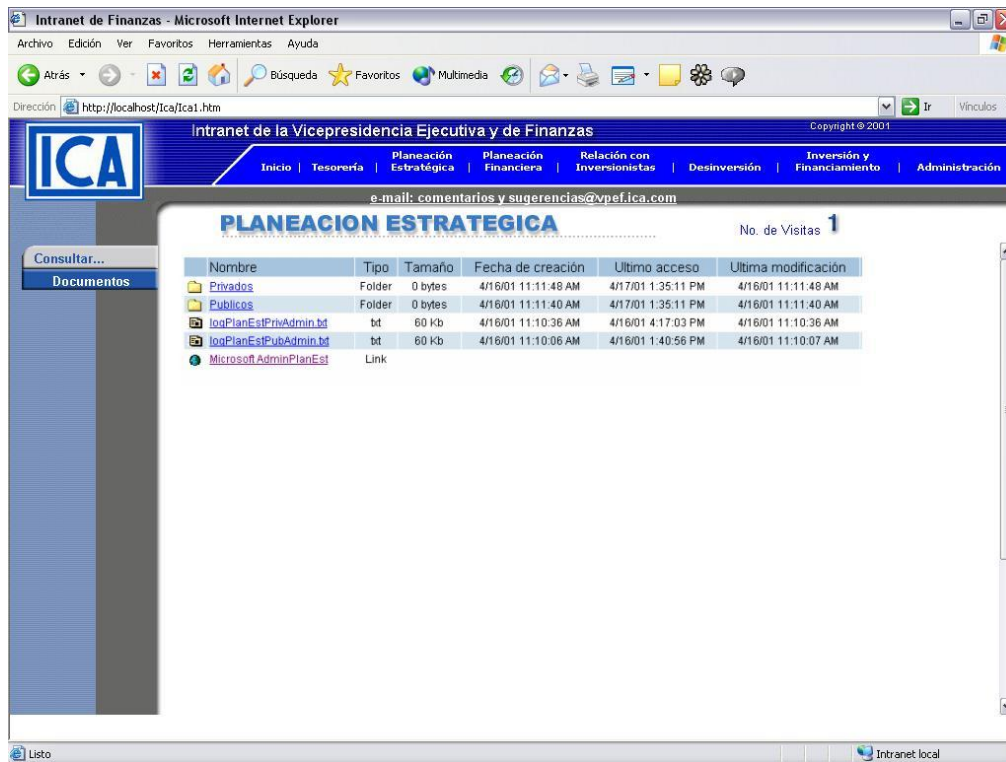


Figura 6.15. Consulta de información.

## Pantalla de Administración de Usuarios

Este módulo sirve para actualizar, agregar, consultar y borrar usuarios que consultan y/o administran los documentos del Áreas de Finanzas.

El acceso a esta funcionalidad es por medio del botón “Administración” del menú principal. Se requiere proporcionar un usuario y contraseña de Superusuario, válidos para acceder esta funcionalidad restringida.

Una vez que se valida al usuario se presenta una pantalla con la lista de opciones, de la cual debe seleccionarse “Usuarios”.

En la lista que se presenta en la Figura 6.16 aparecen los datos de los usuarios dados de alta, la opción para borrarlos, el identificador del usuario, la fecha de alta y la opción para editar sus privilegios y/o datos generales.

Borrar	Id Usuario	Nombre Usuario	Fecha de Alta	Editar
	<a href="#">super</a>	super super super	29/03/2001	
	<a href="#">rinv</a>	rinv rinv rinv	10/04/2001	
	<a href="#">t</a>	tesoreria tesoreria tesoreria	10/04/2001	
	<a href="#">e</a>	estrategia estrategia estrategia	10/04/2001	
	<a href="#">f</a>	Financiera Financiera Financiera	10/04/2001	
	<a href="#">ad</a>	ad ad ad	16/04/2001	
	<a href="#">adinteso</a>	AdminTeso Tesoreria Prueba	16/04/2001	
	<a href="#">adminplaneest</a>	AdminPlanEst Planeacion Estrategica Prueba	16/04/2001	
	<a href="#">adminplanfin</a>	AdminPlanFin Planeacion Financiera Test	16/04/2001	
	<a href="#">adminrelinv</a>	AdminRelInv Relacion con Inversionistas Prueba	16/04/2001	
	<a href="#">admindes</a>	AdminDes Desinversion Prueba	16/04/2001	
	<a href="#">super1</a>	superusuario1 superusuario1 superusuario1	16/04/2001	
	<a href="#">usrtesoreria</a>	UsrTesoreria Tesoreria Prueba	16/04/2001	
	<a href="#">usrplaneest</a>	UsrPlanEst Planeacion Estrategica Prueba	16/04/2001	
	<a href="#">usrrelinv</a>	UsrRelInv Relacion con Inversionistas Prueba	16/04/2001	
	<a href="#">usrdes</a>	UsrDesinversion Desinversion Prueba	16/04/2001	
	<a href="#">x</a>	xxx	16/04/2001	
	<a href="#">usrplanfin</a>	UsrPlanFin Planeacion Financiera Prueba	16/04/2001	
	<a href="#">a</a>	a a a	16/04/2001	

Figura 6.16. Administración de usuarios.

**Consultar datos del usuario.** Debe hacer *click* sobre el *ID Usuario* correspondiente. Con esto se muestran los datos completos del usuario. En esta pantalla no se puede modificar ningún valor. Para salir de esta pantalla se pulsa el botón de *Cerrar*.

**Agregar un usuario.** Se utiliza el botón de *Agregar*. Posteriormente se llenan los datos que se solicitan y para guardar el registro con esos datos se utiliza el botón de *Agregar* (Figura 6.17.). Para cancelar la operación se utiliza la opción de *Cerrar*.

**Modificar datos del usuario.** Se coloca el puntero del ratón en la columna de Editar del registro a modificar que tienen un icono de un lápiz (✎), se pulsa un *click* y posteriormente se muestran los datos. En esta pantalla (Figura 6.18.) se pueden hacer los cambios pertinentes y seleccionar *Cambiar* para guardarlos. Los campos que no se pueden modificar son la *Fecha de Alta* y el *User ID*. Para cancelar la operación se pulsa el botón de *Cerrar*.

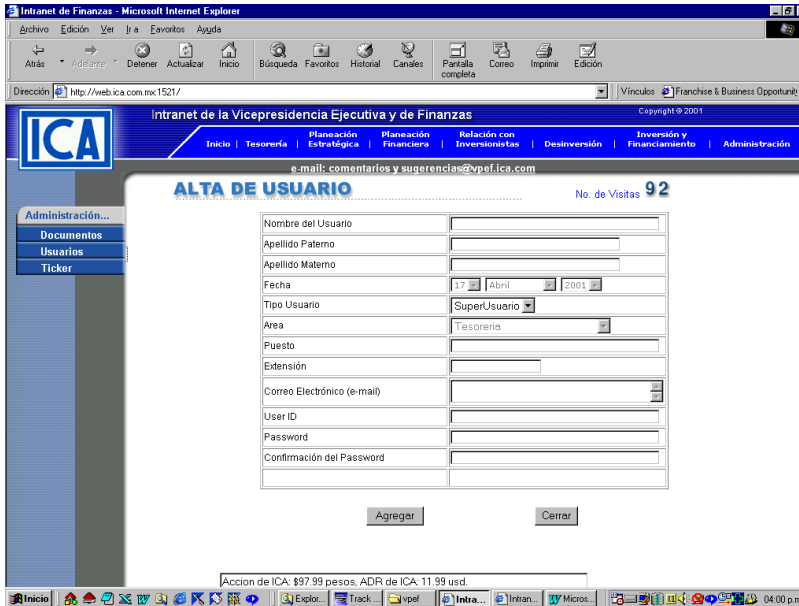


Figura 6.17. Pantalla de Alta de Usuario.

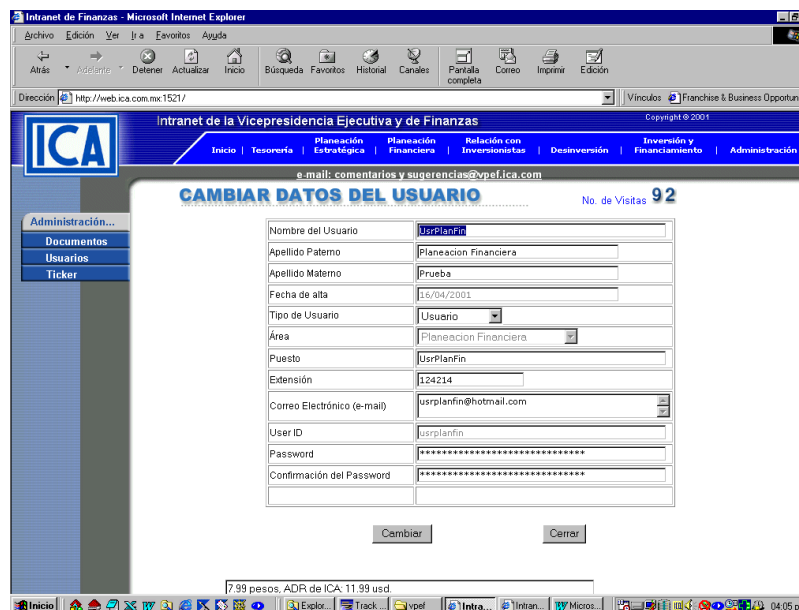


Figura 6.18. Pantalla de Modificación de datos del Usuario.



**Eliminar los datos del usuario.** Se coloca el puntero del ratón en la columna de Borrar del registro a eliminar que tiene un icono de un bote (🗑️), se pulsa un *click* y se muestra una pantalla de confirmación de eliminación (Figura 6.19). Para cancelar la operación se pulsa el botón de *Cancel*.



Figura 6.19. Pantalla de confirmación de eliminación de Usuario.

Hemos observado que cada tipo de usuario tiene diferentes privilegios en el sistema, es conveniente explicarlos con mayor detalle.

### Rol de los Usuarios

Superusuario. Tiene permisos para acceder todas las pantallas de Administración. Puede administrar los Documentos de cualquiera de las seis áreas, para crear, borrar o modificar *folders*, documentos y *links*. Es el responsable de administrar el Catálogo de los Usuarios y el texto contenido en la marquesina.

Administrador. Este usuario puede acceder la parte de Administración de Documentos exclusivamente del área a la que pertenece para crear, borrar o modificar *folders*, documentos y *links*.

Usuario. Este usuario podrá acceder a aquellos documentos, folders y ligas privados y no privados a los cuales se le haya otorgado permiso en la administración de documentos.

### Pantalla de Administración de los documentos

Este módulo sirve para actualizar, agregar y borrar documentos y, en su caso, también *folders* y *links* en el sistema de directorios del servidor *web* identificado por el Área de Finanzas.

El acceso a esta funcionalidad es por medio del botón “Administración” del menú principal. Se requiere proporcionar un usuario y contraseña de Superusuario o Administrador válidos para acceder esta funcionalidad restringida.

En caso de introducir una cuenta de Superusuario, se mostrará primero una pantalla de selección del área a la que se quiere acceder y posteriormente la lista de los documentos, *folders* y *links* disponibles dentro del área seleccionada (Figura 6.20). En

caso contrario, si es una cuenta de Administrador, directamente se presentan los elementos disponibles dentro del área a la que pertenezca el usuario.

Describiremos brevemente las diferentes opciones que un usuario puede seleccionar en pantalla, como agregar un documento o archivo, crear un folder, una liga, cambiar los datos o las rutas de acceso al documento y los accesos a las diferentes pantallas.

Una vez que el Administrador puede consultar los documentos, puede hacer cambios en los mismos, ya sea para agregarlos, editarlos o borrarlos.



Figura 6.20. Pantalla de Lista de Documentos.

**Agregar Archivo.** Esta funcionalidad agregará un archivo a la lista. Es muy importante destacar que se adicionará en el nivel (*folder*) actual en que esté posicionado. Se presenta una pantalla donde se puede utilizar el botón de navegación para seleccionar el documento (Figura 6.21). Éste se desplegará en la lista con el mismo nombre que tenga, en orden alfabético. Se puede seleccionar si el documento es Público o Privado, el valor predeterminado es Público. En caso de seleccionarlo como Privado, se deberán seleccionar los usuarios que tendrán acceso a ese documento, en la lista que se despliega en la parte de abajo. Para seleccionar múltiples usuarios, se debe mantener presionada la tecla de Control durante la selección.

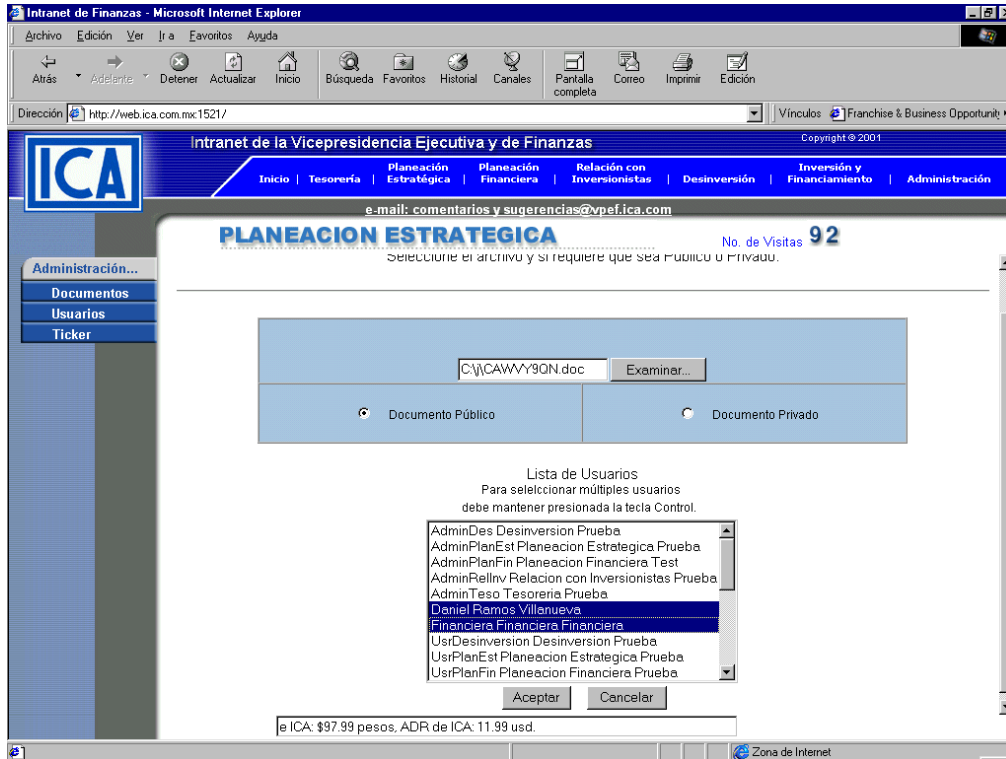


Figura 6.21. Pantalla de Alta de Archivo.

**Agregar Folder.** Esta funcionalidad agregará un nuevo *folder* a la lista. Es muy importante destacar que se adicionará en el nivel (*folder*) actual en que esté posicionado. Se presenta una pantalla donde se pide que se introduzca el nombre del nuevo *folder*, tal como se desplegará en la lista (Figura 6.22). La restricción en el nombre del *folder* es que no puede contener ninguno de los siguientes caracteres: / \ : \* ? " < > |.

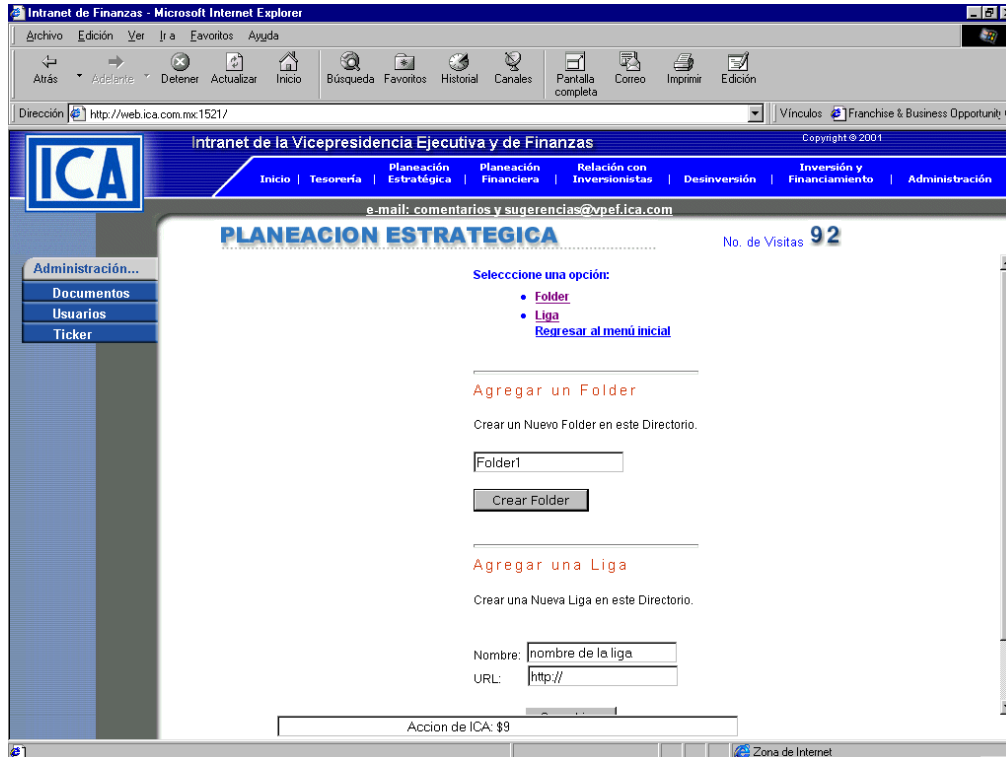


Figura 6.22. Pantalla de Agregar Folder.

**Agregar Liga.** Se solicita el nombre de la liga, mismo que se desplegará en pantalla y la dirección URL (Figura 6.23).

Se debe escribir la dirección completa, incluyendo “ http://”.

**Borrar (🗑).** (Figura 6.24) Esta opción eliminará el elemento seleccionado de la lista. Si se trata de un *folder* se borrará también todo su contenido (*subfolders*, documentos y links). Esta acción no puede ser revocada.

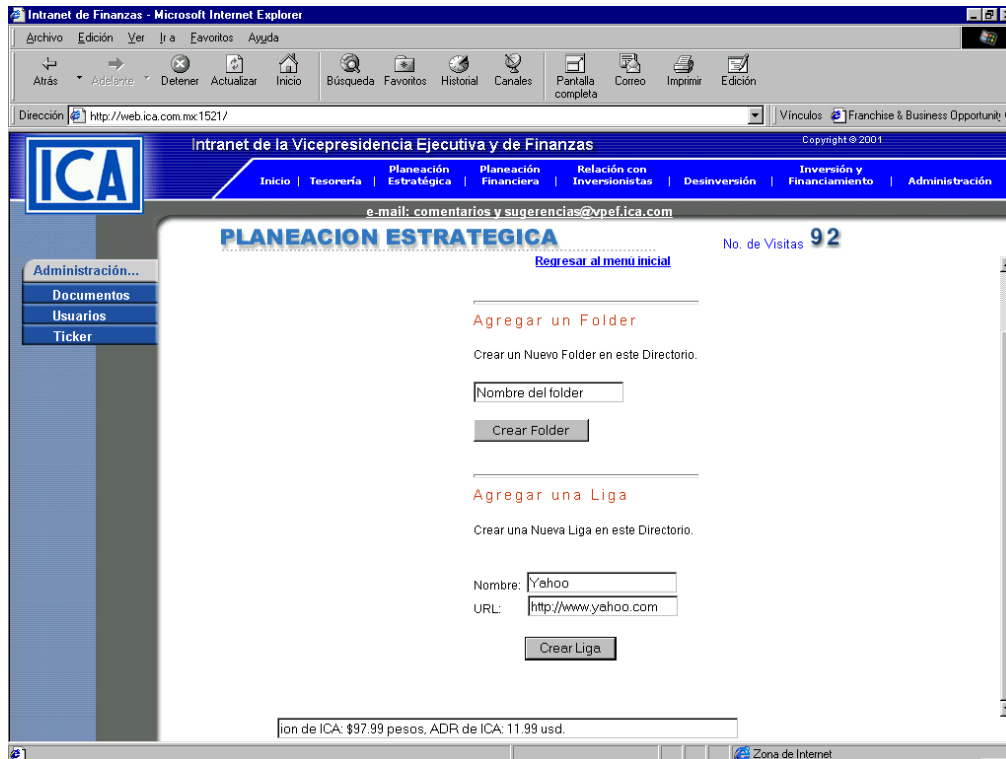


Figura 6.23. Pantalla de Agregar Liga.

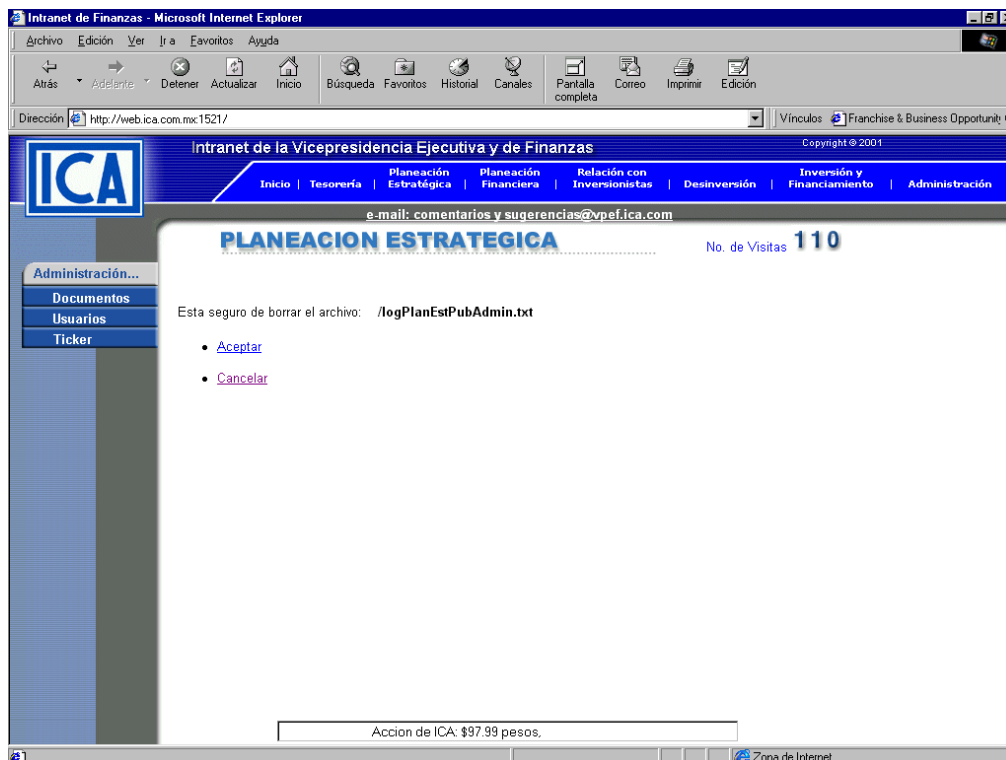


Figura 6.24. Pantalla de Confirmación de Eliminación de Documentos.

**Actualizar** (✎). Esta opción permite actualizar el elemento seleccionado de la lista. Los elementos pueden ser Documentos, *Folders* y *Links*, éstos se explican a continuación. La acción de actualizar no puede ser revocada.

Documentos. En esta pantalla se podrá reemplazar el archivo publicado (Figura 6.25). Al seleccionar un nuevo archivo se elimina el documento actual y se pide que se proporcione la ruta y nombre del nuevo elemento a colocar en la lista. Así mismo, en caso de que el archivo sea Privado, se pueden modificar los usuarios que van a tener acceso al documento. Aparecerán seleccionados aquellos usuarios que fueron elegidos al dar de alta el documento. Para hacer una nueva selección de usuarios se debe presionar la tecla *Control* y marcarlos en la lista.

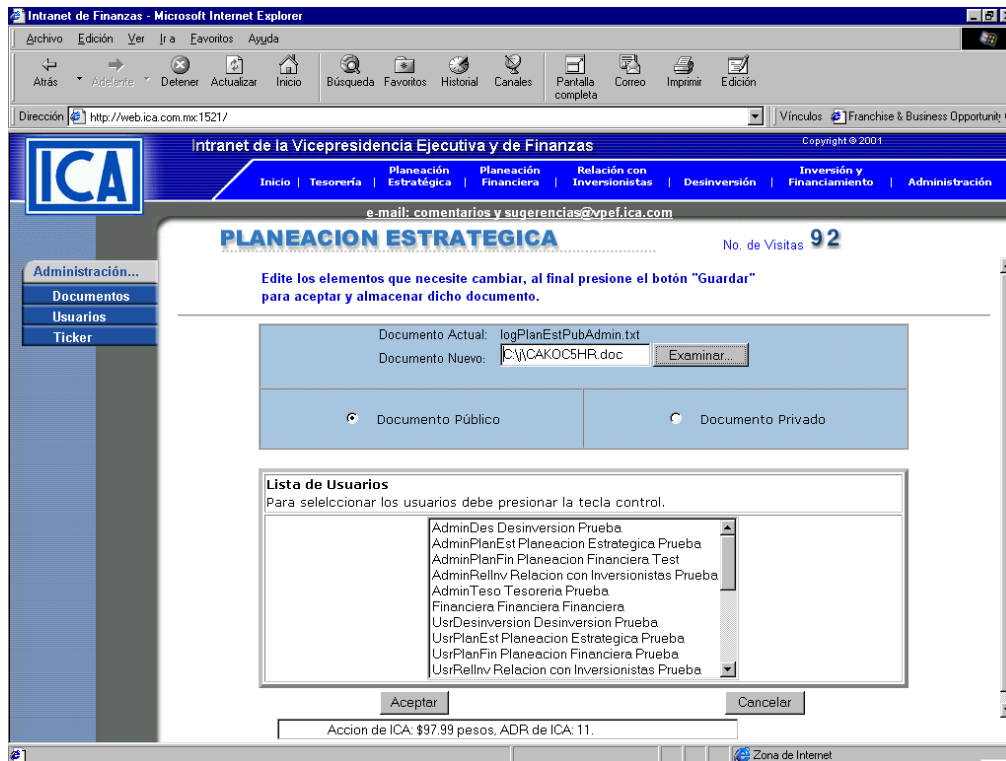


Figura 6.25. Pantalla de Actualización de Documentos.

Folders. Con esta opción se puede cambiar el nombre del folder. Se presenta una pantalla donde indica el nombre actual y se solicita el nuevo nombre (Figura 6.26).

Links. Con esta opción se puede cambiar tanto el nombre desplegado del *link* como la dirección a la que hace referencia la URL (Figura 6.27).

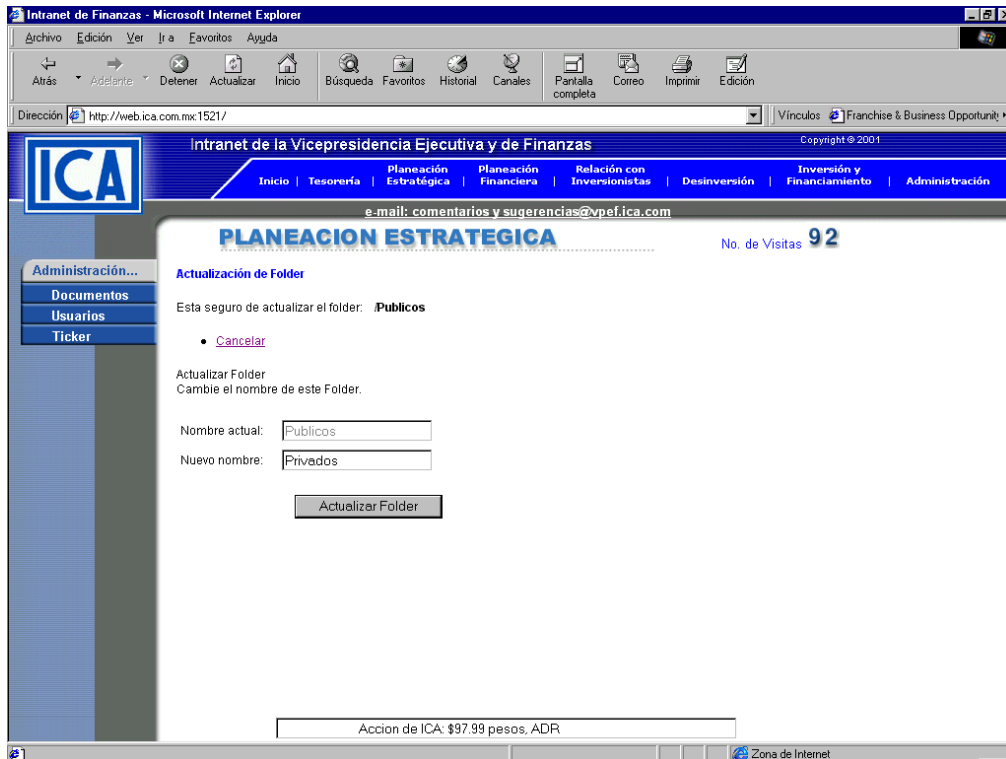


Figura 6.26. Pantalla de Cambio de Folder.

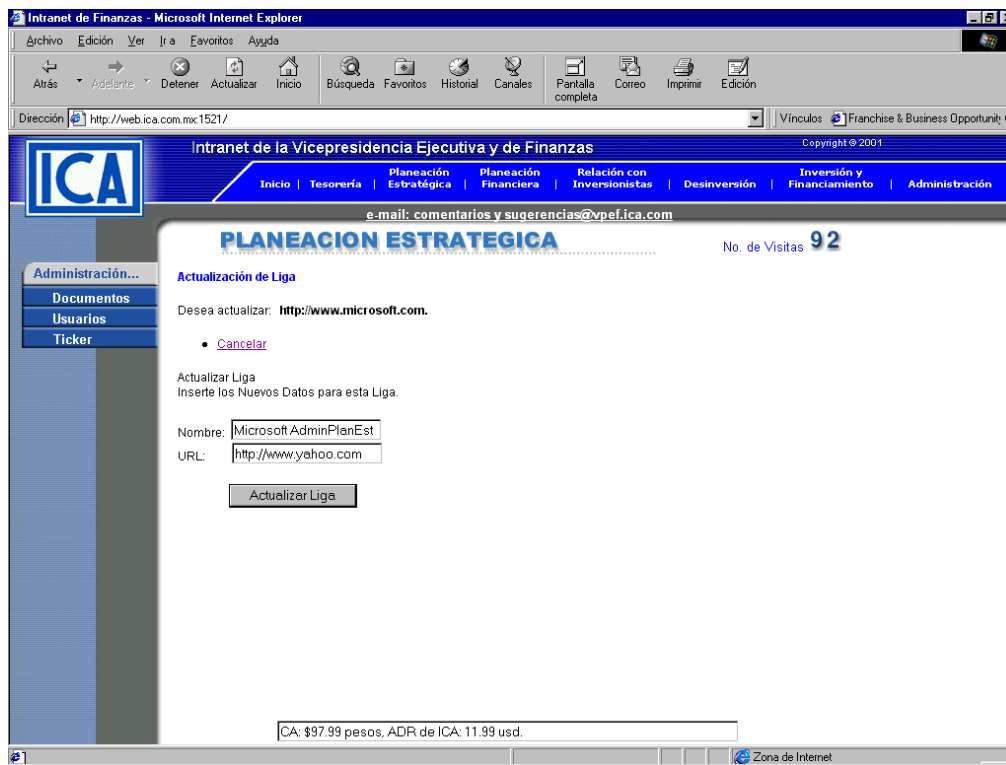


Figura 6.27. Pantalla de Cambio de Link.

Si se habla de documentación, ésta debe ir acompañada de una capacitación. Aquí se muestran los tipos de capacitación.

### 6.1.7. Capacitación

Para una buena operación del sistema es necesario brindar la capacitación necesaria al Administrador del sistema y a todo usuario que tenga relación con él.

Para los usuarios se contempla la siguiente capacitación:

- Funcionamiento del Sistema de Consulta (Documentación para el usuario).

El administrador del sistema deberá tomar la capacitación que se les brinde a los usuarios del sistema y adicionalmente se le brindará la siguiente capacitación:

- Características y ventajas principales de SQL Server 2000.
- Desarrollo e implementación del sistema.
- Administración y mantenimiento del programa.

Una parte fundamental para el buen funcionamiento y posible crecimiento de un sistema se basa en el mantenimiento que a éste se le brinde, a continuación se presenta la descripción de algunos tipos de mantenimiento.

### 6.1.8. Tipos de Mantenimiento

Existen diversos tipos de mantenimientos, entre los que se encuentran el perfecto, preventivo, adaptativo y correctivo, los cuales se muestran en seguida.

Mantenimiento Perfectivo. El mantenimiento perfecto comprende los cambios solicitados al programador del sistema. A medida que se usa el software, se reciben de los usuarios recomendaciones sobre nuevas posibilidades acerca de modificaciones a funciones ya existentes. Para satisfacer estas peticiones se lleva a cabo el mantenimiento perfecto.

Mantenimiento Preventivo. En este tipo de mantenimiento se previenen errores. Este mantenimiento se da cuando se realizan cambios en el software con el fin de mejorar algún proceso. Se puede considerar el mantenimiento a la información que se maneja para garantizar que los resultados dados por el sistema sean los correctos.

Mantenimiento adaptativo. El mantenimiento adaptativo se debe a cambios en el ambiente del programa y a la adaptación de nuevas unidades o módulos. La vida útil estimada del software de aplicación puede fácilmente sobrepasar los diez años, pero considerando la evolución del ambiente, en la práctica éste puede volverse obsoleto en menor tiempo. De este mantenimiento pueden derivarse los siguientes mantenimientos:



*Mantenimiento Aumentativo.* Este tipo de mantenimiento se da cuando se incluyen nuevas funciones, que no se contemplan al inicio del desarrollo del sistema y surgen como una necesidad del usuario.

*Mantenimiento Tecnológico.* Se da debido a los cambios importantes en la informática (Hardware y Software), es decir, al avance en los sistemas operativos, lenguajes de programación y nuevas generaciones de hardware, con lo cual se mejoran las herramientas de trabajo, las versiones de software, equipos periféricos y otros elementos de sistemas.

Mantenimiento correctivo. El proceso que incluye el diagnóstico y corrección de uno o más errores en el sistema se denomina mantenimiento correctivo. Durante el uso del sistema se encontrarán errores, los cuales deben ser informados al equipo de desarrollo. La primera actividad del mantenimiento se da ocasionalmente cuando la prueba del software no haya descubierto todos los errores latentes de un sistema.

El Mantenimiento del Sistema de Seguridad de Archivos de Información Empresarial (SISEA) será en un principio del tipo perfectivo, dadas las características del programa y ya que éste tendrá forzosamente un período de adaptabilidad para su posterior aceptación total por parte del usuario, es decir, con base a las recomendaciones de los usuarios. Tratando de satisfacer estos requerimientos se realizarán los cambios necesarios.

Una vez que el sistema esté trabajando en plenitud, es decir, que tenga el éxito deseado, se realizarán mantenimientos preventivos, con la finalidad de garantizar la información, los resultados óptimos del sistema y previendo posibles errores.

Dadas las necesidades cambiantes de la tecnología y al medio informático en general, el sistema tendrá que adecuarse a las nuevas modificaciones y/o cambios, para lo cual deberán realizarse mantenimientos adaptativos.

En cuanto a los mantenimientos correctivos, éstos serán realizados en cuanto se presente un problema o error en el sistema, para lo cual se pide a los usuarios del sistema informen cualquier anomalía.

Vistos los tipos de mantenimiento analizaremos el costo del sistema.

Con este capítulo se da por terminada la instalación y la liberación del sistema, ahora presentaremos los resultados y las conclusiones.

# CAPÍTULO VII

## RESULTADOS Y CONCLUSIONES

En este capítulo se verán los resultados obtenidos del desarrollo del sistema, así como nuestras conclusiones.

Los resultados obtenidos del desarrollo del SISEA son:

En la fase de análisis se detectaron las necesidades de información a cubrir, el producto que esperaba obtener el usuario y la forma en como quería que funcionara el sistema. Se observó la forma como se llevaba a cabo el proceso para la consulta de archivos en el Área de Finanzas, lo que nos permitió detectar problemas, duplicidades e ineficiencias desde el punto de vista del usuario. Con la información recaudada, evaluamos la conveniencia de optimizar el proceso creando un sistema que ayudara a consultar los archivos de una manera rápida y segura.

En el diseño se visualizó la organización general del sistema, mostrando en él las entradas, los procesos, las entidades de datos y las salidas de acuerdo a la información recopilada en la fase de análisis. Se identificaron los procedimientos necesarios que se llevarían a cabo en el desarrollo del sistema. Se diseñaron y describieron los procesos y rutinas generales que utiliza el sistema. Se definió el modelo de datos, en el cual se establecieron los nombres para las entidades de datos, las llaves de dichas entidades y con ello, se obtuvo el diagrama entidad-relación del sistema.

En el desarrollo del sistema se generaron las pantallas, programas, módulos y rutinas generales. Así mismo, se realizó la revisión de código y se llevaron a cabo las pruebas unitarias, con la intención de encontrar problemas funcionales en la lógica, así como problemas técnicos en el código de cada uno de los programas y procesos desarrollados.

En las pruebas y reportes se prepararon los datos para las pruebas y se llevaron a cabo éstas, los usuarios hicieron solicitudes de cambio y una vez atendidos dichos requerimientos se obtuvo su visto bueno para la fase de liberación del sistema. Se revisó que los reportes generados por el sistema dieran datos verídicos y que su presentación fuera la correcta.

Por último, en la liberación del sistema se generó la documentación de usuarios, se llevó a cabo la capacitación técnica y aplicativa, y se dio soporte y seguimiento a la instalación.

Podemos concluir que se creó un sistema que ayudará a los usuarios del Área de Finanzas de ICA en la consulta de archivos que les auxilien en la toma de decisiones, que les ofrece la seguridad de que sólo ellos y las personas asignadas consultarán sus archivos.

Un punto importante que hay que resaltar es que el usuario puede hacer uso de este sistema en cualquier momento, además puede hacerlo desde cualquier parte de la República o cualquier parte del mundo siempre que se tenga la infraestructura necesaria.

El sistema al estar desarrollado con lenguaje ASP lo limita a trabajar sobre una sola plataforma, que en nuestro caso es Windows, ya que ASP es dependiente de la plataforma donde se ejecuta, no como otros lenguajes como JSP que son independientes de la plataforma.

Consideramos que el sistema desarrollado puede ser utilizado en todas las áreas de la empresa, con sus adecuaciones correspondientes.

El costo del sistema desarrollado es bajo, donde se considera lo siguiente:

- Número de líneas de código.
- Por Hora/Hombre (Analista, Diseñador, Programador).

El costo de una línea de código oscila entre los 20 y 30 dólares americanos (USD). Para el Sistema de Seguridad de Archivos de Información Empresarial se tienen aproximadamente 6500 líneas de código.

El costo del Sistema obtenido mediante esta opción y tomando el valor por línea de 30 USD sería de: \$195,000.00 USD, al tipo de cambio actual.

El tiempo de desarrollo prácticamente se basa en las horas/hombre utilizadas para el análisis, desarrollo e implementación del sistema, por lo cual realizamos lo siguiente:

El tiempo de desarrollo del sistema fue de 8 meses, desde el análisis, diseño y programación hasta la implementación del sistema.

El analista trabajó durante 4 meses, mientras el diseñador lo hizo durante 6 meses, 8 horas al día. Considerando los días hábiles únicamente, se tienen por mes 22 días, por lo tanto el total de horas-hombre por mes es de 176.

Los programadores (2) trabajaron durante 6 meses, 8 horas al día, por lo tanto el número de horas por los 6 meses laborados es de 1056 horas/hombre, por cada uno de ellos, dando un total de 2112 horas/hombre.

El costo por hora/hombre para los analistas, diseñadores, programadores y administradores de sistemas es de aproximadamente \$50.00 USD.

A continuación se presenta el análisis de precios realizado por horas-hombre.

<b>Concepto</b>	<b>Costo Unitario *</b>	<b>Unidad de Medición</b>	<b>Cantidad</b>	<b>Costo por Concepto *</b>
Analista	\$50.00	Hora-Hombre	704	\$35,200.00
Diseñador	\$50.00	Hora-Hombre	1056	\$52,800.00
Programadores(2)	\$50.00	Hora-Hombre	2112	\$105,600.00
<b>TOTAL</b>				<b>\$193,600.00</b>

\* *Costos en dólares americanos.*

La empresa a la cual se le está desarrollando el sistema ya tiene la infraestructura de red necesaria para la implementación del sistema, así como el *hardware* y *software* necesarios, por lo tanto no se incluyen estos conceptos en el análisis de costos.

Para empresas que no cuenten con la estructura de red, el *hardware* (Servidor) y *Software* necesarios para la implementación del sistema, estos conceptos se agregarán al costo del sistema.

Si consideramos lo anterior, la empresa ICA sólo invirtió en el sueldo de los programadores, debido a que se aprovecharon los recursos con lo que ya se contaba como licencias, servidores e infraestructura de red.

Una vez concluido este trabajo esperamos sirva de referencia a todos aquellos interesados en el desarrollo de sistemas. Los conocimientos y la formación académica adquiridos en la Facultad de Ingeniería nos permitieron el desarrollo de este trabajo, que es el resultado directo de contar con un sistema como el

Programa de Apoyo a la Titulación (PAT) ofrecido por el Departamento de educación Continua de la Facultad de Ingeniería.

La realización de esta tesis nos deja apreciar sin lugar a dudas la importancia del trabajo en equipo, de una disciplina y planeación en el desarrollo del mismo y de una visión y objetivo a seguir con posibilidades de cambiar los medios pero no el fin, para desarrollar un buen trabajo.

Finalmente el desarrollo de este trabajo de tesis destacó nuestras habilidades personales y el compañerismo de cada uno de nosotros, ayudándonos a concluir este requisito y con ello, conseguir esta meta, que por un momento tan sólo fue un proyecto. También nos da la satisfacción de haber avanzado un escalón más en nuestro desarrollo personal, que al ponerlo en perspectiva, nos ofrece una oportunidad de brindar un mejor servicio a nuestra comunidad y a nuestro país.

# APÉNDICE A

## CÓDIGO

Este apéndice contiene el código de alguno de los procesos más importantes desarrollados para el sistema. La intención de éste es dar una idea de la extensión de la programación desarrollada, ya que por cantidad no es posible incluirla en su totalidad en el presente trabajo.

**Ica.htm**

Esta página es la principal de acceso en donde podemos navegar libremente sobre el sitio de Finanzas del grupo ICA.

```

<html>
<head>
<script language='JavaScript'>
function getBrowserName()
    {
        if (navigator.appName=='Microsoft Internet
Explorer')
            document.write ("<frameset
rows='125,* ,25' frameborder='NO' border='0'
framespacing='0'> <frame name='top'
src='header.asp' frameborder='No' scrolling='No'
noresize marginwidth='10' marginheight='10'>
        <frameset cols='160,*'> <frame name='left'
src='menulzquierdo.html' frameborder='0'
scrolling='Auto' marginwidth='10' marginheight='10'>
<frame name='main' src='blank.html' frameborder='0'
scrolling='Auto' marginwidth='10'
marginheight='10'></frameset> <frame name='down'
src='ticker.asp' frameborder='0' scrolling='No' noresize
marginwidth='10'
marginheight='10'></frameset></html>")
        else
            document.write ("<frameset
rows='125,* ,50' frameborder='NO' border='0'
framespacing='0'> <frame name='top'
src='header.asp' frameborder='No' scrolling='No'
noresize marginwidth='10' marginheight='10'>
        <frameset cols='160,*'> <frame name='left'
src='menulzquierdo.html' frameborder='0'
scrolling='Auto' marginwidth='10' marginheight='10'>
<frame name='main' src='blank.html' frameborder='0'
scrolling='Auto' marginwidth='10'
marginheight='10'></frameset> <frame name='down'
src='ticker.asp' frameborder='0' scrolling='No' noresize
marginwidth='10'
marginheight='10'></frameset></html>")
    }
</script>

<title>Intranet de Finanzas</title>
</head>

<script
language='JavaScript'>getBrowserName()</script>

<!--
<frameset rows='125,* ,35' frameborder='NO'
border='0' framespacing='0'>

```

```

<frame name='top' src='header.asp'
frameborder='No' scrolling='No' noresize
marginwidth='10' marginheight='10'>

```

```

<frameset cols='160,*'>
    <frame name='left'
src='menulzquierdo.html' frameborder='0'
scrolling='Auto' marginwidth='10' marginheight='10'>
    <frame name='main' src='blank.html'
frameborder='0' scrolling='Auto' marginwidth='10'
marginheight='10'>
</frameset>

```

```

<frame name='down' src='ticker.asp'
frameborder='0' scrolling='No' noresize
marginwidth='10' marginheight='10'>
</frameset>

```

```

</html>
-->

```

## Filesystem\_admin.asp

Esta página es la principal de acceso en donde se crea el ambiente de creación de objetos, folder, ligas y carpetas.

```
<!--#include file = "_private/Ejecuta_Query.inc"-->

<% Response.Buffer = True%>
<%

    id_area=Request.form ("area")
    ruta1 =request.form("ruta")

'response.write id_area
'response.write ruta1

                IF id_area="" THEN
id_area=Request.QueryString ("id_area")

'Response.Write "id_area = " & id_area & "<BR>"
                END IF

if id_area <> "" then
    Session("Id_Area")= id_area
'Response.Write "asigna session"
    else
        id_area = Session("Id_Area")
'Response.Write "asigna id"
    end if

'Response.Write "id_area = " & id_area & "<BR>"

vsql="select * from Areas where id_area="&id_area
'Response.Write(vsql) & "<BR>"
Set Rs1 = Cnn1.Execute (vsql)
area_nom = Rs1("area_nom")
'Response.Write area_nom
'response.end

usuario=session("usuariold")
'Response.Write "usr = " & id_area & "<BR>"

if id_area = "" then

                sql="select * from usuarios where
id_usr = "&usuario&""
'Response.Write(sql)
set RSid= Cnn1.Execute (sql)
id_area = RSid("id_area")
'Response.Write(id_area)

    end if

if trim(id_area) <> "" then
```

```
                vsql="select * from Areas where
id_area="&id_area&""
                else
                    vsql="select * from Areas"
                end if
'Response.Write(vsql)
set RSid= Cnn1.Execute (vsql)
area_nom = RSid("area_nom")

'RutaActual = Request("RutaActual")

%>

<!--#Include File="filesystem_lib_w.asp"-->

<%
'valida la parte del texto en el header.
If id_area= 1 THEN
%>
<SCRIPT LANGUAGE=javascript>
<!--
        parent.frames[0].document.images['img_ilust
racion'].src = 'images/tesoreria.gif';
//-->
</SCRIPT>
<%
End if

If id_area= 2 THEN
%>
<SCRIPT LANGUAGE=javascript>
<!--
        parent.frames[0].document.images['img_ilust
racion'].src = 'images/ESTRATEGICA.gif';
//-->
</SCRIPT>
<%
End if

If id_area= 3 THEN
%>
<SCRIPT LANGUAGE=javascript>
<!--
        parent.frames[0].document.images['img_ilust
racion'].src = 'images/P_FINANCIERA.gif';
//-->
</SCRIPT>
<%
End if

If id_area= 4 THEN
%>
<SCRIPT LANGUAGE=javascript>
<!--
        parent.frames[0].document.images['img_ilust
racion'].src = 'images/INVERSIONISTAS.gif';
//-->
</SCRIPT>
```



```

<%
  End if

  If id_area= 5 THEN
%>
<SCRIPT LANGUAGE=javascript>
<!--
      parent.frames[0].document.images['img_ilustracion'].src = 'imagenes/desinversion.gif';
-->
</SCRIPT>

<%
  End if

  If id_area= 6 THEN
%>
<SCRIPT LANGUAGE=javascript>
<!--
      parent.frames[0].document.images['img_ilustracion'].src = 'imagenes/INVER_FINA.gif';
-->
</SCRIPT>

<%
  End if

  If id_area= 7 THEN
%>
<SCRIPT LANGUAGE=javascript>
<!--
      parent.frames[0].document.images['img_ilustracion'].src = 'imagenes/administracion.gif';
-->
</SCRIPT>

<%
  End if
%>

<html>
<head>
<title>FileSystem</title><b></b>
<link rel="stylesheet" type="text/css"
href=" ../styles/common.css">
<meta http-equiv="pragma" content="no-cache">
<script>
<!--
function valida_folder( forma ) {
  var Valid = "\\.*?\"<>|";
  for ( var i = 0; i < forma.newname.value.length; i++ )
    if ( Valid.indexOf( forma.newname.value.charAt( i )
    >= 0 ) {
      alert("A file or folder name cannot contain any of
the following characters:\n\n\t\t \\..*?\"<>|");
      forma.newname.focus();
      return(false);
    }
  }
-->
</script>

```

```

<link rel="stylesheet" type="text/css"
href=" ../styles/common.css">

</head>
<body class="letbody" bgcolor="#FFFFFF">
<%
Dim fs
Dim sAction
Dim sFile, sPath, sFolder, sFileType
Dim scriptname
Dim Archivo
Dim ArchivoEscr
Dim fileobject
Dim filecollection
Dim file
Dim lineid
Dim bgcolor
Dim bgcolor_on
Dim bgcolor_off
Dim foldercollection
Dim folder
Dim errornum
Dim errorcode
Dim ruta
Dim dirArchivosRaiz
Dim raiz

'raiz = "C:/inetpub/wwwroot/ica/doc_areas/"&
area_nom
'raiz =
"d:/inetpubica/wwwroot/pags/empresas/vpef/doc_areas/"& area_nom
raiz = "d:/inetpubica/docs/doc_areas/" & area_nom
dirArchivosRaiz = "doc_areas/" & area_nom
'Response.Write raiz & "<br>"
'Response.Write area_nom & "<br>"
'Response.end

' Reinicia los valores de error
errornum = 0
errorcode = ""

scriptname=Request.ServerVariables("Script_Name")
sAction = Request.QueryString("action")
sFileType = Request.QueryString("filetype")

'Response.Write sFileType
'Response.End

' mipatru es la unica variable de entrada de afuera
If Request.QueryString("mipatru") <> "" Then
  session("mipatrulla") = Request("mipatru")
end if

If Request.QueryString("path") = "" Then
  sPath = session("mipatrulla")

Else

```

```

If Mid(Request.QueryString("path"), 1,
Len(session("mipatrulla")) = session("mipatrulla") and
Len(Request.QueryString("path"))>Len(session("mipatrulla")) Then
    sPath = Request.QueryString("path")

else
    sPath = session("mipatrulla")
end if

If InStr(sPath, "../") Then
    errornum = errornum+1
    errorcode = errorcode & " <li><b>Illegal use
of ""../""</b>. To protect the directory structure outside
of this web folder, you are restricted to utilizing the
links provided by the application.</li>"
End If
End If

If sPath="/" Then
If Request.QueryString("file") = "" Then
    sFile = sPath & Request.Form("file")
Else
    sFile = sPath & Request.QueryString("file")
End If
If Request.QueryString("folder") = "" Then
    sFolder = sPath & Request.Form("folder")
Else
    sFolder = sPath & Request.QueryString("folder")
End If
Else
If Request.QueryString("file") = "" Then
    sFile = sPath & "/" & Request.Form("file")
Else
    sFile = sPath & "/" & Request.QueryString("file")
End If
If Request.QueryString("folder") = "" Then
    sFolder = sPath & "/" & Request.Form("folder")
Else
    sFolder = sPath & "/" & Request.QueryString("folder")
End If
End If

```

```

' Valida que no haya errores y que no se realicen
acciones indebidas
If errornum < 1 Then
    Set fs =
Server.CreateObject("Scripting.FileSystemObject")

```

```

Select Case sAction

```

```

    Case "VerFolder"
        MostrarLista

```

```

    Case "NuevoFolder"
        CrearFolder

```

```

    Case "BorrarArchivo"
        BorrarArchivo

```

```

Case "BorrarFolder"
    BorrarFolder

Case "AgregarLiga"
    AgregarLiga

Case "BorrarLiga"
    BorrarLiga

Case "ActualizarArchivo"
    ActualizarArchivo

Case "ActualizarFolder"
    ActualizarFolder

Case "actualizarLiga"
    actualizarLiga

Case Else
    MostrarLista
End Select
Set fs = Nothing

Else
    DisplayErrors

End If
%>
</body>
</html>

```



```

{
    var sVar
    var iError
    iError=0;

    for (i = 0; i <
sTipo.length; i++)
    {
        sVar=
sTipo.substring(i,i+1);
    }

    if (sVar == "C")
    {
        //Validando comilla simple
        if
(event.keyCode == 39)
        {
            iError++;
        }
    }

    if (sTipo.length ==
iError )
    {
        event.returnValue=false;
    }
    else
    {
        event.returnValue=true;
    }
}
//Fin de valida caracteres

</script>

<SCRIPT ID=clientEventHandlersVBS
LANGUAGE=vbscript>
<!--
'Sub window_onfocus
'    log_in.usuariold.focus
'End Sub

'function numEmple_OnFocus
'    log_in.usuariold.select()
'End function

-->
</SCRIPT>

<SCRIPT LANGUAGE=jsript>
<!--
function numEmple_OnFocus()
{
    //document.passwordform.login.focus();
    document.log_in.usuariold.focus();
}
-->
</SCRIPT>

</head>
<body bgcolor="white" language="JavaScript"
o!nfocus="return window_onfocus()" onload =
"numEmple_OnFocus()">

<form name="log_in" action="ValidaUsuarios.asp"
method="post" language=javascript>

<p>&nbsp;</p>
<p>&nbsp;</p>

<div id="showimage"
style="position:absolute;width:250px;left:200;top:130"
>
<table border="2" width="400" bgcolor="#000080"
cellspacing="0" cellpadding="2" height="150">
<tr>
<td width="100%">
<table border="0" width="100%"
cellspacing="0" cellpadding="0"
height="150">
<tr>
<td onMousedown="initializedragie()"
style="cursor:hand" width="100%" align=middle
height="20">
<ilayer width="100%"
onSelectStart="return false">
<layer
width="100%"
onMouseover="dragswitch=1;drag_dropns(showimag
e)" onMouseout="dragswitch=0">
<font
face="Verdana" color="#FFFFFF">
<strong>
<small>Verificación de Acceso a la Intranet
de Finanzas</small>
</strong>
</font>
</layer>
</ilayer>
</td>
<td style="cursor:hand">
<a href="#"
onClick="hidebox();return false">

```



## Login\_arch.asp

Página principal para validar la entrada a la parte de consulta de documentos.

```
<!--#include file = "_private/Ejecuta_Query.inc"-->
<%
session("usuariold") = ""
session("password") = ""
session("usuarioTipo") = ""
session("nombre_archivo1")= ""
session.Timeout = 60

Dim nombre_archivo
nombre_archivo =
trim(Request.QueryString("nombre_archivo"))
session("nombre_archivo1")=
trim(Request.QueryString("nombre_archivo"))

'if session("nombre_archivo10") = "" then
'session("nombre_archivo10") = ""
'session("nombre_archivo10") =
trim(Request.QueryString("nombre_archivo"))

'response.write session("nombre_archivo10")
'response.end
'End if

'response.write session("nombre_archivo10")
'response.end
'response.write session("nombre_archivo10")

'if not session("nombre_archivo10") = "" then

vSQL = "select * from documentos where doc_priv
='N' and doc_nom=" & nombre_archivo & ""
'response.write vSQL
'response.end

set RsDoc =
server.CreateObject("ADODB.RecordSet")
RsDoc.Open vSQL, cnn1

if not RsDoc.EOF then
'response.write vSQL
'response.end

Dim ruta
ruta =
Replace(RsDoc.Fields("doc_ruta").value," ","%20")
'response.write
ruta

'response.End
```

```
RsDoc.Close

'session("nombre_archivo10") = ""
%>

<form
method="post" action="home/presenta/fra_indx1.asp"
id="form1" name="form1">

<input type="hidden" name="input2"
value="<%=ruta%>" language="javascript">

</input>

</form>

<script>
//alert("El
archivo se registro exitosamente !");

//document.arch.action='Sube_arch.asp';

//document.arch.submit();

//history.go(-3);

//var x;
//x =
document.form1.
//x =
document.form1.('ruta');

descripcion = document.form1.input2.value;
document.location.assign(descripcion);
</script>

<% Response.End

Else
end if

'end if

RsDoc.Close
%>

<html>
<head>
<meta name="VI60_defaultClientScript"
content=VBScript>
<meta NAME="GENERATOR" Content="Microsoft
Visual Studio 6.0">
<script language="JavaScript1.2"
src="windowTable.js"></script>
<style>
td {font-family:Verdana;font-size:12}
</style>
```

```

<script ID="clientEventHandlersJS"
LANGUAGE="javascript">

        function Checar(){
                                //Checando
nombre
                                if(document.log_in.usuariold.value==""){
                                        alert('Favor de
ingresar su nombre de usuario para poder validar su
acceso al documento.');
```

document.log\_in.usuariold.focus();  
return;

}

if(document.log\_in.password.value==""){  
 alert('Favor de  
ingresar su password para poder poder validar su  
acceso al documento.');

document.log\_in.password.focus();  
return;

}

document.log\_in.submit ();  
 }

```

                                }
//-->
</script>

        <script language="JavaScript">
                //Necesitamos validar que los
campos que se van a actualizar no acepten comilla
simple.

                function funcion(c)
                {
                                //pone el foco en
un campo tipo TEXTO
                                c.select();
                } //Fin de funcion

                function
valida_caracter(sTipo)
                {
                                var sVar
                                var iError
                                iError=0;

                                for (i = 0; i <
sTipo.length; i++)
                                {
                                        sVar=
sTipo.substring(i,i+1);

                                if (sVar == "C")
                                {

                                        //Validando comilla simple
                                        if
(event.keyCode == 39)
                                        {
                                                iError++;
                                        }

                                if (sTipo.length ==
iError )
                                {
                                        event.returnValue=false;
                                }
                                else
                                {
                                        event.returnValue=true;
                                }
                                } //Fin de valida caracteres
</script>

<SCRIPT LANGUAGE=jscrip>
<!--
```





## Filesystem\_lib\_w.asp

Página principal de los procesos a realizar de acuerdo a cada una de las peticiones de lado administrativos, por ejemplo, borrar ligas, folders, etc.

```

<!--#include file = "_private/Ejecuta_Query.inc"-->
<%

' Subproceso: AgregarLiga

Sub AgregarLiga
  Dim miCad
  Dim cambiacadena
  Session("pagAnterior") =
Request.ServerVariables("HTTP_REFERER")

  Set Archivo = fs.OpenTextFile(Replace(raiz &
request("path") & "/" & request("file"), "/", "\"))

  if Archivo.AtEndOfStream <> True then
    miCad = Archivo.ReadAll
  end if

  ' concatena las ligas actuales y las nuevas

  cambiacadena = Replace(Request.Form("nombre"),
Chr(32), "_")

'response.write nuevacadena
'response.end

  'miCad = miCad & Request.Form("liga")
  &vbCrLf & Request.Form("nombre") &vbCrLf
  miCad = miCad & Request.Form("liga")
  &vbCrLf & cambiacadena &vbCrLf

'response.write miCad
'response.end

  Set ArchivoEscr = fs.CreateTextFile(Replace(raiz &
request("path") & "/" & request("file"), "/", "\"), true)
  ArchivoEscr.Write miCad
  ArchivoEscr.Close

Response.Redirect("filesystem_admin.asp?action=Ve
rFolder&path=" & server.URLEncode(request("path")))
End Sub

```

```

' Subproceso: BorrarLiga

Sub BorrarLiga
  If Request.QueryString("commit") <> "si" Then
    Dim miLiga
    miLiga = Request.QueryString("liga")
    Session("pagAnterior") =
Request.ServerVariables("HTTP_REFERER")
    Session("sFile") = sFile
    Response.Write "<font
face='Arial,Helvetica, Geneva, Swiss, SunSans-Regular'
size=2>"

    'sFile = mid(sFile,
Len(session("mipatrulla")), Len(sFile)-1)
    Response.Write "<br><br><p>Esta seguro
de querer borrar:&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<b> " &
Request("nameLink") & "</b>"
    'Response.Write "<br><br><p>Esta
seguro de querer borrar:&nbsp; <b> " & miLiga &
".</b> "
    Response.Write "<br><br><b><br>Esta
acción es irreversible.</b></p>"
    Response.Write "<UL>"
    Response.Write "<LI><a href="" &
scriptname & "?action=BorrarLiga&commit=si&path="
& server.URLEncode(request("path")) &
"&file=misligas.txt&liga=" &
server.URLEncode(miLiga) & "">Aceptar</a></LI>"
    Response.Write "<br><br>"
    Response.Write "<LI><a href=" &
Session("pagAnterior") & ">Cancelar</a></LI>"
    Response.Write "</UL>"

  Else
    Dim cad1, cad2
    Response.Write Replace(raiz &
request("path") & "/" & request("file"), "/", "\")
    Set Archivo =
fs.OpenTextFile(Replace(raiz & request("path") & "/" &
request("file"), "/", "\"))

    do while not Archivo.AtEndOfStream
      cad1 = Archivo.ReadLine
      Response.Write cad1

&vbCrLf

      Response.Write
Request.QueryString("liga") &vbCrLf
      if StrComp(cad1,
Request.QueryString("liga")) <> 0 then
        cad2 = cad2 & cad1

&vbCrLf

        cad2 = cad2 &
Archivo.ReadLine &vbCrLf
      else
        cad1 = Archivo.ReadLine
      end if

    loop

    Archivo.Close

```



```

pos & "ZZZZ"           'Response.write
                        'Response.end
    If pos = 0 Then
        If yapaso
            nuevacadena = nuevacadena +
corresp
        Else
            nuevacadena = nuevacadena +
newname
        End If
    exit do
    Else
        corresp1 = Mid(Trim(corresp), pos +
1, pos + cadena2)

        If Trim(corresp) = "" Then
        Else
            corresp =
Mid(Trim(corresp), 1, pos - 1)
        End If

        'Response.write corresp1
        'Response.write corresp
        'Response.write oldname
        'Response.end

        p = InStr(corresp, oldname)

        'Response.write p & "XXXX"
        'Response.end

        If p = 1 Then
            nuevacadena = nuevacadena +
newname + Chr(32)
            yapaso = 1
            'Response.Write nuevacadena

```

```

                        'Response.End
    Else
        nuevacadena = nuevacadena +
corresp + Chr(32)
        'Response.Write nuevacadena
        'Response.End
    End If

        corresp =
trim(corresp1)
        End If
        Loop
        'response.end
        nuevacadena =
Replace(nuevacadena, Chr(32),Chr(3))
        nuevacadena =
Replace(nuevacadena, Chr(6),Chr(32))
        cad1 =
Replace(nuevacadena, Chr(3),vbCrLf)
        cad1 = cad1 &vbCrLf
        'Response.Write cad1
        'Response.end

    else
        cad1 = Replace(cad1,
Request.Form("oldname"),
Request.Form("newname"))
        cad1 = Replace(cad1,
Request.Form("oldliga"), Request.Form("newliga"))
    End if

    end if

        'Response.Write cad1
        'Response.End

        Archivo.Close

        Set ArchivoEscr =
fs.CreateTextFile(replace(raiz & request("path") & "/"
& request("file"),"/","\"), true)
        ArchivoEscr.Write cad1
        ArchivoEscr.Close

        'Response.Write cad1
        'Response.End

        Response.Redirect("'" &
Session("pagAnterior") & "'")
    End If

```

```

End Sub

.....
' Subproceso: CrearFolder
.....

Sub CrearFolder

Response.Write "<font
face='Arial,Helvetica,Geneva,Swiss,SunSans-Regular'
size=2>"

If fs.FolderExists(Replace(raiz & request("path") & "/"
& trim(request("folder")),"/","\")) Then
    response.write "<a
href='javascript:history.go(-1);'>El folder llamado <b>"
& sFolder & "</b> ya existe!</a><br>"
Else
    response.write Replace(raiz & request("path")
& "/" & trim(request("folder")),"/","\")
    fs.CreateFolder(Replace(raiz &
request("path") & "/" & trim(request("folder")),"/","\"))

' crea el archivo de las ligas
Set fs =
CreateObject("Scripting.FileSystemObject")
Set ArchivoEscr =
fs.CreateTextFile(Replace(raiz & request("path") & "/"
& trim(request("folder")),"/","\") & "\misligas.txt", True)
ArchivoEscr.Close

Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom""><img src='Images/folder1.gif'
width=18 height=18 border=0 alt='Document'></td>"

' Se regresa a mostrar el folder que acaba de
crear:
Response.Redirect("filesystem_admin.asp?a
ction=VerFolder&path=" &
server.URLEncode(request("path")))
End If
End Sub

```

```

.....
' Subproceso: BorrarArchivo
.....

Sub BorrarArchivo
Response.Write "<font
face='Arial,Helvetica,Geneva,Swiss,SunSans-Regular'
size=2>"
If Request.QueryString("commit") <> "si" Then
    Session("pagAnterior") =
Request.ServerVariables("HTTP_REFERER")
    Session("sFile") =
server.URLEncode(sFile)
    'Response.Write
"<br><br><p>Desea borrar: " & sFile & ". "

```

```

sFile = mid(sFile,
Len(session("mipatrulla")+1, Len(sFile))
Response.Write "<br><br><p>Esta
seguro de borrar el archivo:&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<b> "
& sFile & "</b>"

If sFileType = "jpg" OR sFileType =
"gif" Then
    Response.Write "<p><img
src=""http://" &
Request.ServerVariables("HTTP_HOST") &
Replace(sfile, " ", "%20") & ""></p>"
End If
'Response.Write "<b><br>This
action cannot be undone.</b></p>"
Response.Write "<UL>"
Response.Write "<LI><a href="" &
scriptname & "?action=BorrarArchivo&path=" &
server.URLEncode(request("Path")) & "&file=" &
server.URLEncode(request("file")) &
"&commit=si">Aceptar</a></LI>"
Response.Write "<br><br>"
Response.Write "<LI><a href=" &
Session("pagAnterior") & ">Cancelar</a></LI>"
Response.Write "</UL>"

Else

Dim nombre_archivo, Rsconsulta

nombre_archivo =
trim(request("file"))

vSQL = "select * from documentos
where doc_nom=" & nombre_archivo & ""

set Rsconsulta =
server.CreateObject("ADODB.RecordSet")
Rsconsulta.Open vSQL, cnn1

Do while not
rsconsulta.EOF

cnn1.execute
"DELETE permisos WHERE id_doc = " &
rsconsulta("id_doc")

rsconsulta.movenext
loop
rsconsulta.Close
Set rsconsulta =

Nothing

cnn1.execute "DELETE
documentos WHERE doc_nom = " &
request("file")&""

fs.DeleteFile(Replace(raiz &
request("path") & "/" & request("file"),"/","\"))

```

```

                Response.Redirect("" &
Session("pagAnterior") & "")

    End If
End Sub

.....
' Subproceso: BorrarFolder
.....

Sub BorrarFolder
    Response.Write "<font
face='Arial,Helvetica, Geneva,Swiss,SunSans-Regular'
size=2>"
    If Request.QueryString("commit") <> "si" Then
        Session("pagAnterior") =
Request.ServerVariables("HTTP_REFERER")
        Session("sFolder") = sFolder
        'Response.Write "<br><br><p>Esta
seguro de eliminar este folder:" & sFolder & ". "
        sFolder = mid(sFolder,
Len(session("mipatrulla))+1, Len(sFolder))
        Response.Write "<br><br><p>Esta
seguro de borrar el folder:&nbsp;&nbsp;&nbsp;&nbsp;<b> "
& sFolder & "</b> "
        'Response.Write "<b><br>This
action cannot be undone.</b></p>"
        Response.Write "<UL>"
        Response.Write "<LI><a href="" &
scriptname & "?action=BorrarFolder&path=" &
server.URLEncode(request("path")) & "&folder=" &
server.URLEncode(request("folder")) &
"&commit=si">Aceptar</a></LI>"
        Response.Write "<br><br>"
        'Response.Write request("folder")
        Response.Write "<LI><a href=" &
Session("pagAnterior") & ">Cancelar</a></LI>"
        Response.Write "</UL>"
    Else
        'Response.Write sPath & "<br>"
        'Response.Write sFile & "<br>"
        'Response.Write sFile & "<br>"
&replace(raiz & request("path") & "/" &
request("folder"),"/","\")

        Dim ruta, Rsconsulta, l

        ruta = replace(dirArchivosRaiz &
request("path") & "/" & request("folder"),"/","\")
        l = len(ruta)
        'response.write l

        SQL = "select * from documentos
where substr(doc_ruta,1,"&l&")="" & ruta & ""
        'response.write SQL

        set Rscarpeta =
server.CreateObject("ADODB.Recordset")
        Rscarpeta.Open SQL,cnn1

```

```

Do while not
Rscarpeta.EOF

                cnn1.execute
"DELETE permisos WHERE ID_DOC = " &
Rscarpeta("id_doc")

                cnn1.execute
"DELETE documentos WHERE ID_DOC = " &
Rscarpeta("id_doc")

                Rscarpeta.movenext
                loop

                Rscarpeta.Close
                Set Rscarpeta =
Nothing

                'Response.Write replace(raiz &
request("path") & "/" & request("folder"),"/","\")
                'Response.End

                fs.DeleteFolder(replace(raiz &
request("path") & "/" & request("folder"),"/","\"))

                Response.Redirect("" &
Session("pagAnterior") & "")
    End If
End Sub

.....
' Subproceso: ActualizarArchivo
.....

Sub ActualizarArchivo
Dim miruta
    Response.Write "<font
face='Arial,Helvetica, Geneva,Swiss,SunSans-Regular'
size=2>"
    If Request.QueryString("commit") <> "si" Then
        Session("pagAnterior") =
Request.ServerVariables("HTTP_REFERER")
        Session("sFile") = sFile
        'Session("sFile") =
server.URLEncode(sFile)
        'Response.Write "<br><br><p>Esta
seguro de actualizar: " & sFile & ". "
        sFile = mid(sFile,
Len(session("mipatrulla))+1, Len(sFile))
        Response.Write "<br><br><p>Esta
seguro de actualizar el
archivo:&nbsp;&nbsp;&nbsp;&nbsp;<b> " & sFile & "</b> "
        If sFileType = "jpg" OR sFileType =
"gif" Then
                Response.Write "<p><img
src=""http://" &

```

```

Replace(Request.ServerVariables("HTTP_HOST"), "
", "%20") & Replace(sfile, " ", "%20") & ""></p>
End If
'Response.Write "<b><br>This
action cannot be undone.</b></p>"
Response.Write "<UL>"
Response.Write "<LI><a href="" &
scriptname & "?action=ActualizarArchivo&path=" &
server.URLEncode(sPath) & "&file=" &
server.URLEncode(request("file")) &
"&commit=si">Aceptar</a></LI>"
Response.Write "<br><br>"
Response.Write "<LI><a href=" &
Session("pagAnterior") & ">Cancelar</a></LI>"
Response.Write "</UL>"

Else
miruta = raiz &
Request.QueryString("path")
miruta = Replace(miruta, "/", "\")
Response.write "<br><br><A
NAME=add_doc><span class='header'><b>Actualizar
Archivo</b></span></A>"
Response.write "<br><font
face='Vedana' size='2'>Cambia este Documento en
este Directorio.</font>"
Response.write "<br>"
Response.write "<form
Name=""fred"" ENCTYPE=""multipart/form-data""
method=""post"" action=""ActualizarArchivo.asp"">"
Response.write "<input
type=""hidden"" name=""rutaserver"" value="" &
miruta & "">"
Response.write "<TABLE
WIDTH=70% CELLPACING=2 CELLPADDING=2
BORDER=0>"
Response.write "<TR><TH
ALIGN=left>"
Response.write "<font
face='Arial,Helvetica, Geneva,Swiss,SunSans-Regular'
size=2>"
Response.write "Nombre del
archivo:</TH>"
Response.write "<TD><INPUT
NAME=newname type=File size=40>"
Response.write "</TD>"
Response.write "</TR>"
Response.write "</TABLE>"
Response.write "<br CLEAR=left>"
Response.write
"<blockquote><blockquote><blockquote><bl
ockquote><INPUT TYPE=""submit""
value=""Actualizar
Archivo""></blockquote></blockquote></blockquote><
/blockquote>"
Response.write "</form>"

fs.DeleteFile(miruta&"\"&request("file"))
End If
End Sub

```

```

' Subproceso: ActualizarFolder
'
'
Sub ActualizarFolder
Dim miruta
Response.Write "<font
face='Arial,Helvetica, Geneva,Swiss,SunSans-Regular'
size=2>"
If Request.Form("commit") <> "si" Then
Session("pagAnterior") =
Request.ServerVariables("HTTP_REFERER")
sFolder = trim(sFolder)
Session("sFolder") = sFolder
'Response.Write
"<br><br><p>Estas seguro de actualizar este
folder:&nbsp;&nbsp;<b> " & sFolder & "</b> "
sFolder = mid(sFolder,
Len(session("mipatrulla))+1, Len(sFolder))

Response.Write "<span
class='letbodyblue'>"
Response.Write "<p>Actualización
de Folder </span>"
Response.Write "<span
class='letbody'>"
Response.Write "<br><br><p>Esta
seguro de actualizar el
folder:&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<b> " & sFolder & "</b>"
Response.Write
"<br><br><b><br>This action cannot be
undone.</b></p>"
Response.Write "<UL>"
Response.Write "<LI><a href=" &
Session("pagAnterior") & ">Cancelar</a></LI>"
Response.Write "</UL>"

miruta = raiz & sPath & "/"
miruta = Replace(miruta, "/", "\")

'Response.Write
Request.Form("rutaserver")
'Response.Write
trim(Request.Form("oldname"))
'Response.Write
trim(Request.Form("newname"))

Response.write "Actualizar Folder"
Response.write "<br>"
Response.write "Cambie el
nombre de este Folder."
Response.write "<br>"
Response.write "<form
Name=""fred"" method=""post"" onsubmit='return
valida_folder(this);' action=" & scriptname &
"?action=ActualizarFolder>"
Response.write "<input
type=""hidden"" name=""rutaserver"" value="" &
miruta & "">"

```

```

Response.write "<TABLE
WIDTH=35% CELSPACING=2 CELLPADDING=2
BORDER=0>"
Response.write "<TR>"
Response.write "<Td>"
Response.write "<span
class='letbody'>Nombre actual:"
Response.write "</td>"
Response.write "<TD><INPUT
NAME=""oldname1"" type=""text"" size=20 value="" &
trim(Request.QueryString("folder")) & "" disabled
onfocus=""blur();"" >"
Response.write "</TD>"
Response.write "</TR>"
Response.write "<TR>"
Response.Write "<td></span>"
Response.write "<span
class='letbody'>Nuevo nombre:"
Response.write "</td>"
Response.write "<TD><INPUT
NAME=""newname"" type=""text"" size=20>"
Response.write "</TD>"
Response.write "</TR>"
Response.write "</TABLE>"
Response.write "<br CLEAR=left>"
Response.write "<input
type=""hidden"" name=""commit"" value=""si"">"
Response.write "<input
type=""hidden"" name=""oldname"" value="" &
trim(Request.QueryString("folder")) & "">"
Response.write
"<blockquote><blockquote><blockquote><IN
PUT TYPE=""submit"" value=""Actualizar Folder""
id=1
name=1></blockquote></blockquote></blockquote>"
Response.write "</span></form>"

'Response.write scriptname
'Response.End

Else

'Response.Write
Request.Form("rutaserver")
'Response.Write
trim(Request.Form("oldname"))
'Response.Write
trim(Request.Form("newname"))
'Response.End

fs.MoveFolder
Request.Form("rutaserver")&
trim(Request.Form("oldname")),
Request.Form("rutaserver")&
trim(Request.Form("newname"))

Application("conecta_ConnectionString") =
"Provider=MSDAORA;DATA
SOURCE=ica3;PASSWORD=ramosd;USER
ID=ramosd;"

```

```

15 Application("conecta_ConnectionTimeout") =
30 Application("conecta_CommandTimeout") =
Application("conecta_CursorLocation") = 3
Application("conecta_RuntimeUserName") =
"ramosd"
Application("conecta_RuntimePassword") =
"ramosd"
'Realiza la conexión a la Base de Datos
Dim cnn3
Set cnn3 =
Server.CreateObject("ADODB.Connection")
'conecta.Open
Application("Connection1_ConnectionString"),
Application("Connection1_RuntimeUserName"),
Application("Connection1_RuntimePassword")
cnn3.Open
Application("Conecta_ConnectionString"),
Application("Conecta_RuntimeUserName"),
Application("Conecta_RuntimePassword")

'On Error Resume Next
'Set Conn =
Server.CreateObject("ADODB.Connection")
' Conn.ConnectionTimeout =
200 Conn.CommandTimeout = 200
' Conn.Open
"SRH_EXP_INTRA","INTRANET","INTRANET"
' Set cta = Conn.Execute
(loc_cuenta)

If cnn3.Errors.Count > 0 Then
Response.Write "<b> Favor
de reintentar. Probablemente el folder esta siendo
ocupado./b>"
End If

'Response.Write
trim(Request.QueryString("folder"))
'Response.End

'Request.Form("rutaserver")

'd:\Inetpubica\docs\doc_areas\Tesoreria\

'trim(Request.Form("oldname"))
'aaa

'trim(Request.Form("newname"))
'yyy

```

```
'Session("pagAnterior")
'http://webica/filesystem_admin.asp

Application("conecta_ConnectionString") =
"Provider=MSDAORA;DATA
SOURCE=ica3;PASSWORD=ramosd;USER
ID=ramosd;"

Application("conecta_ConnectionTimeout") =
15

Application("conecta_CommandTimeout") =
30

Application("conecta_CursorLocation") = 3

Application("conecta_RuntimeUserName") =
"ramosd"

Application("conecta_RuntimePassword") =
"ramosd"

'Realiza la conexión a la Base de Datos
Dim cnn2
Set cnn2 =
Server.CreateObject("ADODB.Connection")
cnn2.Open
Application("Conecta_ConnectionString"),
Application("Conecta_RuntimeUserName"),
Application("Conecta_RuntimePassword")

dim registros
Set registros =
Server.CreateObject("ADODB.Recordset")
'.....
'Response.write
'Response.End
'.....

'Se arma la cadena para obtener la
ruta para el select
corresp =
Request.Form("rutaserver")

corresp=replace(corresp, "\", "/")

cadena2 = Len(corresp)
contador = 0

'Se toma el nombre de foder viejo
folder_viejo =
trim(Request.Form("oldname"))

'Para cada uno de los registros
Do While Not contador = 3
pos = 1
pos = InStr(pos,
Trim(corresp), "/", 0)
```

```
corresp1 =
Mid(Trim(corresp), pos + 1, pos + cadena2)
contador = contador + 1

corresp = corresp1

Loop

corresp = corresp1 + folder_viejo

'se realiza el SELECT
'str = "select * from documentos
where doc_ruta like '%" & corresp & "%"
registros.open "select * from
documentos where doc_ruta like '%" & corresp &
"%",cnn2

'corresp =
registros.Fields("doc_ruta")
'Response.Write corresp
'Response.End

Do While not registros.EOF

'valor de foder nuevo
folder_nuevo =
trim(Request.Form("newname"))

'LOS REGISTROS OBTENIDOS
DEL RECORDSET
corresp =
registros.Fields("doc_ruta")

'Response.Write corresp
'Response.End

corresp= replace(corresp,
trim(Request.Form("oldname")),trim(Request.Form("n
ewname")))

'La actualización del folder
strsql1= "Update documentos set
doc_ruta = '" & corresp & "' & " where id_doc = " &
registros.Fields("id_doc") & "'

'Response.Write strsql1
'Response.End

dim cmdTemp
Set cmdTemp =
Server.CreateObject ("ADODB.Command")

cmdTemp.ActiveConnection = cnn2
cmdtemp.CommandText =
strsql1

cmdtemp.execute

registros.MoveNext
```







```

valign=""bottom"" >" & folder.datecreated &
"</font></td>" & vbCrLf
    Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"" >" & folder.datelastaccessed &
"</td>" & vbCrLf
    Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"" >" & folder.datelastmodified & "</td>"
& vbCrLf
    Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"" ><a href="" & scriptname &
"?action=BorrarFolder&path=" &
server.URLEncode(sPath) & "&folder=" &
server.URLEncode(folder.name) & "" style='font-
family: Verdana;font-size: 8 pt;color:
rgb(129,15,172)'><img src='Images/del.gif'
alt='Eliminar' width=10 border=0></a></td>" & vbCrLf
    Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"" ><a href="" & scriptname &
"?action=ActualizarFolder&path=" &
server.URLEncode(sPath) & "&folder=" &
server.URLEncode(folder.name) & "" style='font-
family: Verdana;font-size: 8 pt;color:
rgb(129,15,172)'><img src='Images/editicon.gif'
alt='Editar' width=10 border=0></a></td>" & vbCrLf
    Response.Write "</tr>" & vbCrLf
    Next
    Set foldercollection=nothing

' Usa la propiedad "Files" para obtener los
archivos en el directorio especificado
Set filecollection = fileobject.Files

    Application("conecta_ConnectionString") =
"Provider=MSDAORA;DATA
SOURCE=ica3;PASSWORD=ramosd;USER
ID=ramosd;"

    Application("conecta_ConnectionTimeout") =
15

    Application("conecta_CommandTimeout") =
30

    Application("conecta_CursorLocation") = 3

    Application("conecta_RuntimeUserName") =
"ramosd"

    Application("conecta_RuntimePassword") =
"ramosd"

'Realiza la conexión a la Base de Datos
Dim cnn2
Set cnn2 =
Server.CreateObject("ADODB.Connection")

'conecta.Open
Application("Connection1_ConnectionString"),

```

```

Application("Connection1_RuntimeUserName"),
Application("Connection1_RuntimePassword")
    cnn2.Open
Application("Conecta_ConnectionString"),
Application("Conecta_RuntimeUserName"),
Application("Conecta_RuntimePassword")

    dim candado
    Set candado =
Server.CreateObject("ADODB.Recordset")

' Despliega los archivos contenidos en el
directorio especificado
For Each file in filecollection

' Despliega todos menos
"misligas.txt"
if StrComp(file.name,"misligas.txt")
<> 0 then

    If lineid = 0 Then
        bgcolor = bgcolor_off
        lineid = 1
    Else
        bgcolor = bgcolor_on
        lineid = 0
    End if
    dim archivo1,archivo2
        archivo1 =

"ActualizarArchivo.asp"
    Response.Write "<tr>" & vbCrLf

        dim str
        str = "select * from documentos
where doc_nom = " & file.name & ""
'Response.Write str
        candado.open "select * from
documentos where doc_nom = " & file.name &
"",cnn2

'Response.End
'Response.Write
candado.Fields("doc_priv")
    dim letra
    letra = trim
(candado.Fields("doc_priv"))
'Response.Write "<br>" & letra
.....

    if letra = "N" then
        Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom""><img
src='../images/transp.gif' width=20 height=20 border=0
alt='Document'><img src='../images/icoDoc.gif'
width=16 height=14 border=0 alt='Document'></td>"

    else

        Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom""><img
src='../images/candado.gif' width=20 height=20

```

```
border=0 alt='Document'><img
src='../images/icoDoc.gif' width=16 height=14
border=0 alt='Document'></td>"
End if
.....
candado.Close
'Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom""><img
src='Images/icoDoc.gif' width=16 height=14 border=0
alt='Document'></td>"

'Response.End
dim archivo
archivo =

"login_arch_admin.asp"

Response.Write "<td
class=letbody bgcolor="" & bgcolor & "" align=""left""
valign=""bottom""><a
href=""&archivo&"?nombre_archivo=""&Replace(file.n
ame," ", "%20")&"">" & file.name & "</a></td>"
&vbCrLf

Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom"">" &
fs.GetExtensionName(file.name) & "</font></td>"
&vbCrLf

Call Size(file.size)
Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom"">" & file.datecreated
& "</td>" &vbCrLf

Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom"">" &
file.datelastaccessed & "</td>" &vbCrLf

Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom"">" &
file.datelastmodified & "</td>" &vbCrLf

Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom""><a href="" &
scriptname & "?action=BorrarArchivo&path=" &
server.URLEncode(sPath) & "&file=" &
server.URLEncode(file.name) & "&filetype=" &
Lcase(fs.GetExtensionName(file.name)) & ""
style='font-family: Verdana;font-size: 8 pt;color:
rgb(129,15,172)'><img src='Images/del.gif'
alt='Eliminar' width=10 border=0></a></td>" &vbCrLf
Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom""><a
href=""&archivo1&"?file=""&server.URLEncode(file.na
me)&"&path=""&server.URLEncode(sPath)&""><img
src='Images/editicon.gif' alt='Editar' width=10
border=0></a></td>" &vbCrLf
```

```
'Response.Write "<td
class=letbody bgcolor="" & bgcolor & ""
align=""center"" valign=""bottom""><a
href=""ActualizarArchivo?action=ActualizarArchivo&pat
h="" & server.URLEncode(sPath) & "&file=" &
server.URLEncode(file.name) & "&filetype=" &
Lcase(fs.GetExtensionName(file.name)) & ""
style='font-family: Verdana;font-size: 8 pt;color:
rgb(129,15,172)'>Actualizar</a></td>" &vbCrLf
Response.Write "</tr>" &vbCrLf
end if

Next

dim miLiga
dim nameLink
sFile = sFile & "misligas.txt"

Set Archivo = fs.OpenTextFile(replace(raiz &
sPath & "/misligas.txt", "/", "\"))

' Despliegas las ligas contenidas en el
directorio especificado
DO WHILE NOT Archivo.AtEndOfStream
miLiga = Archivo.ReadLine

If lineid = 0 Then
    bgcolor = bgcolor_off
    lineid = 1
Else
    bgcolor = bgcolor_on
    lineid = 0
End if
Response.Write "<tr>" &vbCrLf
nameLink = Archivo.ReadLine
Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom""><img src='Images/icoLink.gif'
width=12 height=13 border=0 alt='Liga'></td>"
Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""left""
valign=""bottom""><a href="" & miLiga & " target='_top'
style='font-family:Helvetica, Arial;font-size: 9 pt;'>" &
nameLink & "</a></td>" &vbCrLf
Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"">Link</td>" &vbCrLf
Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"">&nbsp;</td>" &vbCrLf
Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"">&nbsp;</td>" &vbCrLf
Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"">&nbsp;</td>" &vbCrLf
Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"">&nbsp;</td>" &vbCrLf
Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom"">&nbsp;</td>" &vbCrLf
Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom""><a href="" & scriptname &
```

```

"?action=BorrarLiga&path=" &
server.URLEncode(sPath) & "&file=misligas.txt&liga="
& server.URLEncode(miLiga) & "&nameLink=" &
server.URLEncode(nameLink) & "" style='font-family:
Verdana;font-size: 8 pt;color: rgb(129,15,172)'><img
src='Images/del.gif' alt='Eliminar' width=10
border=0></a></td>" &vbCrLf
    'Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom""><a href="" & scriptname &
"?action=BorrarLiga&path=" &
server.URLEncode(sPath) & "&file=misligas.txt&liga="
& server.URLEncode(miLiga) & "" style='font-family:
Verdana;font-size: 8 pt;color:
rgb(129,15,172)'>Borrar</a></td>" &vbCrLf
    Response.Write "<td class=letbody
bgcolor="" & bgcolor & "" align=""center""
valign=""bottom""><a href="" & scriptname &
"?action=actualizarLiga&path=" &
server.URLEncode(sPath) & "&file=misligas.txt&liga="
& server.URLEncode(miLiga) & "&nameLink=" &
server.URLEncode(nameLink) & "" style='font-family:
Verdana;font-size: 8 pt;color: rgb(129,15,172)'><img
src='Images/editicon.gif' alt='Editar' width=10
border=0></a></td>" &vbCrLf
    Response.Write "</tr>" &vbCrLf
LOOP

    Response.Write
"<tr><td><br>&nbsp;</td></tr>" &vbCrLf
    Response.Write "</table>" &vbCrLf

End Sub

.....
' Subproceso: DisplayErrors
.....

Sub DisplayErrors
    Response.Write "You have attempted to
perform " & errornum & " prohibited action(s)."
    Response.Write "<ul>" & errorcode & "</ul>"
& vbCrLf
End Sub
%>

```

## Ticker.asp

Módulo principal para el manejo del ticker.

```

<!--#include file = "_private/Ejecuta_Query.inc"-->
<%

Set rs3= Server.CreateObject ("adodb.recordset")
rs3.open "select * from Banner",cnn1
buznumber = rs3.Fields("bann_texto")

%>

<html>
<head>
</head>
<title>ticker</title>
<script language="JavaScript"
src="ticker.js">
</script>

</head>
<body topmargin=0 leftmargin=0
background="images/fondogrisgal.gif">
<form method="post" name="formReporte"
id="formReporte">
<input type= "hidden"
name="checatoggle" value="<%=buznumber%>">
</input>

<div align="center">
</form>
<script>
var descripcion="";
descripcion =
document.formReporte.checatoggle.value;
ticker(descripcion);

/*ticker(checatoggle.value);*/
</script>
</div>

</body>
</html>

```

## ActualizarArchivo.asp

Módulo principal para actualizar un archivo dentro de la parte administrativa.

```

<!--#include file = "_private/Ejecuta_Query.inc"-->
<%

id_area=Request.form ("area")
ruta1 =request.form("ruta")

IF id_area="" THEN
id_area=Request.QueryString ("id_area")

'Response.Write "id_area = " & id_area & "<BR>"
END IF

if id_area <> "" then
Session("Id_Area")= id_area
'Response.Write "asigna session"
else
id_area = Session("Id_Area")
'Response.Write "asigna id"
end if

'Response.Write "id_area = " & id_area & "<BR>"

vsq="select * from Areas where id_area="&id_area
'Response.Write(vsq) & "<BR>"
Set Rs1 = Cnn1.Execute (vsq)
area_nom = Rs1("area_nom")
'Response.Write area_nom
'response.end

usuario=session("usuarioid")
'Response.Write "usr = " & id_area & "<BR>"

if id_area = "" then

sql="select * from usuarios where
id_usr = "&usuario&""
'Response.Write(sql)
set RSid= Cnn1.Execute (sql)
id_area = RSid("id_area")
'Response.Write(id_area)

end if

if trim(id_area) <> "" then
vsq="select * from Areas where
id_area="&id_area&""
else
vsq="select * from Areas"
end if
'Response.Write(vsq)
set RSid= Cnn1.Execute (vsq)

```









```

set rsusuario =
server.CreateObject("ADODB.Recordset")

'vSQL =
"select * from usuarios"

vSQL =
"select * from usuarios order by
usu_nombre,usu_apaterno,usu_amaterno"

rsusuario.Open vSQL,cnn1,3,3

dim a_usuarios
a_usuarios = ""

dim i, permiso, snombre
i=0

do while not
Rsusuario.EOF

i=i+1

a_usuarios=
a_usuarios & "usuarios[" & i & ",1] = " & chr(34) &
rsusuario.Fields("id_usr").Value & chr(34) & chr(13)

snombre =
rsusuario.Fields("usu_nombre")

if not
isnull(rsusuario.Fields("usu_apaterno")) then

snombre =
snombre & " " & rsusuario.Fields("usu_apaterno")

end if

if not
isnull(rsusuario.Fields("usu_amaterno")) then

snombre =
snombre & " " & rsusuario.Fields("usu_amaterno")

```

```

end if

a_usuarios=
a_usuarios & "usuarios[" & i & ",3] = " & chr(34) &
snombre & chr(34) & chr(13)

'Selecciona a los usuarios con permisos

vSQL =
"select count(*) from permisos where id_doc=" &
id_doc & " and id_usr=" & rsusuario.Fields("id_usr") &
""

set RsPer
= server.CreateObject("ADODB.RecordSet")

RsPer.Open vSQL, cnn1

if
RsPer.fields(0) = "0" then

permiso = 0

Response.Write "<OPTION id=optusr_" &
rsusuario.Fields("id_usr") & " value=" &
rsusuario.Fields("id_usr") & ">" & snombre &
"</OPTION>" & chr(13)

Else

permiso = 1

Response.Write "<OPTION id=optusr_" &
rsusuario.Fields("id_usr") & " value=" &
rsusuario.Fields("id_usr") & " selected >" & snombre &
"</OPTION>" & chr(13)

End if

a_usuarios= a_usuarios & "usuarios[" & i &
",2] = " & permiso & chr(13)

```

```

rsusuario.movenext

RsPer.close

loop

Rsusuario.close

Response.Write
chr(13) & "<script language=JavaScript>" & chr(13)

Response.Write "//
Matriz de Usuarios" & chr(13)

Response.Write
"var usuarios = new Array(" & i & ", 3)" & chr(13)

Response.Write
a_usuarios

Response.Write
"</script>"

Set rsusuario =
Nothing

%>

</select>

</td>
</tr>
</table>
</center>
</div>

<div align="center">
<center>
<table width="713" border="0">
<tr>
<td align=center
width="365">
<input
TYPE="button" VALUE="Aceptar" onClick="Checa()"
name="Aceptar">

```

```

</td>
<td align=center
width="334">
<input
TYPE="button" VALUE="Cancelar"
onClick="javascript:history.go(-1);" name="Cancelar">
</td>
</tr>
</table>
</center>
</div>
</form>
</body>
</html>

```

# BIBLIOGRAFÍA

Para la realización del presente trabajo consultamos los siguientes libros y direcciones web.

Atre, S. DATA BASE: STRUCTURED TECHNIQUES FOR DESIGN PERFORMANCE AND MANAGEMENT. John Wiley & Sons. 1980.

Booch, Grady. OBJECT-ORIENTED ANALYSIS AND DESIGN WITH APPLICATIONS. Benjamín /Cummings Publishing Company, Inc. Segunda edición. Estados Unidos de América, 1994. 589 pp.

Booch, Grady. Rumbaugh, James. Jacobson, Ivar. THE UNIFIED MODELING LANGUAGE / USER GUIDE. Addison-Wesley. Estados Unidos de América, 1998. 482 pp.

Campderrich, B. TÉCNICAS DE BASES DE DATOS. Editores Técnicos Asociados, 1986.

Codd E.F. THE RELATIONAL MODEL FOR DATABASE MANAGEMENT VERSION 2. Addison Wesley. 1990.

Cohen, Daniel. Asín, Enrique. SISTEMAS DE INFORMACIÓN PARA LOS NEGOCIOS. 3ª Edición. Mc Graw Hill.

Date, C. J. INTRODUCCIÓN A LOS SISTEMAS DE BASES DE DATOS. Vol. I, 5ª Edición. Addison Wesley Iberoamericana. 1990.

---

De Miguel, A. Piattini, M. CONCEPCIÓN Y DISEÑO DE BASES DE DATOS. DEL MODELO E/R AL MODELO RELACIONAL. Ra-Ma. 1993.

Elmasri, R.; Navathe, S.B. SISTEMAS DE BASES DE DATOS. CONCEPTOS FUNDAMENTALES. Addison-Wesley Iberoamericana. 1997.

Fernández, C. EL MODELO RELACIONAL DE DATOS: DE LOS FUNDAMENTOS A LOS MODELOS DEDUCTIVOS. Díaz de Santos. 1987.

Fernández, C.B. EL MODELO RELACIONAL DE DATOS. DE LOS FUNDAMENTOS A LOS MODELOS DEDUCTIVOS. Diaz de Santos. 1987.

Gardarin, G. DOMINAR LAS BASES DE DATOS. Ediciones Gestión 2000. 1993.

Gardarin, G.; Valduriez, P. RELATIONAL DATABASES AND KNOWLEDGE BASES. Addison-Wesley. 1989.

Hansen, G.W.; Hansen, J.V. DISEÑO Y ADMINISTRACIÓN DE BASES DE DATOS. Prentice-Hall. 1997.

Hursch, C.; Hursch, J. SQL. EL LENGUAJE DE CONSULTA ESTRUCTURADO. Ra-Ma. 1998.

Khoshafian S. OBJECT-ORIENTED DATABASES. Wiley Professional Computing. 1993.

Kim W. MODERN DATABASE SYSTEMS: THE OBJECT MODEL, INTEROPERABILITY, AND BEYOND. ACM press. 1995.

Korth, H. F.; Silberschatz, A. FUNDAMENTOS DE BASES DE DATOS. 3ª edición. McGraw-Hill. 1998.

Korth, H.; Silberschatz, A. FUNDAMENTOS DE BASES DE DATOS. McGraw-Hill. 1987

Kroenke, D:M. PROCESAMIENTO DE BASES DE DATOS. FUNDAMENTOS, DISEÑO E INSTRUMENTACIÓN. Prentice-Hall. 1996.

Larman, Craig. APPLYING UML AND PATTERNS AN INTRODUCTION TO OBJECT-ORIENTED ANALYSIS AND DESIGN. Prentice Hall. Estados Unidos de América, 1998. 507 pp.

Moran Mejía, Ma. Esther. MÉTODOS PARA EL ANÁLISIS Y DISEÑO DE SISTEMAS ORIENTADOS A OBJETOS. Tesina UNAM ENEP Acatlán. 1998. 98 pp.

Rumbaugh, James. Blaha, Michael. Premerlani, William. Eddy, Frederick. Lorencen, William. MODELADO Y DISEÑO ORIENTADO A OBJETOS y METODOLOGÍA OMT. Prentice Hall. España, 1996.

---

Silberschatz A. DATABASE SYSTEM CONCEPTS. 3rd edition. McGraw Hill. 1997.

Ullman, J. D. PRINCIPLES OF DATABASE SYSTEMS. 2ª edición. Computer Science Press. 1988.

Ullman, J.D. DATABASE AND KNOWLEDGE-BASE SYSTEM. Computer Science Press. 1988.

Vossen, G. DATA MODELS, DATABASE LANGUAGES AND DATABASE MANAGEMENT SYSTEMS. Addison-Wesley, 1990.

Waldén, Kim. Nerson, Jean-Marc. SEAMLESS OBJECT-ORIENTED SOFTWARE ARCHITECTURE ANALYSIS AND DESIGN OF RELIABLE SYSTEMS. The object-Oriented Series. Prentice Hall. Gran Bretaña, 1995. 438 pp.

Sitios *Web* consultados:

- [http://msdn.microsoft.com/library/default.asp?url=/library/en-us/architec/8\\_ar\\_cs\\_3jar.asp](http://msdn.microsoft.com/library/default.asp?url=/library/en-us/architec/8_ar_cs_3jar.asp)  
Tutorial de asp
- <http://whatis.techtarget.com/>  
Diccionario de términos tecnológicos
- <http://www.123aspx.com/directory.aspx?dir=164>  
Ligas útiles a portales sobre programación en ASP
- <http://www.asp.net/>  
Programación en ASP
- <http://www.cisco.com/univercd/cc/td/doc/cisintwk/idg4/nd2001.htm>  
Redes alámbricas en cisco
- <http://www.sql-server-performance.com/>  
Características de SQL server
- <http://www.timedancer.com/guide/webguide.html>  
Guía de configuración de servidores web
- <http://www.webopedia.com>  
Diccionario de términos tecnológicos