

56



**UNIVERSIDAD NACIONAL AUTONOMA
DE MEXICO**

FACULTAD DE INGENIERIA

SISTEMA DE INFORMACION DEPORTIVA

T E S I S

QUE PARA OBTENER EL TITULO DE:
INGENIERO EN COMPUTACION
P R E S E N T A N :

ARTURO JIMENEZ NAVARRETE
CRUZ SERGIO AGUILAR DIAZ
FRANCISCO JAVIER MONTOYA CERVANTES
GILBERTO RIVERA NAVARRETE

DIRECTOR DE TESIS: ING. POBLANO ORDOÑEZ SEBASTIAN



CIUDAD UNIVERSITARIA

**TESIS CON
FALLA DE ORIGEN**

2002



Universidad Nacional
Autónoma de México

Dirección General de Bibliotecas de la UNAM

Biblioteca Central



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

AGRADECIMIENTOS.

A la Universidad Nacional Autónoma de México por haberme formado cultural, profesional y socialmente.

A la Facultad de Ingeniería y a todos nuestros maestros de la misma, por que supieron transmitirnos sus conocimientos y experiencias con el único afán de forjarnos como buenos ingenieros.

Un agradecimiento especial a nuestro director de tesis, Ing. Sebastián Poblano Ordoñez, por el inmenso apoyo y guía para la obtención de este logro.

Y a todos aquellos que de una u otra manera colaboraron mediante observaciones, pláticas y conocimientos en la realización de esta tesis.

A todos ellos gracias.

Gilberto

Arturo

Javier

Sergio.

AGRADECIMIENTO

A nuestro director de tesis, Ing. Sebastián Poblano Ordóñez, por su inmenso apoyo y guía para la obtención de este trabajo.

Gracias
por la paciencia que nos tuvo,
por el tiempo que nos dedicó,
por transmitirnos su experiencia,
por todo lo que nos enseñó,
y sobre todo
por su amistad que nos brindó.

Gracias
A la familia Poblano Palma,
Por su comprensión y tiempo brindado.

Sergio, Javier, Arturo y Gilberto.

Dedico esta tesis

A mis padre, Dario y Amalia:

Quienes sin escatimar esfuerzo alguno, han sacrificado gran parte de su vida para formarme y educarme.

A mis once hermanos:

*Por todos los momentos que hemos compartido y su apoyo incondicional que siempre me han brindado.
Por el ejemplo a seguir que representan para mí.*

A mi hermano Satur:

Por el gran apoyo que me brindo durante los años de estudiante, es algo que nunca olvidaré.

Un reconocimiento muy especial:

Al Ing. Sebastián Poblano y familia.

A mis amigos y compañeros de tesis Sergio, Arturo y Javier.

Mis más sinceros agradecimientos a todos ellos.

Gilberto.

Dedico esta tesis:

A mis Padres, Irene y Samuel

Por el apoyo incondicional durante toda mi vida y durante todo el desarrollo de mi carrera. Ya que este logro es más de ustedes.

A mi Esposa, Claudia

Que de no haber sido por su apoyo este logro no se hubiera dado, gracias por tu confianza.

A mis hermanos y amigos

Que siempre me apoyaron en todos los aspectos de mi vida y carrera, gracias por todos los momentos que compartimos.

A mi Director de Tesis: Ing. Sebastián Poblano Ordóñez

Por todo el apoyo y sobre todo la paciencia que me tuvo, ya que su ayuda y experiencia fue parte fundamental en la obtención de este logro.

A mis compañeros de tesis: Javier, Gilberto y Sergio

Por su apoyo y colaboración para el desarrollo de este trabajo.

Sinceramente.

Arturo

Dedico esta tesis:

A mi madre: Antonia Díaz E.

Quien con muchos sacrificios y esfuerzos, me dió la oportunidad de formarme y educarme para lograr algo en la vida.

A mi Esposa: Rosa

*Quien siempre me motivó y confió en mí, le doy las gracias por los hijos que me dió ya que para mí son una gran motivación y un gran aliciente para levantarme y luchar cada día. Ana Rosa y Sergio Andrés
Gracias*

A mis hermanos y amigos.

Que siempre me apoyaron de muchas maneras y por todos los momentos que compartimos.

A mi Director de Tesis: Ing. Sebastián Poblano Ordóñez

Por todo el apoyo y paciencia que me tuvo, ya que su ayuda y experiencia fue parte fundamental en la obtención de este logro.

A mis compañeros de tesis: Javier, Arturo y Gilberto

Por su apoyo y colaboración para el desarrollo de este trabajo.

A todos los mis Amigos de UNICA y UACT.

Quienes siempre me brindaron incondicionalmente su apoyo y amistad, ya que durante la realización de este trabajo siempre pude contar con ellos.

Con mucho cariño y respeto.

Sergio

Dedico esta tesis:

A mi madre: Antonia Díaz E.

Quien con muchos sacrificios y esfuerzos, me dió la oportunidad de formarme y educarme para lograr algo en la vida.

A mi Esposa: Rosa

*Quien siempre me motivó y confió en mí, le doy las gracias por los hijos que me dió ya que para mí son una gran motivación y un gran aliciente para levantarme y luchar cada día. Ana Rosa y Sergio Andrés
Gracias*

A mis hermanos y amigos.

Que siempre me apoyaron de muchas maneras y por todos los momentos que compartimos.

A mi Director de Tesis: Ing. Sebastián Poblano Ordóñez

Por todo el apoyo y paciencia que me tuvo, ya que su ayuda y experiencia fue parte fundamental en la obtención de este logro.

A mis compañeros de tesis: Javier, Arturo y Gilberto

Por su apoyo y colaboración para el desarrollo de este trabajo.

A todos los mis Amigos de UNICA y UACT.

Quienes siempre me brindaron incondicionalmente su apoyo y amistad, ya que durante la realización de este trabajo siempre pude contar con ellos.

Con mucho cariño y respeto.

Sergio

Dedico esta tesis:

A mis Padres: Margarita y Javier

Este logro es más de ustedes por el apoyo que me brindaron y los sacrificios otorgados.

A mi Esposa: Lilia

Quien me motivó y confió en mí, para lograr esta meta.

A mis hermanos: Virginia, Ema y familia, César y Familia, Xochitl y Paco.

Que siempre me apoyaron en todos los aspectos de mi vida y carrera, gracias por todo.

A mi Director de Tesis: Ing. Sebastián Poblano Ordóñez

Por todo el apoyo y paciencia que me tuvo, ya que su ayuda y experiencia fue parte fundamental para llevar a cabo realización de este trabajo.

A mis compañeros de tesis: Sergio, Arturo y Gilberto

Aunque tarde pero lo hicimos.

Mis más sinceros agradecimientos a todos ellos.

Javier

INDICE

INTRODUCCIÓN	1
I. La Dirección General de Actividades Deportivas y Recreativas (DGADyR)	3
1.1 Historia	3
1.2 Misión, estructura y funciones	3
1.2.1 Facultades de la DGADyR	6
II. Identificación de la problemática de la DGADyR	8
2.1 Necesidades	8
2.2 Problemática	8
2.3 Organización actual	9
2.4 Objetivo	11
III. Fundamentos teóricos	13
3.1 Proceso de Desarrollo de un Sistema Informático	13
3.1.1 Metodologías más utilizadas	14
a) Ciclo de vida clásico	15
b) Construcción de prototipos	16
c) Modelo en espiral	16
d) Ciclo de vida clásico modificado	17
3.2 Técnicas de desarrollo de sistemas	17
3.2.1 Técnicas estructuradas	18
a) Programación estructurada	19
b) Diseño estructurado	20
c) Análisis estructurado moderno	22
d) Modelado de datos	25
e) Ingeniería de información	28
3.2.2 La técnica del desarrollo conjunto de aplicaciones (DCA)	28
3.2.3 Las técnicas de prototipos y desarrollo rápido	30
3.3 ¿Qué metodología y técnicas elegir?	33
a) Ciclo de vida clásico	33
b) Construcción de prototipos	33
c) Modelo en espiral o evolutivo	34
IV. Método de diseño estructurado de Yourdon	36
4.1 Marco para el modelado de procesos	37
4.1.1 Diagrama de flujo de datos	37
4.1.1.1 Características del DFD	38
a) Enfoque descendente	39
b) Cubrir los detalles	39
c) Dibujar de nuevo los diagramas	40
4.1.2 Diccionario de datos (DD)	40
4.1.2.1 ¿Cómo formar un diccionario de datos?	42
4.1.3 Diagrama de transición de estados	42
4.1.4 Especificación estructurada y tablas de decisión	43
4.1.4.1 Especificación Estructurada	44
4.1.4.2 Tablas de decisión	44
4.1.5 Diagramas o mapas de estructura	45
4.1.5.1 Diagrama de estructura	45
4.1.5.2 Propiedades	47
a) Modularidad	47
b) Cohesión	47

c) Acoplamiento	48
4.2 Marco para el modelado de datos	49
4.2.1 Base de datos	50
4.2.2 Tipos de consulta	53
4.2.3 Modelo Entidad-Relación (DER)	53
4.2.4 Normalización	55
4.3 Etapas del ciclo de vida	56
4.3.1 Estudio de viabilidad	57
4.3.2 Análisis del Sistema	57
4.3.3 Diseño del Sistema	57
4.3.4 Implementación o producción	58
4.3.5 Pruebas del sistema	58
a) Pruebas de unidad	58
b) Pruebas de integración	59
c) Pruebas de volumen	60
d) Pruebas de validación	60
4.3.6 Control de calidad	61
4.3.7 Mantenimiento	62
V. Análisis y diseño estructurado de Yourdon	64
5.1 Análisis del Sistema	64
5.2 El DFD del SIDUNAM	65
5.3 Diagrama Entidad-Relación (DER)	68
5.4 Diccionario de Datos del SIDUNAM	69
5.5 Diseño de módulos del SIDUNAM	78
VI. Análisis de la tecnología para la implementación del sistema (SIDUNAM)	84
6.1 RedUNAM	84
6.2 Red de Telecomunicaciones de la UNAM	84
6.2.1 Características principales de la Red de Telecomunicaciones	87
6.2.2 Servicios de la Red de Telecomunicaciones	88
6.3 El protocolo de comunicación TCP/IP	90
6.3.1 Historia del protocolo de comunicación TCP/IP	91
6.3.2 ¿Cómo trabaja el protocolo TCP/IP?	92
6.3.3 Direcciones IP	93
6.3.4 Principales características del protocolo TCP/IP	95
6.4 Plataformas de hardware	95
6.4.1 Procesadores CISC	96
6.4.2 Procesadores RISC	97
6.4.3 Multiproceso	98
6.4.4 Multiprocesamiento Simétrico-Asimétrico	99
6.4.5 El equipo ideal (Cliente)	101
6.4.5.1 Características deseables en los equipos Cliente	102
6.5 Plataformas de Software	107
6.5.1 Sistemas Operativos de Red	107
6.5.1.1 Linux	108
6.5.1.2 Windows NT	110
6.5.2 Sistema Administrador de Base de Datos Relacional (RDBMS)	116
6.5.2.1 Sybase	116
6.5.2.2 MS SQL Server	117
VII. Desarrollo e implantación del sistema (SIDUNAM)	122
7.1 Arquitectura Cliente/Servidor (C/S)	122

7.2 Lenguaje de programación	127
7.2.1 MS Visual Basic 6.0	127
7.2.2 Programación orientada a eventos	128
7.3 Creación de la Base de Datos para el SIDUNAM	131
7.4 Acceso a datos con MS Visual Basic 6.0	137
7.5 Seguridad del Sistema SIDUNAM	141
7.6 Distribución del SIDUNAM vía FTP	144
7.7 Propuesta de implantación	148
7.8 Alcances	151
VIII. Pruebas realizadas al sistema (SIDUNAM)	152
a) Pruebas de unidad	152
b) Pruebas de integración	152
c) Pruebas de volumen	153
d) Pruebas de validación	153
CONCLUSIONES	155
BIBLIOGRAFÍA.	159

INTRODUCCION

En los siguientes párrafos que constituyen ésta introducción, se expone la estructura general de la tesis que se presenta titulada: **Sistema de Información Deportiva de la Universidad Nacional Autónoma de México (SIDUNAM)**.

Capítulo I.- Trata de la historia, misión, estructura y funciones de hoy en día de la Dirección General de Actividades Deportivas y Recreativas (DGADyR). Este capítulo es la base para el desarrollo de éste proyecto.

Capítulo II.- Expone las necesidades de la comunidad universitaria en cuanto a una formación integral. Se plantea la problemática existente en la DGADyR, y cómo se está atacando, así como el objetivo a cumplir con el sistema a realizar.

Capítulo III.- Examina los fundamentos teóricos de las metodologías y técnicas más utilizadas para el desarrollo de un sistema informático, ventajas y desventajas en su uso, así como la relación que existe entre éstas; de igual manera, en este capítulo se propone la metodología a seguir para el desarrollo del sistema (SIDUNAM).

Capítulo IV.- Describe la metodología estructurada de desarrollo de sistemas de Yourdon, que es la metodología seleccionada para el desarrollo del sistema en cuestión.

Capítulo V.- Se aplica la metodología estructurada de Yourdon al problema, la cual se ocupa para diseñar y analizar las funciones y las relaciones de cada uno de los módulos que componen el sistema (SIDUNAM), por ejemplo, el Diagrama Entidad-Relación (DER), el Diagrama de Flujo de Datos (DFD), el Diccionario de Datos (DD), etc.

Capítulo VI.- Analiza la infraestructura tecnológica de la UNAM en cuanto a equipo de cómputo y telecomunicaciones se refiere. Se hace un estudio de la **Red-UNAM** y del protocolo de comunicación estándar de la UNAM (TCP/IP). De igual forma

se analizan el *software* (Sistema Operativo de Red y el Sistema Administrador de Base de Datos Relacional) y el *hardware* (el equipo Servidor y los equipos Cliente), sobre los cuales se pondrá en operación el sistema a realizar. Se hace un estudio comparativo de mercado de las diferentes plataformas de *hardware* y *software*, y se propone la opción más adecuada para la realización del sistema.

Capítulo VII.- Examina la arquitectura Cliente/Servidor, las características y capacidades del lenguaje de programación para llevar a cabo el sistema, así también, trata sobre la creación de la base de datos, acceso a los datos de la misma y sobre la seguridad y distribución de la aplicación. Por otra parte se da la propuesta de implantación y los alcances del sistema final.

Capítulo VIII.- Menciona el tipo de pruebas que se llevaron a cabo al sistema para asegurar y garantizar la calidad del mismo, una vez concluido éste. Finalmente se dan las conclusiones acerca de la realización del SIDUNAM.

I. LA DIRECCIÓN GENERAL DE ACTIVIDADES DEPORTIVAS Y RECREATIVAS (DGADyR)

1.1. Historia

Desde su fundación, la Universidad Nacional Autónoma de México, además de su actividad académica fundamental, se impuso la tarea de apoyar el desarrollo humano con un área especializada de formación deportiva. De esta manera se incluyó en su organización a la Dirección General de Actividades Deportivas y Recreativas, que desde entonces ha estado presente paralelamente a la vida académica no sólo de los alumnos, sino de toda la comunidad universitaria. Hasta 1998 había cumplido con el papel de la promoción y administración de las actividades deportivas de la Universidad, sin embargo, a raíz de la reestructuración de la Secretaría de Asuntos Estudiantiles ambas dependencias se replantearon el mismo objetivo, de manera que cada una trabajaría de forma complementaria con la otra. Por la misma razón, se inició un programa ambicioso de modernización en los servicios que cubría la DGADyR, que hasta hoy se lleva a cabo.

1.2. Misión, estructura y funciones

La Dirección General de Actividades Deportivas y Recreativas (DGADyR) originalmente fue una dependencia de la Secretaría de Asuntos Estudiantiles de la UNAM, y en la actualidad depende de la Secretaría de Servicios a la Comunidad Universitaria, que tiene como Misión:

"Hacer de la cultura física, deportiva y recreativa elementos que coadyuven a la formación integral del estudiante universitario, favoreciendo el desarrollo de valores tales como honradez, lealtad, autoridad, disciplina y superación personal, trabajo en equipo, respeto y sentido de identidad con su escuela con su Universidad y con su país. Propicia, asimismo, condiciones de recreación que favorezcan el tránsito exitoso de los alumnos por la Universidad"¹.

¹ UNAM, DGADyR, p 3.

La estructura de la DGADyR está integrada por las siguientes áreas:

- La Dirección General
- La Subdirección de Deportes
- La Subdirección de Logística
- La Subdirección de Investigación y Medicina del Deporte
- La Subdirección de Planeación y Desarrollo Académico
- La Subdirección de Recreación
- La Coordinación de Comunicación Social
- La Coordinación de Informática
- La Coordinación Jurídica
- La Unidad Administrativa

Las cuales tienen la tarea de la Planeación, Organización, Ejecución y Supervisión de los programas sustantivos de la Dependencia, cada Subdirección, Coordinación y Unidad Administrativa, se estructura internamente por acuerdo del Director General conforme al plan anual de actividades que se aprueba y que considera el número de deportes, equipos, eventos que se organizan y en los que participan integrantes de la comunidad a los que se les brindaran los servicios, investigaciones a realizar y programas recreativos a desarrollar, entre otros.

Los planes, programas, cursos y seminarios de formación, capacitación y actualización deportiva y recreativa se efectúan por medio del Centro de Educación Continua de Estudios Superiores del Deporte, dependiente de la Unidad de Planeación y Desarrollo Académico, conforme a los lineamientos y las normas complementarias que expide el Consejo Asesor dentro del marco de la Legislación Universitaria.

Conforme a los acuerdos emitidos por las autoridades universitarias, la Dirección General de Actividades Deportivas y Recreativas tiene las siguientes funciones.

- Promover entre la comunidad estudiantil universitaria, la práctica de actividades físico-deportivas y recreativas que contribuyan al desarrollo del bienestar biopsico-social y con esto, elevar la calidad de vida.
- Planear, organizar, desarrollar y supervisar los programas de actividades físico-deportiva para los estudiantes universitarios en las diferentes disciplinas que se llevan a cabo en la UNAM.
- Promover, reglamentar, avalar y supervisar la integración, preparación, registro y participación de los equipos deportivos estudiantiles que representan a la UNAM en los eventos Nacionales e Internacionales en sus diferentes ramas y categorías.
- Designar, remover al personal adscrito que se ocupe del desarrollo de los programas de la Dependencia.
- Coordinar y colaborar en el desarrollo y superación de las actividades de los entrenadores en jefe, presidentes de asociaciones deportivas y coordinadores deportivos de las Facultades y Escuelas de la UNAM.
- Promover y otorgar premios, distinciones y estímulos que se instituyan para los estudiantes deportistas y personal que apoye en forma excepcional al deporte universitario.
- Programar e impartir cursos para la formación, capacitación y actualización de recursos humanos en disciplinas de cultura física y deportes, metodologías, técnicas de enseñanza y entrenamiento deportivo, así como de disciplinas y ciencias aplicadas a la cultura física, el deporte y la recreación.

- Publicar y difundir los resultados de los estudios e investigaciones que se realicen en la Dependencia, así como de los avances del conocimiento, en los ámbitos Nacional e Internacional.
- Promover el intercambio académico con otras instituciones Nacionales y Extranjeras.
- Representar a la institución en el ámbito del deporte conforme lo dispongan las autoridades universitarias.
- Administrar eficientemente los recursos humanos, físicos y financieros que sean encomendados a la dependencia.
- Asesorar y colaborar con otras instituciones en el desarrollo de programas afines a los de la DGADyR.

1.2.1. Facultades de la DGADyR

Las Facultades de esta dirección le son otorgadas por la Ley Orgánica de la UNAM y la Normatividad Interna de las Facultades. Éstas son:

- Proponer al Rector, con la aprobación del Secretario de Servicios a la Comunidad Universitaria de la UNAM, el nombramiento de los funcionarios de la Dependencia.
- Proponer los nombramientos del personal académico y administrativo de la Dirección General conforme a las disposiciones vigentes en los estatutos y reglamentos universitarios, así como nombrar y remover a los entrenadores y personal auxiliar adscrito a la Dependencia.

- Organizar, dirigir y supervisar las actividades y los programas de trabajo de la Dependencia.
- Proponer a las autoridades universitarias lineamientos y políticas generales, así como los convenios en materia de cultura física, deportiva y recreación.

II. IDENTIFICACIÓN DE LA PROBLEMÁTICA DE LA DGADyR

2.1. Necesidades

Como se ha mencionado en el capítulo anterior, la Secretaría de Asuntos Estudiantiles fue reestructurada, haciéndola corresponsable de la formación integral de los estudiantes universitarios. Por ello, la Dirección General de Actividades Deportivas y Recreativas como un organismo dependiente de la primera en un principio; se replanteó la necesidad de adaptar su estructura interna a las nuevas necesidades que al paso del tiempo fueron surgiendo en la comunidad universitaria.

Para llevar a cabo este propósito se sistematizó y reorientó el desarrollo de las áreas de la Dirección, a fin de enfocar el servicio hacia un mayor número de estudiantes universitarios; paralelamente a ello, se inició la búsqueda de nuevas formas de obtención de financiamiento con el objetivo de apoyar las actividades deportivas y el mantenimiento de las instalaciones donde éstas se realizan.

2.2. Problemática

En el presente, al realizar un diagnóstico sobre la estructura y funciones de la DGADyR se encontró una desvinculación de Información con su entorno universitario y externo. Por lo tanto la DGADyR se impuso la tarea de buscar mecanismos para mejorar y ordenar su administración así como orientar sus funciones. Dichos mecanismos deberían de contener en esencia la prioridad de brindar un mejor servicio y que éste, estuviera guiado hacia los estudiantes de la UNAM.

Con el propósito de que la DGADyR tenga una administración transparente, con áreas funcionales y acordes a los lineamientos que en la actualidad le marca la Secretaría de Servicios a la Comunidad Universitaria, se empezó una reestructuración y sistematización de las funciones. Debido a ello, se pudieron crear, sin costo para la administración, Coordinaciones como la de Informática y la de Asuntos Jurídicos que en

colaboración con el área de Planeación, fueran las encargadas de organizar y estructurar programas de acción e interacción de los elementos de la Dependencia.

En esta etapa de reorganización se detectó que en algunas Dependencias se manejaba información de los Deportistas y en otras no se contaba con dicha información. Siendo que muchos de los Deportistas practicaban una o más disciplinas deportivas o bien recreativas que competen a este organismo universitario. Por lo tanto, se pretende homogeneizar dicha información y que cualquier Dependencia de la DGADyR pueda contar con ella, ya sea para consultarla o bien actualizarla en cualquier momento, desde las distintas instalaciones de la universidad.

2.3. Organización actual

Con la cooperación de la Coordinación de Informática, se instaló en el área administrativa un programa computarizado que ayudó a registrar, controlar y administrar los recursos económicos, materiales y humanos. Además fue la gestora de la adquisición de equipo de cómputo actualizado para el mismo fin. Sin embargo, era necesario crear un programa que lograra la vinculación integral entre todas las Dependencias y sus actividades deportivas.

En cuanto a la desvinculación con el entorno, se iniciaron mecanismos de relación con los Colegios y Escuelas del Nivel Medio Superior y Superior, a fin de beneficiar a un mayor número de estudiantes universitarios con los servicios que esta Dependencia ofrece. De igual manera, se realizaron acciones buscando una comunicación más abierta entre quienes dirigen el deporte universitario y los estudiantes integrantes de los equipos representativos, así como con Organismos Públicos y Privados.

Así mismo, las acciones deportivas empezaron a regirse bajo el propósito de llevar estas actividades a las diversas Dependencias Académicas, privilegiando la práctica masiva de las mismas, así como la de la actividad recreativa. Para ello, se

estableció una estrategia de trabajo basada en la coordinación y descentralización de las actividades de la Dirección y en el establecimiento de programas concretos que permitan la distribución de las tareas en todos los Centros Educativos de la Universidad.

En el área de Fútbol Americano se inició una revisión a fondo de su estructura y funcionamiento, sus alcances y costos; su reorganización propició una mayor participación de alumnos, orientó sus mayores esfuerzos a fomentar e incrementar la práctica de este deporte en diferentes Escuelas y Facultades.

Las actividades de recreación, se agruparon en tres programas: el Programa de Desarrollo de la Recreación que contiene actividades permanentes como la Ludoteca, el Acondicionamiento Físico General, Rítmico-Aeróbico, Físico-Acuático, Recreación Psicomotriz, Masaje y Ajedrez. Este último se realizó en coordinación con la Dirección General de Atención a la Comunidad Estudiantil.

A la vez, el Programa de Promoción Recreativa planea y organiza las actividades requeridas por la comunidad universitaria, que van desde Competencias Especiales en Torneos Recreativos en diversas Ramas Deportivas, hasta Caminatas, Talleres de Guitarra y Papiroflexia entre otros.

El área de Medicina del Deporte cumple una función de apoyo a las actividades deportivas y recreativas, en cuanto a atención Médico-Deportiva, además de promover una cultura del cuidado físico y del sano desarrollo corporal; para ello contempla tres programas: Servicio Asistencial, Apoyo a Eventos Deportivos y Evaluaciones Físicas. Las actividades asistenciales hacia los estudiantes deportistas abarcan áreas como la Psicológica, Odontológica, Nutricional y Clínica General.

Finalmente, las tareas y los lineamientos antes planteados son productos del compromiso adquirido el 13 de febrero de 1998, en donde se marcó para el Plan de Desarrollo de esta Dirección, un nuevo modelo de acción en torno al deporte y la

recreación universitarios, en el cual sus tareas se enfocarán a la atención de sus estudiantes universitarios en la rama deportiva y recreativa, así como vincular esta función con la sociedad a través de actividades de extensión.

Las necesidades de información que ha de cubrir este programa que logre vincular todas las actividades descritas de manera sistemática y ordenada, pueden concentrarse en los siguientes datos:

1. Datos Generales
2. Datos Académicos
3. Datos Deportivos
4. Datos Médicos
5. Datos Laborales
6. Datos de Identificación
7. Historia Deportiva
8. Estadísticas Físicas

Por tanto, el problema a resolver consiste en crear un sistema para computadora, el cual maneje la Información de todos los Deportistas, y como se mencionó anteriormente dicha información pueda ser consultada o bien actualizada, desde cualquier instalación de la universidad, por ejemplo: ENEP Zaragoza, ENP No. 2, CCH Sur, etc., ya que se cuenta con una gran cantidad de Información de los Deportistas en cada una de las 35 disciplinas deportivas de la Dirección General de Actividades Deportivas y Recreativas de nuestra Universidad Nacional Autónoma de México (UNAM).

2.4. Objetivo

El objetivo principal de este trabajo es la creación de un sistema seguro y eficiente para el manejo de información por computadora que nos permita controlar al Deportista y a todas las personas que estén involucradas en el deporte universitario, en

Escuelas, Facultades y las diversas Instalaciones que se encuentren dentro o fuera de ciudad universitaria, además este sistema deberá funcionar bajo la arquitectura Cliente/Servidor, estando este último en las oficinas de la DGADyR y los Clientes en las Escuelas y Facultades de la UNAM. También éste sistema deberá tener la posibilidad de obtener reportes impresos, en pantalla o en archivos a disco. De igual forma, el sistema contemplará alternativas de consulta, ya sea por Tipo de Persona, Dependencia o Subdirección, y sus datos correspondientes; así como su utilización en línea en forma simultánea desde los diferentes planteles de nuestra institución. Esta información tendrá que ser manejada a través de RedUNAM e Internet tanto para consulta como para actualización.

En los siguientes capítulos se define cómo se atacará el problema para tener el sistema necesario para el manejo de la gran cantidad y gran variedad de información que se tiene en la DGADyR de la UNAM.

III. FUNDAMENTOS TEÓRICOS

3.1. Proceso de desarrollo de un sistema informático

Un *sistema*, según el diccionario "Webster" corresponde a:

1. Un conjunto u ordenación de cosas relacionadas de tal manera que forman una unidad o un todo orgánico; 2. Un conjunto de hechos, principios, reglas, etc., clasificados y ordenados de tal manera que muestran un plan lógico uniendo las diferentes partes; 3. Un método o plan de clasificación u ordenación; 4. Una forma establecida de hacer algo; un método, un procedimiento...

Considerando las definiciones anteriores, podemos indicar que un *sistema basado en computadora* es:

"Un conjunto de elementos organizados para llevar a cabo algún método, procedimiento o control mediante el procesamiento de la información"¹.

Los elementos de un *sistema basado en computadora*, son²:

- *Software*. Incluye a los programas de computadora, las estructuras de datos y la documentación asociada.
- *Hardware*. Son los dispositivos electrónicos propios de la computadora y los dispositivos electromecánicos para la comunicación con el exterior.
- *Gente*. Involucra a los usuarios, operadores del software y los operadores del hardware.
- *Base de datos*. Conjunto ordenado de información la cual es accesible mediante el software.

¹Pressman, Roger S., Ingeniería del software, Editorial Mc Graw-Hill, México, 1993, pp. 140-141

²ibidem, pp. 142

- *Documentación.* Información escrita que explica la operación y uso del sistema.
- *Procedimientos.* Los pasos que definen el uso específico de cada elemento del sistema.

Para el desarrollo de un sistema basado en computadora, existen varias metodologías disponibles en la actualidad:

- Ciclo de vida clásico.
- Construcción de prototipos.
- Modelo en espiral o evolutivo.
- Ciclo de vida clásico modificado.

El proceso de desarrollo de un sistema (independientemente de la metodología que se ha seleccionado) contiene tres fases genéricas:

- Descripción y análisis de la situación actual.
- Diseño (que incluye el uso de base de datos, codificación y empleo de un lenguaje de programación y pruebas del sistema).
- Mantenimiento.

3.1.1. Metodologías más utilizadas

Antes que nada, definamos lo que es una metodología.- Una *metodología* es una versión amplia y detallada de un ciclo de vida *completo* del desarrollo de sistemas que incluye: (1) tareas paso a paso para cada fase; (2) funciones individuales y en grupo desempeñadas en cada tarea; (3) productos resultantes y normas de calidad para cada tarea, y (4) técnicas de desarrollo que se utilizarán en cada tarea.

Recordemos el propósito del ciclo de vida: planear, ejecutar y controlar el proyecto de desarrollo de un sistema. El ciclo de vida define las fases y las tareas

esenciales para el desarrollo de sistemas, sin importar el tipo de envergadura del sistema que se intenta construir.

Ya que se definió que es una metodología, entonces analizaremos las más ampliamente utilizadas:

a) Ciclo de vida clásico

El ciclo de vida clásico presenta un enfoque sistemático o secuencial del desarrollo de un sistema que comienza en el nivel de la ingeniería de sistemas y progresa a través del análisis, diseño, codificación, prueba y mantenimiento.

Ingeniería de sistemas. Dado que el software forma parte de un sistema mayor, se debe comenzar por establecer los requisitos de todos los elementos del sistema y luego asignarlos al software.

Análisis. Se recopilan los requisitos necesarios para el desarrollo del sistema, además se propone su función, su rendimiento y las interfaces requeridas.

Diseño. Traduce los requisitos en una representación de software. Se documenta y forma parte de la configuración del software.

Codificación. El diseño se traducirá en forma legible para la máquina; si el diseño se realizó detalladamente, la codificación podrá realizarse mecánicamente.

Prueba. La prueba se centra en la lógica interna del programa, comprobando sentencias, funciones externas, entradas y resultados esperados.

Mantenimiento. El mantenimiento del software aplica cada uno de los pasos precedentes del ciclo de vida a un programa existente en vez de a uno nuevo.

b) Construcción de prototipos

La *construcción de prototipos* es un modelo que facilita la creación del sistema a construir mediante una de las tres formas siguientes:

- Un prototipo en papel o un modelo basado en computadora que describa la interacción hombre – máquina.
- Un prototipo que implemente algunos subconjuntos de la función requerida del programa deseado.
- Un programa existente que ejecute parte o toda la función deseada, pero que tenga otras características que deban ser mejoradas en el nuevo trabajo de desarrollo.

Como en el ciclo de vida clásico, se comienza con la recolección de los requisitos, luego se produce un "diseño rápido", el cual se enfoca sobre la representación de los aspectos del software visibles al usuario, de esta manera, el "diseño rápido" conduce a la construcción del prototipo que será evaluado por el cliente/ usuario, y que será utilizado para refinar los requisitos del software a desarrollar. Se produce un proceso interactivo en el que el prototipo se va afinando para satisfacer las necesidades del cliente, al mismo tiempo que va ayudando al que lo desarrolla.

c) Modelo en espiral

El *modelo en espiral* se ha desarrollado para cubrir las mejores características tanto del ciclo de vida clásico, como de la creación de prototipos añadiendo un nuevo elemento: el análisis de riesgo.

Esta metodología define cuatro actividades:

- *Planificación.* Determinación de objetivos, alternativas y restricciones.

- *Análisis de riesgo.* Análisis de alternativas e identificación y/o resolución de riesgos.
- *Ingeniería.* Desarrollo del producto de "siguiente nivel".
- *Evaluación del cliente.* Valoración de los resultados de la ingeniería.

En cada iteración alrededor de una espiral se construyen sucesivas versiones del software, cada vez más completas. Durante la primera vuelta se definen los objetivos, las alternativas y las restricciones, y se analizan e identifican los riesgos. El cliente evalúa y basándose en los comentarios se produce la fase de planificación y de análisis de riesgo. Si el análisis de riesgo indica que hay una incertidumbre en los requisitos, se puede usar la creación de prototipos para refinar los requisitos.

En cada bucle alrededor de la espiral, la culminación del análisis de riesgo resulta en una decisión de "seguir o no seguir". Cada vuelta alrededor de la espiral requiere ingeniería que se puede llevar a cabo mediante el enfoque del ciclo de vida clásico o de la creación de prototipos.

d) Ciclo de vida clásico modificado

Tomando en cuenta los elementos del ciclo de vida clásico previamente expuestos, y con la intención de hacerlo aplicable a la mayoría de las situaciones, surge el *ciclo de vida clásico modificado* que proporciona más flexibilidad entre las fases, permitiendo iteraciones entre ellas, según sean necesarias; trata de acomodar cambios en los requerimientos y disminuir los efectos de errores en cada etapa del desarrollo.

3.2. Técnicas de desarrollo de sistemas

Definamos lo que es una técnica.- Una *técnica* es un método que aplica herramientas y reglas específicas para completar una o más fases del ciclo de vida del desarrollo de sistemas. Uno de sus sinónimos habituales es el *paradigma*.

Las técnicas son, en su mayoría, sólo aplicables a una parte del ciclo de vida total. Por tanto, no pueden, por sí mismas, reemplazar al ciclo de vida. Algunas de las más conocidas técnicas de desarrollo de sistemas pretenden introducir en el ciclo de vida una precisión y un rigor propios de la ingeniería.

Son de destacar dos puntos importantes. En primer lugar, una auténtica metodología debe acompañar al ciclo de vida completo del desarrollo de sistemas, incluido el soporte de sistemas. Y segundo, la mayor parte de las metodologías modernas incluyen el uso de varias técnicas de desarrollo (con sus herramientas asociadas).

3.2.1. Técnicas estructuradas

Las técnicas estructuradas son métodos formales de división de un problema de empresas en fragmentos y relaciones manejables, y la ulterior reunión de estos fragmentos y relaciones (posiblemente, con añadidos y eliminaciones) en una solución de empresa (e informática) útil para resolver el problema. Uno de sus sinónimos es *métodos estructurados*.

En cierto sentido, las técnicas estructuradas utilizan el método de "divide y vencerás" para resolver problemas relacionados con el desarrollo de software y sistemas. A continuación se mencionarán las técnicas más populares entre los desarrolladores de sistemas:

- Programación estructurada.
- Diseño estructurado (varios métodos).
- Análisis estructurado moderno.
- Modelado de datos.
- Ingeniería de información.

La mayoría de las técnicas estructuradas de hoy en día se centran en dos visiones de los sistemas: **Actividades y Datos**. Recordemos que los bloques elementales **Actividades** se basaban en el concepto entrada-proceso-salida (EPS). Como el proceso es el núcleo de la transformación entrada-salida, estas técnicas reciben con frecuencia el nombre de **técnicas orientadas a procesos**. Las técnicas de procesos elaboran modelos de sistemas que se basan en el estudio de los procesos y/o sus entradas y salidas. El diseño estructurado y el análisis estructurado moderno son ejemplos de estas técnicas.

Las **técnicas orientadas a datos** establecen modelos de sistemas que se basan en una organización y un acceso ideales a los datos del sistema, independientemente de cómo se utilicen estos datos para satisfacer las necesidades de información (salidas). El modelado de datos y la ingeniería de información son ejemplos de técnicas orientadas a datos.

Veamos a detalle estas técnicas estructuradas:

a) Programación estructurada

La programación estructurada es una técnica orientada a procesos para el diseño y la escritura de programas con mayor claridad y consistencia. En esencia, la programación estructurada sugiere que el esquema lógico de cualquier programa debería ser escrito con un conjunto limitado de estructuras de control.

Esta técnica tiene que ver sólo con la lógica y la codificación de los programas. Propone que los programas deberían diseñarse de manera que pudieran leerse de principio a fin con un mínimo de ramificaciones. En particular, los programas bien estructurados se escriben exclusivamente con diversas combinaciones de tres **estructuras de control limitadas**. Estas tres estructuras básicas son:

- Una *secuencia* de instrucciones o grupo de instrucciones.

- Una *selección* de instrucciones o grupo de instrucciones basadas en ciertos criterios de decisión (esta construcción recibe a menudo el nombre de *if-then-else* o construcción de casos).
- Una *iteración* de instrucciones o grupo de instrucciones que se repite sobre la base de ciertos criterios (construcción que se da en dos formas básicas *repeat-until* y *do-while*).

Esta construcción puede repetirse (o anidarse) en otras construcciones. Una característica importante de las estructuras es que cada construcción debe tener la propiedad de disponer de una sola entrada y una sola salida. Ello quiere decir que sólo puede haber un punto de entrada y un punto de salida en una estructura dada. El código estructurado se lee como ésta página, de arriba abajo, sin referencias hacia atrás. Esto hace que sea más fácil de leer, probar, depurar y mantener.

Como técnica estructurada, la programación estructurada sólo da apoyo a algunas partes de las fases de diseño, implantación y soporte del ciclo de vida. Los programas estructurados son más fáciles de escribir, probar y depurar, y mucho más sencillos de mantener.

b) Diseño estructurado

Las técnicas de diseño estructurado ayudan a las personas que hacen desarrollos a abordar programas complejos y de grandes dimensiones. El diseño estructurado es una técnica orientada a procesos utilizada para fragmentar un programa grande en un conjunto jerarquizado de módulos y obtener un programa informático más fácil de implantar y mantener (cambiar), entre sus sinónimos (aunque técnicamente inadecuados) se encuentran *diseño descendente de programas* y *programación estructurada*.

La idea es simple. Diseñar un programa como una distribución jerárquica descendente de módulos. Un módulo es un grupo de instrucciones: un párrafo, un

bloque, un subprograma o una subrutina. La estructura descendente de estos módulos se desarrolla conforme a diversas reglas y directrices de diseño.

Diferentes escuelas de pensamiento han desarrollado su propia técnica para conseguir diseños bien estructurados. Entre estas escuelas se incluyen las siguientes:

- **Yourdon-Constantine.** Esta técnica obtiene la estructura de software ideal por el estudio de flujo de datos a través de las funciones de programa necesarias. (Nota: Esta técnica describe la estructura descendente de los módulos en forma de un árbol invertido.)
- **Warnier-Orr.** Esta técnica obtiene la estructura de software ideal por el estudio del contenido de las salidas y las entradas. (Nota: Esta técnica describe la disposición jerárquica descendente en forma de series de llaves de izquierda a derecha.)
- **Jackson.** Esta técnica también obtiene la estructura de software ideal por el estudio del contenido de las salidas y las entradas. (Esta técnica, extendida en Europa, describe la disposición jerárquica descendente de los módulos en forma de un árbol invertido, similar al de Yourdon-Constantine.)

Todas estas técnicas se consideran técnicas de procesos, dado que su propósito es diseñar procesos y, específicamente, procesos de software.

El diseño estructurado de Yourdon, cuenta con las siguientes propiedades:

- Los módulos deben tener una fuerte *cohesión*; es decir, cada módulo debería comportar una y solo una función. Así, los módulos serán reutilizables en futuros programas.
- Los módulos deben estar debidamente *acoplados*; es decir, han de tener una dependencia mínima unos de otros. Ello reduce al mínimo el efecto que futuros cambios en un módulo pueden producir en otros módulos.

El modelo de software obtenido del diseño estructurado de Yourdon recibe el nombre de **diagrama de estructuras**. El diagrama de estructuras se obtiene del estudio del flujo de datos a través del programa. El diseño estructurado se lleva a cabo durante la fase de diseño del ciclo de vida. No cubre todos los aspectos del diseño; por ejemplo, el diseño estructurado no servirá de ayuda para diseñar las entradas, la base de datos o los archivos. Las ventajas obtenidas del diseño estructurado son numerosas. En primer lugar, los programas que se desglosan conforme al diseño estructurado son más fáciles de leer y de probar por equipos de programadores múltiples, ya que las interfaces entre los módulos también estarán bien definidas y limitadas por las reglas, los módulos cuya prueba sea correcta deberían funcionar igual de bien cuando se integraran en el sistema global. En segundo lugar, los sistemas y los programas desarrollados con diseño estructurado son más fáciles de mantener. En tercer lugar, una de las ventajas, con frecuencia olvidada, del diseño estructurado es que los módulos de programas desarrollados conforme a la técnica suelen ser reutilizables. Ello se debe a que han sido escritos con cohesión.

c) Análisis estructurado moderno

El análisis estructurado es la técnica centrada en procesos más popular y ampliamente practicada, que se utiliza para realizar modelos de las necesidades de usuario en un sistema. El análisis estructurado divide un sistema en procesos, entradas, salidas y archivos. Elabora modelos del tipo entrada-proceso-salida orientados a flujos para un problema o una solución de empresa.

Los principales impulsores del análisis estructurado son Tom DeMarco, Chris Gane, Trish Sarson y Ed Yourdon.

La técnica del análisis estructurado es sencilla en su concepto. Un nuevo modelo del sistema evoluciona a partir de una serie de diagramas orientados a flujos denominados **diagramas de flujo de datos** o DFD. Como los diagramas de flujo de datos describen el flujo de datos entre procesos, el análisis estructurado recibe también

a veces el nombre de *método de flujo de datos*. Por otra parte, en su mayoría, los especialistas consideran que los DFD son *modelos de procesos* y centran su análisis en dichos procesos (de ahí que el análisis estructurado sea en realidad un método orientado a procesos).

El analista debería producir varios DFD en su análisis estructurado. Estos DFD pueden diferir en lo que se refiere a:

- Si el modelo corresponde al sistema actual o al sistema propuesto.
- Si el modelo corresponde a los detalles de implantación del sistema (a veces denominado sistema *físico*) o a los fundamentos del sistema (lo que se llama, en ocasiones, sistema *lógico*).

Así pues, el modelo obtenido con los DFD ha de corresponder a: (1) la implantación del sistema actual; (2) las bases del sistema actual; (3) las bases del sistema propuesto, o (4) la implantación del sistema propuesto.

El concepto de **sistema lógico**, a veces denominado **sistema esencial**, fue creado por el análisis estructurado, para el cual es vital. En general, solemos recortar nuestra propia creatividad al pensar, antes de tiempo, en todo nuevo sistema según *cómo* debería diseñarse e implementarse (el **sistema físico**). El análisis estructurado anima al analista (o le exige) a que defina *qué* debería hacer un sistema (el **sistema lógico**) antes de decidir *cómo* diseñarlo o implementarlo. Por tanto, esta técnica obliga al analista a considerar primero la "solución de empresa" y sólo más tarde, la "solución técnica". Sus defensores insisten en que, al separar las cuestiones lógicas y física, se obtienen las siguientes ventajas:

- El analista identifica con mayor precisión las necesidades de la empresa y de los usuarios finales, al no preocuparse antes de tiempo por las cuestiones tecnológicas.

- El analista muestra mayor tendencia a concebir soluciones alternativas más creativas, en lugar de soluciones basadas en el sistema existente (evitando el síndrome "siempre lo hemos hecho así"). Las soluciones creativas pueden promover ventajas del tipo competitivo en la compleja situación económica actual.

Recientemente, Ed Yourdon introdujo una versión mejorada del análisis estructurado denominada **análisis estructurado moderno**. Este método elimina el modelado detallado del sistema actual (tanto lógico como físico).

Adviértase que el interés central de todas las etapas se ha desplazado claramente hacia el nuevo sistema, prestándose escasa atención al sistema actual. Las diferentes variedades de análisis estructurados se encuadran o bien en el método original o bien en el método moderno. En la actualidad, la mayoría de los expertos admite que, como mínimo, debería dedicarse menos tiempo al modelado detallado del sistema actual. Ello se debe a que: (1) la cantidad de tiempo a menudo no justifica el costo; (2) el sistema actual tiende a condicionar o limitar la creatividad cuando cambia el contexto desde el sistema actual al nuevo sistema, y (3) los usuarios y los directivos muestran poca paciencia a la hora del modelado de un sistema que, según las expectativas, será modificado o sustituido.

El análisis estructurado fue la primera técnica estructurada que se utilizó para estudiar de modo específico las fases del análisis de sistemas del ciclo de vida. Pero sólo examina dichas fases desde el punto de vista de las especificaciones o el modelado. El analista debe acudir a otras técnicas para recoger los hechos, analizar los problemas y las oportunidades del sistema, identificar las necesidades de los usuarios, identificar las soluciones candidatas y evaluar la viabilidad de estas soluciones.

Debería advertirse que el análisis estructurado y el diseño estructurado son técnicas integradas. El método de diseño estructurado de Yourdon proporciona estrategias formales para obtener diagramas de estructuras de programas a partir de

diagramas de flujo de datos elaborados adecuadamente durante el análisis estructurado. En su conjunto, esta técnica se suele también llamar *ingeniería de software*.

d) Modelado de datos

En la actualidad, las técnicas orientadas a procesos han recibido el complemento de las técnicas orientadas a datos.

El **modelado de datos** es una técnica orientada por los datos que representa un sistema en función de sus datos, independientemente de cómo se procesen dichos datos para producir información.

En esta ocasión de igual manera, el concepto es sencillo. Si se capturan los datos y se almacenan en estructuras flexibles de archivos y base de datos, todas las necesidades actuales y futuras de información pueden satisfacerse por medio del uso de dichos datos. El modelado de datos ha evolucionado a partir de las técnicas utilizadas por los diseñadores de base de datos.

Las técnicas de modelado de datos se describen brevemente del modo siguiente. Primero, identificar los "entes" de empresa (denominados *entidades*) en torno a los cuales captan datos las aplicaciones o la empresa. Estas entidades pueden ser:

- Cosas tangibles (por ejemplo, materiales, suministros, máquinas, vehículos y productos).
- Funciones (como clientes, proveedores, empleados y accionistas).
- Sucesos (como pedidos, devoluciones, contratos, viajes, accidentes o pagos).
- Lugares (como oficinas o almacenes).

A continuación, identificar aquellos *atributos* que describen una presencia de cada entidad. Por ejemplo, consideremos la entidad ESTUDIANTE. Algunos de los atributos que describen una presencia de ESTUDIANTE son: NÚMERO DE IDENTIFICACIÓN DEL ESTUDIANTE, NOMBRE DEL ESTUDIANTE, DIRECCIÓN EN EL CAMPUS, TELÉFONO EN EL CAMPUS, DIRECCIÓN DE SU DOMICILIO, TELÉFONO DE SU DOMICILIO, CLASIFICACIÓN DEL SEMESTRE, MEDIA DE CALIFICACIONES, etc. Eventualmente, puede desearse almacenar estos atributos, o campos, en un archivo o base de datos de estudiantes. También eventualmente, se diseñarían entradas para capturar estos atributos, y podrían diseñarse numerosas variedades de salidas para resumir informaciones sobre estos y otros atributos.

Después, identificar las actividades de empresa que tienen lugar entre las entidades; por ejemplo, los ESTUDIANTES "SE INSCRIBEN EN" CURSOS. ESTUDIANTE y CURSOS son entidades. "SE INSCRIBEN EN" es una actividad de empresa que asocia presencias de las entidades ESTUDIANTE y CURSO. ¿Dónde radica la importancia de estas asociaciones?. No sólo queremos almacenar datos sobre ESTUDIANTES y CURSOS, sino también conocer qué ESTUDIANTES hay en cada CURSO. Normalmente, el analista compone una imagen o **modelo de datos** de estas entidades, relaciones y atributos.

En algunos métodos de modelado de datos, el analista utiliza también técnicas formales para asegurar que el modelo de datos sea lo suficientemente flexible para adaptarse a las necesidades actuales y futuras que se basen en los mismos datos. Esta técnica es la normalización, que dice lo siguiente: La normalización es un método de análisis de datos que organiza los atributos de datos de manera que se agrupen entre sí para formar entidades estables, flexibles y adaptables.

Algunos métodos de modelado requieren también que el analista identifique los sucesos de empresa que dan origen a las presencias de las entidades de datos que se van a crear, borrar o modificar. Por ejemplo, los datos referidos a un ESTUDIANTE pueden haber sido "creados" cuando: (1) se ha admitido a un nuevo estudiante; (2) se

ha readmitido a un estudiante antiguo, o (3) se ha trasladado a un estudiante desde otro campus. Éstos son tres sucesos distintos de empresa que pueden darse en una escuela universitaria. Este análisis de sucesos resulta útil para identificar eventualmente modelos de procesos que puedan capturar y almacenar adecuadamente los datos representados en el modelo de datos.

Las ventajas extraídas del modelado de datos son importantes. Si los archivos y la base de datos de los nuevos sistemas se construyen de acuerdo al modelo de datos, poseerán las siguientes propiedades:

- Contendrán datos actualizados y precisos.
- Satisfarán todas las necesidades actuales.
- Satisfarán los requisitos futuros sin necesidad de cambios drásticos en el sistema, ya que los datos ya se han añadido, o pueden añadirse fácilmente, a las entidades apropiadas.

En otras palabras, los procesos y las aplicaciones pueden evolucionar en torno al modelo de datos, con la realización de ligeros cambios en el propio modelo.

Muchos libros representan el modelado de datos sólo como una técnica de *base de datos*. En realidad, funciona igualmente bien en el caso de archivos convencionales y datos distribuidos. La mayoría de los libros aboga por la construcción de base de datos y sistemas corporativos de gran envergadura. En la práctica, la técnica funciona con los mismos buenos resultados en base de datos de aplicaciones de menor tamaño que pueden ser fácilmente integradas en fecha posterior.

Aunque el modelado de datos es bastante útil, antes o después habrán de diseñarse también los modelos de procesos. Pueden utilizarse herramientas y técnicas orientadas a procesos, como el análisis estructurado moderno, para estudiar estas necesidades de procesos y, de este modo, servir de complemento a las técnicas de modelado de datos.

e) Ingeniería de información

Además de combinar el modelado de datos y de procesos, la ingeniería de información pone un especial acento en la importancia de la planificación de sistemas. La **ingeniería de información** es una técnica basada en los datos, pero también sensible a los procesos, aunque la técnica propone un equilibrio entre los métodos orientados a datos y orientados a procesos, se basa claramente en los datos; primero se elaboran los modelos de datos y, después, los de procesos. Esta es la primera técnica estructurada de las que hemos abordado que cubre casi todo el ciclo de vida. En realidad, la única fase no incluida en la ingeniería de información es el soporte de sistemas. Particular interés en la ingeniería de información es su tratamiento de la planificación de sistemas. La ingeniería de información define la **planificación de sistemas** como la mejora de una organización a través de la tecnología de información.

En la ingeniería de información, el centro primordial son los datos almacenados. Los analistas, los programadores y otros profesionales de la informática son responsables de diseñar todos los almacenes de datos y de asegurar que dichos datos sean capturados, almacenados y mantenidos adecuadamente. También diseñan e implantan las principales salidas de información del sistema. Los usuarios finales satisfacen muchas necesidades adicionales de información por medio del estudio y la aplicación de informes fáciles y agradables, y de lenguajes de consulta. Las técnicas de ingeniería de información parecen estar adquiriendo cada vez mayor fuerza entre los desarrolladores de sistemas, ello se debe a su capacidad para equilibrar e integrar verdaderamente las perspectivas de los datos y los procesos.

3.2.2. La técnica del desarrollo conjunto de aplicaciones (DCA)

Mientras que las técnicas anteriormente tratadas ponen especial acento en los procesos y los datos, una técnica cada vez más popular denominada desarrollo conjunto de aplicaciones centra su interés principalmente en las *personas*.

El desarrollo conjunto de aplicaciones (DCA) es una forma de trabajo altamente estructurada que lleva a los usuarios, los directivos y los especialistas en sistemas de información a definir y especificar conjuntamente las necesidades de los usuarios, las opciones técnicas y los diseños externos (entradas, salidas y pantallas).

El DCA intenta requerir mayor participación de los usuarios y los directivos en el ciclo de vida del desarrollo de sistemas. Del aumento de esta participación pueden obtenerse numerosas ventajas:

- El DCA suele mejorar la relación entre usuarios, directivos y profesionales en sistemas de información.
- El DCA suele ampliar la cultura informática de los usuarios y los directivos, así como la cultura en la empresa y sus aplicaciones, y de los especialistas en sistemas de información.
- El DCA sitúa la responsabilidad de la resolución de los conflictos donde corresponda (haciéndola recaer tanto en los usuarios como en los directivos).
- El DCA reduce por lo general el tiempo transcurrido en el desarrollo de los sistemas, gracias a la síntesis de entrevistas múltiples en el trabajo conjunto estructurado.
- El DCA reduce normalmente el costo del desarrollo de sistemas al conseguir definir las necesidades y establecer sus prioridades correctamente desde el principio.
- El DCA consigue, por lo general, sistemas de más valor y mejoras en la satisfacción de los usuarios/directivos. Hace aumentar la confianza de los usuarios y los directivos y su apoyo al proyecto.
- El DCA consigue normalmente sistemas menos costosos de mantener, dado que su primera versión ya satisface las necesidades.

Para tener éxito, el DCA impone algunos requisitos previos. Primero, los directivos deben mostrar la voluntad de liberar a sus subordinados de su trabajo cotidiano (o retribuir las posibles horas extras) para que tomen parte en las sesiones. Segundo, los directivos deben ofrecerse ellos mismos para participar en las sesiones. Los directivos han de promover también un ambiente de cooperación y escucha con sus subordinados durante las sesiones. Finalmente, los encargados de dirigir y registrar las sesiones deben tener una formación suficiente para centrar y reordenar las conversaciones, así como mediar en los conflictos y en las disputas.

3.2.3. Las técnicas de prototipos y desarrollo rápido

Las últimas técnicas a las que nos referiremos, son los prototipos y el desarrollo rápido de aplicaciones (DRA). Empecemos con los prototipos:

- Los **prototipos de viabilidad** se utilizan para probar la viabilidad de una tecnología específica aplicable a un sistema de información. Sobre la base de los resultados del prototipo, puede adoptarse una decisión sobre si se sigue adelante o no con el desarrollo de la solución técnica.
- Los **prototipos de necesidades** (a veces denominados *prototipos de descubrimiento*) se utilizan para “descubrir” las necesidades de los usuarios con respecto a la empresa. Pretenden simular la forma de pensar de los usuarios. Su base es sencilla: los usuarios reconocerán sus necesidades cuando las vean.
- Los **prototipos de diseño** (también llamados *prototipos de comportamiento*) se utilizan para simular el diseño del sistema de información final. Mientras que los prototipos de necesidades se centran sólo en el contenido, los prototipos de diseño lo hacen en la forma y el funcionamiento del sistema deseado.
- Los **prototipos de implantación** (a veces denominados *prototipos de producción*) constituyen una extensión de los prototipos de diseño donde el prototipo evoluciona directamente hacia el sistema de producción. En

principio, los prototipos de implantación omiten normalmente detalles como la edición de datos, las seguridades y los mensajes de ayuda. Estos detalles pueden añadirse posteriormente, si se desea que el prototipo evolucione hacia un sistema de producción. Los prototipos de implantación adquirieron notoriedad con el auge del uso de lenguajes de cuarta generación (L4G) y generadores de aplicaciones. Estos lenguajes de base de datos y generadores suministran potentes herramientas para la rápida generación de prototipos de archivos, base de datos, pantallas, informes, y similares.

Las ventajas que se obtienen del uso de prototipos son numerosas:

- Los usuarios se hacen participantes más activos en el desarrollo del sistema. Suelen mostrarse más interesados en los prototipos de trabajo que en las especificaciones de diseño en papel.
- La definición de necesidades se simplifica por el hecho de que muchos usuarios finales no comprenden o no son capaces de enunciar detalladamente sus necesidades hasta que ven un prototipo.
- La probabilidad de que los usuarios finales aprueben un diseño y luego rechacen su implantación se reducirá notablemente.
- Según se dice, el diseño mediante prototipos reduce el tiempo de desarrollo; sin embargo, algunos expertos cuestionan este ahorro de tiempo.

Pero el uso de prototipos comporta también sus riesgos. ¿Cuáles son sus desventajas?

- Los prototipos suelen pasar a las fases de análisis y diseño con demasiada rapidez. Ello empuja al analista a pasar demasiado rápido a la codificación, sin haber comprendido bien las necesidades y los problemas.

- Los prototipos pueden desalentar la consideración de soluciones técnicas alternativas. El analista suele seguir adelante con el primer prototipo alternativo que suscita una reacción razonablemente favorable en los usuarios.
- Los sistemas implantados desde prototipos son con frecuencia poco flexibles para adaptarse a los cambios en las necesidades, ya que han sido desarrollados "por encima".
- No siempre es fácil cambiar los prototipos. Varios expertos han advertido un crecimiento en las bibliotecas de código L4G con deficiencias de diseño, no estructurado, imposible de leer e inadecuadamente documentado.
- Los prototipos rara vez se pulen. La tecnología utilizada puede impedir, en la práctica, su comprensión por el usuario y, en consecuencia, desalentar su participación.

Los prototipos deben actuar de complemento a las técnicas de especificación apropiadas. La técnica de desarrollo rápido de aplicaciones da respuesta a este problema.

Por último, revisaremos la técnica de desarrollo rápido de aplicaciones:

El **desarrollo rápido de aplicaciones (DRA)** es una combinación de diversas técnicas estructuradas (en especial, la ingeniería de información basada en datos) con técnicas de prototipos y de desarrollo conjunto de aplicaciones cuyo fin es acelerar el desarrollo de sistemas. Las técnicas DRA requieren el uso interactivo de técnicas estructuradas y prototipos para definir las necesidades de usuario y diseñar el sistema final. Por el uso de técnicas estructuradas, el equipo de desarrollo primero construye los modelos previos de datos y de procesos de las necesidades. Los prototipos ayudan entonces al analista y a los usuarios a verificar dichas necesidades, y a pulir formalmente los modelos de procesos y de datos. El ciclo de hacer modelos, luego

prototipos, luego modelos, más tarde prototipos, y así sucesivamente, produce en último término una relación combinada de necesidades de empresa y de diseño técnico.

3.3. ¿Qué metodología y técnicas elegir?

La demanda del Software continua creciendo. Por ello se han desarrollado día a día nuevos métodos más eficientes y menos complejos para la elaboración de nuevos sistemas.

A partir del estudio anterior, sobre las técnicas y metodologías del desarrollo de sistemas, a continuación seleccionaremos de entre las más utilizadas la metodología y las técnicas con las cuales implementaremos nuestro sistema:

a) Ciclo de vida clásico

El ciclo de vida clásico proporciona un modelo que sirve para dos propósitos; el primero, permite representar los procesos de concepción y producción en una forma gráfica y lógica; y el segundo, proporciona un marco de trabajo alrededor del cual las actividades de aseguramiento de calidad pueden ser construidas en una manera disciplinada.

b) Construcción de prototipos

La construcción de prototipos como metodología, puede ser problemática porque el cliente ve funcionando lo que parece ser la primera versión del sistema e ignora que no se han considerado todos los aspectos de calidad, por lo que solicita desesperadamente se apliquen todas las mejoras necesarias. También puede suceder que, para crear el prototipo se utilice un Sistema Operativo o un Lenguaje de Programación inapropiados, simplemente porque ya está disponible y es conocido. A pesar de esto, el modelo funciona adecuadamente si se definen las reglas del juego desde el principio; el cliente y el desarrollador deben estar de acuerdo en que el

prototipo sirve como un mecanismo de definición de requisitos y posteriormente será descartado para construir el sistema real.

c) Modelo en espiral o evolutivo

El modelo en espiral utiliza tanto el enfoque basado en prototipos como los pasos sugeridos por el ciclo de vida de clásico pero incorporándolo dentro de un marco de trabajo interactivo. Pero, por otro lado, puede resultar difícil convencer a los usuarios de que el modelo en espiral es controlable; se requiere una considerable habilidad para la valoración del riesgo y, por último, este modelo es el más reciente y no se ha usado tanto como el ciclo de vida o la creación de prototipos.

Por lo antes escrito, para el desarrollo del presente proyecto, nos apoyaremos fundamentalmente en el ciclo de vida clásico (Fig. 3.1.) que es una metodología ampliamente usada y que tiene un lugar definido e importante dentro del trabajo que se emplea en el desarrollo de sistemas, suministra una plantilla en la que pueden colocarse los métodos para el análisis, diseño, codificación, prueba y mantenimiento.

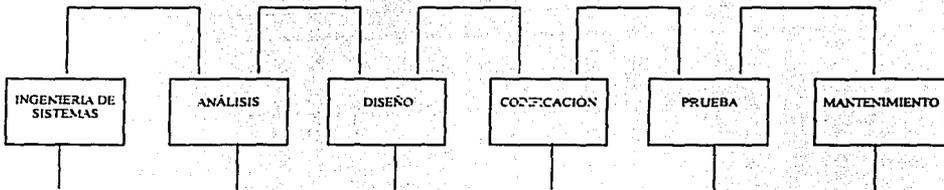


FIG. 3.1.

Además, los pasos de esta metodología son muy similares a los pasos genéricos aplicables a todas las metodologías de desarrollo de sistemas.

En cuanto a las técnicas de desarrollo de sistemas, y basándonos en el estudio anterior, tenemos que las técnicas estructuradas de diseño y análisis moderno de Yourdon (orientadas a los procesos) y la técnica de modelado de datos (orientada a los datos), son las más conocidas y practicadas, por lo tanto; nuestro proyecto se centrará en las antes mencionadas. Estas técnicas nos permitirán la construcción correcta de un sistema, mejorarán la calidad del mismo, reducirán costos, etc. Desde el punto de vista de eficiencia y de acuerdo a la problemática que se ha planteado, nos resultan las más adecuadas para implementar nuestro sistema. En el próximo capítulo se estudiarán éstas.

En la mayoría de los casos, las técnicas pueden y deben combinarse de forma que sea posible utilizar las ventajas de cada una en un mismo proyecto, y cumplir de igual manera con el ciclo de vida completo del desarrollo de sistemas.

IV. TÉCNICAS ESTRUCTURADAS DE ANÁLISIS Y DISEÑO DE YOURDON

Tal como se mencionó en el capítulo anterior, se analizarán a detalle las técnicas de Yourdon. Dentro del análisis y el diseño, el enfoque estructurado permite modelar los componentes del sistema por medio de símbolos gráficos, estableciendo una atención especial en los aspectos lógicos del sistema. Este enfoque formula las especificaciones funcionales para los módulos de software. La técnica estructurada de Yourdon, aporta una teoría que recoge todos los avances de la programación estructurada que durante décadas se ha venido instrumentando. Entre las características del proceso se encuentran las siguientes:

- Modelado de procesos.
- Modelado de datos.
- Etapas del ciclo de vida.

En la técnica de Yourdon el análisis y el diseño de un sistema se dividen en diversas etapas. Si examinamos su estructuración se puede observar una mayor precisión en cuanto a los pasos a seguir a diferencia de otros métodos. A lo largo de sus obras Yourdon describe técnicas para la realización del análisis y el diseño estructurado de sistemas basado principalmente en los siguientes conceptos:

- Diagramas de flujo de datos para la representación de procesos.
- Diccionario de datos como base o soporte de información del sistema.
- Diagramas de transición de estados para la representación estructurada de las funciones a realizar en los procesos.
- Especificaciones de programas basadas en lenguaje estructurado y tablas de decisión.

- Diagramas o mapas de estructura para la representación modular de los procesos y las variables intercambiadas entre ellos.
- Modelo Entidad-Relación para la representación conceptual de datos.

Yourdon tiene el mérito de describirnos un proceso sistematizado y a la vez flexible en cuanto a la aplicación de las técnicas estructuradas. Particularmente, las técnicas utilizadas por Yourdon para el desarrollo de cualquier sistema, las explicamos a continuación:

4.1. Marco para el modelado de procesos

La etapa de *análisis y especificación estructurada*, examina los problemas mediante el modelo entrada-proceso-salida, es decir, mediante el establecimiento del flujo de la información. Esta etapa crea una representación del problema real con el objeto de establecer una solución apoyándose en herramientas como: el diagrama de flujo de datos, el diccionario de datos, diagramas de transición de estados, las mini - especificaciones y las tablas de decisión; herramientas que serán explicadas a continuación.

4.1.1. Diagramas de flujo de datos

El enfoque de flujo de datos permite crear una descripción gráfica del problema, esta descripción se llama *diagrama de flujo de datos (DFD)* y establece los caminos que seguirán los datos a través de los procesos del sistema a desarrollar. Vamos a ver los formalismos de representación de los conceptos que adoptó Yourdon.

Al utilizar el DFD es posible beneficiarse de tres maneras:

1. Contar con rapidez de una implantación técnica del sistema.
2. Comprender la relación existente entre los procesos del sistema.

3. La comunicación a los usuarios del estado actual del sistema, mediante los DFD's.

A continuación se define la simbología utilizada por Yourdon para facilitar la comprensión del DFD de un sistema, ver la Fig. 4.1.



FIG. 4.1.

La *entidad* representa algo que da y recibe datos del sistema; cada entidad se identifica por medio de un nombre apropiado.

El *flujo de datos* representa el movimiento de los datos, con la punta se señala el destino de los datos, un flujo de información bidireccional puede representarse por medio de dos flechas paralelas. Cada flecha también se define con un nombre apropiado.

El *proceso* indica la existencia de una acción de transformación. Los procesos siempre denotan una transformación de los datos, por lo tanto, el flujo de información que sale de un proceso, siempre tendrá un nombre diferente del que tenía al entrar.

El *almacén de datos* representa el almacenamiento de la información. Se permiten anotaciones entre ambas líneas verticales del símbolo correspondiente. Es conveniente señalar que los DFD's pueden particionarse para mostrar mayor detalle.

4.1.1.1. Características del DFD

Para la elaboración eficaz de un DFD se debe atender a las siguientes características sistemáticas:

1. Desarrollar el DFD mediante el enfoque descendente (top-down).
 - Hacer una lista de las entidades, los flujos de datos, los procesos y los almacenes de datos; esto determina los límites del sistema que se desarrolla.
 - Dibujar un DFD básico que ilustre exclusivamente los aspectos generales.
2. Cubrir los detalles.
 - Añadir, por pasos, los detalles a cada proceso.
 - Indicar excepciones, cuando sea necesario.
3. Dibujar de nuevo los diagramas y volver a definir todos los símbolos por medio de nombres significativos.

A continuación los puntos mencionados anteriormente, se verán a detalle:

a) Enfoque descendente

Se puede iniciar un DFD a partir de la narración del sistema organizacional, haciendo uso de cuatro categorías: entidad, flujo de datos, proceso y almacén de datos. Cuando se haya compilado la lista de los datos elementales se puede elaborar un diagrama inicial.

El primer diagrama (nivel cero), incluirá lo básico de las entradas, los procesos y las salidas. Este será el diagrama más general.

b) Cubrir los detalles

Posteriormente, los DFD's deberán incluir los detalles de cada uno de los procesos. Pero en el primer diagrama se deben definir las entradas y las salidas, éstas se mantienen constantes en los diagramas consecutivos. El manejo de las excepciones se ignora en los primeros dos o tres niveles del dibujo del DFD.

El resto del diagrama original se descompone en diagramas que detallan los procesos, agregando en cada nivel inferior nuevos almacenes de datos y nuevos flujos de datos.

c) Dibujar de nuevo los diagramas

Después que los diagramas de flujo se aclaran, lo que sigue es volver a dibujarlos y rotularlos de manera significativa. Es decir, los rótulos no deberán ser muy genéricos. También se debe pensar en que las personas poco familiarizadas con el sistema sean capaces de comprender su contenido.

La efectividad de los DFD's se debe a su consistencia: cada vez que sea posible, hay que unificar términos.

Los diagramas de flujo se emplean para el análisis y diseño de procesos. Utilizando DFD desde el principio se pueden establecer los requisitos de la información y de esta manera, obtener una visión global del flujo de los datos del sistema. En este punto, bastará con dibujar los DFD's a mano y de manera somera.

Al detallar los DFD's, éstos pueden ser útiles para la interacción con los usuarios, puesto que ellos pudieran opinar sobre la realización del proceso y así obtener una mayor precisión en los diagramas. Al final, los DFD's sirven para la documentación del sistema ya que ayudan a concebir la lógica de los flujos de datos de la organización. Para evitar el dibujo manual de los DFD's se pueden emplear herramientas integradas (*workbench*) o instrumentos CASE (*Computer Assisted Software Engineering*).

4.1.2. Diccionario de Datos (DD)

El diccionario de datos es una referencia útil para localizar nombres y atributos de los datos utilizados en el sistema, este deberá actualizarse cada vez que se hagan

cambios en el sistema. Yourdon lo utiliza también para la descripción de los diferentes datos a través de los siguientes símbolos:

Símbolo	Representación
=	Está compuesto de
+	Para unir campos elementales
()	Campos opcionales
[]	Selección de una entre varias opciones
{ }	Iteraciones
**	Comentarios
@	Identificación o clave de un almacenamiento
	Separación de un almacenamiento

Mediante esta notación se puede representar la sintaxis del sistema.

Los registros del diccionario de datos deben contener información referente a:

1. El nombre y el alias del dato.
2. Las descripciones del dato.
3. Los flujos de datos que se relacionan con el término.
4. La longitud disponible en caracteres.
5. Cualquier otra información pertinente.

1. *Nombres y Alias.* Es necesario incluir la manera más común de denominar el dato en los programas del sistema, además de su alias. El registro de estos datos, facilita la comunicación y la referencia cruzada entre esos programas.

2. *Descripción.* La descripción incluirá un texto breve, pero informativo del dato elemental.

3. *Flujos de datos.* Establecer el origen y el destino de la transferencia de datos.

4. *Longitud del dato.* En el diccionario de datos, también se establece la longitud permitida para el acceso de un dato. La longitud siempre se da en función del número de caracteres impresos.

5. *Información adicional.* Deberá agregarse al diccionario de datos cualquier información útil extra.

4.1.2.1. ¿Cómo formar un diccionario de datos?

Se pueden observar los siguientes pasos en la elaboración de un diccionario de datos:

1. **Flujos de datos y almacenes de datos.** En este paso se catalogan los flujos que entren y salgan de los procesos ya señalados por los DFD's. Además, se catalogan los almacenes de datos empleados en cada proceso.
2. **Describir la estructura de datos.** Aquí, se describen los grupos relacionados de datos elementales que existan dentro del sistema. La estructura de datos es alimentada por los flujos y los almacenes de datos.
3. **Desglosar la estructura de los datos en datos elementales.** En esta parte, hay que trabajar con los significados más simples del sistema, es decir, con los datos elementales.

4.1.3. Diagramas de transición de estados

Mediante estos diagramas se representan las diferentes funciones a realizar, indicando la secuencia y las condiciones que manejan su ejecución.

Cada proceso se ve representado por una serie de acciones enmarcadas en rectángulos. Se pasa de una acción a la siguiente a través de una conexión en la que se representa el par condición–acción, es decir, la condición que se debe cumplir para seguir la secuencia por ese lado del diagrama y la acción a realizar en ese caso. (Fig. 4.2.)

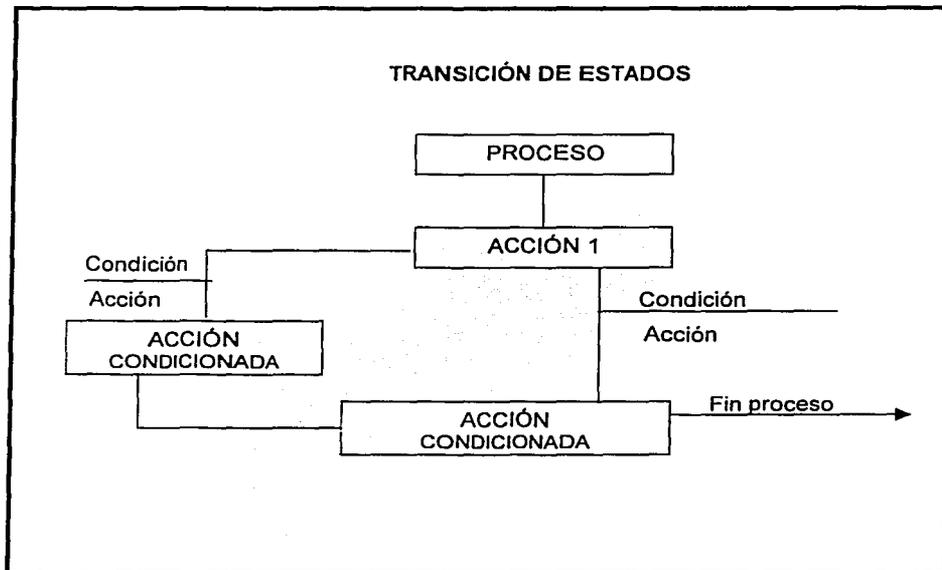


FIG. 4.2

4.1.4. Especificación estructurada y tablas de decisión

En particular la programación estructurada, sugiere que el esquema lógico de cualquier programa debería ser escrito con un conjunto limitado de estructuras de control. También el programador estructurado diseña la lógica de sus programas

mediante herramientas de modelado como las tablas de decisión, a continuación, se dará una explicación más a detalle de esto.

4.1.4.1. Especificación estructurada

Las *especificaciones* son sentencias que existirán por cada proceso especificado en los DFD's. Se componen de verbos imperativos y de palabras reservadas para expresar la lógica del funcionamiento de cada proceso.

La sintaxis empleada se limita a oraciones declarativas, estructuras de decisión y estructuras de iteración:

- PROCEDIMIENTO (PROCEDURE).
- SI-ENTONCES-DE OTRA MANERA (IF-THEN-ELSE).
- HACER...MIENTRAS (DO...WHILE).
- HACER...HASTA (DO...UNTIL).

Una herramienta que puede emplearse para hacer las especificaciones es la constituida por las tablas de decisión.

4.1.4.2. Tablas de decisión

Las *tablas de decisión* se emplean para traducir acciones y condiciones a una forma tabular. La organización de una tabla de decisión se muestra en la Fig. 4.3.

La tabla está dividida en cuatro cuadrantes; el superior izquierdo, contiene la lista de todas las condiciones; el inferior izquierdo, la lista de todas las acciones que se pueden producir basándose en combinaciones de las condiciones; en los cuadrantes de la derecha se indican las combinaciones de las condiciones y las correspondientes acciones que se han de producir para cada combinación específica. De esta manera, cada columna se interpreta como una regla de procesamiento.

Números de reglas	1	2	3	4	5	6	7	8	9	10
Filas de condiciones										
Filas de acción										

R E G L A S

FIG. 4.3.

Para desarrollar una tabla de decisión se aplican los siguientes pasos:

1. Listar las acciones que puedan asociarse con un proceso.
2. Listar las condiciones o decisiones durante la ejecución del procedimiento.
3. Asociar conjuntos específicos de condiciones con acciones específicas, eliminando las combinaciones de condiciones imposibles.
4. Definir reglas indicando qué acciones ocurren para un conjunto de condiciones.

4.1.5. Diagramas o mapas de estructura

En ésta etapa, el *diseño estructurado* se apoya en las herramientas del análisis y especificación estructurada y en el mapa de estructura para interconectar los datos con los procesos que en conjunto constituyen el sistema que en gran medida solucionan la problemática de la situación en consideración. El diseño estructurado, además, tiene en cuenta conceptos clave como son: la modularidad, la cohesión y el acoplamiento; explicados más adelante

4.1.5.1. Diagrama de estructura

Mediante ellos se representa gráficamente la estructura de un proceso descomponiéndolo en módulos que se intercambian y se encadenan hasta completar

las funciones a realizar. Podemos confeccionar, partiendo de este tipo de unidad, estructuras complejas que representen la secuencia de ejecución de funciones. Estas estructuras son similares al de técnicas estructuradas de análisis y diseño de programación, aunque en este tipo de diagramas se añaden los parámetros o variables que se intercambian los diferentes módulos.

Cada unidad de un diagrama de estructura se representaría de la siguiente manera, según la notación Yourdon. (Fig. 4.4.)

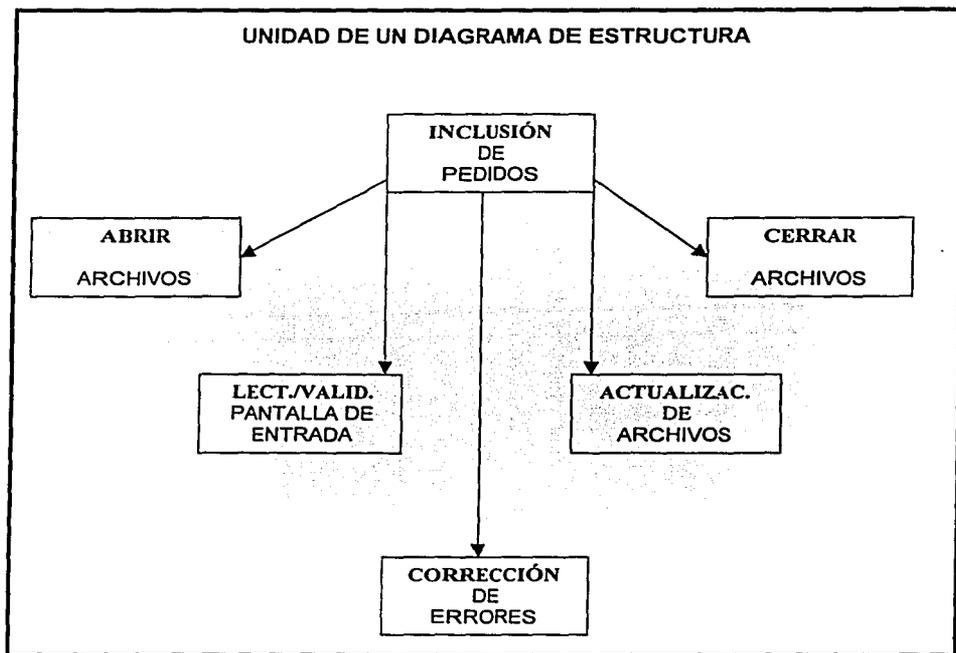


FIG. 4.4.

4.1.5.2. Propiedades

El diseño estructurado de Yourdon pretende dividir un programa en un conjunto jerárquico de módulos en sentido descendente, dotado de las siguientes propiedades:

a) Modularidad

El concepto de *modularidad* implica que el software se divide en componentes con nombres y ubicaciones determinados, denominados *módulos* y que se integran para satisfacer los requisitos del problema. Es decir, los módulos son los componentes procedimentales (procesos) del sistema.

b) Cohesión

Los módulos deben tener una fuerte *cohesión*; es decir, la cohesión se refiere a la fuerza de unión entre las funciones que realiza un módulo y se mide considerando una escala que va desde la más débil hasta la más fuerte (la más deseada) en el siguiente orden:

1. *Cohesión coincidental*. Ocurre cuando los elementos dentro de un módulo no tienen relación aparente entre cada uno de ellos; esto puede suceder cuando se segmenta arbitrariamente un programa en varios módulos.
2. *Cohesión lógica*. Este tipo de cohesión implica algunas relaciones entre los elementos de un módulo, como por ejemplo, en uno que desempeñe todas las funciones de entrada y salida, o en otro dedicado a la edición general de los datos. Un módulo unido lógicamente por lo común combina varias funciones relacionadas en una forma compleja e interrelacional.
3. *Cohesión temporal*. En los módulos que presentan cohesión temporal, todos los elementos son ejecutados en un momento dado sin requerir de ningún parámetro o lógica alguna para determinar qué elemento debe ejecutarse. Un ejemplo de esta cohesión se observa en un módulo dedicado a la inicialización de un sistema.

4. *Cohesión en la comunicación.* Los elementos de un módulo que contiene cohesión en la comunicación se refieren al mismo conjunto de datos de entrada y de salida; por ejemplo la instrucción "imprima el archivo de salida" presenta una cohesión en la comunicación.
5. *Cohesión secuencial.* Esta clase de cohesión se presenta cuando la salida de un elemento es la entrada para el siguiente; por ejemplo, "lea la siguiente transacción y actualice el archivo maestro" se encuentra unida secuencialmente.
6. *Cohesión funcional.* Este tipo de cohesión representa un tipo fuerte, y por lo tanto deseable, de la relación entre los elementos de un módulo, debido a que todos ellos se encuentran encausados al desempeño de una sola función. Un ejemplo de un módulo de este tipo, es el que ejecuta la instrucción "escriba el registro en el archivo de salida".
7. *Cohesión informacional.* Ocurre cuando un módulo contiene una estructura de datos compleja, así como varias rutinas que manejan dicha estructura; cada rutina del módulo presenta unión funcional; esta cohesión es la realización total de la abstracción de los datos. La cohesión informacional solamente requiere que el segmento con cohesión funcional sea ejecutado al ser llamado el módulo.

No es indispensable tener siempre el nivel más alto de cohesión, sin embargo, lo importante es intentar conseguir una cohesión alta y saber reconocer la cohesión baja, de forma que sea posible modificar el diseño del software para que disponga de una mayor independencia funcional. de tal forma que los módulos serán reutilizables en futuros programas.

c) Acoplamiento

Los módulos deben estar débilmente *acoplados*; es decir, han de tener una dependencia mínima unos de otros. Ello reduce al mínimo el efecto que futuros cambios en un módulo puedan producir en otros módulos.

El *acoplamiento* es una medida de la interconexión entre los módulos de un sistema. El acoplamiento depende de la complejidad de las interfaces entre los módulos, del punto en el que se hace una entrada o referencia a un módulo y de los datos que pasan a través de esas interfaces. En el diseño del software se busca el más bajo acoplamiento posible. La conectividad sencilla entre módulos da como resultado un software que es más fácil de comprender y menos propenso a errores que aparecen en una posición y que se propagan a lo largo del sistema. El acoplamiento entre módulos puede ser considerado dentro de una escala del más fuerte al más débil (el más deseable) de la siguiente manera:

1. Acoplamiento del contenido. Ocurre cuando un módulo modifica los valores locales o las instrucciones de algún otro módulo; este acoplamiento es posible en los programas en lenguaje ensamblador.
2. Acoplamiento de zonas compartidas. En este acoplamiento, los módulos son conjuntados por medio de zonas globales para las estructuras de datos, por ejemplo, cuando un programa se referencia a un bloque común de datos.
3. Acoplamiento de control. Este acoplamiento incluye el paso de banderas de control, ya sea como parámetros o en forma global entre los módulos, de tal forma que un módulo controla la secuencia de proceso de otro.
4. Acoplamiento por zonas de datos. Este tipo de acoplamiento es similar al de zonas compartidas, excepto que los elementos globales son compartidos en forma selectiva entre los diversos módulos que requieren los datos.
5. Acoplamiento de datos. Incluye el uso de listas de parámetros para pasar los elementos entre los módulos.

4.2. Marco para el modelado de datos

Ya que analizamos el marco del modelado de procesos, ahora se analizará el modelado de datos. En la actualidad las técnicas orientadas a procesos han recibido el

complemento de las técnicas orientadas a datos. La técnica de modelado de datos representa las necesidades de usuario para un sistema en función de los datos del sistema, independientemente de cómo se procesen o utilicen dichos datos para producir información. Las técnicas de modelado de datos representan al bloque elemental Datos.

4.2.1. Base de Datos

Es posible considerar dos tipos de almacenamiento de datos en un sistema de información basado en computadora: archivos individuales o elaborar una base de datos. Un sistema que hace uso de los archivos convencionales implica que los datos almacenados lleguen a ser redundantes, la actualización de los archivos lleva más tiempo y existirán problemas de integración de datos, ya que los cambios en un archivo, requerirán la modificación de ciertos datos en otros archivos.

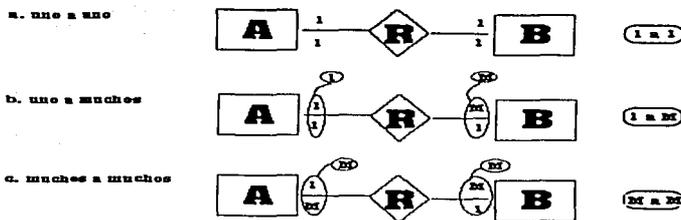
Una base de datos es una colección de datos clasificados y estructurados. Los datos de una base de datos están clasificados en campos y registros. Por ejemplo piense en los libros de una biblioteca. Los Datos propios de un determinado libro (título, autor, editorial, etc.) son campos que agrupamos en un registro.

Por otro lado, una *base de datos* es una fuente central de datos significativos que pueden ser compartidos por varios usuarios y por varias aplicaciones (si ese fuera el caso) y asegura que los datos requeridos estén siempre disponibles. además permite el mantenimiento preciso y consistente de los mismos. Es oportuno en este momento, establecer los conceptos básicos necesarios en el manejo de una base de datos.

Entidades. Una *entidad* es cualquier objeto o evento, que sea fuente de datos.

Relaciones. Las *relaciones* son asociaciones de datos entre entidades. Existen tres tipos de relaciones: *relación de uno a uno* (1:1), para una entidad existe otra sola entidad; *relación de uno a muchos* (1:M), para una entidad existen muchas otras

entidades, o para muchas entidades existe otra sola entidad y finalmente, la *relación de muchos a muchos* (M:M), para muchas entidades existen otras varias entidades, como se ve en la Fig. 4.5.



Tipos de relaciones entre entidades

FIG. 4.5.

Atributos. Un atributo es una característica de una entidad, y pueden describirse varios atributos para una entidad. Ejemplo clave, nombre, domicilio, etc.

Registros. Un registro es una colección de datos elementales que tienen algo en común con la entidad correspondiente. Los datos son las unidades más pequeñas en una base de datos. La palabra dato, puede aquí considerarse como sinónimo de atributo, los tipos de los datos pueden ser alfabéticos, numéricos o alfanuméricos (todos ellos de longitud variable).

Ocasionalmente, un dato puede ser llamado como un campo, sin embargo, un campo representa algo físico y no lógico, por lo tanto esta nominación es incorrecta; además, un grupo de datos puede agruparse en un campo, por ejemplo, el campo fecha contiene los datos día, mes y año, véase la Fig. 4.6. Un registro puede esquematizarse de la siguiente manera:

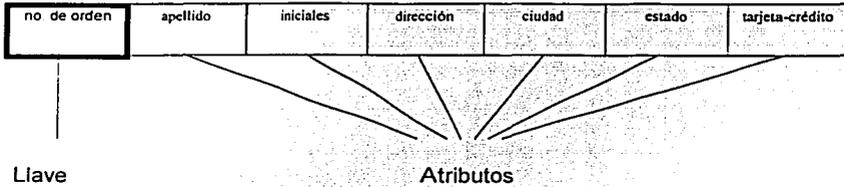


FIG. 4.6.

Tabla. Una *tabla* es una colección de datos presentada en forma de matriz bidireccional, donde las filas son los registros y las columnas los campos.

Llaves. Una *llave* es un dato en un registro que se emplea como identificador para éste. Pueden mencionarse dos tipos de llaves:

Llave primaria: Columna o grupo de columnas que identifican de manera exclusiva a un registro. Símbolo PK. Debe cumplir con:

- NN: No aceptar valores nulos.
- ND: No pueden existir dos renglones con el mismo valor.
- NC: No puede cambiarse el valor.
- PK asignada por el usuario(PK UA): De esta forma el usuario proporciona el valor para la llave primaria del registro insertado. Ejemplo: RFC del trabajador.
- PK asignada por el sistema(PK SA): De esta forma el sistema proporciona automáticamente el valor para la llave primaria cuando el usuario inserta un registro. Ejemplo: Número de factura.

Todas las tablas deben contener solo una llave primaria, a cual puede estar compuesta de uno más campos.

Llave foránea: Un atributo que no es llave primaria en una tabla pero si lo es en otra.

Nulo. Un nulo es ausencia de dato.

Consulta. Una consulta es el mecanismo que recoge una pregunta, simple o compuesta, que se formula a la base de datos con el fin de obtener los datos de una o más tablas que cumplan las condiciones enunciadas.

4.2.2. Tipos de consulta

El almacenamiento empleando base de datos facilita la emisión de reportes adecuados a los usuarios mediante el empleo de consultas, las cuales son preguntas acerca de la información contenida en dicha base. Pueden mencionarse algunos tipos de consultas básicas, involucrando a las entidades, atributos y valores, en cada uno de los casos, se conocen dos de esos elementos y se intenta obtener el tercero.

- Tipo 1: Se cuenta con la entidad E y uno de sus atributos A, se desea buscar el valor correspondiente: $V = (E, A)$.
- Tipo 2: $E = (V, A)$.
- Tipo 3: $A = (V, E)$.
- Tipo 4: $V = (E, \text{todos los } A)$.
- Tipo 5: Todas las $E = (V, \text{todos los } A)$.
- Tipo 6: Todos los $A = (V, \text{todas las } E)$.

Empleando operadores aritméticos, comparativos y booleanos, las consultas pueden ser más complejas y precisas respecto a sus tipos básicos.

4.2.3. Modelo Entidad-Relación (DER)

Para realizar el modelo conceptual de datos, Yourdon propone un esquema de entidades relaciones del sistema, pasando después a normalizar esta estructura.

Veamos el ejemplo de la relación entre PROFESORES y ALUMNOS: un profesor pueden tener varios alumnos y un alumno varios profesores.

Los diagramas E-R son una técnica para representar gráficamente la estructura lógica de una base de datos. Como tal, ofrecen una forma sencilla de comunicar los rasgos prominentes del diseño de cualquier base de datos. Básicamente en un diagrama del modelo E-R se deben representar los conceptos siguientes:

Entidad. La componen una serie de datos que agrupados, tienen un cierto significado para la empresa. Se presenta por medio de un rectángulo en cuyo interior figura el nombre de la entidad, (Fig. 4.7.)

REPRESENTACIÓN DE UNA ENTIDAD

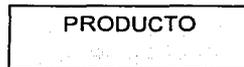


FIG. 4.7.

Cada entidad está compuesta por una serie de atributos que se representarán por una serie de círculos enlazados con la entidad.

Relaciones. Constituyen el enlace entre las diferentes entidades del sistema. Se representa mediante líneas que unen las entidades o a través de un rombo en cuyo interior se coloca el nombre de la relación.

Así, entre un SERVICIO, por ejemplo, y un RESPONSABLE existirá la relación "Es servido por " (Fig. 4.8.)

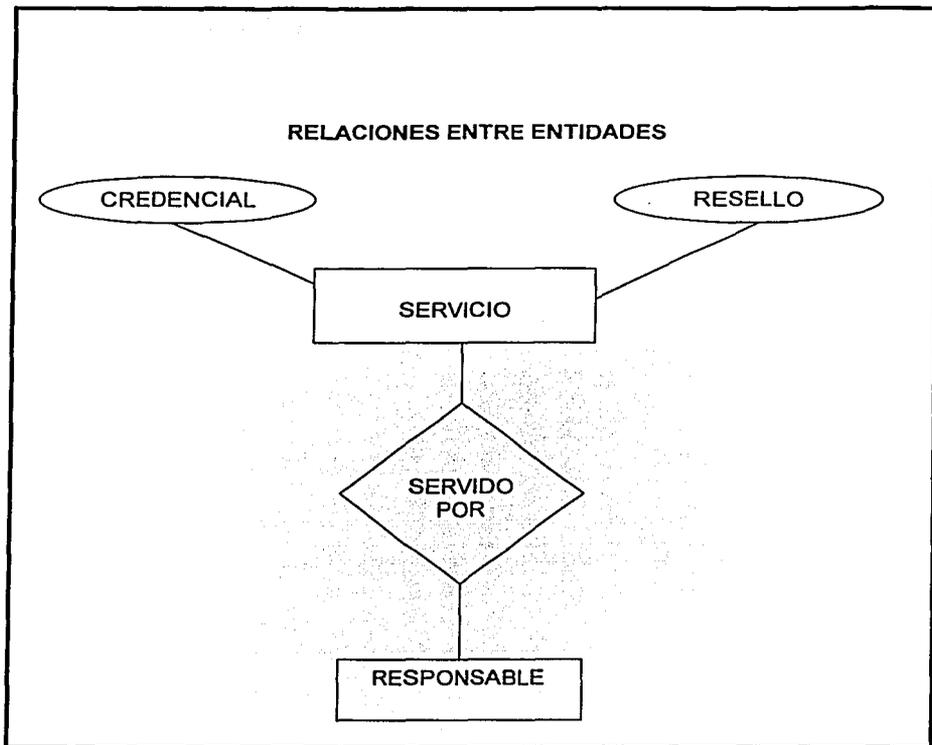


FIG. 4.8.

4.2.4. Normalización

La normalización es la simplificación de las relaciones existentes entre los datos de las aplicaciones y los almacenamientos usados por los usuarios; de tal manera, que se establezcan estructuras de datos de menor tamaño. Se emplean tres pasos para lograr esta simplificación de los datos, llamados formas normales (FN).

La *primera forma normal (FN1)* elimina los grupos de datos repetidos e identifica la llave primaria, en este punto puede ser necesario desglosar alguna relación en dos o más relaciones.

La *segunda forma normal (FN2)* se aplica para asegurar que todos los atributos no llave sean dependientes de la llave primaria. Si existen dependencias se colocarán en otra relación.

La *tercera forma normal (FN3)* elimina cualquier dependencia transitoria (la dependencia en la cual sus atributos no llave son dependientes de otros atributos no llave).

4.3. Etapas del ciclo de vida

Ed Yourdon define las siguientes etapas y niveles en el ciclo de vida de los sistemas informáticos: (Vea Fig. 4.9.)

CICLO DE VIDA DE YOURDON

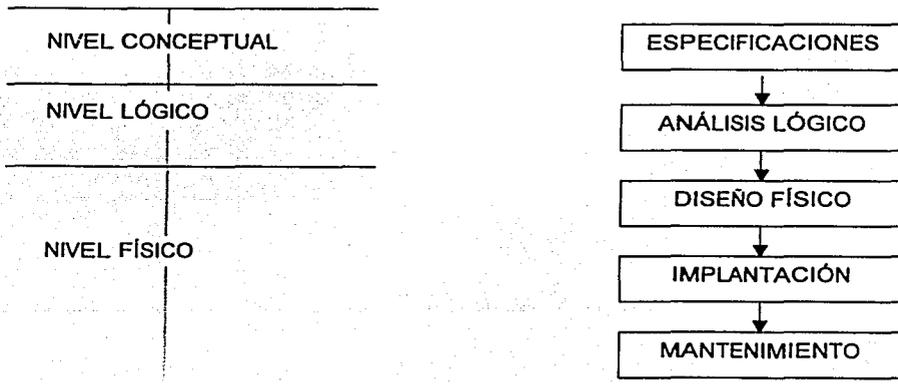


FIG. 4.9.

Finalmente, de las técnicas descritas a lo largo de estas etapas, se distinguen las siguientes actividades a realizar que ya se han planteado como bases de la programación estructurada, pero que son retomadas por el método de Yourdon:

4.3.1. Estudio de viabilidad

Debe hacer un estudio de la situación actual, representando la misma a través de DFD de primer nivel o diagramas de contexto en los que de forma simple, se indiquen los procesos más relevantes. La *descripción de la situación actual* del problema en estudio es indispensable para identificar las necesidades a considerar: el analista se entrevista con el cliente para definir los objetivos del sistema, la información que se desea obtener, la información que se va a suministrar, las funciones y el rendimiento requeridos. Es útil establecer la diferencia entre lo que "necesita" el cliente y lo que "quiere" el cliente.

En este estudio de la situación actual se identificarán las deficiencias del mismo como puntos a resolver por el nuevo sistema. Esta actividad debe ocupar entre un 5 y un 10% del total del proyecto.

4.3.2. Análisis del Sistema

En esta fase se debe representar mediante las técnicas ya vistas de diagramas de flujo, modelo Entidad-Relación, diagramas de transición de estado, etc., el sistema a desarrollar. En esta parte también se debe tener en cuenta los requerimientos de los usuarios relativos a cambios o funcionalidad del sistema y esta labor es facilitada por las herramientas CASE de diagramación y por técnicas de prototipos del sistema.

4.3.3. Diseño del Sistema

- En esta tercera fase, se pasa del nivel conceptual descrito anteriormente a un nivel de representación lógica de los datos mediante un diseño dependiente del modelo de base elegida, y una estructuración de los procesos utilizando diagramas de

estructura de los mismos y generando las especificaciones de programa correspondientes.

4.3.4. Implementación ó producción

Comprende la generación de código y el ensamblaje e integración de todos los módulos.

4.3.5. Pruebas del sistema

Las pruebas se tienen que realizar con la totalidad del sistema hasta llegar a la aceptación del mismo por parte del usuario. En esta fase se harán pruebas de integración y de funcionamiento conjunto de programas y cadenas.

La prueba del proyecto es un elemento importantísimo para garantizar la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación. La prueba debe considerar los siguientes objetivos:

- La prueba es un proceso de ejecución de un programa con la intención de descubrir errores.
- La prueba es adecuada si se cuenta con una alta probabilidad de mostrar un error no descubierto antes.

a) Pruebas de unidad

La *prueba de unidad*, centra el proceso de verificación en la menor unidad del diseño, es decir, en el módulo. Las pruebas que se realizan son de:

- *Interfaz*. Para asegurar que la información fluye de forma adecuada hacia y desde la unidad del programa que está siendo probada.

- *Estructuras de datos.* Para asegurar que los datos que se mantienen temporalmente conservan su integridad durante todos los pasos de ejecución del algoritmo.
- *Condiciones llímite.* Para asegurar que el módulo funciona correctamente en los límites establecidos como restricciones de procesamiento.
- *Caminos independientes.* Para asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez.

También deben diseñarse pruebas para detectar errores en las estructuras de datos locales de cada módulo, en las siguientes categorías:

1. Tipificación impropia o inconsistente.
2. Iniciación o valores implícitos erróneos.
3. Nombres de variables incorrectos.
4. Tipos de datos inconsistentes.
5. Desbordamiento de direccionamiento.

De la misma manera, deben establecerse caminos para detectar errores debidos a cálculos o comparaciones incorrectos o flujos de control inadecuados:

1. Comparaciones entre tipos de datos distintos.
2. Operadores lógicos o de precedencia incorrectos.
3. Variables o comparadores incorrectos.
4. Terminación inapropiada o inexistente de bucles.

b) Pruebas de integración

La *prueba de integración*, se establece para construir la estructura del programa mientras que, simultáneamente se realizan pruebas para detectar errores asociados con la interacción. Para "poner juntos" a los módulos se debe seguir la técnica

denominada integración incremental, que consiste en probar pequeños segmentos para que los errores sean más fáciles de aislar y de corregir.

c) Pruebas de volumen

La *prueba de volumen*, intenta detectar errores en el software debidos a condiciones anormales, demandando recursos en cantidad, frecuencia o volúmenes anormales. Algunos ejemplos de pruebas para estos casos son:

1. Generar diez interrupciones por segundo, cuando las normales son una o dos.
2. Incrementar las frecuencias de datos de entrada en magnitud.
3. Casos en los que se requieran el máximo de memoria o de otros recursos.
4. Casos para probar la memoria virtual.
5. Casos de excesiva búsqueda de datos en disco.

d) Pruebas de validación

La validación del software se consigue demostrándose la conformidad con los requisitos, es decir, que el software funciona de acuerdo con las expectativas razonables del cliente.

Cuando se ha realizado el software para un cliente, es necesario realizar *la prueba de validación*, para que el cliente establezca conformidad del sistema con todos los requisitos.

Para realizar esta prueba, es necesaria la instalación del producto terminado, posteriormente, se lleva a cabo la capacitación, tanto al personal técnico como a los propios usuarios. Se entiende por producto terminado, al sistema listo para ser empleado en el equipo para el cual fue diseñado, junto con el manual de usuario. El sistema deberá estar listo para comenzar su operación de manera inmediata.

4.3.6. Control de calidad

El objeto de esta actividad es garantizar los controles de calidad del software que puedan estar definidos para la empresa. Esta fase complementa la anterior de forma que el producto final sea de un buen nivel de calidad y cumpla los estándares fijados.

El factor más importante de la calidad de un sistema es su utilidad, en otras palabras, que el sistema satisfaga las necesidades del usuario; la planeación cuidadosa, el análisis y la participación del cliente son obligatorios para el desarrollo de productos de programación útiles.

El concepto de calidad implica que: Los requisitos del software son la base de las medidas de la calidad, hay ciertos seguimientos especificados que definen los criterios de desarrollo y existe un conjunto de requisitos implícitos que frecuentemente no se mencionan, como el deseo de un buen mantenimiento. Se debe resaltar el hecho de que la garantía de la calidad es un proceso de protección que se aplica a lo largo de todo el ciclo de vida del sistema y que esta garantía se basa en *las revisiones técnicas formales* cuyo objetivo es encontrar errores durante un proceso para evitar que se propaguen al paso siguiente.

Las revisiones técnicas formales garantizan la calidad del software mediante:

- Revisar el producto, no al productor.
- Fijar una agenda y mantenerla.
- Limitar el debate y las impugnaciones.
- Enunciar áreas de problemas, pero no intentar resolver cualquier problema que se ponga de manifiesto.
- Tomar notas escritas.
- Limitar el número de participantes e insistir en la preparación anticipada.
- Desarrollar una lista de comprobación para cada módulo que haya de ser revisado.

- Disponer recursos para las revisiones técnicas formales.
- Llevar a cabo un buen entrenamiento de todos los revisores.
- Repasar las revisiones anteriores.

4.3.7. Mantenimiento

El término *mantenimiento* se ha aplicado al proceso de modificar un programa después de que ha sido desarrollado y está en uso. Las modificaciones implican simples cambios para corregir errores de codificación, cambios más extensos para corregir errores de diseño o cambios drásticos para corregir errores de especificación o para acomodarse a nuevos requerimientos.

Es imposible producir sistemas de cualquier tamaño que no necesiten mantenimiento. A lo largo de la vida útil de un sistema, sus requerimientos originales serán modificados para reflejar las necesidades cambiantes. Ya que el mantenimiento es inevitable, los sistemas deben ser diseñados e implementados de tal manera que los problemas de mantenimiento sean minimizados.

El mantenimiento del software cae en tres categorías:

- Mantenimiento perfectivo.
- Mantenimiento adaptativo.
- Mantenimiento correctivo.

El *mantenimiento perfectivo* significa cambios que mejoran el sistema de alguna manera sin cambiar su funcionalidad.

El *mantenimiento adaptativo* es el mantenimiento que se requiere debido a los cambios en el ambiente del programa.

El *mantenimiento correctivo* modifica de alguna manera el funcionamiento de ciertas partes del programa.

El proceso de mantenimiento se inicia por un conjunto de solicitudes de cambio por parte de los usuarios del sistema o por parte de los administradores del mismo. Esta petición, generalmente involucra elementos del mantenimiento adaptativo, correctivo y perfectivo. Los cambios se implementan y son validados.

Finalmente, se libera una nueva versión del sistema. El proceso de esta manera se vuelve iterativo con un nuevo conjunto de cambios propuestos para la nueva liberación.

El mantenimiento es una iteración del proceso de desarrollo y se deben aplicar los estándares de comparación. Los nuevos requerimientos deben ser validados, los componentes del sistema deben ser rediseñados e implementados y parte, o la totalidad del sistema, debe ser probada.

V. ANALISIS Y DISEÑO ESTRUCTURADO YOURDON

5.1. Análisis del Sistema

Basados en los antecedentes del capítulo I, donde se trato el funcionamiento de la DGADyR y de acuerdo con la problemática tratada en el capítulo II, se realizó una recopilación de información de datos para hacer un análisis de la información que debería de manejar el Sistema a desarrollar.

La Información que el Sistema deberá de manejar será la siguiente: Contemplar las distintas Subdirecciones, las distintas Disciplinas Deportivas, las Actividades realizadas en el Área de Recreación y Fútbol Americano (Logística), los distintos tipos de Datos que pueda tener una persona (Deportista), así como las Instalaciones fuera de la ciudad universitaria tales como CCH's, ENEP's, etc.

Además también se debe de contar con un módulo de usuarios, realizar reportes en pantalla, a disco o bien a impresora, así como contar con un módulo de información común (Catálogos).

Basándose en lo anterior se realizó un diseño general para el manejo y captura de la información, se contemplaron también los distintos tipos de Usuarios que pueden acceder al Sistema, los Privilegios o Permisos que tendrán, así como la forma de obtener reportes de una manera sencilla.

Las Subdirecciones que más información manejan son tres: Deportes que contiene 35 disciplinas, Logística que es el Fútbol Americano y Recreación que comprende actividades de recreación orientadas básicamente a niños para motivarlos e invitarlos a practicar alguna actividad física.

A continuación se describirán y se aplicaran tanto la metodología como las técnicas seleccionadas para la desarrollo e implementación del Sistema, desde un nivel lógico-conceptual.

5.2. El DFD del SIDUNAM

El Diagrama de Flujo de Datos, trata de mostrar de una manera clara los procesos que se realizan dentro del Sistema y el almacenamiento de los datos recopilados por el Sistema (Fig. 5.1).

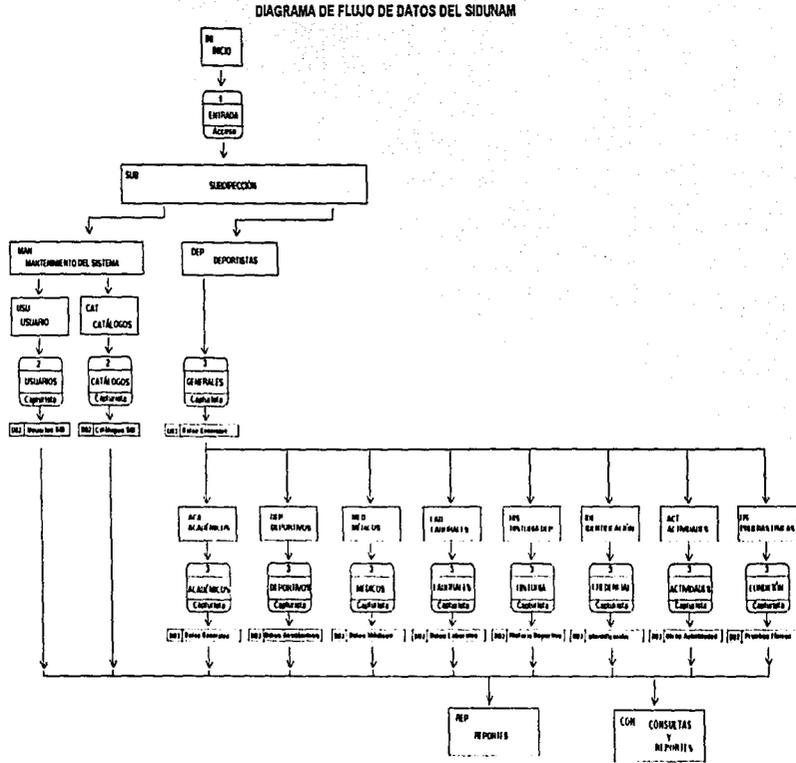


Fig. 5.1.

El Sistema está dividido en tres bloques:

- Acceso al Sistema
- Mantenimiento del Sistema
- Información del Deportista.

Acceso al Sistema. El primer bloque, nos muestra la parte donde se verifica el tipo de Usuario que entrará al Sistema y de acuerdo con los permisos que tenga podrá acceder para consultar o actualizar la información, así como también podrá o no dar de alta usuarios, entrar a una Subdirección o bien actualizar los Catálogos del Sistema, véase la siguiente figura (Fig. 5.2).

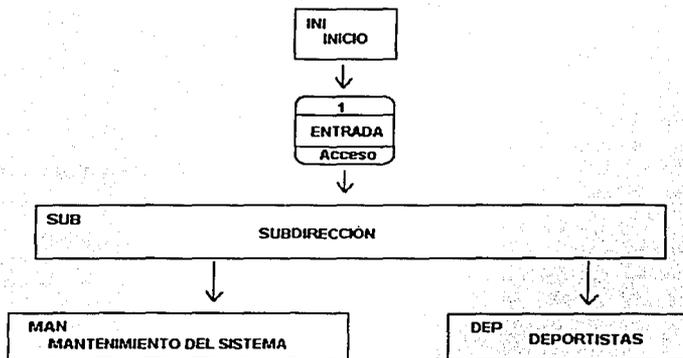


Fig. 5.2.

Mantenimiento del Sistema. A este segundo bloque, generalmente podrán entrar los Administradores de cada una de las Dependencias de la DGADyR, es decir, que habrá un responsable de la información por cada Dependencia, para tener un mejor control del Sistema y de la Información con la cual se alimentarán los Catálogos. Por otra parte el responsable de la Dependencia solicitará al Administrador del Sistema las altas de Usuarios y el tipo de permisos para los mismos, véase la figura siguiente (Fig. 5.3).

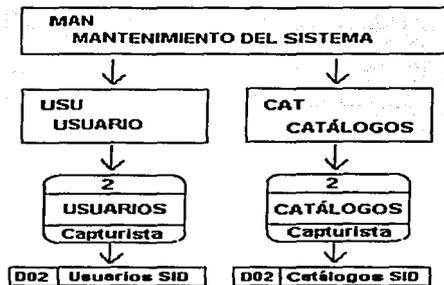


Fig. 5.3.

Información del Deportista. Aquí se capturarán inicialmente los datos generales de un Deportista para posteriormente poder capturar los demás datos, ya sean académicos, deportivos, laborales, médicos, etc. Para así poder consultar, actualizar o bien obtener reportes de dicha información, véase la figura siguiente (Fig. 5.4).

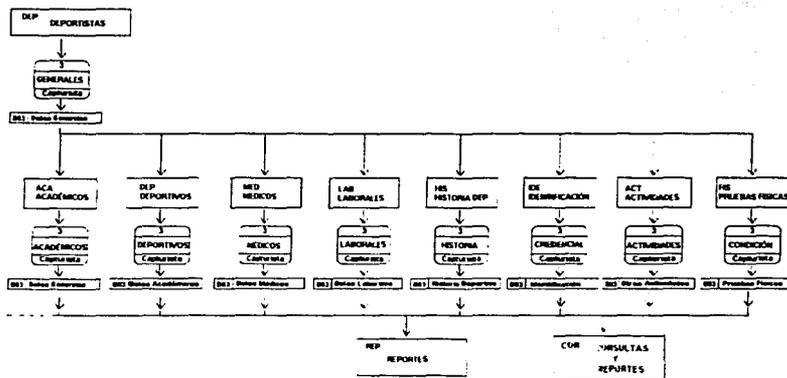


Fig. 5.4.

5.3. Diagrama Entidad-Relación (DER)

Retomando los fundamentos teóricos descritos en el capítulo III y IV, y aplicándolos a nuestro Sistema tendremos como resultado final el Diagrama Entidad-Relación (DER) para el SIDUNAM, mostrado en la (Fig. 5.5).

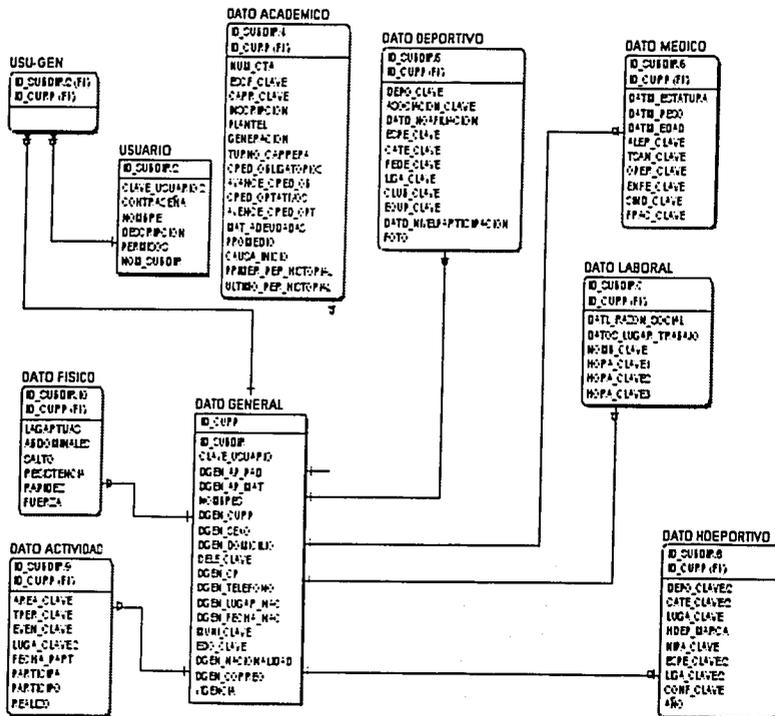


Fig. 5.5.

5.4. Diccionario de datos del SIDUNAM

El diccionario de datos del SIDUNAM recopila los nombres de las tablas principales y atributos de las mismas, así como las características de cada uno de ellos. Además de establecer la relación con cada uno de los catálogos utilizados por el Sistema.

Tabla de Control

La Tabla de Control llevará la llave principal, y se generará automáticamente para optimizar el Sistema, además manejará una llave externa que es la que indicará cuál Subdirección se mostrará. Para el manejo de los registros eliminados se maneja un campo llamado *vigencia* que tendrá dos valores 1/0, para que en determinado momento si se requiere recuperar un registro eliminado en forma accidental se le avise al Administrador General del Sistema y éste cambie el valor del campo.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CONTROL_SID	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3	PK	ID_CLAVE	VARCHAR	18	NO
	4		ID_VIGENCIA	CHAR	1	NO

Tablas de datos del SIDUNAM

A continuación describiremos el bloque 1, el cual está formado por las tablas encargadas de recibir los distintos tipos de datos que se manejan dentro del Sistema y que posteriormente tendrán que crearse las pantallas apropiadas de captura para cada una de éstas, así como las de consulta. Los datos aquí manejados, como se mencionó anteriormente son pensando en un esquema general y se utilizarán para cualquier Dependencia, ya sea un Deportista o bien una persona que participe en una actividad recreativa, así como también una persona que tenga su centro de trabajo en Medicina Deportiva, Planeación, etc., de igual manera, para todas aquellas personas que practiquen algún deporte dentro de la UNAM.

Datos Generales

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
DATOS_GENERALES	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3		DGEN_AP_PAT	VARCHAR	25	NO
	4		DGEN_AP_MAT	VARCHAR	25	NO
	5		NOMBRES	VARCHAR	25	NO
	6		DGEN_CURP	VARCHAR	18	SI
	7		DGEN_SEXO	CHAR	1	SI
	8		DGEN_DOMICILIO	VARCHAR	80	SI
	9		DELE_CLAVE	CHAR	3	SI
	10		DGEN_CP	CHAR	5	SI
	11		DGEN_TELEFONO	VARCHAR	13	SI
	12		DGEN_LUGAR_NAC	VARCHAR	25	SI
	13		DGEN_FECHA_NAC	DATE		SI
	14		MUNI_CLAVE	CHAR	3	SI
	15		EDO_CLAVE	CHAR	3	SI
	16		DGEN_NACIONALIDAD	VARCHAR	15	SI
	17		DGEN_CORREO	VARCHAR	50	SI

Datos Académicos

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
DATOS_ACADEMICOS	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3		NUM_CTA	NUMERIC	18	NO
	4		ESCF_CLAVE	CHAR	6	NO
	5		CARR_CLAVE	CHAR	6	NO
	6		INSCRIPCION	CHAR	1	SI
	7		PLANTEL	CHAR	3	SI
	8		GENERACION	CHAR	4	SI
	9		TURNO_CARRERA	CHAR	3	SI
	10		CRED_OBLIGATORIOS	CHAR	10	SI
	11		AVANCE_CRED_OB	CHAR	3	SI
	12		CRED_OPTATIVOS	CHAR	10	SI
	13		AVANCE_CRED_OPT	CHAR	5	SI
	14		MAT_ADEUDADAS	CHAR	3	SI
	15		PROMEDIO	REAL	4	SI
	16		CAUSA_INICIO	CHAR	2	SI
	17		PRIMER_PER_HISTORIAL	CHAR	5	NO
	18		U_TIMO_PER_HISTORIAL	CHAR	5	SI

Datos Deportivos

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
DATOS_DEPORTIVOS	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3		DEPO_CLAVE	CHAR	3	SI
	4		ASOD_CLAVE	CHAR	3	SI
	5		DATO_NOAFILIACION	CHAR	3	SI
	6		ESPE_CLAVE	CHAR	3	SI
	7		CATE_CLAVE	CHAR	3	SI
	8		FEDE_CLAVE	CHAR	3	SI
	9		LIGA_CLAVE	CHAR	3	SI
	10		CLUB_CLAVE	CHAR	3	SI
	11		EQUIP_CLAVE	CHAR	3	SI
	12		DATO_NIVELPARTICIPACION	VARCHAR	25	SI
	13		FOTO	IMAGE	16	SI

Datos Médicos

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
DATOS_MEDICOS	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3		DATM_ESTATURA	VARCHAR	8	SI
	4		DATM_PESO	VARCHAR	8	SI
	5		DATM_EDAD	VARCHAR	8	SI
	6		ALER_CLAVE	CHAR	3	SI
	7		TSAN_CLAVE	CHAR	3	SI
	8		OPER_CLAVE	CHAR	3	SI
	9		ENFE_CLAVE	CHAR	3	SI
	10		SIND_CLAVE	CHAR	3	SI
	11		FRAC_CLAVE	CHAR	3	SI

Datos Laborales

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
DATOS_LABORALES	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3		DATL_RAZON_SOCIAL	VARCHAR	25	SI
	4		DATL_LUGAR_TRABAJO	VARCHAR	25	SI
	5		NOMB_CLAVE	CHAR	3	SI
	6		HORA_CLAVE1	CHAR	3	SI
	7		HORA_CLAVE2	CHAR	3	SI
	8		HORA_CLAVE3	CHAR	3	SI

Historia Deportiva

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
DATOS_HDEPORTIVA	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3		DEPO_CLAVE	CHAR	3	SI
	4		CATE_CLAVE	CHAR	3	SI
	5		LUGA_CLAVE	CHAR	3	SI
	6		HDEP_MARCA	VARCHAR	10	SI
	7		NIPA_CLAVE	CHAR	3	SI
	8		ESPE_CLAVE	CHAR	3	SI
	9		LIGA_CLAVE	CHAR	3	SI
	10		CONF_CLAVE	CHAR	3	SI
	11		AÑO	CHAR	4	SI

Otras Actividades

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
DATOS_ACTIVIDADES	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3		AREA_CLAVE	NUMERIC	18	SI
	4		TPER_CLAVE	CHAR	3	SI
	5		EVEN_CLAVE	CHAR	3	SI
	6		LUGA_CLAVE	CHAR	3	SI
	7		FECHA_PART	DATE	8	SI
	8		PARTICIPA	VARCHAR	25	SI
	9		PARTICIPO	VARCHAR	25	SI
	10		REALIZO	VARCHAR	25	SI

Pruebas Físicas

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
DATOS_FISICOS	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3		LAGARTIJAS	VARCHAR	25	SI
	4		ABDOMINALES	VARCHAR	25	SI
	5		SALTO	VARCHAR	25	SI
	6		RESISTENCIA	VARCHAR	25	SI
	7		RAPIDEZ	VARCHAR	25	SI
	8		FUERZA	VARCHAR	25	SI

TESIS CON
FALLA DE ORIGEN

Aquí describimos las partes que forman el bloque 2, el cual está formado por dos partes, la primera es la de Usuarios que como ya se menciona son los que podrán entrar al Sistema y la segunda son los Catálogos que se usarán dentro del mismo.

Usuarios del SIDUNAM

Usuarios

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
USUARIOS_SIDUNAM	1	PK	ID_SID	NUMERIC	9	NO
	2	PK	ID_SUBDIR	CHAR	6	NO
	3		CLAVE	VARCHAR	10	NO
	4		CONTRASEÑA	VARCHAR	10	NO
	5		NOMBRE	VARCHAR	50	SI
	6		DESCRIPCION	VARCHAR	25	SI
	7		PERMISOS	VARCHAR	6	NO

Catálogos del SIDUNAM

Catálogo de Áreas

Utilizado en Actividades del Deportista, referente al área de alguna actividad deportiva.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATA_AREAS	1	PK	AREA_CLAVE	CHAR	3	NO
	2		AREA_DESCRIPCION	VARCHAR	25	SI

Catálogo de Carreras

Utilizado en datos Académicos del Deportista.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATA_CARRERAS	1	PK	CARR_CLAVE	CHAR	3	NO
	2		CARRERA	VARCHAR	25	S
	3		CVE_ESCFAC	NUMERIC	3	NO

Catálogo de Escuelas y Facultades

Utilizado en datos Académicos del Deportista.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATA_ESCFAC	1	PK	ESCF_CLAVE	CHAR	3	NO
	2		ESCUELA/FACULTAD	VARCHAR	25	S

Catálogo de Eventos

Utilizado en Actividades del Deportista, relativo a un evento de alguna actividad deportiva.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATA_EVENTOS	1	PK	EVEN_CLAVE	CHAR	3	NO
	2		EVEN DESCRIPCION	VARCHAR	25	SI

Catálogo de Subdirecciones

Dependencia a la que pertenece o en la que participa. Utilizado en casi todas las tablas del Sistema.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATA_SUBDIRECCIONES	1	PK	ID_SLBDIR	CHAR	6	NO
	2		SUBD DESCRIPCION	VARCHAR	25	SI

Catálogo de Personas

El tipo de persona que es, es decir, Deportista, Coordinador, Juez, Médico, Entrenador, etc. Utilizado en Actividades del Deportista.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATA_PERSONAS	1	PK	TPER_CLAVE	CHAR	3	NO
	2		TPER DESCRIPCION	VARCHAR	25	SI
	3		ID SUBDIR	CHAR	6	NO

Catálogo de Asociaciones

Utilizado en Datos Deportivos, nos indica si el Deportista está afiliado en alguna asociación deportiva.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATD_ASOCIACIONES	1	PK	ASOD_CLAVE	CHAR	3	NO
	2		ASOD DESCRIPCION	VARCHAR	25	SI
	3		ASOD FEDERACIONES	VARCHAR	25	SI
	4		ID SUBDIR	CHAR	6	NO

Catálogo de Categorías

Utilizado en Datos Deportivos, y nos indica la categoría en la cuál está participando de su disciplina deportiva.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATA_CATEGORIAS	1	PK	CATE_CLAVE	CHAR	3	NO
	2		CATE DESCRIPCION	VARCHAR	25	SI
	3		ID SUBDIR	CHAR	6	NO

Catálogo de Clubes

Utilizado en Datos Deportivos, nos indica si el Deportista está afiliado en algún club deportivo.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATD_CLUBS	1	PK	CLUB_CLAVE	CHAR	3	NO
	2		CLUB_DESCRIPCION	VARCHAR	25	SI
	3		ID_SUBDIR	CHAR	6	NO

Catálogo de Deportes

Utilizado en Datos Deportivos, y nos indica el deporte que practica.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATD_DEPORTES	1	PK	DEPO_CLAVE	CHAR	3	NO
	2		DEPO_DESCRIPCION	VARCHAR	25	SI
	3		ID_SUBDIR	CHAR	6	NO

Catálogo de Equipos

Utilizado en Datos Deportivos, nos indica el equipo con el cuál participa.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATD_EQUIPOS	1	PK	EQUPE_CLAVE	CHAR	3	NO
	2		EQUPE_DESCRIPCION	VARCHAR	25	SI
	3		FEDE_CLAVE	CHAR	3	SI
	4		ASOD_CLAVE	CHAR	3	SI
	5		LIGA_CLAVE	CHAR	3	SI
	6		CLUB_CLAVE	CHAR	3	SI
	7		ID_SUBDIR	CHAR	6	NO

Catálogo de Especialidades

Utilizado en Actividades del Deportista, y nos indica su especialidad.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATD_ESPECIALIDADES	1	PK	ESPE_CLAVE	CHAR	3	NO
	2		ESPE_DESCRIPCION	VARCHAR	25	SI
	3		ID_SUBDIR	CHAR	6	NO

Catálogo de Federaciones

Utilizado en Datos Deportivos, nos indica la federación en la que está inscrito.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATD_FEDERACIONES	1	PK	FEDE_CLAVE	CHAR	3	NO
	2		FEDE_DESCRIPCION	VARCHAR	25	SI
	3		FEDE_CODEME	VARCHAR	25	SI
	4		FEDE_CONADE	VARCHAR	25	SI
	5		ID_SUBDIR	CHAR	6	NO

Catálogo de Ligas

Utilizado en Datos Deportivos, nos indica si está inscrito en alguna liga, participando en equipo o individualmente.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATD_LIGAS	1	PK	LIGA_CLAVE	CHAR	3	NO
	2		LIGA_DESCRIPCION	VARCHAR	25	SI
	3		FEDE_CLAVE	CHAR	3	SI
	4		ASOD_CLAVE	CHAR	3	SI
	5		ID_SUBDIR	CHAR	6	NO

Catálogo de Delegaciones

Utilizado en Datos Generales, para capturar su Delegación Política.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATG_DELEGACIONES	1	PK	DELE_CLAVE	CHAR	3	NO
	2		DELE_DESCRIPCION	VARCHAR	25	SI

Catálogo de Estados

Utilizado en Datos Generales, para capturar su Estado en el cuál radica.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATG_ESTADOS	1	PK	EDO_CLAVE	CHAR	3	NO
	2		EDO_DESCRIPCION	VARCHAR	25	SI

Catálogo de Municipios

Utilizado en Datos Generales, para capturar el Municipio de su domicilio si es que lo tiene y depende del Estado.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATG_MUNICIPIOS	1	PK	MUNI_CLAVE	CHAR	3	NO
	2		MUNI_DESCRIPCION	VARCHAR	25	SI
	3		EDO_CLAVE	CHAR	3	NO

Catálogo de Lugares

Utilizado en su Historia Deportiva, y nos indica el lugar en el cuál se realizó su participación deportiva.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATH_LUGARES	1	PK	LUGA_CLAVE	CHAR	3	NO
	2		LUGA_DESCRIPCION	VARCHAR	25	SI

Catálogo de Nivel de Participación

Utilizado en Historia Deportiva, y nos indica el nivel o lugar que obtuvo en algún evento que participó.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATI_NIVEL_PARTICIPACION	1	PK	NIPA_CLAVE	CHAR	3	NO
	2		NIPA_DESCRIPCION	VARCHAR	25	SI

Catálogo de Horarios

Utilizado en Datos Laborales, sirve para capturar su horario de trabajo, si es que el Deportista trabaja.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATL_HORARIOS	1	PK	HORA_CLAVE	CHAR	3	NO
	2		HORA_DESCRIPCION	VARCHAR	25	SI

Catálogo de Nombramientos

Utilizado en Datos Laborales, nos indica si trabaja en la UNAM su nombramiento, si no el puesto que desempeña en su lugar de trabajo.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATL_NOMBRAMIENTOS	1	PK	NOMB_CLAVE	CHAR	3	NO
	2		NOMB_DESCRIPCION	VARCHAR	25	SI

Catálogo de Alergias

Utilizado en Datos Médicos, se utiliza para capturar si el Deportista tiene alergias.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATM_ALERGIAS	1	PK	ALER_CLAVE	CHAR	3	NO
	2		ALER_DESCRIPCION	VARCHAR	25	SI

Catálogo de Enfermedades

Utilizado en Datos Médicos, se utiliza si el Deportista ha tenido alguna enfermedad.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATM_ENFERMEDADES	1	PK	ENFE_CLAVE	CHAR	3	NO
	2		ENFE_DESCRIPCION	VARCHAR	25	SI

Catálogo de Fracturas

Utilizado en Datos Médicos, se utiliza para indicar si ha estado fracturado.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATM_FRACTURAS	1	PK	FRAC_CLAVE	CHAR	3	NO
	2		FRAC_DESCRIPCION	VARCHAR	25	SI

Catálogo de Operaciones

Utilizado en Datos Médicos, se utiliza para indicar si se le ha practicado alguna operación.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATM_OPERACIONES	1	PK	OPER_CLAVE	CHAR	3	NO
	2		OPER_DESCRIPCION	VARCHAR	25	SI

Catálogo de Síndromes

Utilizado en Datos Médicos, se utiliza para indicar si se tiene algún síndrome.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATM_SINDROMES	1	PK	SIND_CLAVE	CHAR	3	NO
	2		SIND_DESCRIPCION	VARCHAR	25	SI

Catálogo de Tipo de Sangre

Utilizado en Datos Médicos, indica el tipo de sangre para en determinado momento si sufre algún accidente la persona, poder ayudarlo.

TABLA	CAMPO	LLAVE	NOMBRE ATRIBUTO	TIPO	LONGITUD	NULO
CATM_TIPOS_SANGRE	1	PK	TSAY_CLAVE	CHAR	3	NO
	2		TSAY_DESCRIPCION	VARCHAR	25	SI

5.5. Diseño de módulos del SIDUNAM

Para el desarrollo del diseño de módulos, como se dijo anteriormente, lo dividiremos en tres bloques:

- Acceso al Sistema
- Mantenimiento del Sistema
- Información del Deportista

1. Acceso al Sistema.

En éste primer bloque manejamos distintos niveles de acceso ya que el usuario tendrá que introducir una Clave y una Contraseña, y en base a ésta tendrá ciertos privilegios, ya que serán almacenados y cada que se necesite pueden ser consultados para permitir el acceso o negarlo a la entrada de un bloque posterior.

En esta parte capturamos sus permisos y la dependencia a la cuál pertenece. Los permisos de un usuario pueden ser de consulta, actualización y mantenimiento del Sistema (Fig. 5.6).

El diagrama muestra una interfaz de usuario con los siguientes elementos:

- Un recuadro superior con el texto: **Organización, nombre del Sistema e Imágenes correspondientes.**
- Un recuadro central izquierdo con el texto: **Área de captura de :
Clave y Contraseña**
- Un recuadro central inferior con el texto: **Área de:
Subdirección y Permisos**
- Un recuadro a la derecha con el texto: **Botones para:
Aceptar
Cancelar
Salir**

Fig. 5.6.

2. Mantenimiento del Sistema.

Este módulo está dividido en dos partes, la primera se refiere a los Usuarios del Sistema y la segunda al Área de los Catálogos.

En el módulo de Usuarios del Sistema se captura la información correspondiente a la Clave, Contraseña, Dependencia, los Permisos, la Descripción y el Nombre del Usuario. Para cada dependencia existirá un responsable que solicitará al responsable del Sistema el alta o baja de uno o varios Usuarios. Todo esto para llevar un control sobre los Usuarios que acceden al Sistema y que permisos tienen; los permisos que tiene un usuario son de consulta (Fig. 5.7).

ESTA TESIS NO SALE
DE LA BIBLIOTECA

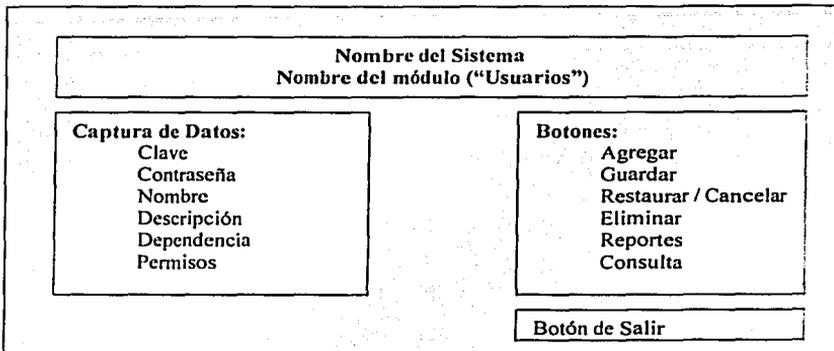


Fig. 5.7.

Ahora bien en el módulo de Catálogos tenemos una parte en donde manejaremos los Catálogos comunes y otras en donde se manejarán por Dependencia, pero el diseño de los módulos y la captura de cada Catálogo será de la misma forma como el anterior de usuarios. La parte superior describe el nombre del Sistema y a cuál Catálogo pertenece, la parte derecha el área de botones y la izquierda los campos de cada Catálogo, por último tendremos un botón de salir (Fig. 5.8).

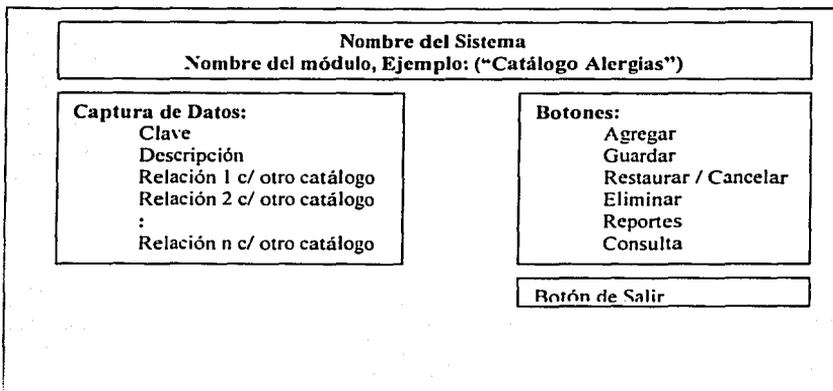


Fig. 5.8.

El responsable de cada Dependencia proporcionará al responsable del Sistema la información correspondiente a su área, para que sea capturada en el Catálogo correspondiente y así pueda alimentar al Sistema con información confiable.

Para navegar entre las distintas dependencias utilizaremos una selección ("menús"), para decidir a cuál dependencia acceder (Fig. 5.9).

Nombre del Sistema
Nombre del módulo "Catálogos"

Área de selección para:
Dependencia y Catálogo

Botón de Salir

El diagrama muestra una interfaz de usuario con un recuadro principal que contiene un sub-recuadro superior con el texto "Nombre del Sistema" y "Nombre del módulo 'Catálogos'". Debajo de esto hay un área de selección con el texto "Área de selección para:" y "Dependencia y Catálogo". En la parte inferior derecha del recuadro principal hay un botón etiquetado como "Botón de Salir".

Fig. 5.9.

3. Información del Deportista

Por último describiremos el bloque 3. Información del Deportista. Esta es la parte más importante ya que fue la que originó el desarrollo del Sistema, aquí manejaremos la Dependencia a la cuál pertenece el Deportista y toda la información que éste pueda tener como son: Datos Generales, Académicos, Deportivos, Médicos, Laborales, su Historial Deportivo, Estadísticas Físicas y sus actividades dentro y fuera de la UNAM (Fig. 5.10).

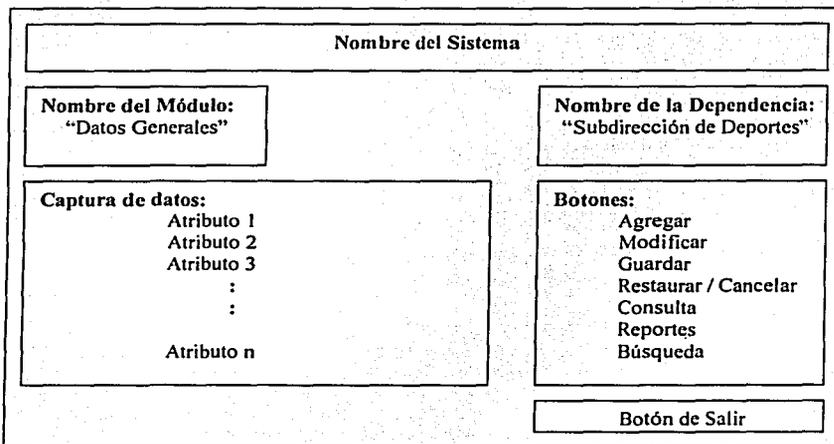


Fig. 5.10.

Una vez capturada toda la información de los Deportistas podremos generar reportes por Dependencia, Escuela o Facultad, Disciplina Deportiva, etc. Así como también podremos generar una credencial deportiva ya que contamos con información suficiente para ello.

Las Instancias que serán incluidas dentro del Sistema son: La Subdirección de Deportes, Logística, Recreación, Medicina del Deporte y Planeación, las Coordinaciones de Comunicación Social e Informática, La Unidad Administrativa y La Dirección.

Las Dependencias que de alguna forma manejarán mayor información son: Deportes ya que tiene 35 disciplinas deportivas entre las cuales están ajedrez, atletismo, buceo, canotaje, handball, natación, etc. Logística por su parte contempla lo relacionado con Fútbol Americano. Así la Subdirección de Recreación, básicamente esta orientada a los niños, ya que su objetivo principal es promover las actividades físicas y mentales, así como invitarlos a que practiquen alguna disciplina de las que se imparten dentro de la UNAM y fomentar el desarrollo integral del niño.

En esta parte también contaremos con una forma de navegación entre Subdependencias y el tipo de datos del Deportista. (Fig. 5.11).

Nombre del Sistema

Área de selección para: *Dependencia y tipo de datos*

Ejemplo: *Subdirección de Recreación Datos Deportivos*

Botón de Salir

Fig. 5.11.

En el capítulo siguiente se analizará la infraestructura de cómputo y comunicaciones dentro de la Universidad, con el fin de poder hacer uso de ella; definiremos el Sistema Operativo y el Manejador de Base de Datos a utilizar, para el desarrollo del Sistema SIDUNAM, así también, se elegirá el equipo de cómputo que se ocupará para echar a andar la aplicación (Servidor y Clientes).

VI. ANÁLISIS DE LA TECNOLOGÍA PARA LA IMPLEMENTACIÓN DEL SISTEMA (SIDUNAM)

Teniendo en mente y llevando a la práctica el aprovechamiento al máximo de toda la infraestructura tecnológica de la UNAM en cuanto a equipo de cómputo, telecomunicaciones, Internet y lo más actual en *software* de desarrollo de sistemas de información; para la elaboración de la presente tesis se abordarán en este capítulo los temas sobre los cuales se pondrá en operación el sistema en cuestión.

6.1. RedUNAM

Para las universidades mexicanas Internet representa una poderosa herramienta para acercar a los estudiantes a un cúmulo de información reciente, estimulando su interés hacia la investigación y la comunicación con jóvenes universitarios e instituciones de alto nivel, buscando concretar una formación educativa activa e integral. Consciente de la tendencia mundial hacia el desarrollo de nuevas tecnologías, la máxima casa de estudios ha creado uno de los proyectos más ambiciosos de toda Latinoamérica: La Red Integral de Telecomunicaciones de la Universidad Nacional Autónoma de México, conocida como RedUNAM.

RedUNAM es el proyecto desarrollado para la transmisión de información (datos, video y voz) entre las facultades, institutos, centros de difusión, coordinaciones y demás dependencias que conforman a la UNAM.

6.2. Red de Telecomunicaciones de la UNAM

Haciendo un poco de historia tenemos que a finales de los años 60's y el principio de la década de los 70's marcaron para la UNAM, la etapa de inicio de las comunicaciones telefónicas y de datos. Es en ese periodo cuando se realizan las primeras conexiones de teletipos hacia una computadora central, utilizando líneas telefónicas de cobre, de la recién instalada red telefónica dentro de la institución.

Rápidamente esta tecnología es usada al interior de la UNAM y difundida al exterior, por ello se efectúan una gran cantidad y diversidad de conexiones, de terminales de caracteres, de graficación e impresión, hasta la interconexión de estaciones de trabajo -remotas todas ellas- manejando líneas telefónicas. A partir de la segunda parte de la década de los 80's surge en la UNAM la búsqueda de cambios en las comunicaciones. Así en 1987, la UNAM establece la primera conexión a la Red Académica BITNET, mediante enlaces telefónicos, desde la Ciudad Universitaria hasta el Instituto Tecnológico de Estudios Superiores de Monterrey (ITESM) y de ahí hasta San Antonio, Texas en los EUA.

Posteriormente, la UNAM buscó consolidar su enlace a esa red internacional mediante la computadora IBM 4381, la cual sirvió como residencia del correo electrónico y otros servicios de BITNET; dentro de ese proceso se inició la conexión de terminales IBM con emulación 3270, estableciéndose además un enlace con la Red TELEPAC de la SCT, bajo la finalidad, nunca lograda, de brindar este servicio a nivel nacional. No fue sino hasta 1989, cuando la UNAM a través del Instituto de Astronomía establece un convenio de enlace a la red de la NSF (*National Science Foundation*) en EUA, el cual se realizó utilizando el satélite mexicano Morelos II entre el Instituto de Astronomía en la UNAM y el UCAR (*University Corporation for Atmospheric Research*)-NCAR (*National Center for Atmospheric Research*) con residencia en Boulder Colorado. Asimismo, se llevó a cabo el primer enlace para conectar las redes de área local, entre el Instituto de Astronomía y la Dirección General de Servicios de Cómputo Académico (DGSCA), utilizando enlaces de fibra óptica.

A partir de ese momento se inició dentro de la UNAM una revolución en las comunicaciones, así como la adquisición masiva de computadoras personales y su interconexión e intercomunicación en redes de área local, principalmente en las dependencias del subsistema de la investigación científica; lo cual permitió desarrollar la infraestructura de comunicaciones con fibra óptica, y establecer más enlaces satelitales hacia Cuernavaca, Mor. y San Pedro Mártir en Ensenada, Baja California Norte, a la par del primer enlace de microondas de alta velocidad entre la Torre II de

Humanidades y la Dirección General de Servicios de Cómputo Académico, DGSCA, sobre la Ciudad de México.

Con esto último, se estableció en definitiva el final de la era del teleproceso, para dar paso a las redes de computadoras y sus enlaces a través de fibra óptica. En 1990 la UNAM, fue la primera institución en Latinoamérica que se incorpora a la red mundial Internet, que enlaza a millones de máquinas y decenas de millones de usuarios en todo el mundo.

Dicha red es producto de un proyecto del Gobierno de los Estados Unidos que data de 1970, y en sus primeras etapas como parte de un programa de investigación militar de DARPA (*Defense Advanced Research Projects Agency*) se logra demostrar la viabilidad de las comunicaciones entre computadoras, por medio de la conmutación de paquetes; lo cual creó la red ARPANET, que enlazó en sus primeros años varias decenas de sitios en una red nacional dedicada a la comunidad de investigación en computación. El concepto de conmutación de paquetes se extendió en muy pocos años para incluir redes satelitales y redes basadas en radio. Su ininterrumpido desarrollo que no tiene límite a la fecha, contempla como elemento fundamental el diseño de una arquitectura para comunicar redes que permita la coexistencia de paquetes de comunicación de diferentes tipos bajo el protocolo TCP/IP; mismo que se mantiene como estándar en la actualidad, dado su funcionalidad y posibilidad de adaptación a los requerimientos que se van presentando. A finales de los 80's se da la apertura al uso comercial, en tanto se limitaba a proporcionar servicios a la comunidad académica. A finales de 1989 se estableció un ambicioso proyecto que debía sustituir los antiguos conmutadores para renovar totalmente el sistema telefónico de la UNAM, de acuerdo con los estándares más modernos y con capacidad de crecer conforme a las necesidades de la institución. Para este proyecto que constituye la parte fundamental del Programa Institucional en informática, en la Dirección General de Servicios de Cómputo Académico se creó la **Dirección de Telecomunicaciones Digitales** cuyo objetivo sería la creación de la **Red Integral de Telecomunicaciones de la UNAM**, la

cual debería ser capaz de transmitir indistintamente datos e imágenes entre las dependencias universitarias independientemente de su ubicación geográfica.

Ante la necesidad de integrar los diferentes servicios y recursos de cómputo como soporte de desarrollo eficiente para la investigación y la docencia, surge el Laboratorio de REDUNAM en 1990 (proyecto del Departamento de Redes y Comunicaciones de la DGSCA) como un espacio para el estudio, análisis de comunicación, topologías de redes, protocolos y servicios, entre otras cuestiones.

6.2.1. Características principales de la Red de Telecomunicaciones

La Red Integral de Telecomunicaciones de la Universidad Nacional Autónoma de México se inaugura oficialmente en 1992, véase la Fig. 6.1.

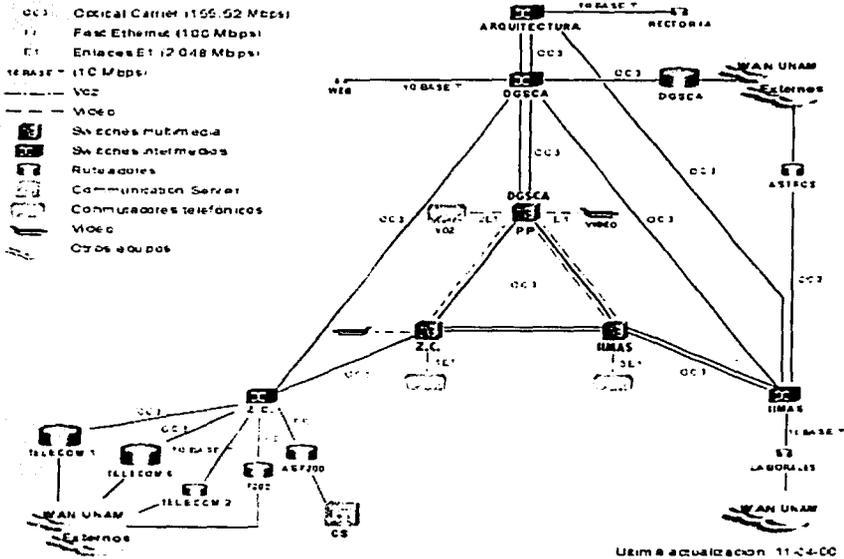


Fig. 6.1. Red Integral de Telecomunicaciones de la Universidad Nacional Autónoma de México

Entre sus principales características destacan hoy en día:

- Transmisión indistinta de datos y video, mediante sistemas digitales basados en normas internacionales que rigen actualmente.
- Integración a la red de las principales instalaciones de la Universidad.

Esto significa, que a nivel bachillerato, licenciatura, posgrado e investigación, alrededor del 95% de sus miembros se encuentran en instalaciones cubiertas por la red, en varias regiones del país desde Ensenada, Baja California; hasta Puerto Morelos en Quintana Roo.

6.2.2. Servicios de la Red de Telecomunicaciones

Los servicios que ofrece REDUNAM son, en primer lugar, aquellos que permiten la comunicación entre los diversos *hosts* (entendiéndose por éstos, las máquinas que fungen como Servidores y como Clientes).

Entre los servicios brindados por REDUNAM destacan los que se muestran a continuación:

- Ruteo
El ruteo se encarga de que los archivos no se extravíen en alguna parte del mundo o lleguen a personas equivocadas. Sin este servicio no sería posible la comunicación con los *hosts* de otras redes. También es posible limitar a usuarios indeseables.
- DNS (Domain Name Service)
Se encarga de resolver la conversión entre las direcciones numéricas y los nombres lógicos de las máquinas, de forma tal que el usuario pueda utilizar el nombre: *medusa.fi-c.unam.mx* para comunicarse con su Servidor de Base de Datos en lugar de la dirección: *132.248.139.169*.

- **NOC (Network Operation Center)**
Se encarga de monitorear el comportamiento de la red, conservar su óptima operación, así como darle el mantenimiento necesario.
- **NIC (Network Information Center)**
Se encarga de distribuir la información de los servicios de red, soportarlos dentro de la **REDUNAM** y dar la capacitación necesaria a los usuarios sobre sus beneficios.
- **Correo Electrónico**
Comunicación entre usuarios a través de mensajes. Es uno de los servicios de mayor demanda, el cual permite la comunicación entre usuarios de cualquier parte del mundo a través de la colocación de mensajes de texto en el buzón electrónico del destinatario.
- **Gopher**
Menús jerárquicos para búsqueda de información en **REDUNAM** e Internet a través de conexiones transparentes al usuario.
- **Gateway (Puerta de Enlace)**
Comunicación entre redes de diferentes clases. Por ejemplo una red de mainframe de IBM a una red de PC's.
- **FTP (Protocolo de transferencia de archivos)**
Transferencia de archivos de una máquina remota a otra local o viceversa.
- **Telnet**
Sesiones remotas a grandes computadores para aprovechar sus altas capacidades de cálculo y otros recursos que difícilmente se encuentran en máquinas menores

- **WWW (World Wide Web)**

Sin duda uno de los servicios más interesantes y completos que se pueden encontrar en Internet. Este servicio aprovecha la tecnología multimedia para presentar información mediante el uso de imágenes, texto y sonido desde cualquier punto de Internet a la computadora del usuario.

- **Submáscaras de Red**

Notación estándar para grupos de direcciones IP, a veces llamada "Dirección de Red". Igual que en un número de teléfono puede ser separado en prefijo de área y el resto, así, podemos separar una dirección IP en el prefijo de red y el resto. En nuestro caso es: *255.255.255.0* y se dice que es de clase "C".

6.3. El protocolo de comunicación TCP/IP

Aunque poca gente sabe lo que es TCP/IP todos lo emplean indirectamente y lo confunden con un solo protocolo cuando en realidad son varios, de entre los cuales destaca y es el más importante el protocolo IP. Bajo este nombre (TCP/IP) se esconde uno de los protocolos más usados del mundo, si bien es cierto que existen diferentes tipos de protocolos, entre los cuales los de mayor uso en empresas pequeñas y medianas, así como en grandes corporativos son los siguientes:

- TCP/IP
- IPX/SPX
- NetBeui
- DLC

Sin lugar a dudas, debido al gran auge y crecimiento que ha tenido Internet en los últimos años, el protocolo que se perfila como estándar mundial es el TCP/IP,

adoptándolo de igual forma dentro de la **REDUNAM**. Por tal motivo la aplicación que se desarrolló en la presente tesis, tomó como base el uso de este protocolo.

6.3.1. Historia del protocolo de comunicación TCP/IP

TCP/IP (*Transmission Control Protocol/Internet Protocol*). En el año de 1973 la DARPA (*Defense Advanced Research Projects Agency*) inició un programa de investigación de tecnologías de comunicación entre redes de diferentes características. El proyecto se basaba en la transmisión de paquetes de información, y tenía por objetivo la interconexión de redes. De este proyecto surgieron dos redes: Una de investigación, ARPANET, y una de uso exclusivamente militar, MILNET. Para comunicar las redes, se desarrollaron varios protocolos: El protocolo de Internet y los protocolos de control de transmisión. Posteriormente estos protocolos se englobaron en el conjunto de protocolos TCP/IP. En 1980, se incluyó en el UNIX 4.2 de BERKELEY, y fue el protocolo militar estándar en 1983. Con el nacimiento en 1983 de INTERNET, este protocolo se popularizó bastante, y su destino va unido al de Internet. ARPANET dejó de funcionar oficialmente en 1990.

El propósito de TCP/IP es proporcionar los protocolos necesarios de transmisión de información independientemente de la red o equipo utilizado, aplicándose en organizaciones que precisaban establecer conexiones entre distintos tipos de LAN. Dada la flexibilidad que caracteriza un protocolo de esa naturaleza, TCP/IP ha sido elevado casi de forma inmediata al dominio de la Internet, ya que el modelo básico en Internet es el modelo Cliente/Servidor. El Cliente es un programa que le solicita a otro que le preste un servicio. El Servidor es el programa que proporciona este servicio. La arquitectura de Internet esta basada en capas. Esto hace más fácil implementar nuevos protocolos. El conjunto de protocolos TCP/IP, al estar integrado plenamente en Internet, también dispone de este tipo de arquitectura. El modelo de capas de TCP/IP es algo diferente al propuesto por ISO (*International Standard Organization*) para la interconexión de sistemas abiertos OSI (*Open System Interconnection*).

6.3.2. ¿Cómo trabaja el protocolo TCP/IP?

El protocolo TCP/IP define una red de intercambio por paquetes, lo que significa que la información se transmite entre las computadoras en partes: una parte está conformada por los paquetes que contienen una "carga" de datos y, la otra, por la información de cabecera que identifica el envío, constituida por la dirección de destino y los códigos de corrección de errores. Si bien algunas órdenes requieren tan sólo un único paquete, las transmisiones de archivos de gran extensión se dividen en múltiples paquetes, debiendo incluir una secuencia de números indicativos del orden según el cual los paquetes se deben re-ensamblar en el sistema de destino.

Una de las características más importantes del protocolo TCP/IP es su esquema de direccionamiento de red, el cual proporciona una forma de direccionar diferentes redes, así como los nodos de las mismas. Este esquema es extensivo, dando cabida a millones de posibles direcciones en una escala global. De hecho, las direcciones Internet son registradas y asignadas por el NIC (*Network Information Center*), a fin de evitar conflictos aunque no es esencial registrar las direcciones, si tan sólo se trata de establecer una conexión entre redes en un ámbito reducido.

Como se dijo en un principio, TCP/IP está compuesto por un conjunto de diferentes protocolos. Cada uno de ellos cae dentro de un esquema de estratos que se compara a menudo con el modelo OSI (*Open System Interconnection*). A continuación se describe cada uno de estos protocolos:

- **TCP (*Transmission Control Protocol*)**. El protocolo TCP proporciona un servicio de comunicación que forma un circuito, es decir, que el flujo de datos entre el origen y el destino parece que sea continuo. TCP proporciona un circuito virtual el cual es llamado una conexión. El protocolo TCP corresponde al estrato de sesión y transporte del modelo OSI. TCP recibe información de las aplicaciones que operan en niveles superiores en la pila de protocolos, y es responsable del empaquetado y la transmisión a través de la red. TCP pasa a IP los paquetes que crea.

- **IP (*Internet Protocol*)**. El protocolo IP corresponde al estrato Red en el modelo OSI. Este se encarga de crear los paquetes de información, y es el responsable de añadir las direcciones IP de origen y destino.

Algo que IP no puede proporcionar es la garantía de que los paquetes llegarán a su destino, y en el orden adecuado. Por tal motivo, TCP añade información a cada uno de los paquetes que este módulo crea, en la que incluye datos de identificación y ubicación. Si IP pierde un paquete, TCP es el responsable de determinar cuál es el paquete perdido, y de volverlo a enviar.

6.3.3. Direcciones IP

La dirección ***Internet Protocol*** (IP) de un nodo es una dirección lógica, que es independiente de la dirección física asignada a la tarjeta de red por el fabricante de la misma. La dirección IP es también independiente de la configuración de la red. La dirección IP tiene la misma forma, no importando el tipo de red que se usa. Este formato único es un valor numérico de 4 bytes (32 bits), que sirve para identificar, tanto a la red, como al nodo de la misma. Cada dirección IP debe ser única y consta de cuatro números decimales, separados por puntos. Como se mencionó anteriormente, las direcciones IP no dependen de ningún tipo de red en concreto. Ello significa que los paquetes pueden atravesar diferentes tipos de redes. En cada uno de los tipos de red, el protocolo TCP/IP se encarga de asignar la dirección IP a un nodo físico, para poder realizar el tránsito. Los paquetes contienen las direcciones IP del remitente, lo cual permite que el destinatario pueda reclamar los datos, en caso de ser necesario. La dirección IP permite identificar tanto la red como el nodo que corresponde al remitente. Una forma de definir direcciones IP consiste en unirse a la comunidad DARPA Internet, contactando con el NIC (***Network Information Center***) mencionado en párrafos anteriores. Si no se está interesado en registrar oficialmente una red, se pueden elegir números arbitrarios que se rijan por el esquema de direccionamiento IP (cuatro números decimales, separados por puntos). Sin embargo, es aconsejable que se utilice el esquema DARPA Internet, por si en el futuro se precisase entrar en contacto con

algún punto externo. La dirección IP de 4 bytes está dividida en dos partes: una de ellas, sirve para identificar a la red, mientras que la otra, identifica a la computadora (nodo). De esta manera, la parte que corresponde a la red, debe ser siempre la misma para todos los nodos que la conforman, en tanto que la parte que identifica al nodo, ha de ser única y diferente para cada equipo conectado a dicha red.

Existen diversos esquemas para la asignación de estos números:

1. Esquema de direccionamiento Clase A.

El primer byte es la dirección de la red, y los tres últimos la del nodo. El rango para el primer byte es de 1 a 126, lo cual permite un total de 126 redes diferentes, con 16 millones de nodos cada una.

2. Esquema de direccionamiento Clase B.

Es el esquema utilizado dentro de la Red UNAM, donde los primeros bytes identifican la red, y los dos últimos, el nodo. El rango para el primer byte es de 128 a 191, quedando el segundo para identificaciones más finas. Esto permite un total de 16.000 redes con 65.000 nodos.

3. Esquema de direccionamiento Clase C.

Los tres primeros bytes identifican a la red, y el último es la dirección del nodo, lo cual permite un total de 2 millones de redes diferentes, con 254 nodos cada una.

Es posible dividir una red en varias subredes, para así poder hacer uso de múltiples tipos de medios, o bien para reducir la congestión, disminuyendo el número de puestos de trabajo de la red. Cuando existen subredes, las direcciones IP constan de una dirección de red, una dirección de subred y una dirección de nodo. La parte de dirección que corresponde al nodo en la dirección IP, se divide para incluir las direcciones de la subred y del nodo. Para las redes externas, la red total seguirá apareciendo como un único conjunto, con una única dirección de red.

6.3.4. Principales características del protocolo TCP/IP

Entre las principales características que posee el protocolo TCP/IP, se pueden mencionar las siguientes:

- Independencia de tecnologías de redes (Soporta múltiples tecnologías).
Ya que no está basado sobre el *hardware* de ningún vendedor en especial. Los protocolos de TCP/IP definen la unidad de transmisión como datagrama y especifican cómo transmitir datagramas en una red.
- Interconexión universal (Puede funcionar en máquinas de cualquier tamaño).
Una red de TCP/IP permite la comunicación de cualquier pareja de computadoras que forman parte de la red. A cada computadora se le asigna una dirección, que es reconocida por todos los equipos integrantes de la red. Cada datagrama lleva la dirección de origen y destino. Las computadoras intermedias usan la dirección de destino para tomar decisiones de ruteo.
- Acuse de recibo entre computadoras que dialogan (Reconocimientos de extremo a extremo).
Los protocolos de TCP/IP proporcionan un acuse de recibo sólo entre las dos computadoras que dialogan y no entre máquinas sucesivas que forman parte de la trayectoria.

6.4. Plataformas de *hardware*

Como al principio del capítulo, en esta ocasión se analizará lo referente a las características del *hardware*, para poder llevar a cabo nuestro objetivo, por lo tanto, nos daremos a la tarea de explicar las particularidades de éste e iremos proponiendo el más viable tanto para el Servidor como para los equipos Clientes. En cuanto al equipo que fungirá como Servidor, se hace necesario exigir, las siguientes características que analizaremos a continuación.

Cabe mencionar que, dada la dimensión del sistema a generar, la plataforma a utilizar debe de ser robusta, confiable, eficiente, escalable y con un alto rendimiento, de tal forma que consienta un buen número de usuarios simultáneamente.

Así, se realizó una investigación comparativa de las tecnologías existentes en el mercado, con el fin de elegir la que mejor se apegue a las necesidades de nuestro sistema.

6.4.1. Procesadores CISC

Durante los años 60, 70 y 80's la evolución de los procesadores se caracterizo por un enriquecimiento del repertorio de instrucciones, tanto en cantidad como en modos de direccionamiento, se creía que la mejor forma de crear un procesador era incrementando su complejidad, como parte de un esfuerzo por emular fielmente las instrucciones proporcionadas por un lenguaje de alto nivel. A este tipo de procesadores se les llama CISC (*Complex Instruction Set Computer*), como por ejemplo los Intel x86, Motorola 68000 y Digital VAX. Un procesador CISC se caracteriza por tener un *set* de instrucciones amplio, una gran variedad de modos de direccionamiento y el uso extensivo de microcódigo.

En la teoría, un procesador CISC parece una buena opción ya que ejecuta directamente en *hardware* las instrucciones de *software*. Sin embargo, el análisis de cientos de programas en código objeto reveló los siguientes hechos:

- Los compiladores muy frecuentemente no hacen uso completo de las instrucciones proporcionadas por el procesador (alrededor del 80% de las operaciones se realizan con sólo un 20% de las instrucciones del repertorio).
- Un diseño complejo lleva a implementaciones ineficientes de las instrucciones.

- El uso de las instrucciones simples como **LOAD**, **STORE**, **ADD**, **SUB** y **BRANCH** son mucho más frecuentes que el de las instrucciones más complejas.

6.4.2. Procesadores RISC

Los procesadores RISC (*Reduced Instruction Set Computer*), como ejemplo la arquitectura SGI MIPS, Sun Sparc, Digital Alpha, PowerPC y HP PaRISC, se basan en simplificar al máximo el conjunto de instrucciones. En este tipo de arquitecturas, la complejidad queda a cargo del compilador. Los principales puntos de diseño de un procesador RISC son:

- Conjunto reducido de instrucciones. Sus instrucciones son sencillas, realizan únicamente las operaciones básicas. Idealmente, se busca que se ejecute una o más instrucciones por cada ciclo de reloj.
- No existen modos de direccionamiento. Tienen arquitectura (**LOAD/STORE**). Esto quiere decir que los accesos a la Memoria Principal (MP) son para extraer instrucciones, datos y para almacenar éstos últimos. Con esto se simplifica el diseño del procesador.
- Existe un mayor espacio para incorporar el chip de memoria.
- Tienen un número grande de registros de CPU.
- Menor consumo de potencia.
- Tiempos más cortos de diseño.

Por lo antes mencionado, concluimos que no es un hecho meramente académico, sino puramente comercial y económico. La "era RISC" ha alcanzado a todos los fabricantes de semiconductores: AMD, Intel, MIPS, Motorola, etc., y todos ellos son productos usados por fabricantes de computadoras y estaciones de trabajo: Apple, DEC, HP, IBM, SUN, etc. y sus correspondientes clónicos.

El tiempo de diseño de estos productos se reduce sensiblemente, lo que disminuye su costo final, además, son globalmente más eficaces, de menores dimensiones y más bajo consumo, ofreciendo siempre claras ventajas técnicas frente a los más avanzados CISC.

6.4.3. Multiproceso

La industria informática, ha tenido siempre un objetivo primordial, repetido a lo largo de toda su cadena (fabricantes de semiconductores, fabricantes de sistemas y usuarios): **La búsqueda de la velocidad**. Para alcanzar este objetivo se han invertido ingentes cantidades de recursos, hasta alcanzar los límites físicos del silicio.

Hoy es posible construir sistemas, que aún teniendo procesadores más "lentos" que otros, ofrezcan un rendimiento significativamente superior. Son los sistemas multiprocesador, que como su denominación indica, incorporan varios procesadores para llevar a cabo las mismas funciones.

Ahora, y en gran medida gracias a los procesadores de arquitectura RISC, el soporte multiprocesador es una solución integrada y fácilmente disponible en estaciones de trabajo de sobremesa, que resuelve, a través de *hardware* VLSI (**Very Large Scale Integration**), los complejos problemas de compartición de recursos (memoria compartida) de aquellas primeras máquinas.

Evidentemente, estas mejoras en el *hardware*, para ser funcionales, requieren importantes desarrollos en el *software*, y de hecho, muchos Sistemas Operativos admiten extensiones multiproceso (SCO, Solaris, System V, Windows NT, Unix, Linux, etc.), que proporcionan paralelismo "en bruto" (asignando múltiples tareas a múltiples procesadores) a nivel del Sistema Operativo.

El multiproceso no es algo difícil de entender: más procesadores significan más potencia computacional. Un conjunto de tareas puede ser completado más rápidamente

si hay varias unidades de proceso ejecutándolas en paralelo; adicionalmente, las plataformas multiproceso son fácilmente escalables.

Ahora bien, en el caso de nuestra aplicación, estamos concientes de que las aplicaciones de Base de Datos son típicamente intensas en manejo de Entradas/Salidas, por lo tanto, las arquitecturas de *hardware* que permiten la utilización de múltiples procesadores comúnmente otorgan un rendimiento superior a aquellas que no lo poseen.

6.4.4. Multiprocesamiento simétrico-asimétrico

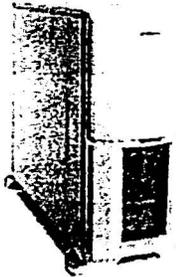
Estas dos arquitecturas tienen diferencias entre sí, la más marcada es el rendimiento que manifiestan cada una de ellas. El multiprocesamiento simétrico, trata a todos los procesadores como iguales. Cualquier procesador puede hacer el trabajo de cualquier otro, y las aplicaciones se dividen en subprocesos que pueden ejecutarse de manera concurrente en cualquier procesador disponible, mejora el rendimiento de la aplicación misma y también el rendimiento total del sistema. Pero lo más importante es que los sistemas simétricos requieren aplicaciones que puedan aprovechar el paralelismo de múltiples subprocesos. Entre los Sistemas Operativos compatibles con el multiprocesamiento simétrico se encuentran UNIX, OS/2 y Windows NT y entre las aplicaciones, el conjunto de aplicaciones Microsoft BackOffice y también los administradores de base de datos SQL Server, Oracle, Sybase e Informix.

Por su parte en el multiprocesamiento asimétrico, hay un procesador (maestro) en el cual se ejecuta el Sistema Operativo, los demás procesadores ejecutan el resto de las tareas. La ventaja de éste es que al aumentar más procesadores se tiene que hacer un cambio mínimo y fácil para el manejo de éstos y en general se eliminan muchos problemas de integridad de datos. La gran desventaja es que al haber sólo una copia del Sistema Operativo en un sólo procesador (maestro) cuando este procesador falla todo el sistema falla porque todos los recursos que son manejados por el Sistema Operativo no pueden ser accedidos.

Dado que la plataforma a utilizar debe de ser confiable, robusta y con un alto rendimiento, para el caso del (SIDUNAM) motivo del presente trabajo, recomendamos seleccionar la arquitectura de multiprocesamiento simétrico ya que mejora el rendimiento de la aplicación así como el rendimiento total del sistema. Hasta aquí, hemos analizado los requerimientos del Servidor sobre el cual se levantará el sistema, por tanto, sugerimos lo siguiente:

- Que el equipo se pueda escalar, tanto en memoria, capacidad en disco duro, así como en el número de procesadores, a medida que crezcan las necesidades del sistema.
- Que acepte multiprocesamiento simétrico.
- Que optimice los procesos de entrada y salida, ya que accesaremos a una base de datos.

De igual forma, considerando el análisis anterior, se realizó una investigación para conocer las marcas de los equipos que mejor se adaptaban a las necesidades del proyecto; el estudio de mercado nos llevó a decidir, que los siguientes equipos cumplieran con los requerimientos técnicos y expectativas que se necesitan para poder llevar a cabo el desarrollo de nuestra aplicación, véanse las Fig. 6.2. y 6.3.



Compaq ProLiant ML 350

Procesador Pentium III 800MHz (capacidad para 2ndo. procesador)

Dos Discos duros de 9.1GB Standard hasta 72.8GB

384 MB SDRAM 133 MHz expandible hasta 2GB

High Speed CD-ROM drive

256-KB de memoria caché integrada

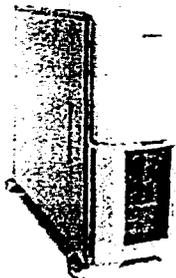
Tarjeta de Red NC3163 Fast Ethernet NIC 10/100 integrada

Fig. 6.2.

Dado que la plataforma a utilizar debe de ser confiable, robusta y con un alto rendimiento, para el caso del (SIDUNAM) motivo del presente trabajo, recomendamos seleccionar la arquitectura de multiprocesamiento simétrico ya que mejora el rendimiento de la aplicación así como el rendimiento total del sistema. Hasta aquí, hemos analizado los requerimientos del Servidor sobre el cual se levantará el sistema, por tanto, sugerimos lo siguiente:

- Que el equipo se pueda escalar, tanto en memoria, capacidad en disco duro, así como en el número de procesadores, a medida que crezcan las necesidades del sistema.
- Que acepte multiprocesamiento simétrico.
- Que optimice los procesos de entrada y salida, ya que accedemos a una base de datos.

De igual forma, considerando el análisis anterior, se realizó una investigación para conocer las marcas de los equipos que mejor se adaptaban a las necesidades del proyecto; el estudio de mercado nos llevó a decidir, que los siguientes equipos cumplían con los requerimientos técnicos y expectativas que se necesitan para poder llevar a cabo el desarrollo de nuestra aplicación, véanse las Fig. 6.2. y 6.3.



Compaq ProLiant ML 350

Procesador Pentium III 800MHz (capacidad para 2ndo. procesador)

Dos Discos duros de 9.1GB Standard hasta 72.8GB

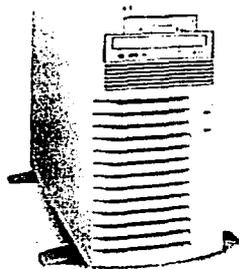
384 MB SDRAM 133 MHz expandible hasta 2GB

High Speed CD-ROM drive

256-KB de memoria cache integrada

Tarjeta de Red NC3163 Fast Ethernet NIC 10/100 Integrada

Fig. 6.2.



El HP netserver E800 es un Servidor con doble procesador Pentium III 800MHz

Dos DD de 9.1GB cada uno

384MB SDRAM, expandible hasta 2GB

High Speed CD-ROM drive

Tarjeta de Red Fast Ethernet 10/100 integrada

Fig. 6.3.

6.4.5. El equipo ideal (Cliente)

La tecnología en general, y especialmente la tecnología informática y de las telecomunicaciones, avanza sin cesar; los equipos compatibles PC, cada vez son más rápidos, tienen más capacidad de almacenamiento y de proceso de la información, pueden realizar más trabajos, nos proporcionan el acceso a grandes cantidades de información y sus precios son accesibles.

En el ámbito educativo de nuestro país, la mayoría de los centros están dotados de equipos compatibles PC, y la mayor parte de los programas didácticos existentes están diseñados para estas máquinas. Por lo antes escrito, consideramos que para el propósito del presente trabajo, se tendrán que aprovechar los equipos PC, existentes hoy en día en las instalaciones de nuestra universidad.

Una vez propuesto el tipo de equipo para poder ejecutar la aplicación en cuestión, queda por determinar las características específicas con las que deberán contar los equipos Cliente, como son: La velocidad con la que procesa la información, cantidad de memoria, capacidad de almacenamiento, el rendimiento gráfico y comunicación con otros equipos.

6.4.5.1. Características deseables en los equipos cliente

Los equipos informáticos actuales son modulares. Están constituidos por múltiples componentes fabricados por distintas empresas pero que funcionan perfectamente adaptados entre sí con la ayuda de las funcionalidades que proporcionan los programas del BIOS (*Basic Input/Output System*) y del Sistema Operativo que se instale (Windows, MS/DOS, Linux, etc.). Esta modularidad facilita en muchos casos su futura ampliación o escalabilidad, permitiéndonos adaptar nuestros equipos a los requerimientos de los nuevos programas mediante la ampliación de su procesador, disco duro y memoria o la conexión de un nuevo periférico. A continuación vamos a analizar las principales características de estos equipos en cuanto al rendimiento que conviene exigir para obtener el máximo aprovechamiento del sistema a realizar.

La velocidad

La velocidad real de trabajo de los equipos está determinada por las características de todos los elementos que configuran el sistema informático, pero depende especialmente de cuatro factores:

a) El tipo de microprocesador (cerebro del equipo), que es capaz de realizar unas determinadas operaciones básicas y dispone de unos registros y buses para conducir la información. En estos momentos los microprocesadores más utilizados son los Pentium III, con registros internos de 64 bits, que comercializa la empresa Intel y los K7 de la empresa AMD de similares características.

Conviene además que el microprocesador esté insertado en un zócalo o *socket* tipo ZIF (*Zero Insertion Force*), que facilitará su sustitución si llega el momento de actualizarlo.

b) La velocidad del reloj interno del microprocesador (se mide en megahercios, MHz), que determina las operaciones que el microprocesador es capaz de realizar cada segundo. Un microprocesador a 900 MHz es capaz de realizar novecientos millones de operaciones cada segundo.

c) La arquitectura de la tarjeta principal que constituye la Unidad Central del equipo y la velocidad de los buses a través de los cuales circula la información.

d) La memoria cache, más rápida que la memoria RAM (*Random-Access Memory*), almacena los datos que utiliza el microprocesador con más frecuencia. Una memoria cache de 512 Kbytes puede ser suficiente.

La memoria y proceso de la información

La capacidad de proceso de la información de un equipo depende especialmente del microprocesador y de la memoria RAM que tenga el equipo. En la memoria RAM es donde se almacenan los programas que se están ejecutando. Un equipo con poca memoria RAM solo podrá ejecutar programas pequeños. Con la llegada del Sistema Operativo Windows y de los entornos multimedia los programas cada vez son más grandes y necesitan más memoria, de manera que los equipos no deberían tener menos 128 MB. de memoria RAM.

Esta memoria debe estar acorde con las características de los buses de la tarjeta principal, y además conviene que sea tipo SDRAM (*Synchronous Dynamic Random Access Memory*) que va en conectores DIMM (*Dual In-line Memory Module*) de 168 contactos que es más rápida que las anteriores memorias EDO (*Extended Data Out*) con conectores SIMM (*Single In-Line Memory Module*) de 72 contactos o SRAM (*Static Random Access Memory*) con conectores SIMM de 30 contactos.

La capacidad de almacenamiento

La capacidad de almacenamiento de los equipos viene dada por las características de las unidades de disco que instalemos. El equipo debe disponer al menos de las siguientes unidades:

a) Unidad lectora de disquetes convencionales de 3½ pulgadas con 1'44 MB. de capacidad máxima (HD, alta densidad), que puede leer también disquetes DD (*Double Density* de 720 K).

b) Disco duro de gran capacidad; por ejemplo un disco duro de 20 G. (1 Gigabyte = 1.000 Megabytes = 1.000.000 Kbytes). De tecnología IDE (*Integrated Drive Electronics*) que si además son Ultra-DMA (*Direct Memory Access*) aumentan muy considerablemente su velocidad de trabajo pudiendo llegar a 66 M/seg. También hay discos duros SCSI (*Small Computer System Interface*) que aún son más rápidos (hasta 80 M/seg), pero su precio es mayor.

c) Lector de CD-ROM (*Compact Disk - Read Only Memory*) o lector de DVD (*Digital Versatile Disc*). Hasta hace muy poco todos los equipos multimedia incluían un lector de CD-ROM, que permite leer discos ópticos de 650 MB., reproducir discos compactos de audio y visualizar fotografías digitales. Una unidad con una velocidad x48 proporciona una transferencia de unos 7.2 M/seg. No obstante hoy en día ya puede ser una mejor opción adquirir un DVD. Los lectores DVD (*Digital Video Disk*) además de leer los discos convencionales de CD-ROM y audio, pueden leer los nuevos discos DVD de 17 G. Un DVD con velocidad x10 proporciona una transferencia de 15 M/seg.

Rendimiento gráfico

El rendimiento gráfico de los equipos depende de dos factores:

a) La tarjeta de video, que controla el monitor, será tipo AGP (**Accelerated Graphics Port**). La tarjeta incluirá 16 MB. o 32 MB. de memoria de vídeo, que asegurará la presentación de imágenes, vídeo y gráficos con una resolución aceptable.

b) El monitor, ha de ser capaz de responder a los requerimientos de la tarjeta de vídeo. Puede ser una pantalla de baja radiación de 15 pulgadas de diámetro y el tamaño de sus puntos será de 0'28 mm. o menor (cuanto menor sea este tamaño mayor será el número de puntos y por lo tanto también será mejor la definición de las imágenes que presente).

Comunicación

Para que nuestro equipo pueda conectarse a una red, necesitamos de un dispositivo electrónico llamado Tarjeta de Red o bien de un Módem, a continuación explicaremos cada uno de éstos:

a) La Tarjeta de Red, es lo que se necesita para conseguir una conexión entre más de un equipo para conformar una red. Normalmente las tarjetas de red modernas para equipos PC, suelen ir conectadas a un *slot* (ranura de expansión) PCI (Peripheral Component Interconnect), y no necesitan de ninguna configuración previa, éstas se conectan a una a una velocidad de 10/100 Mbps; de igual forma las hay antiguas que van conectadas a un *slot* ISA (Industry Standard Architecture), una desventaja es que a éste tipo de tarjetas se le tiene que configurar las interrupciones y las direcciones de Entrada/Salida, pero por otro lado son más baratas que las PCI, éstas se conectan a una a una velocidad de 10 Mbps.

b) El Módem es un dispositivo de Entrada/Salida por medio del cual la computadora puede establecer contacto con otras computadoras a través de la línea telefónica. Para realizar el proceso de comunicación, es necesario que cada computadora tenga su módem el cual debe estar programado con la misma velocidad de transmisión. El término Módem procede de Modulador/Demodulador que resume la función del módem, es decir, los datos que una computadora debe enviar están formados por bits, estos bits se transmiten de uno en uno por el puerto serie al módem,

éste convierte estos datos digitales en señales analógicas de modo que puedan circular por la línea telefónica, modula los datos. El módem que se encuentra en el otro extremo de la línea telefónica y recibe estas señales de frecuencia las convierte en señales digitales, bits, decimos que demodula los datos, y los transmite por el puerto serie de uno en uno al PC. La mayoría de los módems utilizan un grupo de ordenes o comandos denominados comandos Hayes o comandos AT, debido a que todos ellos empiezan con las letras AT (por ejemplo ATDT significa realizar la marcación por tonos o ATDP por pulsos).

En resumen, las características deseables para los equipos Cliente son las siguientes:

- Que el equipo se pueda escalar, tanto en memoria, capacidad en disco duro, así como en el procesador, a medida que crezcan las necesidades del sistema y que éste tenga capacidad de comunicación con otros equipos.

Es importante mencionar que el sistema que se creará puede ser instalado en un equipo de cómputo con un procesador 486 o superior, con una memoria RAM de 32 MB. y un disco duro de 200 MB., lógicamente mientras el equipo Cliente sea más potente, obtendremos un mejor aprovechamiento del sistema a realizar, siendo así que nosotros proponemos el siguiente equipo para los Clientes, véase la Fig. 6.4.



HP VL400

Procesador: Intel Celeron 766 MHZ.
RAM: 32MB 64MB o 128MB. Caché: 512 KB.
Disco Duro: 20 GB.

Fig.6.4.

6.5. Plataformas de *software*

Una vez descrito y seleccionado el *hardware*, pasamos a seleccionar el Sistema Operativo de Red y el Sistema Administrador de Base de Datos Relacional (RDBMS) para el Servidor sobre el que se ejecutará el SIDUNAM; para ello tenemos en cuanto a Sistemas Operativos de Red, dos alternativas que "a priori" parecen válidas para Servidores PC, éstas son Linux y Windows NT; para el caso del RDBMS (*Relational DataBase Management System*), se verán Sybase y SQL Server. A continuación estudiaremos estas herramientas teniendo en cuenta que ambas pueden aprovechar al máximo las tecnologías que se han estudiado a lo largo de este capítulo; por consiguiente describiremos las capacidades de éstas.

6.5.1. Sistemas Operativos de Red

El Sistema Operativo, es el instrumento indispensable para hacer de la computadora un objeto útil. Un Sistema Operativo es el encargado de brindar al usuario una forma amigable y sencilla de operar, interpretar, codificar y emitir las ordenes al procesador central para que este realice las tareas necesarias y específicas para completar una orden.

Un Sistema Operativo (SO) es en sí mismo un programa de computadora. Sin embargo, es un programa muy especial, quizá el más complejo e importante en una computadora. El SO despierta a la computadora y hace que reconozca a la CPU (*Central Processing Unit*), la memoria, el teclado, el sistema de vídeo, las unidades de disco, los puertos y conectores (serie y paralelo) de entrada y salida, los cuales son vías por medio de las cuales la computadora se comunica con los dispositivos externos antes mencionados. Además, proporciona la facilidad para que los usuarios se comuniquen con la computadora y sirve de plataforma a partir de la cual se corran programas de aplicación.

Los Sistemas Operativos de Red empleados normalmente son UNIX, Linux, Macintosh OS, OS/2, Windows 2000 y Windows NT. El UNIX y sus clones permiten múltiples tareas y múltiples usuarios. Otros SO multiusuario y multitarea son OS/2, desarrollado inicialmente por Microsoft e IBM, Windows NT y Windows 2000 desarrollados por Microsoft. El SO multitarea de Apple se denomina Macintosh OS.

Como se dijo anteriormente analizaremos los Sistemas Operativos de Red Linux y Windows NT.

6.5.1.1. Linux

Linux es una versión de Unix libremente distribuible e independiente, para plataformas con procesadores X86, Motorola 68k, Digital Alpha, Sparc, Mips y Motorola Power PC.

En la actualidad, este Sistema Operativo es utilizado por miles de usuarios para desarrollo de *software*, redes y para plataformas de usuarios finales. Linux, entre los miles de Sistemas Operativos alternos que existen, se ha convertido en una opción interesante, independientemente de que estas vengan de UNIX o de las más conocidas donde se encuentra Windows 2000 y NT.

Linux está disponible en Internet en cientos de Servidores FTP, pues el programa es libre. Algunas distribuciones son: Caldera, Debian, Slackware, Red Hat, etc.

Linux, maneja los archivos de forma jerárquica, de la misma forma que DOS (*Disk Operating System*), con la diferencia que el DOS está diseñado para procesadores X86 que no soportan verdaderas capacidades de múltiples tareas.

a) Ventajas

1. Precio.
2. Estabilidad, no se traba a cada rato.
3. Seguridad, es mucho más seguro que otros Servidores.
4. Compatibilidad, reconoce la mayoría de los otros Sistemas Operativos en una red.
5. Velocidad, es mucho más veloz para realizar las tareas.
6. Posee el apoyo de miles de programadores a nivel mundial.
7. El paquete incluye el código fuente, lo que permite modificarlo de acuerdo a las necesidades del usuario.
8. Ideal para la programación, ya que se puede programar en Linux para distintas plataformas.
9. Un sistema de crecimiento rápido.
10. Se puede usar en casi cualquier computadora, desde una 386.
11. Multitareas REAL.
12. Puede manejar múltiples procesadores. Incluso hasta 16 procesadores.
13. Casi libre de virus (intentos de virus), se conoce un virus para Linux.
14. Maneja discos duros de hasta 16 TeraBytes.
15. Se consiguen parches con facilidad, además de ser gratuitos.
16. Se posee el apoyo de millones de usuarios a nivel mundial.
17. Los fabricantes de *Hardware* le están dando su apoyo, como IBM y COMPAQ.
18. Vendedores y desarrolladores implementan un sistema de certificación para Linux.
19. La corporación DATA Internacional predice que el crecimiento de este programa será del orden de un 25 por ciento anual en el nuevo milenio.

b) Desventajas

- Linux no cuenta con una empresa que lo respalde, por lo que no existe un verdadero soporte como el de otros Sistemas Operativos.

- Linux corre el riesgo de llegar a fragmentarse como fue el caso de UNIX.
- Algunas empresas pueden llegar a ayudar a Linux con la intención de mejorar sus relaciones públicas, aunque en el fondo no tengan ninguna intención de utilizarlo fielmente.

6.5.1.2. Windows NT

Con Windows NT, Microsoft ha expresado su dedicación a escribir *software* no sólo para PC de escritorio sino también para poderosas estaciones de trabajo y Servidores de Red y Bases de Datos. Microsoft Windows NT no es necesariamente un sustituto de DOS ni una nueva versión de éste; es, en conjunto, un nuevo SO diseñado desde sus bases para las máquinas más modernas y capaces disponibles.

Windows NT de Microsoft ofrece características propias que ningún otro SO para PC ofrece, con excepción de Unix. Además de las características tradicionales de estricta seguridad de sistema, red interconstruida, servicios de comunicación y correo electrónico interconstruidos, herramientas de administración y desarrollo de sistema y una GUI (Graphical User Interface), Windows NT puede correr directamente aplicaciones de Windows de Microsoft y de Unix.

Windows NT, es un SO de 32 bits, que puede hacer completo uso de los procesadores de estas características. Esto le permite sacar ventaja de los microprocesadores más avanzados, como los Intel 80486 y posteriores, así como de las computadoras con tecnología RISC. Además de ser multitarea, está diseñado para tomar ventaja del multiproceso simétrico.

El rendimiento general, es decir, la combinación del rendimiento del microprocesador, transferencia de datos y acceso a memoria es, en definitiva, lo que este Sistema Operativo ha venido a mejorar sensiblemente.

- Linux corre el riesgo de llegar a fragmentarse como fue el caso de UNIX.
- Algunas empresas pueden llegar a ayudar a Linux con la intención de mejorar sus relaciones públicas, aunque en el fondo no tengan ninguna intención de utilizarlo fielmente.

6.5.1.2. Windows NT

Con Windows NT, Microsoft ha expresado su dedicación a escribir *software* no sólo para PC de escritorio sino también para poderosas estaciones de trabajo y Servidores de Red y Bases de Datos. Microsoft Windows NT no es necesariamente un sustituto de DOS ni una nueva versión de éste; es, en conjunto, un nuevo SO diseñado desde sus bases para las máquinas más modernas y capaces disponibles.

Windows NT de Microsoft ofrece características propias que ningún otro SO para PC ofrece, con excepción de Unix. Además de las características tradicionales de estricta seguridad de sistema, red interconstruida, servicios de comunicación y correo electrónico interconstruidos, herramientas de administración y desarrollo de sistema y una GUI (Graphical User Interface), Windows NT puede correr directamente aplicaciones de Windows de Microsoft y de Unix.

Windows NT, es un SO de 32 bits, que puede hacer completo uso de los procesadores de estas características. Esto le permite sacar ventaja de los microprocesadores más avanzados, como los Intel 80486 y posteriores, así como de las computadoras con tecnología RISC. Además de ser multitarea, está diseñado para tomar ventaja del multiproceso simétrico.

El rendimiento general, es decir, la combinación del rendimiento del microprocesador, transferencia de datos y acceso a memoria es, en definitiva, lo que este Sistema Operativo ha venido a mejorar sensiblemente.

La multitarea significa que el Sistema Operativo puede realizar varias tareas al mismo tiempo. En Windows NT existe la posibilidad de concurrencia de diversas tareas, de manera que si, por ejemplo, una de ellas queda parcialmente parada como consecuencia de un lento acceso al disco, el microprocesador puede concentrar su atención en otras tareas. Básicamente, lo que se consigue es que no se malgasten ciclos de procesamiento. El beneficio de esto es que los usuarios pueden realizar otras tareas mientras tiene lugar un acceso al disco en segundo plano, o bien, se prepara un trabajo de impresión.

El multiprocesamiento simétrico es una característica exclusiva de Windows NT, que permite sacar todo el partido de los procesadores múltiples. Windows NT es el primer Sistema Operativo que está preparado para realizar un uso eficaz de este tipo de tecnología. Los Sistemas Operativos anteriores se limitaban a asignar tareas específicas a cada microprocesador individual, como por ejemplo, para la Entrada/Salida en una red local.

El multiprocesamiento asimétrico, en donde un microprocesador se dedica exclusivamente a una tarea específica, da lugar a que éste permanezca inactivo en cuanto finaliza su tarea. En el multiprocesamiento simétrico, el Sistema Operativo puede asignar diferentes tareas a un mismo microprocesador; de esta manera, si uno de ellos termina su trabajo antes que otro, el Sistema Operativo podrá ocuparlo en otra actividad. por esto, ofrece un rendimiento superior.

Las características de trabajo en red local con las que cuenta Windows NT permiten compartir archivos de un sistema con cualquier usuario en un entorno de red, así como conectarse a directorios compartidos con otros sistemas.

El Sistema Operativo de Red Windows NT presenta una arquitectura del tipo Cliente/Servidor (ésta arquitectura la analizaremos ampliamente en el próximo capítulo), ya que es un requisito indispensable que tendrá que cumplir el sistema a desarrollar en el presente trabajo.

En cuanto a la seguridad en Windows NT, se tienen dos cosas básicamente: El control total en el acceso al sistema y a los archivos o subdirectorios que hay en el sistema.

En el primer punto, el control sobre el acceso al sistema se refiere al manejo de user names y passwords para poder acceder al Sistema Operativo, de esta manera se mantienen a los usuarios sin autorización fuera del sistema. El siguiente nivel de seguridad en cuanto a este punto se refiere, son los privilegios que tiene un usuario, todos los usuarios o grupos de usuarios a los directorios y archivos del sistema, por ejemplo, el acceso a los archivos del sistema de NT está estrictamente limitado al Administrador del Sistema, mientras que las aplicaciones comunes como lo son hojas de cálculo o procesadores de palabras pueden ser accesados por todos los usuarios.

Windows NT cuenta con un extenso sistema de control de seguridad para el acceso a archivos. El propósito de la seguridad en Windows NT es brindarle el acceso sólo a aquellos usuarios que están autorizados, controlar el acceso concurrente a archivos, a los directorios y a los recursos del sistema. La seguridad en los sistemas Windows NT debe ser configurada por el Administrador del Sistema siendo necesario para todos los sistemas un Administrador (incluyendo los sistemas monousuarios). El Administrador establece los nombres de usuario, crea grupos de usuarios, asigna los usuarios a los grupos, controla los passwords, permite los niveles de acceso a las funcionalidades del sistema; en pocas palabras el Administrador controla todos los puntos de acceso al sistema.

El Administrador puede controlar el acceso específico a ciertas funciones del sistema, especialmente aquellas que afectan el funcionamiento del mismo, este sistema de control es llamado la política de derechos del usuario. De esta manera el Administrador a través de esta política puede controlar las labores que efectúa un usuario tanto local como remotamente.

La Integridad del sistema en NT.- Entendamos por integridad del sistema a la habilidad del mismo de permanecer activo cuando una de sus aplicaciones falla. Windows NT está diseñado para prevenir la caída catastrófica del sistema en caso de que algunas de sus aplicaciones fallen y para esto establece los siguientes cuatro mecanismos de protección de memoria:

1) Espacio de direcciones separado: Cada proceso maneja sus propias direcciones virtuales y el sistema prohíbe el acceso a espacios de memoria de otros procesos. 2) Modos de Kernel y usuarios separados: Todas las aplicaciones corren en modo de usuario, por lo tanto, está prohibido el acceso o modificación del código o datos del sistema que residan en el kernel. 3) Banderas de páginas: Cada página de la memoria virtual tiene una bandera la cual determina cómo puede ser accesada en modo usuario y en modo kernel. 4) Seguridad de los Objetos: El manejador virtual de la memoria crea un tipo especial de objeto llamado objeto-sección el cual funciona como una ventana hacia la memoria virtual, por lo tanto, cada vez que un proceso accesa un objeto-sección el sistema determina si el proceso tiene los permisos de lectura y/o escritura sobre éste. Dentro de la integridad del sistema Windows NT establece políticas y procedimientos de protección, el acceso a recursos de esta manera protege a los procesos de caer en estados muertos cuando compiten por recursos.

Windows NT Server, cuenta con el Microsoft Internet Information Server que es un Servidor Web que nos permite publicar información en una Intranet de la organización o en Internet. El IIS (*Internet Information Server*) transmite la información mediante el Protocolo de Transferencia de Hipertexto (HTTP). Internet Information Server (IIS) puede configurarse también para proporcionar servicios de Protocolo de Transferencia de Archivos (FTP) y Gopher. El servicio FTP permite que los usuarios transfieran archivos a y desde su sitio Web. El servicio Gopher utiliza un protocolo controlado por menús para encontrar documentos. El protocolo Gopher ha quedado en buena parte reemplazado por el protocolo HTTP (*HyperText Transfer Protocol*). El propósito de Windows NT es el de ser un Sistema Operativo diseñado para correr en distintas plataformas soportando los siguientes procesadores:

- La familia Intel X86
- De motorola 680x0
- El MIPS 400
- El ALFA de Dec.
- El HP-PA de Hewlett Packard
- Los SPARC RISC processors de Sun Microsystems.
- El RS/6000 de IBM
- Unas futuras versiones del Powerpc (Apple, IBM y Motorola).

La independencia de plataforma está basada en el concepto de desarrollar un kernel específico para cada uno de los distintos procesadores que sirva de interfaz entre el *hardware* específico y las llamadas al sistema de NT.

Desde hace unos años atrás está existiendo un debate continuo en el campo de los Servidores: Linux o NT. Ambas plataformas, Linux y NT son totalmente opuestas en todos los niveles: Visualmente son diferentes, corren de manera diferente, soportan diferente *hardware*, y tienen costo diferente, véase la Fig. 6.5.

Comparación de precios

Software	Windows NT	Linux
Sistema Operativo	Windows NT Server con 10 usuarios: \$981	Red-Hat Linux con usuarios ilimitados: \$50
Servidor Web	IIS: \$0 (dentro de él),	Apache: \$0 (dentro de él)
Servidor para Base de Datos	Microsoft SQL Server: \$1075	PostgreSQL: \$0, Sybase: \$50 (Gratis con RedHat)
Servidor Proxy (ilimitado)	WinGate: \$949	IP-Masq: \$0 (dentro de él)
Total:	\$3005	\$50

Fig.6.5.

La más obvia diferencia entre NT y Linux es que la interfaz de NT con su Servidor Web IIS intenta traer el interfaz de Windows, más familiar con la mayoría de

usuarios, al entorno de Servidores. Idealmente un Webmaster podría mantener un sistema NT usando su ratón y haciendo click . Lo que podemos ver sin discusión es su facilidad de manejo. Sí, es un Sistema Operativo muy sencillo de usar, quien tenga bases en la administración de una red de computadoras lo puede manejar perfectamente sin problemas, y todo gracias a su entorno gráfico que con el mouse maneja casi todo, lanzar programas, minimizar, maximizar y cerrarlos con solo apretar con el mouse en los respectivos íconos que se encuentran sobre la ventana. Prueba de ello, es su uso masivo, alrededor del 90% a nivel mundial.

Ahora bien, dada la magnitud del sistema a desarrollar, se requiere de un Sistema Operativo robusto; capaz de soportar a un gran número de usuarios en forma simultanea; que maneje adecuadamente la memoria y el multiprocesamiento simétrico, y, sobre todo, lo más importante, que aproveche al máximo el equipo Cliente (con la interfaz gráfica Windows 3.11, 95, 98, 2000 y NT), existente hoy en día, en prácticamente todas las áreas administrativas de nuestra universidad. Es por esto, que nosotros proponemos que el Sistema Operativo de red Windows NT Server sea el elegido para poder desarrollar bajo esta plataforma el SIDUNAM.

Haciendo alusión a la figura 6.5., que compara las características principales de los Sistemas Operativos Linux y NT, podemos decir que; en cuanto a costo definitivamente Linux es la mejor opción, pero por otro lado, debemos estar concientes de que Linux no cuenta con un verdadero soporte como es el caso de Windows NT, así también, sabemos que NT es el primer Sistema Operativo que está preparado para realizar un uso eficaz del multiprocesamiento simétrico, por lo tanto, NT permite el aprovechamiento al máximo de los procesadores múltiples; es por todo esto que insistimos en seleccionar a Windows NT como el Sistema Operativo para poder llevar a cabo nuestro proyecto.

Una vez identificados y analizados los posibles Sistemas Operativos a utilizar, se requiere hacer también un análisis de los Sistemas Administradores de Base de Datos Relacionales (RDBMS), que se acoplan a los Sistemas Operativos antes mencionados; estos RDBMS¹⁵ deben de ser robustos y tienen que manejar altos niveles de seguridad,

usuarios, al entorno de Servidores. Idealmente un Webmaster podría mantener un sistema NT usando su ratón y haciendo click . Lo que podemos ver sin discusión es su facilidad de manejo. Si, es un Sistema Operativo muy sencillo de usar, quien tenga bases en la administración de una red de computadoras lo puede manejar perfectamente sin problemas, y todo gracias a su entorno gráfico que con el mouse maneja casi todo, lanzar programas, minimizar, maximizar y cerrarlos con solo apretar con el mouse en los respectivos iconos que se encuentran sobre la ventana. Prueba de ello, es su uso masivo, alrededor del 90% a nivel mundial.

Ahora bien, dada la magnitud del sistema a desarrollar, se requiere de un Sistema Operativo robusto; capaz de soportar a un gran número de usuarios en forma simultanea; que maneje adecuadamente la memoria y el multiprocesamiento simétrico, y, sobre todo, lo más importante, que aproveche al máximo el equipo Cliente (con la interfaz gráfica Windows 3.11, 95, 98, 2000 y NT), existente hoy en día, en prácticamente todas las áreas administrativas de nuestra universidad. Es por esto, que nosotros proponemos que el Sistema Operativo de red Windows NT Server sea el elegido para poder desarrollar bajo esta plataforma el SIDUNAM.

Haciendo alusión a la figura 6.5., que compara las características principales de los Sistemas Operativos Linux y NT, podemos decir que; en cuanto a costo definitivamente Linux es la mejor opción, pero por otro lado, debemos estar concientes de que Linux no cuenta con un verdadero soporte como es el caso de Windows NT, así también, sabemos que NT es el primer Sistema Operativo que está preparado para realizar un uso eficaz del multiprocesamiento simétrico, por lo tanto, NT permite el aprovechamiento al máximo de los procesadores múltiples; es por todo esto que insistimos en seleccionar a Windows NT como el Sistema Operativo para poder llevar a cabo nuestro proyecto.

Una vez identificados y analizados los posibles Sistemas Operativos a utilizar, se requiere hacer también un análisis de los Sistemas Administradores de Base de Datos Relacionales (RDBMS), que se acoplan a los Sistemas Operativos antes mencionados; estos RDBMS^s deben de ser robustos y tienen que manejar altos niveles de seguridad,

mecanismos de respaldo, entre otras características. De tal manera que a continuación, analizaremos los RDBMS[®], basados en los Sistemas Operativos antes descritos.

6.5.2. Sistema Administrador de Base de Datos Relacional (RDBMS)

Empezaremos dando las características principales de un RDBMS: 1) Representación de los datos por medio de tablas relacionadas. 2) Utiliza lenguaje de cuarta generación. 3) Flexibilidad: La modificación de los datos y los cambios a la estructura de la base de datos resulta muy sencilla. Además el RDBMS es el responsable de: 1) Mantener las relaciones entre la información y la base de datos. 2) Asegurarse de que la información es almacenada correctamente, es decir, que las reglas que definen las relaciones entre los datos no sean violadas. 3) Recuperar toda la información en un punto conocido en caso de que el sistema falle. Por lo tanto, procederemos a realizar un análisis de los RDBMS siguientes:

6.5.2.1. Sybase

Sybase es una base de datos relacional diseñada para aplicaciones críticas y en línea, que requieren un alto rendimiento escalable, se basa en el modelo relacional. El lenguaje de consultas básicas es SQL. Sybase emplea una arquitectura *multithreaded* (multitareas) en la que la base de datos, más que el Sistema Operativo, es la que programa y controla las tareas para obtener una máxima eficiencia. Múltiples sentencias SQL pueden aumentarse con la programación de constructores, tales como lógica condicional, llamadas a procedimientos y variables locales, estos pueden combinarse en un objeto de base de datos llamado un procedimiento de almacenamiento. Los procedimientos pueden regresar hileras de datos y mensajes de error, además de regresar valores en variables de programación en el programa de aplicación. El Servidor Sybase también soporta disparadores como objetos independientes, estos tienen las capacidades de los procedimientos con tres extensiones importantes.- 1) Ellos no pueden ejecutarse directamente, sólo responden al cumplimiento de una condición. 2) Un disparador puede restaurar o modificar los

resultados de una transacción del usuario. 3) El disparador puede ver los cambios hechos a los datos.

Los procedimientos de almacenamiento (*stores procedures*) y los disparadores (*triggers*) convierten a Sybase en un Servidor programable.

Las rutinas de mantenimiento de la base de datos son realizadas mientras que el sistema está totalmente operando. Sybase también se protege contra fallas del sistema con su propio mecanismo de tolerancia a fallas implantadas en el *software*. La interoperabilidad de Sybase permite a cualquier aplicación Cliente trabajar cooperativamente con cualquier fuente de datos a través de diferentes plataformas de *hardware*, red local y Sistema Operativo. El API (*Application Program Interface*) del *Open Client* de Sybase es un método bastante simple y flexible para acceder una gran variedad de fuentes de datos. El *Open Server* de Sybase permite a cualquier fuente de datos responder a las peticiones de los Clientes y comportarse como si fuera un Servidor inteligente. Los procedimientos remotos de Sybase RPC's (*Remote Procedure Calls*) son un mecanismo de mensajes para la comunicación entre cualquier Cliente y el Servidor.

6.5.2.2. MS SQL Server

El MS SQL Server es un sistema Administrador de Base de Datos Relacional inteligente, basado en la arquitectura Cliente/Servidor (como se mencionó ésta arquitectura se analizará a fondo en el próximo capítulo). Entre las ventajas que tiene MS SQL Server están su perfecta integración con el Sistema Operativo Windows NT Server. Este Servidor de Base de Datos es un programa que almacena datos estructurados en forma de tablas relacionales, escucha un puerto TCP/IP a través del cual acepta conexiones de Clientes autenticados, admite comandos en lenguaje SQL y devuelve al Cliente a través de la red los datos resultantes del procesamiento de los comandos SQL, es decir, SQL Server puede recibir conexiones (Clientes que desean manipular datos) desde cualquier equipo conectado a Internet.

La seguridad y garantía del procesamiento de transacciones en SQL Server lo hacen ideal para las más exigentes aplicaciones de misión crítica. Las capacidades avanzadas de SQL Server tales como: Triggers, Procedimientos Almacenados, Reglas, Defaults y su tecnología de llamado a procedimiento remoto (RPC), hacen de SQL Server la herramienta ideal para el desarrollo de aplicaciones dorsales (*back-ends*) en estaciones de trabajo, minicomputadoras y mainframes. Su arquitectura diseñada para el alto rendimiento, multitarea, asegura un rendimiento consistente y estable que se mantiene aún cuando el número de usuarios aumenta. Los puentes SQL Server (*gateways*), también proveen la tecnología clave para la integración de microcomputadoras con minicomputadoras y mainframes de larga escala. Una de las grandes ventajas de SQL Server para Windows NT es que al igual que la mayoría de los sistemas desarrollados por la compañía Microsoft, se ejecutan sobre ambientes gráficos que ayudan tanto al Cliente para el desarrollo de aplicaciones, como al Administrador para tener mejor controlado el sistema.

Características principales de SQL Server:

- Administración remota de los recursos del sistema.
- Arquitectura de 32 bits.
- Capacidad para generación automática de copias de seguridad.
- Ejecución de procedimientos almacenados en forma remota.
- Un robusto soporte de tolerancia a fallas.
- El reforzamiento centralizado de la integridad de los datos minimiza el mantenimiento, programación y reduce los costos, asegurando que los datos son correctos.
- Interoperabilidad dinámica con la red local.
- Monitoreo gráfico del rendimiento del sistema.
- Interfaces de programación remota como: **DB-Library** y **ODBC**.
- Un complejo y robusto manejo de transacciones con capacidad de *roll-forward* y *roll-back* que permiten la recuperación automática de transacciones al reinicio del sistema.

En el pasado, las diferencias entre las distintas clases de *software* y *hardware* eran relativamente claras: *mainframe*, minicomputadora y PC. En la medida en que el *software* RDBMS se ha vuelto más poderoso y el *hardware* basado en PC para multiprocesamiento simétrico de alto rendimiento (SMP) más económico, los usuarios han encontrado mejores opciones para elegir la mejor plataforma de aplicaciones Cliente/Servidor. La razón más importante para elegir Microsoft SQL Server es la escalabilidad.

Como SQL Server se basa en un núcleo paralelo de base de datos *multithreaded* (multitareas), puede aprovechar procesadores adicionales. Hasta hace muy poco, los sistemas de *hardware* de multiprocesamiento eran costosos y complejos. En muchos casos era necesario emplear paralelamente *software* especial de Bases de Datos y Sistemas Operativos para obtener un mayor soporte de multiprocesamiento simétrico (SMP). En los últimos cinco años ha aumentado el poder del procesamiento disponible para los usuarios en el *hardware* de bajo costo. Hoy en día, SMP es una arquitectura dominante. SQL Server se enfoca a estas plataformas, porque predecimos que este tipo de *hardware* será el predominante en los próximos años.

En combinación con sistemas de *hardware* poderosos y de bajo costo de HP, Compaq, Digital y otras marcas, Microsoft SQL Server y Windows NT Server proporcionan un excelente rendimiento.

En resumen, después de haber analizado y propuesto la plataforma a utilizar para el desarrollo del SIDUNAM, a lo largo del presente capítulo; concluimos que la plataforma que mejor se adapta a los requerimientos técnicos y económicos es el Servidor HP E800, con el Sistema Operativo de Red Windows NT Server versión 4.0 y como Manejador de Base de Datos se seleccionó a MS SQL Server en su versión 7.0., véase la Fig. 6.6., que esquematiza el equipo Servidor a utilizar:

HP NetServer E800

Hardware

- El HP netserver E800 es un Servidor con doble procesador Pentium III 800MHz
- Dos DD de 9.1GB cada uno
- 384MB SDRAM, expandible hasta 2GB
- High Speed CD-ROM
- Unidad de Cinta

Software

- MS Windows NT 4.0
- MS SQL Server 7.0

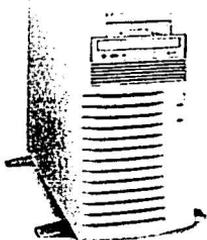


Fig. 6.6.

Por lo tanto, el **software** del Cliente a utilizar, tiene que ser totalmente compatible con el **software** seleccionado hasta este momento para el Servidor, a fin de garantizar la operación y manejo del SIDUNAM por parte de los usuarios Cliente; de igual forma proponemos que el Sistema Operativo sea un ambiente gráfico como lo es Windows, así que nuestro Cliente sería el siguiente, véase la Fig. 6.7.

HP VL400

Hardware

- Procesador: Intel Celeron 766 MHZ.
- RAM: 32MB, 64MB o 128MB. Cache: 512 KB.
- Disco Duro: 10 GB.

Software

- Windows (3.11, 95, 98, 2000 Millenium o NT).



Fig. 6.7.

En el siguiente capítulo analizaremos la arquitectura Cliente/Servidor, el Lenguaje de Programación para el desarrollo del sistema, el cual deberá ser perfectamente compatible con el Sistema Operativo y con el RDBMS, así también, se

creará la Base de Datos del sistema, se analizará la forma de acceder a los datos de la misma, se verá la seguridad en el sistema y la distribución del mismo.

VII. DESARROLLO E IMPLANTACIÓN DEL SISTEMA (SIDUNAM)

En el capítulo anterior, se analizó la infraestructura tecnológica existente para poder decidir sobre que plataformas de *hardware* y *software*, se realizaría el sistema, ahora el presente capítulo tratará lo relacionado a la arquitectura Cliente/Servidor, de aquí en adelante (C/S); ya que uno de los requisitos para llevar a cabo este proyecto, es que el manejo del SIDUNAM, sea sobre un ambiente Cliente/Servidor, además de que se pueda acceder a la información vía RedUNAM e Internet. De igual manera en éste capítulo se trata lo referente al lenguaje de programación utilizado para el desarrollo del sistema, seguridad del mismo y su distribución vía Internet.

7.1. Arquitectura Cliente/Servidor (C/S)

En el pasado capítulo se dijo que Windows NT y SQL Server, están basados en la arquitectura C/S, esto nos permite separar la carga de trabajo en tareas que corran en computadoras tipo Servidor y tareas que corran en computadoras tipo Cliente:

- El Cliente es responsable de la parte lógica y de presentar la información al usuario.
- El Servidor distribuye los recursos disponibles (tales como procesadores, memoria, operaciones de disco, etc.) entre las múltiples peticiones del Cliente. Véase la Fig. 7.1.

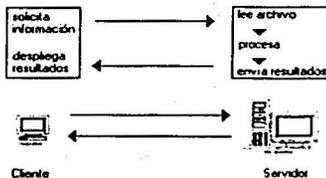


Fig. 7.1.

Además la arquitectura C/S, es el nuevo paradigma de la programación. Dentro de poco todo será C/S, como antes lo fue "orientado a objetos". La programación Cliente/Servidor se utiliza cuando queremos realizar aplicaciones que utilicen redes y que comuniquen entre sí a varios equipos de cómputo. Ambas partes de la aplicación se comunican utilizando algún protocolo de red, en nuestro caso, el protocolo es el TCP/IP, que es el estándar en la UNAM. La justificación de este paradigma es la minimización del tráfico de red. El ancho de banda de las redes de datos, sobre todo de Internet, es actualmente el recurso más precioso por lo escaso y caro que es. Si hay que optimizar un programa para algo, ese algo es utilizar el menor tráfico de red posible para evitar que la red se haga lenta y de esta manera economizar ancho de banda.

La arquitectura computacional Cliente/Servidor es la primera solución en tecnología informática que satisface las presiones de costo-rendimiento de una empresa actual de alto nivel. Por esto se observa una clara tendencia de migración hacia este modelo.

Los motivos que nos llevaron a evaluar la posibilidad de implantar ésta arquitectura Cliente/Servidor dentro del desarrollo del sistema pueden ser varios, pero los principales pueden resumirse de la siguiente manera:

- Flexibilidad para satisfacer las necesidades del negocio.
- Reducción de costos, en cuanto a la adquisición y aprovechamiento del equipo de cómputo y comunicaciones.
- Identificación de la solución que se adecuó más a las necesidades de la empresa y que crezca a la par de los requerimientos.
- Incremento a la productividad del personal.
- Aprovechamiento del equipo existente que no se utiliza al 100% de su capacidad.
- Consolidación de los procesos más confiables, ya que los recursos de cómputo pueden distribuirse en varios Servidores y los Clientes pueden configurarse de modo tal que cualquier aplicación o archivo de datos

pueda ser accesado desde cualquier parte. Además, el aislamiento a fallas que se pueden incluir en este ambiente reduce la pérdida de tiempo.

Una ventaja adicional en el caso del proyecto en cuestión, es que se puede obtener un aprovechamiento total de los recursos dado que todos los equipos (Servidores y Clientes), a pesar de no estar físicamente en el mismo lugar, se encuentran interconectados a través de **RedUNAM**, lo cual hace a la Dirección General de Actividades Deportivas y Recreativas de la UNAM ponerse a la vanguardia en el uso y explotación de los canales de comunicación.

Ventajas de la Arquitectura Cliente/Servidor

- Con la arquitectura Cliente/Servidor se pueden integrar diferentes tipos de equipos y Sistemas Operativos en un ambiente único de procesamiento.
- Anteriormente, la capacidad de cómputo de una empresa se medía directamente por la capacidad del Mainframe o de las Minis con las que contaba. En el ambiente Cliente/Servidor la capacidad está construida por todos los equipos interconectados, entre los que se incluyen las PC's.
- Otra ventaja de esta arquitectura es la considerable reducción en el tráfico de la red, ya que las operaciones de procesamiento se jerarquizan y la red da prioridad a las necesidades del usuario, transmitiéndole sólo la información realmente útil. Esto constituye un estímulo al uso de sistemas abiertos dado que tanto Clientes como Servidores corren en diferentes plataformas de *hardware* y *software*, lo cual permite que las empresas compren productos de diferentes proveedores, sin interferir en el desempeño de las demás aplicaciones y equipos instalados en la red.

El ambiente Cliente/Servidor permite una estructuración, para que los recursos de cómputo se utilicen de la mejor manera posible, con lo cual se reducen los costos de manera considerable. Para lograr esto, se deben llevar a cabo ciertas medidas, las cuales se mencionan a continuación:

1. Separación de Tareas.- Las funciones de cómputo se pueden sincronizar con los recursos de tal manera que el Servidor procese sólo las que son más complejas o que se requieren operaciones intensas, liberando al Cliente de esa labor, dándole la oportunidad de ejecutar otras tareas simultáneas. Con esta configuración puede integrarse una estrategia que equilibre la combinación y costo de los dispositivos de cómputo.
2. Comportamiento de Periféricos.- La combinación de estos dispositivos, también puede realizarse al hacer que sus características coincidan con las funciones que se desea ejecutar y como la arquitectura C/S es abierta, permite la incorporación de un dispositivo especializado para realizar tareas muy específicas.
3. Compartimiento de Herramientas.- Los Clientes pueden compartir diversas aplicaciones con la gran ventaja de que éstas pueden ser controladas de manera centralizada por un mismo Administrador. Con esta ventaja, se elimina la necesidad de tener que instalar, configurar y dar de alta la aplicación en cada Cliente cada vez que surja una nueva versión de *software*.
4. Acceso a la Información.- A pesar de que una empresa se encuentre geográficamente separada, la arquitectura C/S permite el acceso a la información a nivel del Servidor de manera transparente, siempre y cuando existan estándares comunes.

En el caso de la Dirección General de Actividades Deportivas y Recreativas (DGADyR), se pretende implantar un estándar para el sistema a crear, en cuanto al *software* que van a utilizar tanto el Servidor como los equipos Cliente. Esto es, se eligió como estándar una familia de productos que contara con una interfase gráfica, sencilla y amigable, para Administradores y Usuarios finales, tal como el ambiente Windows.

Por tal motivo, como ya se mencionó en el capítulo anterior, el *software* ha utilizar para éste proyecto es de un sólo fabricante, la empresa Microsoft. En las estaciones Cliente, deberá instalarse MS Windows para Trabajo en Grupo 3.11 ó MS Windows 95/98/2000; mientras que los Servidores contarán con MS Windows NT

Server 4.0 y MS SQL Server 7.0 como Sistema Operativo de Red y Manejador de Base de Datos, respectivamente.

La esencia de la arquitectura C/S, está precisamente en que se aprovecha lo que ya se tiene, con la única variante de que ahora se contará con nuevas funcionalidades.

En esta arquitectura, el Cliente maneja la interfase del usuario, construye los requerimientos del Servidor y controla la presentación de resultados, incluso puede contener la lógica de la aplicación. Cuando el Servidor recibe una solicitud de servicio, realiza el proceso, regresa al resultado al Cliente y provee el servicio de administración de datos. Tanto el Cliente como el Servidor pueden ejecutar el trabajo, así como la lógica de la aplicación. De esta manera, el sistema a desarrollar para la Dirección General de Actividades Deportiva y Recreativas, se desempeñará como se describe a continuación: el Servidor se encarga de ejecutar todos aquellos procesos que para el Cliente podrían representar mucho tiempo de proceso, de tal forma que el Servidor se encarga de realizar el trabajo pesado y el Cliente se dedica a presentar los resultados a los usuarios finales.

Desventajas de la Arquitectura Cliente/Servidor

- Hay una alta complejidad tecnológica al tener que integrar una gran variedad de productos. Por una parte, el mantenimiento de los sistemas es más difícil pues implica la interacción de diferentes partes de hardware y de software, distribuidas por distintos proveedores, lo cual dificulta el diagnóstico de fallas.
- Es más difícil asegurar un elevado grado de seguridad en una red de Clientes y Servidores que en un Sistema Centralizado. Se deben hacer verificaciones en el Cliente y en el Servidor. También se puede recurrir a otras técnicas como el encriptamiento.

- En un ambiente Cliente/Servidor, los recursos de cómputo de la red enfrentan cargas de trabajo muy grandes respecto a los Sistemas Centralizados, por lo que el rendimiento no es el mismo.

Por lo que respecta al SIDUNAM, la arquitectura Cliente/Servidor arrojará resultados muy favorables, ya que se va a explotar la infraestructura de RedUNAM al máximo, a su vez se pretende que el Servidor se encuentre dentro de las instalaciones de la (DGADyR), y que los Clientes se encuentren en cualquier parte, no importando que sea fuera del campus universitario.

7.2. Lenguaje de programación

Ya que seleccionamos el Sistema Operativo de Red y el Manejador de Base de Datos Relacional, nos queda por elegir la herramienta de desarrollo, para lo cual, ocuparemos MS Visual Basic 6.0, que como las herramientas anteriores son de la misma empresa Microsoft, así aseguramos una perfecta integración entre las mismas y por lo tanto, tomar ventajas de sus características al desarrollar el sistema SIDUNAM. A continuación estudiaremos las características más sobresalientes de esta herramienta visual de desarrollo.

7.2.1. MS Visual Basic 6.0

MS Visual Basic 6.0 es uno de los Lenguajes de Programación que más entusiasmo despierta entre los programadores de PC's. Visual Basic 6.0 es también llamado lenguaje de cuarta generación (4GL), esto quiere decir, que un gran número de tareas se realizan sin escribir código, simplemente con operaciones gráficas realizadas con el ratón sobre la pantalla. Visual Basic 6.0 es también un programa basado en objetos, aunque no orientado a objetos como C++ o Java. La diferencia está en que Visual Basic utiliza objetos con propiedades y métodos, pero carece de mecanismos de herencia y polimorfismo propios de los verdaderos lenguajes orientados a objetos como C++ o Java.

Visual Basic, es un programa orientado a eventos, es decir, cuando Visual Basic se ha ejecutado, lo único que hace es quedarse a la espera de las acciones del usuario, que en este caso son llamadas **eventos**. El usuario dice si quiere abrir y modificar un archivo existente, o bien comenzar a crear un archivo desde el principio. Visual Basic, pasa la mayor parte del tiempo esperando las acciones del usuario (**eventos**) y respondiendo a ellas. Las acciones que el usuario puede realizar en un momento determinado son variadísimas, y exigen un tipo especial de programación: **la programación orientada a eventos**, Visual Basic la hace especialmente sencilla y agradable. Visual Basic 6.0 está orientado a la realización de programas para **Windows**, pudiendo incorporar todos los elementos de este entorno informático: ventanas, botones, cajas de dialogo, cajas de texto, botones de opción, botones de selección, barras de desplazamiento, gráficos, menús, etc.

Se mencionó la programación orientada a eventos, entonces, a continuación se introducirán algunos conceptos de este tipo de programación.

7.2.2. Programación orientada a eventos

Modo de Diseño y Modo de Ejecución.- La aplicación Visual Basic de Microsoft puede trabajar de dos modos distintos: en modo de diseño y en modo de ejecución. En modo de diseño el usuario construye interactivamente la aplicación, colocando controles en el formulario, definiendo sus propiedades, y desarrollando funciones para gestionar eventos.

La aplicación se prueba de modo de ejecución. En este caso el usuario actúa sobre el programa (introduce **eventos**) y prueba cómo responde el programa. Hay algunas propiedades de los controles que deben establecerse en modo de diseño, pero muchas otras pueden cambiarse en tiempo de ejecución desde el programa escrito en Visual Basic 6.0. También hay propiedades que sólo pueden establecerse en modo de ejecución y que no son visibles en modo de diseño.

Todos los conceptos mencionados anteriormente y otros los explicaremos a continuación en los apartados siguientes:

- **Formularios y Controles.**- Cada uno de los elementos gráficos que pueden formar parte de una aplicación típica de Windows 95/98/2000 es un tipo de **control**: los botones, las cajas de diálogo, las cajas de texto, las cajas de selección desplegables, los botones de opción, los botones de selección, los menús y otros tipos de elementos son controles para Visual Basic 6.0. Cada control debe tener un nombre a través del cual se puede hacer referencia a él en el programa. Visual Basic proporciona nombres por defecto que el usuario puede modificar.

En la terminología de Visual Basic 6.0 se llama **formulario (form)** a una ventana. Un formulario puede ser considerado como una especie de contenedor para los controles. Una aplicación puede tener varios formularios, pero un único formulario puede ser suficiente para las aplicaciones más sencillas. Los formularios deben tener también un nombre, que pueda crearse siguiendo las mismas reglas para los controles.

- **Objetos y propiedades.**- Los formularios y los distintos tipos de controles son entidades genéricas de las que pueden haber varios ejemplares concretos en cada programa. Cada formulario y cada tipo de control tienen un conjunto de **propiedades** que definen su aspecto gráfico (tamaño, color, posición en la ventana, tipo y tamaño de letra, etc.) y su forma de responder a las acciones del usuario (si está activo o no, por ejemplo). Cada propiedad tiene un **nombre** que ya viene definido por el lenguaje. Por lo general, las propiedades de un **objeto** son datos que tienen valores lógicos (*true*, *false*) o numéricos concretos, propios de ese objeto y distintos de las de otros objetos de su clase. Así pues, cada clase, tipo de objeto o control tiene su conjunto de propiedades, y cada objeto o control concreto tiene unos valores determinados para las propiedades de su clase. Se puede acceder a una propiedad de un objeto por medio del

nombre del objeto a que pertenece, seguido de un punto y el nombre de la propiedad.

- Nombres de objetos.- En principio cada objeto de Visual Basic 6.0 debe tener un nombre, por medio del cual se hace referencia a dicho objeto. El nombre puede ser el que el usuario desee, e incluso Visual Basic 6.0 proporciona nombres por defecto para los diversos controles. Los nombres por defecto no son adecuados porque hace referencia al tipo de control, pero no al uso que de dicho control está haciendo el programador. Existe una convención ampliamente aceptada que es la siguiente: *se utilizan siempre tres letras minúsculas que indican el tipo de control, seguidas por otras letras (la primera mayúscula, a modo de separación) libremente escogida por el usuario, que tienen que hacer referencia al uso que se va a dar a ese control.*
- Eventos.- Ya se ha dicho que las acciones del usuario sobre el programa se llaman eventos. Son eventos típicos el oprimir un botón del ratón, el hacer doble clic sobre el nombre de un archivo para abrirlo, el arrastrar un icono, o simplemente mover el ratón. Cada vez que se produce un evento sobre un determinado tipo de control, Visual Basic 6.0 arranca una determinada función o procedimiento que realiza la acción programada por el usuario para ese evento concreto. Estos procedimientos se llaman con un nombre que se forma a partir del nombre del objeto y el nombre del evento, separados por el carácter (`_`), como por ejemplo `txtBox_click`, que es el nombre del procedimiento que se ocupará de responder al evento `click`, en el objeto `txtBox`.
- Métodos.- Los *métodos* son funciones que también son llamadas desde programa, pero a diferencia de los procedimientos no son programadas por el usuario, sino que vienen ya pre-programadas con el lenguaje. Los métodos realizan tareas típicas, previsibles y comunes para todas las aplicaciones. De ahí que vengan con el lenguaje y que se libere al usuario de la tarea de programarlos. Cada tipo de objeto o de control tiene sus propios métodos.

- **Proyectos y archivos.**- Cada aplicación que se empieza a desarrollar en Visual Basic 6.0 es un nuevo **proyecto**. Un proyecto comprende otros componentes más sencillos, por ejemplo los **formularios** (que son las ventanas de la interfase de usuario de la nueva aplicación) y los **módulos** (que son conjuntos de funciones y procedimientos sin interfase gráfica de usuario). Un proyecto se compone siempre de **varios archivos** (al menos dos) y hay que preocuparse de guardar cada uno de ellos en el directorio adecuado y con el nombre adecuado. Existe siempre un archivo con extensión ***.vbp** (*Visual Basic Project*), éste archivo del proyecto contiene toda la **información de conjunto**. Además hay que crear **un archivo para cada formulario y por cada módulo** que tenga el proyecto. Los archivos de los formularios, con la extensión ***.frm** y los segundos con la extensión ***.bas** y si son módulos de clase con la extensión ***.cls**.

En el SIDUNAM, Visual Basic 6.0, se ha utilizado como lenguaje de desarrollo final (**front-end**), esto es, la aplicación desde el Cliente hace consultas hacia un Servidor de base de datos (SQL Server 7.0); éste último realiza las tareas que le piden los Clientes autenticados y regresa resultados a los mismos para que sean desplegados por los usuarios finales; así estamos aprovechando al máximo la arquitectura Cliente/Servidor entre equipos remotos.

7.3. Creación de la base de datos para el SIDUNAM

En el presente proyecto, primero creamos la Base de Datos con el Administrador de Base de Datos SQL Server 7.0., para después desarrollar el sistema con Visual Basic 6.0.

El almacenamiento de los datos es la esencia de los sistemas de información. Los objetivos generales del diseño de la organización del almacenamiento de datos son que se logre:

- Disponibilidad de los datos
- Almacenamiento eficiente de los datos
- Actualización eficiente
- Recuperación dirigida de la información

Para que el almacenamiento de los datos sea eficiente, debe vigilar el cumplimiento de algunos requisitos. Primero, los datos deben estar disponibles para cuando el usuario final desee usarlos. Segundo, los datos deben de ser precisos y consistentes (la base de datos debe tener una integridad). Tercero, el almacenamiento, actualización y grabado de los datos debe de ser ágil, sencillo y confiable.

La esencia de una base de datos es el Sistema Administrador de la Base de Datos, el cual permite la creación, modificación y actualización de la información; la recuperación de los datos y la emisión de los reportes. A la persona responsable de asegurar que la información satisfaga los objetivos programados se le denomina Administrador de la Base de Datos. Una de las ventajas de las bases de datos, es que al compartir los datos significa que éstos deben almacenarse una sola vez, esto apoya la idea de que se mantenga la integridad de la información, ya que el cambio de los datos se realizará de manera más sencilla y confiable si éstos aparecen una vez y no en varios archivos, dado que esto ocasionaría redundancia en los datos. Finalmente el enfoque de base de datos, tiene la ventaja de permitir que los usuarios expongan sus puntos de vista sobre los datos, sin necesidad de preocuparse de la estructura presente de la base de datos o de su ubicación física. No obstante los beneficios del enfoque de base de datos, también deben considerarse las desventajas que presenta, entre estas últimas, las que resaltan son las siguientes: los datos deben almacenarse en un sólo lugar, lo que trae como consecuencia que los datos son más vulnerables a accidentes y, por tanto, se hará necesario un proceso de respaldo completo y constante. Además, también existe el riesgo de que la persona que administre la base de datos pueda convertirse en el único privilegiado o habilitado para estar cerca de la información, esto justificable hasta cierto punto, por cuestiones de seguridad.

Finalmente, el modelo lógico de la base de datos debe transformarse en su correspondiente diseño físico, que es lo que nos ocupa en éste momento; el diseño físico considera la forma de almacenamiento de los datos y de sus interrelaciones, así como la mecánica de acceso a los datos.

Existen tres tipos básicos de base de datos con una estructuración lógica: el tipo jerárquico, el de red y el de relación. Para el SIDUNAM, se utilizó el enfoque relacional, ya que además de ser el más conocido y utilizado, es el que mejor se adapta a las características del *software* que elegimos para el desarrollo del presente proyecto, en este caso es el RDBMS (SQL Server 7.0).

Una base de datos relacional consiste en una o más tablas bidimensionales, las cuales se refieren como relaciones. Los renglones de las tablas representan los registros y las columnas contienen los atributos. El mantenimiento de las tablas de una base de datos relacional es bastante simple al compararlo con el mantenimiento de una estructura jerárquica o de red. Una de las ventajas principales de la estructura relacional, es que las consultas específicas se manejan de una manera muy eficiente.

Con el fin de que las estructuras relacionales sean útiles y manejables, las tablas relacionales primero deben "normalizarse". Existen varios niveles de normalización que se aplicaron a las tablas de la base de datos para el SIDUNAM; la normalización es el proceso de transformación de los almacenamientos de datos (tablas), en conjuntos estables de estructuras de datos de menor tamaño, además de ser más sencillas, tales estructuras son más estables. Las estructuras de datos normalizadas son más fáciles de mantener. Una vez que están normalizadas las tablas, el siguiente paso dentro del proceso de desarrollo del sistema SIDUNAM, fue construir la base de datos, para esto, nos basamos en el diccionario de datos que se presento en el capítulo V. Para construir la base de datos, se utilizó una herramienta de SQL Server llamada SQL Enterprise Manager, la cual cuenta con una interfaz gráfica, vía esta herramienta, realizamos toda la administración de las bases de datos del sistema y las de los usuarios como es la

que a continuación vamos a construir, para efectos del SIDUNAM. Para poder llevar a cabo dicha tarea, fue necesario realizar los siguientes pasos:

1) Crear los dispositivos (devices) de almacenamiento y respaldo.

Los devices son los archivos donde va a ser almacenada la base de datos. Existen tres tipos diferentes de devices: el de datos, el de log y el de respaldo. En los dos primeros residen los datos de la base de datos, mientras que en el tercero se ocupa para crear los respaldos. Con base en el número de tablas que contiene la base de datos y en la cantidad de información que se piensa manejar, para el SIDUNAM se creó un *device* de datos de 500 MB y un *device* de log de 50 MB. Para realizar esta operación, únicamente se tiene que hacer uso de la herramienta de administración y escribir los datos de acuerdo a las necesidades tal como lo muestra la Figura siguiente:

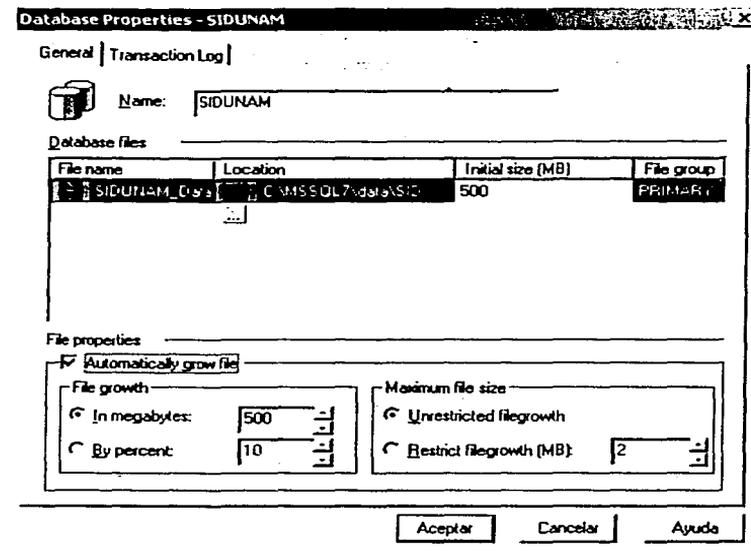


Fig. 7.2.

2) Generar la base de datos.

Una vez que los *devices* fueron creados, se procede a la creación de la base de datos, donde únicamente se le tiene que indicar qué *device* se utilizará para guardar los datos y cuál para el log.

3) Crear las tablas.

Una vez generada la base de datos se tiene que crear, una a una, cada tabla que utiliza el sistema. Esto es tan tardado como las tablas que se vayan a ocupar en el sistema. La siguiente figura es un ejemplo:

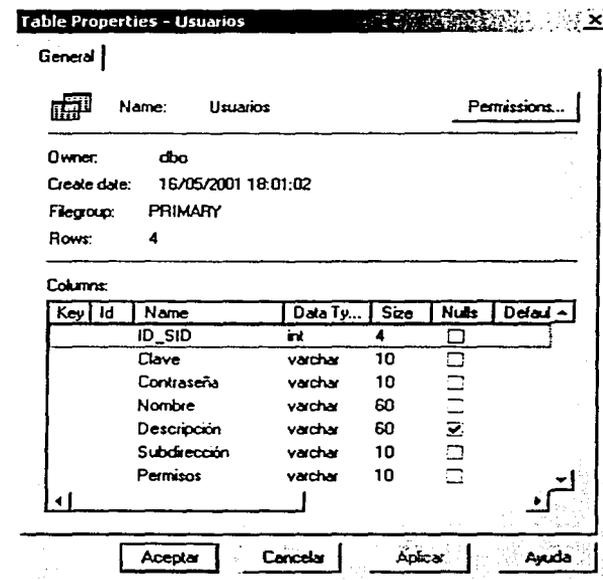


Fig. 7.3.

4) Poner datos en las tablas.

Para agregar datos a la tabla hay dos formas, con el asistente o bien con el comando *Insert*.

5) Crear usuarios y definir los grupos.

Al igual que en el Sistema Operativo (Windows NT), en el RDBMS, se deben dar de alta los usuarios y los grupos de usuarios que tendrán acceso a los diferentes objetos de bases de datos que existan en el Servidor, véase la Figura 7.4. Para nuestro sistema se crearon 2 tipos de Usuarios por Subdirección: 1) El Administrador que tiene todos los privilegios de su propia Subdirección y 2) El Usuario Normal que solamente tiene ciertos privilegios en el manejo de la información de la Subdirección a la cual pertenece. Por lo tanto, estos 2 tipos de Usuarios se integraron dentro de los grupos AdmSIDUNAM y UsusIDUNAM respectivamente, esto último, con el fin de realizar la administración de los permisos de acceso a la información, utilizando únicamente el nombre del grupo, lo cual facilita la administración de la base de datos.

The image shows a screenshot of the 'SQL Server Login Properties - New Login' dialog box. The 'General' tab is selected, showing fields for 'Name', 'Authentication' (with radio buttons for 'Windows NT authentication' and 'SQL Server authentication'), 'Domain', 'Security access' (with radio buttons for 'Grant access' and 'Deny access'), 'Password', and 'Defaults' (with fields for 'Database' and 'Language'). The 'Database' is set to 'master' and 'Language' is set to '<Default>'. At the bottom, there are 'Aceptar', 'Cancelar', and 'Ayuda' buttons.

Fig. 7.4.

Para el manejo de claves de usuario dentro del sistema, se creó una tabla llamada *Usuarios*, donde se tiene la información más importante relacionada a cada Usuario como Clave, Contraseña, Dependencia a la cual pertenece, Permisos, etc. La creación de dicha tabla, obedeció fundamentalmente a que no era conveniente crear una cuenta de usuario al nivel de Windows NT o SQL para cada usuario que fuera accesar al sistema. Esto se debe a que con el uso de ODBC, el Cliente puede comunicarse a la base de datos utilizando cualquier herramienta que haga uso de ODBC, lo que afectaría severamente el carácter confidencial de la información.

6) Asignar permisos a los objetos (base de datos, tablas, etc.).

Los permisos son las acciones que un usuario o un grupo de usuarios, tienen permitido efectuar para un determinado objeto de la base de datos. Un usuario puede tener permiso para *SELECT* e *INSERT* sobre registros en una tabla determinada, pero puede carecer de permisos para *DELETE* o *UPDATE*. Para otorgar permisos a los objetos y establecer un esquema de seguridad en una base de datos, se utiliza el comando *GRANT* con el que se especifica qué usuarios tienen qué permisos sobre qué objetos de la base de datos.

7.4. Acceso a datos con MS Visual Basic 6.0

Ya que tenemos la base de datos y en ésta información, el primer paso para tener acceso a la información de una base de datos consiste en establecer una conexión con el origen de datos.

Visual Basic, puede acceder los datos contenidos en una base de datos, vía dos tecnologías. A continuación se explica que es cada una de ellas y cuales son sus ventajas en su uso:

- DAO.- DAO (*Data Access Objects*), es el primer modelo de objetos de acceso a datos que presentó Visual Basic en su versión 3. Esta tecnología permite una completa administración de la base de datos, muy en especial del formato (MDB Access) nativo. Cuenta con capacidades transaccionales, multiusuarios, seguridad (suficiente en muchos casos), y de red. Su uso es recomendado para bases de datos locales y en entornos de red donde la concurrencia de usuarios no sobrepase los 15 ó 20, además de que la cantidad de datos no sobrepase 1 GB.
- ADO.- ADO (*ActiveX Data Objects*), es el sucesor de DAO. ADO es un modelo más fino que DAO. Muchas de las funcionalidades de DAO están consolidadas en un solo objeto y de ésta manera ADO ofrece un modelo de objetos más simple. Accesar datos a través de ADO es la visión de Microsoft para el futuro. ADO es mejor para aplicaciones Cliente/Servidor, también permite un fácil acceso a muchas fuentes de bases de datos. Visual Basic 6.0 y SQL Server 7.0, trabajan con la tecnología ADO de una manera natural.

ActiveX Data Objects (ADO) es una tecnología amigable y de fácil uso para agregar acceso a bases de datos. Utilizamos ADO para escribir secuencias de comandos compactas y escalables que conecten con bases de datos compatibles con *Open Database Connectivity* ODBC (Conectividad Abierta de Base de Datos) y orígenes de datos compatibles con *Objet Linking & Embedding Data Base* (OLE DB).

ADO, sin embargo, no se comunica directamente con la base de datos, sino que accede a ella a través de una interfase intermediaria, estas interfases son ODBC y OLE-DB. ODBC, fue creado para acceder en lo posible a cualquier tipo de datos siempre y cuando sea del tipo base de datos relacional, mientras que OLE-DB, puede acceder a cualquier tipo de datos, incluido, claro esta, las bases de datos relacionales. El proveedor OLE-DB, también permite acceder a bases de datos a través de ODBC, que es el caso del presente proyecto, siendo así que, ocuparemos el proveedor "OLE-

- DAO.- DAO (*Data Access Objects*), es el primer modelo de objetos de acceso a datos que presentó Visual Basic en su versión 3. Esta tecnología permite una completa administración de la base de datos, muy en especial del formato (MDB Access) nativo. Cuenta con capacidades transaccionales, multiusuarios, seguridad (suficiente en muchos casos), y de red. Su uso es recomendado para bases de datos locales y en entornos de red donde la concurrencia de usuarios no sobrepase los 15 ó 20, además de que la cantidad de datos no sobrepase 1 GB.
- ADO.- ADO (*ActiveX Data Objects*), es el sucesor de DAO. ADO es un modelo más fino que DAO. Muchas de las funcionalidades de DAO están consolidadas en un solo objeto y de ésta manera ADO ofrece un modelo de objetos más simple. Accesar datos a través de ADO es la visión de Microsoft para el futuro. ADO es mejor para aplicaciones Cliente/Servidor, también permite un fácil acceso a muchas fuentes de bases de datos. Visual Basic 6.0 y SQL Server 7.0, trabajan con la tecnología ADO de una manera natural.

ActiveX Data Objects (ADO) es una tecnología amigable y de fácil uso para agregar acceso a bases de datos. Utilizamos ADO para escribir secuencias de comandos compactas y escalables que conecten con bases de datos compatibles con *Open Database Connectivity* ODBC (Conectividad Abierta de Base de Datos) y orígenes de datos compatibles con *Objet Linking & Embedding Data Base* (OLE DB).

ADO, sin embargo, no se comunica directamente con la base de datos, sino que accede a ella a través de una interfase intermediaria, estas interfases son ODBC y OLE-DB. ODBC, fue creado para acceder en lo posible a cualquier tipo de datos siempre y cuando sea del tipo base de datos relacional, mientras que OLE-DB, puede acceder a cualquier tipo de datos, incluido, claro esta, las bases de datos relacionales. El proveedor OLE-DB, también permite acceder a bases de datos a través de ODBC, que es el caso del presente proyecto, siendo así que, ocuparemos el proveedor "OLE-

DB provider for ODBC", el cual nos permite conectarnos a cualquier fuente de datos ODBC.

Antes de crear secuencias de comandos para bases de datos se tiene que proporcionar una forma para que ADO encuentre, identifique y se comuniquen con una base de datos; esto es, se debe crear un archivo de acceso a datos, que no es otra cosa que, el controlador de bases de datos (programas que pasan información desde una aplicación, para nuestro proyecto en Visual Basic, a una base de datos, en este caso SQL Server), éstos utilizan un *Data Source Name* DSN (Nombre de Origen de Datos), para encontrar e identificar una base de datos ODBC en particular. Normalmente, los DSN contienen información de configuración de la base de datos, seguridad de usuarios y ubicación, y puede tener la forma de una entrada en el registro del sistema de Windows 95, 98, 2000 y NT o de un archivo de texto.

A saber, hay tres tipos de DSN: *Usuarios*, *Sistema* o *Archivo*; enseguida se explica en que consiste cada uno de éstos.

Los DSN de *usuario* y de *sistema* residen en el registro del sistema de Windows 95,98, 2000 y NT. Los DSN de *sistema* permiten que todos los usuarios que han iniciado una sesión en un Servidor concreto tengan acceso a una base de datos, siempre y cuando tengan permisos de Administrador; mientras que los DSN de *usuario* limitan la conectividad con la base de datos a los usuarios que hayan creado su propio DSN, es decir, van a poder acceder a la base de datos aquellos usuarios que dispongan de una Clave y Contraseña válidas en SQL Server. Los DSN de *archivo*, que tienen la forma de archivos de texto, proporcionan acceso a varios usuarios y son fácilmente transferibles entre un Servidor y otro mediante la copia de los archivos DSN. Para el SIDUNAM, ocuparemos los DSN de *usuario*, ya que nos garantizan seguridad en el acceso al sistema, o sea que, solamente tiene acceso a la base de datos quien tenga con una Cuenta o Clave y Contraseña válidas en el sistema.

Como se mencionó anteriormente, el primer paso para tener acceso a una base de datos es, establecer una conexión con el origen de datos. ADO, requiere un proveedor (*data consumer*) que pueda entender la estructura de los datos a leer y nos la entregue en formato "recordset" para nuestra manipulación. ADO proporciona el objeto **Connection**, que es el objeto que se utilizará para establecer y administrar las conexiones, de nuestra aplicación (SIDUNAM) y la base de datos SQL Server. El objeto **Connection** incorpora diversas propiedades y métodos que utilizamos para abrir y cerrar conexiones con la base de datos, y para enviar consultas de actualización de la información, estas consultas, se realizan con el lenguaje de consulta estructurado (SQL). Aunque el objeto **Connection** simplifica la tarea de conectar con una base de datos y emitir una consulta, el objeto **Connection** tiene sus limitaciones. En concreto, con el objeto **Connection** no se pueden crear secuencias de comandos que recuperen y presenten información de una base de datos; tiene que saber exactamente las modificaciones que desea realizar en la base de datos y después implementar las modificaciones como consultas.

Para recuperar datos, examinar resultados y modificar la base de datos del SIDUNAM, ADO tiene el objeto **Recordset**. El objeto **Recordset** tiene las funciones necesarias para, dependiendo de las restricciones de las consultas, recuperar y presentar un conjunto de filas, o *registros*, de una base de datos. El objeto **Recordset** mantiene la posición de cada registro devuelto por la consulta, lo que permite "recorrer" los resultados de uno en uno.

Concluyendo, para la realización del presente trabajo, se ocupó el lenguaje de programación MS Visual Basic 6.0 con la tecnología de acceso a datos ADO, como herramienta de desarrollo, esto nos presentó ventajas, ya que se pudo obtener el máximo provecho de la tecnología propuesta en los capítulos anteriores, como es el Sistema Operativo de Red y el RDBMS, así como aprovechar toda la infraestructura de comunicaciones de RedUNAM y la arquitectura Cliente/Servidor, utilizando el protocolo de comunicación TCP/IP; sencillamente la justificación de la utilización de la tecnología antes mencionada, es por razones de rendimiento y minimización del tráfico en la red.

Ya que agregamos, modificamos y analizamos la información de la base datos, se hace necesario, recuperarla y presentarla en pantalla, o bien de una forma impresa y/o almacenarla en disco; para esto, Visual Basic cuenta con una herramienta generadora de reportes que es el *Microsoft Data Report V.6.0*, que nos permite realizar reportes sencillos de una manera rápida; en el presente proyecto, dada la gran cantidad de reportes a generar se tuvo que trabajar con una herramienta más especializada como *Crystal Reports V.8.0*, que es un reporteador popular y aparte poderoso, para muchas bases de datos, como SQLServer, Oracle, Informix, DB2, etc., y fue con esta herramienta que diseñamos los reportes de nuestra aplicación. *Crystal Reports V.8.0*, utiliza una interfase gráfica a partir de donde podemos construir cualquier reporte que necesitemos para el sistema SIDUNAM.

7.5. Seguridad del sistema SIDUNAM

Las consideraciones sobre seguridad en los sistemas informáticos son un componente cada vez más importante en el desarrollo de los sistemas de información de la actualidad. Los cambios generados por las redes informáticas y las nuevas tecnologías introducen nuevos enfoques, riesgos y requerimientos que hacen necesario incorporar la seguridad en el proceso del diseño.

Debido a que el sistema trabaja sobre **RedUNAM**, la información de la base de datos está expuesta a todos los usuarios de Internet, tanto nacionales como Internacionales, y puede existir alguien que con malas intenciones intente filtrarse, por tal motivo, es de suma importancia que ninguna persona ajena, pueda consultar, modificar o borrar información de la base de datos del sistema.

SQL Server está integrado con el sistema de seguridad de Windows NT. Esta integración permite acceder tanto a Windows NT como a SQL Server con la misma *Clave* y *Contraseña*, si así lo decide el Administrador del Sistema. Además SQL Server tiene las características de encriptación que tiene Windows NT para la seguridad en red.

SQL Server valida a los usuarios con 2 niveles de seguridad; autenticación de la clave y validación de permisos en la base de datos de cuentas de usuario y de roles. La autenticación identifica al usuario que está usando una cuenta y verifica sólo la habilidad de conectarse con SQL Server. El usuario debe tener permiso para acceder a determinada base de datos en el Servidor. Los permisos controlan las actividades que el usuario tiene permitido realizar en la base de datos del SQL Server.

1) Autenticación de la clave.- Un usuario debe tener una cuenta para conectarse al SQL Server. Este reconoce 2 mecanismos de autenticación: Autenticación de SQL Server y de Windows NT. Cada uno tiene un diferente tipo de cuenta.

Autenticación de SQL Server.- Cuando se usa, un Administrador del Sistema SQL Server, define una Clave y una Contraseña, para determinado usuario que necesite acceder a la base de datos. El usuario debe suministrar tanto la Clave como la Contraseña cuando se conecta al SQL Server, vía ODBC.

Autenticación de Windows NT.- Cuando se usa, el usuario no necesita de una cuenta de SQL Server, para conectarse a la base de datos, siempre y cuando esta cuenta tenga privilegios de Administrador, de igual manera esto es mediante ODBC. Véase la siguiente figura.

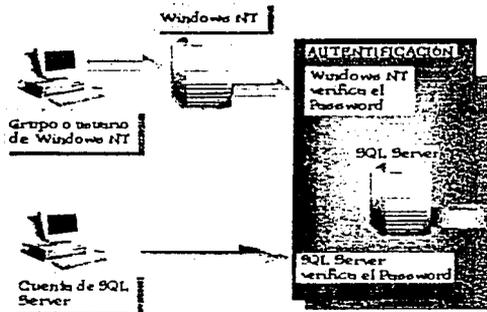


Fig. 7.5.

SQL Server valida a los usuarios con 2 niveles de seguridad; autenticación de la clave y validación de permisos en la base de datos de cuentas de usuario y de roles. La autenticación identifica al usuario que está usando una cuenta y verifica sólo la habilidad de conectarse con SQL Server. El usuario debe tener permiso para acceder a determinada base de datos en el Servidor. Los permisos controlan las actividades que el usuario tiene permitido realizar en la base de datos del SQL Server.

1) Autenticación de la clave.- Un usuario debe tener una cuenta para conectarse al SQL Server. Este reconoce 2 mecanismos de autenticación: Autenticación de SQL Server y de Windows NT. Cada uno tiene un diferente tipo de cuenta.

Autenticación de SQL Server.- Cuando se usa, un Administrador del Sistema SQL Server, define una Clave y una Contraseña, para determinado usuario que necesite acceder a la base de datos. El usuario debe suministrar tanto la Clave como la Contraseña cuando se conecta al SQL Server, vía ODBC.

Autenticación de Windows NT.- Cuando se usa, el usuario no necesita de una cuenta de SQL Server, para conectarse a la base de datos, siempre y cuando esta cuenta tenga privilegios de Administrador, de igual manera esto es mediante ODBC. Véase la siguiente figura.

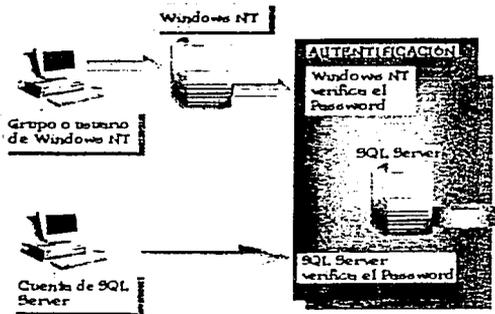


Fig. 7.5.

Cuando SQL Server está corriendo en Windows NT, como es el caso del SIDUNAM, en el sistema Administrador de SQL Server podemos especificar que modo de autenticación vamos a utilizar, estos modos son los siguientes:

- Modo de autenticación de Windows NT.- Sólo está autorizada la autenticación de Windows NT. Los usuarios no pueden usar cuentas de SQL Server.
- Modo Mixto.- Cuando se usa este método de autenticación, los usuarios se pueden conectar a SQL Server con la autenticación de Windows NT o con las de SQL Server.

2) Validación de permisos en la base de datos de claves de usuario y de roles.-

Después de que los usuarios han sido autenticados, y por consiguiente se les ha permitido conectarse al SQL Server, deben tener Claves válidas en la base de datos. Las claves de usuario y los roles, identifican permisos para ejecutar tareas. Las cuentas de usuario son específicas para cada base de datos existentes en el SQL Server. Los roles nos permiten reunir a los usuarios en una sola unidad a la cual se le pueden aplicar permisos. Dentro de cada base de datos, se asignan permisos a las Cuentas de usuarios y a los roles para permitir o limitar ciertas acciones. SQL Server acepta comandos después de que un usuario ha accedido a la base de datos. Véase la siguiente figura.



Fig. 7.6.

SQL Server realiza los siguientes pasos cuando valida permisos: 1) Cuando el usuario realiza una acción, tal como ejecutar un comando, o elegir una opción de un menú, los comandos son enviados al SQL Server. 2) Cuando SQL Server recibe un comando, chequea que el usuario tenga permisos de ejecutar dicha instrucción. 3) Después, SQL Server realiza cualquiera de las siguientes acciones.- a) Si el usuario no tiene los permisos adecuados, SQL Server devuelve un error. b) Si el usuario tiene los permisos adecuados, SQL Server realiza la acción.

7.6. Distribución del SIDUNAM vía FTP

Ahora bien, el sistema hasta aquí, cuenta con la seguridad necesaria, para que ningún intruso pueda acceder a la base de datos sin los permisos necesarios. Por tanto, está listo para que nuestros Clientes puedan instalar en sus sistemas locales la aplicación (SIDUNAM). Dado que el sistema tiene que trabajar en RedUNAM e Internet se hace necesario que el Servidor que contiene nuestra base de datos (SIDUNAM), sea un Servidor FTP, para que los Clientes en donde quiera que estén, puedan conectarse a él y copiar los archivos necesarios en sus sistemas locales, para posteriormente, instalar la aplicación. Aquí tenemos que poner dos cosas en claro; una es el cómo vamos a hacer el Servidor FTP y dos cómo es que vamos a empacar y distribuir los archivos necesarios, para que los Clientes puedan instalar la aplicación en cuestión.

Primero, vamos a ver lo referente al Servidor FTP. Antes que nada, al momento de estar instalando el Sistema Operativo Windows NT, se nos pregunta si queremos instalar el IIS (*Internet Information Server*), y únicamente hay que seleccionar el o los servicios que queramos habilitar como son Gopher, FTP, WWW. FTP fue uno de los primeros protocolos usados en las redes TCP/IP y en Internet. FTP se utiliza para transferir archivos de un equipo de una red a otro equipo de cualquier red, para copiar archivos desde un equipo Cliente a un Servidor sólo se puede usar FTP. En nuestro caso, los Clientes remotos necesitan transferir archivos en ese sentido, así que necesitamos usar FTP. Además, como queremos que los archivos de la aplicación estén disponibles para todos los usuarios remotos, FTP es un servicio extremadamente

fácil de instalar y mantener. Una vez que lo instalamos, asignamos el servicio FTP a los archivos; y no es necesaria ninguna configuración adicional. Los archivos disponibles mediante FTP pueden tener cualquier formato, como archivos de documentos, archivos multimedia o archivos de aplicación. El servicio FTP requiere que los usuarios inicien una sesión para poder usar el servicio. Como podemos apreciar, desde el momento en el que los Clientes quieren copiar los archivos de la aplicación, en sus sistemas locales, tienen que contar con una Clave y Contraseña válidos para poder acceder al Servidor de FTP y contenedor de la base de datos. De igual manera, podemos configurar el número de conexiones simultáneas permitidas y el periodo de tiempo permitido para las conexiones. Para configurar los requisitos de inicio de sesión para el servicio FTP se usa el Administrador de servicios de Internet (IIS). Si el servicio FTP lo configuramos para inicios de sesiones anónimas, los Clientes pueden iniciar sesiones con el nombre de usuario "anonymous", cosa que no nos conviene por cuestiones de seguridad. Los Clientes de FTP también pueden iniciar sesión con un nombre de Usuario y una Contraseña de Windows NT, que es la manera como podemos acceder al Servidor de FTP que contiene nuestra aplicación, esto es, en **Propiedades del servicio FTP** del Administrador de Servicios de Internet (IIS), desactivamos las casilla de verificación **Permitir conexiones anónimas** y **Permitir sólo conexiones anónimas**, para evitar que usuarios ajenos al sistema accedan al Servidor IIS. Véase la Figura 7.7.

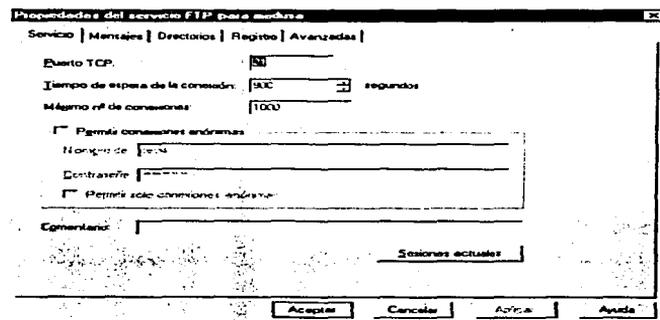


Fig. 7.7.

fácil de instalar y mantener. Una vez que lo instalamos, asignamos el servicio FTP a los archivos; y no es necesaria ninguna configuración adicional. Los archivos disponibles mediante FTP pueden tener cualquier formato, como archivos de documentos, archivos multimedia o archivos de aplicación. El servicio FTP requiere que los usuarios inicien una sesión para poder usar el servicio. Como podemos apreciar, desde el momento en el que los Clientes quieren copiar los archivos de la aplicación, en sus sistemas locales, tienen que contar con una Clave y Contraseña válidos para poder acceder al Servidor de FTP y contenedor de la base de datos. De igual manera, podemos configurar el número de conexiones simultáneas permitidas y el periodo de tiempo permitido para las conexiones. Para configurar los requisitos de inicio de sesión para el servicio FTP se usa el Administrador de servicios de Internet (IIS). Si el servicio FTP lo configuramos para inicios de sesiones anónimas, los Clientes pueden iniciar sesiones con el nombre de usuario "anonymous", cosa que no nos conviene por cuestiones de seguridad. Los Clientes de FTP también pueden iniciar sesión con un nombre de Usuario y una Contraseña de Windows NT, que es la manera como podemos acceder al Servidor de FTP que contiene nuestra aplicación, esto es, en **Propiedades del servicio FTP** del Administrador de Servicios de Internet (IIS), desactivamos las casilla de verificación **Permitir conexiones anónimas** y **Permitir sólo conexiones anónimas**, para evitar que usuarios ajenos al sistema accedan al Servidor IIS. Véase la Figura 7.7.

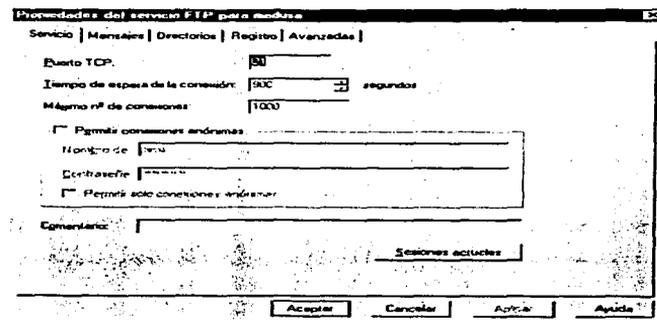


Fig. 7.7.

Al desactivar estas casillas de verificación, cualquier cuenta que sea "anónima" no podrá iniciar sesión. Esto es útil por cuestiones de seguridad; de tal forma que sólo las cuentas válidas en el sistema, que para tal efecto haya creado el Administrador del Servidor, tendrán acceso al Servidor IIS. Además estas cuentas deben ser validadas en el Administrador de Usuarios de Windows NT, debiendo contener ciertos privilegios, para poder hacer uso del servicio de FTP.

La manera de configurar los directorios de FTP, es la siguiente: De forma predeterminada, todos los subdirectorios del Directorio Particular están disponibles para el servicio de FTP, de tal manera, que sólo debemos poner los archivos o subdirectorios en el Directorio Particular que queramos compartir con nuestros Clientes, para nuestro caso compartiremos los archivos de la aplicación y todo lo que está en torno a ella. Algunos exploradores requieren que la lista de FTP tenga el estilo del formato UNIX. Para obtener la máxima compatibilidad con los exploradores, en el SIDUNAM, definimos el estilo de lista de directorios de FTP como formato UNIX.

Por otro lado, el servicio de FTP, cuenta con los permisos de lectura y escritura para el Directorio Particular de forma predeterminada, para el caso de la aplicación (SIDUNAM), se manejó el permiso de lectura, ya que los Clientes no tienen porque modificar la información que se les comparte, únicamente tienen que copiarla a sus sistemas locales, para posteriormente instalar, si así lo desean el SIDUNAM, en sus sistemas locales.

En segundo término veamos cómo empaquetaremos y distribuiremos la aplicación en los Clientes remotos.

Visual Basic 6.0, cuenta con un Asistente de empaquetado y distribución, el cual nos permite crear paquetes de instalación para nuestra aplicación (SIDUNAM); es necesario haber guardado, compilado y creado el ejecutable, antes de ejecutar este asistente. El asistente de empaquetado y distribución ofrece numerosas opciones para la distribución de nuestra aplicación, incluyendo la creación de:

- Paquetes estándar, diseñados para ser instalados mediante un programa de instalación (setup.exe), que es el caso del SIDUNAM.
- Paquetes de Internet, diseñados para ser descargados desde un sitio Web.
- Archivos de dependencia que puede distribuir con sus componentes.

El asistente comprime los componentes de la aplicación en un paquete formado por archivos "cabinet" (.cab). Se pueden crear uno o más archivos (.cab), dependiendo del tamaño de la aplicación.

Empaquetar una aplicación consiste en crear un paquete que permita instalar la aplicación, en el equipo del usuario final. Por otro lado la distribución del paquete supone la creación de los medios de distribución y copia de todos los archivos necesarios a una ubicación donde los usuarios puedan tener acceso a ellos, para esto, está nuestro Servidor FTP, el cual es nuestro Servidor de Distribución y es ahí donde también se aloja la base de datos, para nuestra aplicación en cuestión.

La primera etapa en la creación de un programa de instalación personalizado es determinar los archivos que deben empaquetarse y distribuirse. Todas las aplicaciones de Visual Basic, requieren un conjunto mínimo de archivos, conocido como archivos de preinstalación, que son necesarios antes de poder instalar la aplicación. Además, todas las aplicaciones de Visual Basic necesitan archivos específicos de la aplicación como un archivo ejecutable (.exe), archivo de datos, controles ActiveX (.ocx) , mapas de bits (archivos .bmp) o archivos de librerías (.dll), es decir, existen tres categorías principales de archivos necesarios para ejecutar y distribuir nuestra aplicación: 1) Archivos de tiempo de ejecución. 2) Archivos de Instalación. 3) Archivos específicos de la aplicación.

Después de llevar a cabo el proceso de empaquetado y de generar el medio de distribución de la aplicación (SIDUNAM), se probó el programa de instalación, asegurándonos de hacerlo en un equipo que no tuviera Visual Basic. Una vez que se

terminó con la instalación, se ejecutó el programa, para comprobar que se comportara de la forma esperada.

7.7. Propuesta de implantación

Dado que el objetivo a alcanzar consiste en la automatización del registro en computadora de la gran cantidad de información, que se tiene de Deportistas en cada uno de los deportes en la Dirección General de Actividades Deportivas y Recreativas de nuestra Universidad Nacional Autónoma de México, y siendo que ésta información tendrá que ser manejada a través de **RedUNAM** e Internet, tanto para consulta como para actualización, entonces, partimos de que para realizar el proyecto SIDUNAM, se aprovechó al máximo toda la infraestructura tecnológica de la UNAM en cuanto a equipo de cómputo y telecomunicaciones se refiere. Para esto a lo largo de los pasados capítulos se ha propuesto el uso de las tecnologías, tanto en *hardware* como en *software*, que más se adecuan a las necesidades, para poder cumplir con todas las expectativas que debe tener el sistema SIDUNAM; por lo anterior, en términos generales nuestra propuesta de solución es la siguiente: 1) Para alcanzar el objetivo, se planteó el desarrollo de un sistema bajo la arquitectura Cliente/Servidor, que contará con un origen de datos sobre un equipo que soportará el multiprocesamiento simétrico utilizando como Sistema Operativo de Red a Windows NT Server y a SQL Server como Administrador de Base de Datos Relacionales y el protocolo TCP/IP de comunicación entre los Clientes y el Servidor. En cuanto al equipo Cliente, se utilizará la **interfase** gráfica Windows en cualquiera de sus versiones (95,98,2000, NT o Millenium) y Visual Basic como Lenguaje de Programación y para generar los reportes Crystal Reports. En este sistema, cuando el Servidor recibe una solicitud de servicio, realiza el proceso, regresa al resultado al Cliente y provee el servicio de administración de datos. De esta manera, el SIDUNAM, se desempeñará como se describe a continuación: El Servidor se encarga de ejecutar todos aquellos procesos que para el Cliente podrían representar mucho tiempo de proceso, de tal forma que el Servidor se encarga de realizar el trabajo pesado y el Cliente se dedica a presentar los resultados a los usuarios finales, mediante la aplicación realizada en Visual Basic. La premisa del SIDUNAM, está precisamente en que se aprovecha lo que ya se tiene, con la única

variante de que ahora se contará con nuevas funcionalidades, como es el compartir los recursos que se tienen, tanto de *software* y *hardware*, en la RedUNAM, y por lo tanto en Internet, esto a su vez, trae como consecuencia que haya un mejor rendimiento y minimización del tráfico en la red. 2) Los siguientes puntos presentan las características a detalle que debe cumplir el SIDUNAM.

- I. Manejo bajo un ambiente Cliente/Servidor.- Originalmente, el sistema se pensó bajo una plataforma C/S, la cual permitiría la correcta distribución de las funciones de una máquina potente (Servidor) y otra con menos potencialidad (Cliente), esto es, los datos debían residir en el Servidor, quien puede procesarlos con más facilidad, mandando al Cliente la información ya procesada y éste último, presentarla al usuario final, quien a su vez tendría la posibilidad de guardarla en un archivo, consultarla en pantalla o bien imprimirla según sus necesidades.
- II. Acceso a la información vía RedUNAM e Internet.- El sistema presentaría, una característica muy importante y especial, ésta es que la aplicación, debía correr sobre cualquier computadora Cliente conectada a Red-UNAM, la única restricción es que los Clientes contaran con una interfase gráfica, como lo es Windows en cualquiera de sus versiones. Por lo anterior, tanto el Servidor como el Cliente debían usar un protocolo de comunicación estándar en la UNAM (TCP/IP).
- III. Soporte multiusuario 450 ó más usuarios.- Otra consideración importante, es que el sistema daría servicio a un gran número de usuarios, por tanto, se realizó un cálculo del número de Dependencias y Subdependencias que existen en la UNAM aproximadamente (139 Y 66 respectivamente), las cuales podrían tener acceso al sistema, de éstas, el Director, el Responsable del Área de Deportes, así como las personas que están enroladas en el Deporte de cada una de las Dependencias, serían las que ocuparían el sistema. Así que con base en dicho cálculo, se estimó que la cantidad de usuarios que utilizarían el SIDUNAM, estaría fluctuando entre los 450-500.

- IV. Medidas de seguridad en el acceso a la información.- La seguridad dentro del sistema juega un papel muy importante y debido a que el sistema trabaja en Internet, la información de la base de datos está expuesta a todos los usuarios de Internet a nivel mundial, por lo tanto, no se descarta la idea de que alguien con malas intenciones intente filtrarse. Es por esto, que el sistema cuenta con una doble seguridad de acceso, una de ellas es la que nos ofrece Windows NT y por otra parte el sistema controla el nivel de acceso a los usuarios válidos, esto es, para poder entrar al sistema se tiene que configurar un ODBC, mismo que pide una Clave y Contraseña que el usuario tiene que saber, aparte dependiendo del tipo de Clave es como va a poder hacer movimientos dentro del sistema. En cuanto al nivel de disponibilidad de la información, para el SIDUNAM, se realizan varias medidas de seguridad, como el monitoreo constante de los movimientos que se realizan en la base de datos; así como un programa de respaldo de la información en cinta diario, del Sistema Operativo y la Base de Datos.
- V. Medidas de seguridad para la integridad de la información.- En los niveles de integridad y consistencia, se protege la información a ser consultada o modificada; además de que el sistema se comporte en la forma esperada, estos mecanismos suelen ser transparentes para los usuarios, ya que desde el diseño de la base de datos, se estructuro de tal manera, que fuera estable y lógica, para evitar redundancia de la información e inconsistencia en los datos.
- VI. Consulta e impresión.- La aplicación tenía que contar con la opción de impresión total o selectiva de la información consultada en pantalla, una vez que el usuario realizara determinada búsqueda en el sistema.
- VII. Respaldo de la información en el Disco Duro o en Disquetes.- La aplicación cuenta con la posibilidad de que la información visualizada en pantalla, pueda ser almacenada en el disco duro de la máquina o bien en disquetes o cualquier parte de la red local del sistema propio.

- IV. Medidas de seguridad en el acceso a la información.- La seguridad dentro del sistema juega un papel muy importante y debido a que el sistema trabaja en Internet, la información de la base de datos está expuesta a todos los usuarios de Internet a nivel mundial, por lo tanto, no se descarta la idea de que alguien con malas intenciones intente filtrarse. Es por esto, que el sistema cuenta con una doble seguridad de acceso, una de ellas es la que nos ofrece Windows NT y por otra parte el sistema controla el nivel de acceso a los usuarios válidos, esto es, para poder entrar al sistema se tiene que configurar un ODBC, mismo que pide una Clave y Contraseña que el usuario tiene que saber, aparte dependiendo del tipo de Clave es como va a poder hacer movimientos dentro del sistema. En cuanto al nivel de disponibilidad de la información, para el SIDUNAM, se realizan varias medidas de seguridad, como el monitoreo constante de los movimientos que se realizan en la base de datos; así como un programa de respaldo de la información en cinta diario, del Sistema Operativo y la Base de Datos.
- V. Medidas de seguridad para la integridad de la información.- En los niveles de integridad y consistencia, se protege la información a ser consultada o modificada; además de que el sistema se comporte en la forma esperada, estos mecanismos suelen ser transparentes para los usuarios, ya que desde el diseño de la base de datos, se estructuro de tal manera, que fuera estable y lógica, para evitar redundancia de la información e inconsistencia en los datos.
- VI. Consulta e impresión.- La aplicación tenía que contar con la opción de impresión total o selectiva de la información consultada en pantalla, una vez que el usuario realizara determinada búsqueda en el sistema.
- VII. Respaldo de la información en el Disco Duro o en Disquetes.- La aplicación cuenta con la posibilidad de que la información visualizada en pantalla, pueda ser almacenada en el disco duro de la máquina o bien en disquetes o cualquier parte de la red local del sistema propio.

VIII. Actualización en línea desde cada área responsable.- Dado que el sistema puede ser accesado desde Internet, se hace indispensable que la actualización se lleve a cabo desde cualquier área responsable, no importando en dónde se encuentre ésta, si en el campus universitario o fuera de éste. A esto responde el sistema de manera satisfactoria, siempre y cuando se cumpla con las autentificaciones necesarias para poder acceder a la base de datos.

7.8. Alcances

Siendo que la DGADyR cuenta con personal en todas las Escuelas y Facultades y en las diversas Instalaciones Deportivas como son la Alberca Olímpica, Frontón Cerrado, Estadio de Prácticas, Pista de Calentamiento, etc., el uso del SIDUNAM en la Universidad estaría facilitando el que las Escuelas y Facultades contaran con un sistema que integre toda la de Información de Deportistas de todos los niveles y disciplinas, así como del personal de apoyo; arrojándonos información a cerca de sus Datos Generales, Académicos, Deportivos, Médicos, Laborales, de Historia Deportiva, etc.; en resumen, se terminaría con la desvinculación de Información con su entorno universitario y externo en materia deportiva; siendo así que se contaría con un sistema seguro y eficiente para el control del Deportista Universitario y de todas aquellas personas que estén involucradas en el deporte universitario a nivel general, en Escuelas, Facultades y las diversas instalaciones que conforman la UNAM.

VIII. PRUEBAS REALIZADAS AL SISTEMA (SIDUNAM)

a) Pruebas de unidad

En primer lugar, la prueba de unidad se realiza sobre los procedimientos contenidos dentro de un módulo, cada uno de manera independiente, a fin de localizar posibles errores de codificación y lógica que estén contenidos en su código. En este momento es cuando se detecta si está bien generada una llamada (cadena) a un procedimiento almacenado del Servidor, así como los datos que éste último regresa.

En el caso del SIDUNAM, los primeros procedimientos que se probaron fueron los de consulta, es decir, todos aquellos que mostraban información directamente en la pantalla del usuario; posteriormente, se trabajó con los que generaban las impresiones y, por último, con aquellos que guardaban en disco la información consultada.

b) Pruebas de integración

En esta etapa se evalúa el conjunto de procedimientos que conforman un módulo, por ejemplo, el de Inscripción de Deportistas. Este módulo está integrado por varios procedimientos que realizan una tarea específica, pero que interactúan dependiendo del buen funcionamiento de los otros procedimientos.

Para llevar a cabo esta evaluación, se realizó una matriz de pruebas para ensayar el ciclo completo (consulta, impresión y almacenamiento) de cada una de las operaciones relacionadas con todos los módulos.

En esta etapa fue donde el SIDUNAM presentó más errores, principalmente en el proceso de impresión, ya que al probar el ciclo completo del módulo surgieron opciones que no se habían contemplado anteriormente.

En esta evaluación se observó cómo funcionaba el sistema completo en un ambiente de producción. Una vez que todos los módulos estaban listos y probados individualmente se podía pensar que el sistema estaba listo, sin embargo, después de este período de pruebas se detectó que ciertos módulos tenían errores cuando trabajaban en interacción con otros. Por ejemplo, el módulo datos generales, en donde se almacenan los procedimientos, funciones y variables globales para todo el sistema, presentó varios problemas de incompatibilidad con el resto de los módulos, corrigiendo esta anomalía enseguida.

c) Pruebas de volumen

Con las pruebas de volumen, se asegura el funcionamiento adecuado del sistema en condiciones de gran demanda de recursos en cantidad y frecuencia. Para este tipo de pruebas, tuvimos que trabajar con volúmenes de información cercanos a los que se piensa pueda tener la DGADyR. Lo anterior se realizó en las Salas de Cómputo de la Facultad de Ingeniería; lo que se hizo fue únicamente instalar la aplicación en los Clientes (Máquinas de las Salas), ya que la Base de Datos siempre ha estado en uno de los Servidores (Medusa) de las mismas, para poder realizar pruebas. También se dio de alta información en cada uno de los módulos que conforman la aplicación, de igual forma, se realizaron consultas e impresiones de determinada información del sistema. Se ocupó la ayuda en línea de la aplicación, para saber como es que se tenía que navegar dentro del sistema. Estas pruebas permitieron comprobar el funcionamiento de cada uno de los módulos, y se pudo constatar que el sistema resultó sin complicaciones.

d) Pruebas de validación

En esta prueba lo que se hizo fue validar la información correcta de cada uno de los módulos, o sea la que el sistema realmente esta solicitando en cada uno de estos módulos y no la que el usuario cree que deba de ir.

Por otra parte en el diseño mismo de las ventanas (interfase de usuario) de la aplicación, se especificó el tipo de datos que deberá aceptar el campo, por ejemplo, caracteres, alfanuméricos, numéricos, de fecha, etc., por lo tanto, el sistema mismo no permitirá que en un campo de fecha, se le teclee un tipo de dato diferente. Además tenemos otros tipos de pruebas que también se realizaron y que enseguida se enlistan.

Se capturó información parcialmente correcta y a continuación terminar la entrada de esta información. Se omitieron todos los campos dentro de nuestras pantallas; simplemente pulsando el icono de salir.

Se proporcionó información correcta, pero con demasiados datos de parámetros. Se generó una interrupción del sistema inmediatamente después de haber introducido información en los campos de nuestras pantallas, no afectando la estabilidad del SIDUNAM.

Los objetivos de las pruebas fueron valorar y mejorar la calidad del sistema, siguiendo la idea de que la mejor manera de minimizar el número de errores en un programa es encontrarlos y eliminarlos durante el análisis y el diseño. La confiabilidad del sistema implica un conjunto de actividades que incluyen diversas pruebas que garantizan el apego a las especificaciones de requisitos, principios generales de facilidad en el manejo del sistema, estándares en el lenguaje de programación, estándares organizacionales de la DGADyR, documentación y expectativas del usuario.

CONCLUSIONES

El análisis, desarrollo, pruebas y mantenimiento de un sistema, requieren un enfoque sistemático que permita asegurar su aceptación por parte de los usuarios. Las necesidades y limitaciones del usuario deben estar determinadas y claramente establecidas para lograr un análisis confiable de la situación a resolver; en el diseño del producto es necesario tener presente a los usuarios ya que ellos trabajarán directamente con el sistema, las pruebas que se realicen a lo largo del desarrollo garantizan la aceptación y calidad del sistema. Y finalmente, las tareas de mantenimiento incluirán solicitudes de análisis de cambio, rediseño y modificación, así como actualización de la documentación.

Para lograr el desarrollo y mantenimiento de un producto de *software*, es necesario utilizar técnicas de resolución de problemas junto con las notaciones adecuadas. Para el caso de nuestro proyecto, realizamos en primer lugar, una descripción de la situación por resolver considerando las restricciones del problema para justificar una estrategia de solución sistematizada que realice las funciones requeridas. La estrategia adoptada se fundamenta en el análisis y diseño estructurado, todo esto enmarcado en el ciclo de vida clásico de desarrollo de sistemas.

Cuando ya se tienen claros los requerimientos y las funciones que realizará el sistema, el siguiente paso consiste en especificar la manera en que esto será estructurado en un producto de *software*; en este punto es necesario emplear herramientas de abstracción que nos permitan trabajar con los requisitos, independientemente del lenguaje de programación que se empleará en una etapa posterior a este proceso. Tales herramientas incluyen a los diagramas de flujo de datos que permiten especificar las actividades de los procesos y las transferencias de datos entre ellos de una manera gráfica, y precisamente, esta características gráficas, facilita al mismo tiempo, la comunicación con los Clientes en esta etapa de desarrollo del sistema. El uso del lenguaje en forma estructurada proporciona especificaciones paso a paso de un algoritmo y; cuando se requiere la especificación de una lógica de decisión más compleja, se hace uso de tablas de decisión; como complemento al flujo de datos,

se emplea el diccionario de datos, que contiene un listado organizado de todos los elementos de datos involucrados en el sistema permitiendo una comprensión precisa de las entradas, salidas, almacenamientos y pasos intermedios desarrollados por los procesos del producto de *software*.

Mientras se diseña el sistema empleando las herramientas ya descritas, es inevitable el refinamiento por pasos para lograr en cada ocasión, una mayor abstracción desde los asuntos generales a los detalles específicos. El diseño estructurado implica la conversión de diagramas de flujo en cartas de estructura, estas últimas, son elementos que documentan la estructura jerárquica y las interconexiones de los procesos de un sistema. Después, las cartas de estructura se redefinen mediante criterios de modularización tales como la cohesión de funcionalidad dentro de los módulos y el acoplamiento entre ellos. Por lo tanto, se puede resaltar que los aspectos más útiles del diseño estructurado son la transición sistemática de los diagramas de flujo de datos a las cartas de estructura y la comprensión del acoplamiento y la cohesión de los módulos.

El desarrollo de un sistema de programación incluye la determinación del tipo de sistema a desarrollar, en nuestro caso, se trata de un sistema de procesamiento de datos, que se apoya de una base de datos para el almacenamiento de la información, dentro de este ámbito existen terminologías y notaciones particulares que permiten manejar las relaciones entre los datos y entidades que requiere el sistema para su funcionamiento, tales relaciones se expresan gráficamente mediante los diagramas Entidad-Relación. En general, se cuenta con un modelo para mostrar cómo se relacionan lógicamente los datos de un registro. El orden de los datos no es significativo, puesto que las tablas relacionales muestran las relaciones lógicas, no físicas. Y al solicitar cierta información, el sistema produce resultados de la consulta que contiene específicamente lo solicitado gracias al empleo de otro concepto importante, el de normalización con el cual se simplifica la relación entre los campos de un registro.

Después del análisis y diseño, es momento de trabajar con un lenguaje de programación para traducir lo desarrollado en instrucciones para la computadora; las

características disponibles en el lenguaje de programación ejercen influencia sobre la estructura del sistema, para nuestro proyecto, Visual Basic constituyó un instrumento útil en el manejo de los conceptos del análisis y diseño estructurado y de bases de datos.

Las actividades que sucederán después de la entrega del producto caen dentro del mantenimiento del sistema, tales actividades incluirán, cuando así sea requerido, mejoras en el producto, adaptaciones a nuevos ambientes y corrección de problemas, todo esto con el objeto de proporcionar nuevas funciones e interacciones o trasladar el *software* a nuevos Sistemas Operativos y/o nuevas máquinas. Claro es que, todas estas posibles tareas se deben efectuar con un enfoque sistemático ordenado que analice los requisitos de las modificaciones cuidando el rediseño, reimplementación, revalidación y redocumentación de los cambios.

Para la DGADyR el presente trabajo, significa el hacer fácil el que las Escuelas y Facultades cuenten con un sistema que integre toda la Información de Deportistas de todos los niveles y disciplinas, así como del personal de apoyo, arrojándonos información a cerca de sus Datos Generales, Académicos, Deportivos, Médicos, Laborales, de Historia Deportiva, etc., siendo así que se terminaría con la desvinculación de Información con su entorno universitario y externo en materia deportiva, además de que se contaría con un sistema seguro y eficiente para el control del Deportista Universitario y de todas aquellas personas que estén involucradas en el deporte universitario a escala general, en Escuelas, Facultades y las diversas Instalaciones que pertenecen a la UNAM. De igual manera, en éste sistema se aprovechó al máximo toda la tecnología en cuanto a comunicaciones se refiere, ocupando la arquitectura Cliente/Servidor y el protocolo de comunicación TCP/IP; sencillamente la justificación de la utilización de la tecnología antes mencionada, es por razones de rendimiento y minimización del tráfico en la red, por consiguiente, contar con un sistema eficiente, para el manejo de la información del Deportista Universitario.

Finalmente, podemos establecer que el objetivo fundamental de este proyecto fue proporcionar un sistema de computadora que facilitará a los usuarios llevar a cabo su trabajo, atendiendo a la calidad del producto final y empleando los métodos, herramientas y técnicas que juzgamos convenientes para la situación por resolver que se presentó en la DGADyR. La enseñanza que obtuvimos, además de la relacionada con la metodología empleada, es que la ingeniería en computación es una actividad que involucra de manera intensiva a la gente, se desarrollo un *software* que será usado por otras personas quienes nos establecieron sus necesidades y expectativas del producto final.

BIBLIOGRAFÍA

1. Pressman, Roger. S., **Ingeniería del Software**, Editorial Mc Graw-Hill, México 1993.
2. Fairley, Richard, **Ingeniería de Software**, Editorial Mc Graw-Hill, México 1988.
3. Kendall K. E., and Kendall, J. E., **Análisis y Diseño de Sistemas**, Editorial Prentice-Hall Hispanoamericana, México 1991.
4. Martin, J., **Fourth Generation Languages**, volumen 1, Editorial Prentice-Hall, USA 1985.
5. Tsai, Alice Y. H., **Sistemas de Base de Datos (administración y uso)**, Editorial Prentice-Hall Hispanoamericana, México 1990.
6. Yourdon, E. **Techniques of program structure and desing**, Editorial Prentice-Hall, USA 1979.
7. Yourdon, E. And Constantine, L., **Structured design**, Editorial Prentice-Hall, USA 1979.
8. Blanchard, B. S. And Fabricky W. J., **Systems Engineering and Analysis**, Editorial Prentice-Hall, USA 1981.
9. Stevens, W., Myers, G. And Constantine, L., **Structured Design**, Editado por el MIT. USA 1959.
10. Myers, G., **The art of software tests**, Editorial Wiley. USA 1979.
11. McCall, J., Richards, P. And Walteres, G., **Factors in softwre quality**, Editado por Ntis. USA 1997.
12. Swanson, E. B., **The dimensions of mantenance**, Editado por el IEEE, USA 1976.
13. Jeffrey L. Whitten, **Análisis y Diseño de Sistemas de Información**, Editorial McGraw Hill. Colombia 1996.
14. William R. Vaughn, **Programación de SQL Server 7.0. con Visual Basic 6.0**. Editorial McGraw Hill. México 1999.

15. Ross Nelson, **Guía Completa de Visual Basic para Windows**, Editorial McGraw Hill. México 1995.
16. Tanenbaum Andrew S. **Redes de Computadoras**, Editorial Prentice hall Tercera Edición. México 1997.
17. Tanenbaum Andrew S. **Sistemas Operativos Modernos**, Editorial Prentice Hall. Primera Edición. Año 1995. México.

Consultas a sitios de Internet

18. Dirección General de Actividades Deportivas y Recreativas,
Objetivo y Funciones de la DGADyR
<http://www.unam.mx/dgadyr/dgadyr.html>
DGADyR 2 Febrero de 1999.
19. Coordinación General de Extensión Universitaria,
Coordinar Supervisar, Planear y Reglamentar el Deporte en la FES-C
<http://www.cuautilan2.unam.mx/actdep.htm>
16 de Enero de 1999.
20. Carlos Lizárraga C.
Comparación NT vs. UNIX costos y características
http://fermat.eup.udl.es/usuarios/cesar/microchof_es/unix-nt.html
Centro de Cómputo Académico 7 de Mayo de 1998.
21. Interactive Programmers Community
Manuales de Programación
<http://www.lawebdelprogramador.com/>
Copyright 2000.
22. Patrick Lambert
Windows NT Vs. Linux
<http://live.linux.org.ve/NTvsLinux/metachart.html>
Tomado de MetaChart.
23. Hewlett-Packard Company
Características de Servidores HP
<http://www.hp.es/cpsd/docs/producto/htm/sere800.htm>
1994-2001.
24. Compaq
Características de Servidores Proliant Server ML350

- <http://www.compaq.com/products/servers/proliantml350/>
1994-2001.
25. ABOX
Software y Hardware
<http://www.abox.com/productos.asp?pid=54/>
1999-2001.
26. Asociación Española de Metodología de las Ciencias del Comportamiento
Diccionario de Acrónimos
<http://www.ub.es/comporta/acronimo.htm>
15 Enero del 2001.
27. Laboratorio Nacional de Informática Avanzada
Políticas y Desarrollo de Internet en México
<http://www.lania.mx/spanish/vinculacion/foro/red-unam.html>
Septiembre de 1996.
28. Centro de Información de Red UNAM
Información sobre Red UNAM
<http://www.nic.unam.mx/>
Noviembre del 2000.
29. Juan Salvador Miravet Bonet
Informática y Sociedad
<http://www4.uji.es/~al019803/Tcpip.htm>
Noviembre de 1999.
30. Página Web de Hewlett Packard
Información sobre productos Hewlett Packard
<http://www.hp.es/>
Noviembre del 2000.
31. Página Web de Compaq
Información sobre productos Compaq
<http://www.compaq.com/>
2001.
32. Jordi Palet
Procesadores RISC, multiproceso y caché
<http://www.consulintel.es/Html/Tutoriales/articulos/risc.html>
Abril de 1995.
33. Cyberlatino.com.mx
Evolución del Hardware de las Pc's
<http://www.cyberlatino.com.mx/info/pc/main.htm#70>

Enero 2001.

34. Raúl Lusky
Maestros del Web, Linux otra opción en sistemas operativos
<http://www.maestrosdelweb.com/editorial/computacion/linux.asp>
Enero 2000.
35. Centro informático Científico de Andalucía
Guía de Seguridad en NT
http://www.cica.es/seguridad/DOCUMENTACION/guia_NT.es.html
Mayo 2000.
36. Ricardo Orduz Marzal
Introducción a la tecnología VLSI
<http://www.maestrosdelweb.com/editorial/computacion/linux.asp>
Diciembre 2000.
37. Paul Boutin and Oscar Montero
Linux vs NT
<http://www.winred.com/colaboradores/colaborador92.htm>
Febrero del 2000.
38. Martín A. Marqués
¿Porqué la gente usa Windows?
<http://math.unl.edu.ar/~martin/articulo.html>
Marzo 2000.
39. José Rubí
ADO ... Cursores, Bloqueos. Como elegirlos?
<http://www.ctv.es/USERS/jrubi/resumen/res00546.htm>
2000 .
40. Compaq,
Servidores
<http://www.computadorasdecalidad.net/>
Venezuela Enero 16, 2001.
41. Yahoo! Inc
Información sobre los Sistemas Operativos actuales
http://www.geocities.com/elplanetamx/Sistema_Operativo.html
Copyright 2001.
42. Ing. Manuel Espinoza Curiel
Sistemas Operativos para redes Cliente /Servidor
<http://www.geocities.com/SiliconValley/8195/noscs.html>
Ultima actualización: 3 de diciembre del 2000.