UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

"Desarrollo de un programa de administración académica de la FES Aragón (Wcronos), dentro de un ambiente (cliente – servidor), utilizando una interfaz RDO (Remote Data Object)"

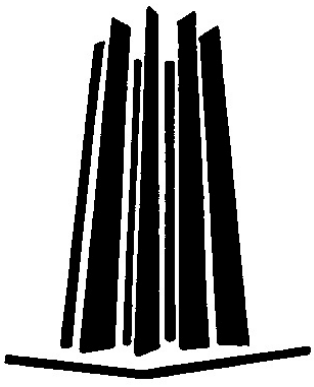
T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

LUIS JAVIER CRUZ MORALES



ASESOR DE TESIS:

ING. VÍCTOR RAÚL VELASCO VEGA

AÑO 2007

AGRADECIMIENTOS

Por la culminación de la carrera de Ingeniería en Computación, agradezco a Dios por darme el don de la vida para llegar hasta este momento.

A mis padres que me dieron la vida y me han brindado cariño, confianza, apoyo moral y parte de su tiempo para salir adelante; guiándome por el camino correcto.

A mi abuelita paterna Josefina (q.e.p.d.), quien veló por mí y mis hermanas, cuando mis padres tenían que trabajar, por enseñarme a ser humilde, justo y dedicado.

A mis abuelos maternos Leonor y Guillermo por brindarme su amistad, cariño y compresión para seguir adelante día a día.

A mis hermanas Silvia Erika y Diana Carolina por la convivencia y el apoyo que nos dimos, más en tiempos difíciles.

A cada uno de mis tíos y tías, los cuales me dieron la motivación para culminar con mis estudios.

A Alejandra, Daygoro Javier y Francisco Javier, los cuales fueron los pilares para soportar los primeros años en la ingeniería, ya que juntos logramos sobresalir y con los dos últimos formamos un trío de Javier.

A mi gran amigo Pascual, el cual me mostró el mejor camino para trabajar en equipo y lograr el mejor rendimiento académico, para cualquier tarea que se tenía que realizar, sin importar el grado de dificultad.

Al Ingeniero Víctor Raúl, por su confianza, apoyo y comprensión, no únicamente para el trabajo, sino para culminar esta tesis.

Al Ingeniero Juan Manuel, el cual no sólo me brindó su amistad y confianza, sino que me mostró como afrontar mi enfermedad, el cual me dio una gran lección al superarse y luchar por la vida.

A cada uno de mis compañeros de informática, con los cuales he vivido muchas experiencias, que me ayudaron en mi superación y desarrollo personal.

A cada uno de mis amigos, amigas, excompañeros y profesores que me brindaron su confianza a lo largo de mi vida para poder llegar hasta este momento.

ÍNDICE

ÍNDICE ..... **II**

OBJETIVOS ..... **V**

INTRODUCCIÓN ..... **VI**

1. TECNOLOGÍA ACTUAL EN BASES DE DATOS **1**

1.1. ANTECEDENTES DE LAS BASES DE DATOS **1**
1.1.1. EL DESARROLLO DE LOS SISTEMAS DE INFORMACIÓN 1
1.1.2. SURGIMIENTO Y PRIMERAS BASES DE DATOS 3
1.1.3. PRIMEROS SISTEMAS DE GESTIÓN DE BASES DE DATOS 3
1.1.4. NUEVOS MODELOS DE SISTEMAS DE GESTIÓN DE BASES DE DATOS 4
1.2. LENGUAJE SQL **6**
1.2.1. CREACIÓN E IMPLEMENTACIÓN 6
1.2.2 COMANDOS BÁSICOS 8
1.3. MODELO CLIENTE-SERVIDOR. **12**
1.3.1. ARQUITECTURA MODELO CLIENTE-SERVIDOR. 12
1.3.2. TIPOS DE CONEXIÓN. 13
1.3.3. INTERFAZ DE COMUNICACIÓN (ODBC). 15
1.3.4. ¿QUÉ ES UNA CONEXIÓN RDO? 16
1.3.5. COMPONENTES. 18
1.3.5.1. Controladores de Cursores. 18
1.3.5.2. Tipo de conjuntos de datos 19
1.3.5.3. Tipos de Bloqueo. 20

2. FUNCIONAMIENTO DE CRONOS **21**

2.1. CONSIDERACIONES PREVIAS **21**
2.2. MÓDULO DE BANCO DE HORAS **24**
2.2.1. ESTRUCTURA DE LA PANTALLA DE CAPTURA DEL BANCO DE HORAS 25
2.2.2. PANTALLA DE REPORTES DEL BANCO DE HORAS 30
2.3. MÓDULO DE HORARIOS **32**
2.3.1. FUNCIONAMIENTO DE LA PANTALLA DE CAPTURA DE HORARIOS 33
2.3.2. MÓDULO DE REPORTES DE HORARIOS 37
2.4. MÓDULO DE EXTRAORDINARIOS **38**
2.4.1. ESTRUCTURA DE CAPTURA DE EXTRAORDINARIOS 39
2.4.2. PANTALLA DE REPORTES DE EXTRAORDINARIOS 42
2.5. MÓDULO DE FINALES **43**
2.5.1. FUNCIONAMIENTO DE LA CAPTURA DE FINALES 43
2.5.2. PANTALLA DE REPORTES DE EXÁMENES FINALES 45
2.6 OTROS PROGRAMAS A CONSIDERAR **46**
2.6.1. PROGRAMA DE CAPTURA DE ORDINARIOS Y EXTRAORDINARIOS DE SERVICIOS ESCOLARES (NEW) 46
2.6.2. PROGRAMA DE SECRETARÍA ACADÉMICA PARA FIRMA DE ASISTENCIA (KARDEX) 49

3. DESARROLLO DE WCRONOS. 52

3.1. RESULTADOS DEL ESTUDIO PREVIO. 52
3.2. ANÁLISIS DEL PROYECTO. 52
3.2.1. IMPLEMENTACIÓN EN EL SERVIDOR 53
3.2.1.1. Programas a utilizar 53
3.2.1.2. Microsoft Windows 2000 Server 54
3.2.1.3. Microsoft SQL Server 2000 55
3.2.1.4. Microsoft Visual Basic 56
3.2.1.5. Programa Administrador 57
3.3. BASES DE DATOS 60
3.3.1. MODELO ENTIDAD-RELACIÓN 60
3.4. CONSTRUCCIÓN DEL PROGRAMA 61
3.4.1. MATERIAS 63
3.4.1.1. Módulo base de inicialización del formulario 65
3.4.1.2. Módulo de conexión y desconexión del entorno RDO 66
3.4.1.3. Módulo base de llenado de un *Combobox*. 67
3.4.1.4. Módulo para dar formato y el llenado de la cuadrícula 68
3.4.1.5. Validación de datos al insertar registro en el formulario 70
3.4.1.6. Módulo de inicialización de la pantalla del formulario 71
3.4.1.7. Módulos de los procesos por botón 72
3.4.1.8. Módulo de intercambio de la cuadrícula con los demás objetos 77
3.4.2. PROFESORES 79
3.4.3. BANCO DE HORAS 82
3.4.4. HORARIOS. 89
3.4.5. EXTRAORDINARIOS. 102
3.4.6. FINALES 108
3.5. PROGRAMA PARA EXPORTAR A LAS DIFERENTES ÁREAS 110

4. GENERACIÓN DE REPORTES CON CRYSTAL REPORTS. 111

4.1. CRYSTAL REPORTS 111
4.2. VISTAS EN LA BASE DE DATOS EN SQL SERVER 2000. 112
4.3. EVALUACIÓN Y MEJORAS DE LOS REPORTES DEL PROGRAMA. 113
4.3.1. MEJORAS EN CADA UNO DE LOS REPORTES. 115
4.3.2. CONSTRUCCIÓN DE LOS REPORTES. 116
4.3.2.1. Banco de horas. 117
4.3.2.2. Horarios 121
4.3.2.3. Extraordinarios 123
4.3.2.4. Finales 124
4.3.3. INTERFASE ENTRE EL PROGRAMA DE CRYSTAL REPORTS Y VISUAL BASIC. 125
4.3.4. FUNCIONAMIENTO DEL VISUALIZADOR DE CRYSTAL REPORTS. 127

5. PERSPECTIVA DE EVOLUCIÓN DEL SISTEMA. 129

5.1. IMPLEMENTACIONES EN EL PROGRAMA EN UN FUTURO. 129
5.1.1. GENERACIÓN DE ESTADÍSTICAS 129
5.1.2. PLANTA FÍSICA 130
5.1.2.1. Visualizador de espacios ocupados 130

5.1.3. EVALUACIONES DE PROFESORES. 131
5.1.3.1. Captura de los cuestionarios de las evaluaciones de profesor..... 132
5.1.3.2. Captura en línea de los cuestionarios de las evaluaciones de profesor 133
5.1.3.3. Generación de reportes y estadísticas. 135
5.1.4. PROPUESTAS DE PROFESORES 135
5.2. BENEFICIOS GENERADOS..... 135
5.2.1. BENEFICIOS AL INTEGRAR WCRONOS 136
5.2.2. ESTANDARIZACIÓN DE LAS BASES DE DATOS...... 136
5.3. BASES PARA CREAR UNA APLICACIÓN DENTRO DEL SERVIDOR. 137

CONCLUSIONES..... 139

BIBLIOGRAFÍA..... 140

GLOSARIO 142

OBJETIVOS

PRIMARIO

- Desarrollar un programa en un ambiente Cliente-Servidor con apoyo de una interfaz RDO (Remote Data Object), que sustituya el programa “Cronos” que trabaja en un ambiente de MS-DOS dentro de una red Novell, el cual es un programa de administración académica de la FES Aragón.

SECUNDARIOS

- Explicar cada uno de los conceptos de los componentes básicos que integra un ambiente Cliente-Servidor.

- Definir en qué consiste una interfaz RDO (Remote Data Object).

- Analizar el funcionamiento de Cronos y los programas con los cuales se puede integrar (New, Kardex).

- Mostrar cómo se construyó el Sistema Gestor de Bases de Datos.

- Construir una estructura de la base de datos de tipo relacional, para ser implantada en la base de datos.

- Crear un modelo de programación para cada uno de los módulos de captura que componen la aplicación del cliente, para que pueda ser fácilmente utilizado por otro programador.

- Integrar una mejor forma para presentar cada uno de los reportes que tiene Cronos.

- Mostrar las posibles perspectivas de crecimiento o evolución del programa.

INTRODUCCIÓN

En todos los ámbitos laborales existen proyectos o actividades que un ingeniero tiene que afrontar día a día; los cuales pueden ser de corto, mediano y largo plazo, el profesional debe estar preparado para afrontarlo, usando como apoyo el perfil de la carrera que estudió.

En mi caso estudié la carrera de Ingeniería en Computación, en donde no sólo adquirí conocimientos, también aprendí a tomar decisiones, a analizar las necesidades de la empresa, y a actualizar los procesos con los que se trabaja.

Se debe elegir la mejor planeación en especial para los proyectos de largo plazo, los cuales conllevan una mayor dificultad, uno de ellos se consideró para realizar esta tesis, por el grado de complejidad y la importancia que representa para la FES Aragón.

El objetivo del proyecto es rediseñar el programa “Cronos”, que trabaja en un ambiente de MS-DOS en una red Novell a un sistema cliente-servidor, el cual es un programa que tiene el propósito de administrar las actividades académicas del personal docente que trabaja en la facultad y de integrarse a las áreas administrativas que requieran la información.

En el primer capítulo se expone los temas que se requiere para el manejo de la migración de sistemas, y así cualquier lector comprenderá como se va a realizar el cambio de éste.

Asimismo los temas se abordan de acuerdo a cómo se fueron desarrollando las bases de datos, desde que era un simple sistema de información hasta los actuales Sistemas Gestores de Bases de Datos, que se requieren para su funcionamiento en el lenguaje “SQL”.

Lo anterior es una pieza fundamental para un ambiente cliente-servidor, donde se explica cada una de sus partes, en especial la interfaz de conexión entre la aplicación y la base de datos de la cual se usa en el programa RDO (Objetos de Datos Remotos).

Dentro del capítulo II se muestra un análisis detallado del funcionamiento y de las partes que comprende el programa “Cronos”, como son:

- Banco de Horas
- Horarios
- Extraordinarios
- Finales

Cada uno se compone de dos fases, la primera es donde se captura la información, considerando todos los detalles de su funcionamiento y otra donde se tienen los diversos tipos de reportes.

También se incluyen otros programas que se relacionan con “Cronos”, como “NEW” de servicios escolares y el de Kardex para la impresión de las tarjetas de asistencias, los cuales se tienen que fusionar con el programa para evitar doble captura de datos.

En el capítulo III se muestra los pasos a seguir para implementar el sistema en un ambiente Cliente-Servidor, considerando todos los recursos con los que se cuenta para primero realizar la implantación del servidor de bases de datos, explicando de manera breve los programas que intervienen (Windows 2000, SQL Server 2000, Visual Basic 6.0 y el programa administrador de las bases de datos).

Se continuará con la construcción de la base de datos que es la parte esencial del proyecto, porque a partir de ella se realizó la construcción del programa. En él se creó un modelo genérico, para que cualquier programador lo pueda modificar en la etapa de captura.

En el capítulo IV se explica cómo se desarrollaron los reportes, con apoyo del programa Crystal Reports que interactúa con Visual Basic y la base de datos, (donde no se usa las tablas, sino las Vistas de las tablas ya relacionadas). Ésta es una herramienta muy poderosa en donde se puede crear o modificar reportes de una manera rápida y sencilla.

Por último, en el capítulo V se contemplan las actualizaciones que se aplicarán ya sea al programa o que tomen la información de la base de datos, además se mencionan los beneficios que se obtuvieron a partir del momento en que se implantó en la FES Aragón y un posible desarrollo de la aplicación dentro del servidor.

En esta tesis se abarcan varios temas relacionados con la aplicación, pero se omiten ciertos lineamientos que se utilizaron para la seguridad del sistema y de la información confidencial que se maneja, ya que el programa está en funcionamiento hasta la fecha de la culminación de esta tesis.

CAPÍTULO

1. TECNOLOGÍA ACTUAL EN BASES DE DATOS

1.1. Antecedentes de las Bases de Datos

Las Bases de Datos son de gran importancia actualmente, ya que manejan grandes cantidades de información que se utilizan para distintos procesos. En este sentido es necesario conocer sus antecedentes y cómo han evolucionado a través del tiempo.

Para ello se tiene que conocer primero lo que es un "Sistema de Información", el fundamento de las primeras bases de datos, y posteriormente los sistemas gestores de bases de datos que son actualmente de gran ayuda para el manejo de grandes cantidades de información, porque son un recurso esencial para su crecimiento, desarrollo e implantación en muchas empresas.

1.1.1. El desarrollo de los Sistemas de Información

Desde los principios de la historia, el hombre ha vivido en comunidades, en cada una de ellas existía un líder que tenía la necesidad de tomar decisiones para poder sobrevivir dentro de su medio ambiente; usando como medio de información las pinturas sobre rocas, los gruñidos y gestos; sin embargo, con el paso del tiempo la interacción verbal se abrió camino en aquellas primitivas civilizaciones.

Conforme transcurrió el tiempo, las civilizaciones crecieron y evolucionaron de tal manera que la cantidad de datos e información también se incrementó. Esto lo llevó a buscar nuevas formas para poder ingresarlos y tener un control sobre ellos. El acontecimiento más importante para dicho desarrollo se debió al descubrimiento del papel. En donde el antecedente más remoto que se tiene de un sistema de información se encontró en los censos romanos.

Más tarde, con apoyo de otro invento que fue la "la imprenta", desarrollada durante la revolución industrial (1440 D. C.), se abrirían las puertas al flujo de información en grandes masas, lo cual representaba un gran paso para el ser humano.

Antes de la Primera Guerra Mundial surgió un período de análisis ya que los científicos eran los poseedores de los conocimientos, y se encargaban de estudiar sus partes y poder mejorar los desarrollos tecnológicos.

En el período comprendido entre las dos guerras mundiales se generó una gran cantidad de información; además de que se hizo necesaria la disminución de los tiempos de respuesta, lo cual favoreció el aceleramiento científico, la creación de nuevos inventos y

conocimientos; uno de los más importantes de la era moderna fue la invención de la computadora.

Desde entonces el procesamiento de información resulta imprescindible y ha sido mejorado año tras año, de tal manera que en la actualidad los sistemas informáticos representan la punta tecnológica para el control de la información.

El uso de la computadora, actualmente, representa el mayor logro para los sistemas informáticos, ya que permite manejar una enorme cantidad de datos, además de que se facilitó el almacenamiento, procesamiento y recuperación de información.

La recuperación de información está vinculada con la representación del conocimiento registrado en documentos en un período; con las necesidades de información de los usuarios del sistema y con el desarrollo de una función, es capaz de comparar ambas y seleccionar los documentos más relevantes para solucionar la necesidad de información y facilitar la toma de decisiones.

Todos los servicios de documentación están inmersos en lo que se llama "Sistemas de Información", los cuales obtienen, almacenan, recuperan y distribuyen datos sobre el conocimiento registrado en un depósito o en una red de documentos. Recoleccionan, almacenan, procesan y recuperan datos, con el fin de disminuir la incertidumbre en la toma de decisiones, mediante el suministro de la información. El equipo de trabajo administrativo es el que diseña y maneja el sistema.

Los sistemas de información pueden analizarse, diseñarse y administrarse como tales mediante los principios generales de diseño. La salida de un sistema de información tiene como objetivo principal que una decisión importante resulte de los datos suministrados, lo que a su vez da por resultado algún tipo de acción que corrige la misma salida, y lleva a otra decisión.

Las necesidades de información se hacen más complejas, en la medida en que se agrandan las operaciones de la organización, así como la forma en que pueden desarrollarse y como mejorar los sistemas de información, a través de la modificación de un sistema de información, un manual o un diseño de un programa de computadora; pero antes se tiene que hacer un análisis tomando en cuenta:

a) Requisitos
b) Conocimientos
c) Objetivos
d) Características del ambiente o sistema
e) Sus operaciones
f) Recursos
g) Políticas
h) Procedimientos

1.1.2. Surgimiento y primeras Bases de Datos

Los primeros sistemas de base de datos aparecieron a finales de los cincuenta. En este período, muchas compañías dieron cuenta de que los sistemas informáticos brindaban la posibilidad de aplicar soluciones mecánicas más baratas y eficientes.

Primero se utilizaron las máquinas que codificaban la información en tarjetas perforadas, con el objetivo de almacenar grandes cantidades de datos en tarjetas y medios magnéticos, sin embargo el proceso era lento, costoso y muy complejo.

Un avance muy importante lo constituyó el comité formado en la *"Conference on Data Systems and Languages"* (CODASYL), y en 1960 se estableció el *"Common Business Oriented Language"* (COBOL) como lenguaje estándar para interrelacionar con datos almacenados en ficheros. Las instrucciones específicas de un programa en COBOL para tratamiento de ficheros eran las de abrir un fichero, leerlo y añadirle un registro.

Pronto los discos magnéticos empezaron a sustituir a las cintas magnéticas, lo cual supuso una reconcepción del proceso de almacenar datos, es decir, se pasó del acceso secuencial al acceso aleatorio.

Durante los sesenta aparecieron distintos modelos de datos, con el objetivo de conseguir una independencia un poco mayor entre las aplicaciones y la organización física de los datos.

1.1.3. Primeros Sistemas de Gestión de Bases de Datos

La tecnología actual con grandes volúmenes de información requiere la delegación de la gestión de varios tipos de aplicaciones de software, denominadas *"Sistemas de Gestión de Base de Datos"* (SGBD) o simplemente, sistemas de bases de datos.

Las funciones básicas de un sistema de gestión de base de datos son:

1. Permitir a los usuarios crear nuevas bases y especificar su estructura, utilizando un lenguaje o interfaz especializado, llamado lenguaje o interfaz de definición de datos.

2. Dar a los usuarios una posibilidad de consultar los datos (es decir, recuperarlos parcial o totalmente) y modificarlo por medio de un lenguaje de consulta o de manipulación de datos.

3. Permitir el almacenamiento de grandes cantidades de datos durante un largo período, manteniendo la seguridad de los datos en cualquier tipo de evento.

4. Controlar el acceso a los datos de muchos usuarios a la vez, evitando que el acceso simultáneo los corrompa.

Entre los modelos más populares en los sesenta se encuentran los siguientes:

El modelo jerárquico se deriva de los sistemas de gestión de información, como el que IBM introdujo (el sistema IMS *Information Management System*) en 1980, derivado del programa Apolo de la NASA sobre sus *System/360*. Se basa en almacenar los datos en una serie de registros, los cuales tienen campos asociados a ellos; para crear enlaces entre los tipos de registros, el modelo utiliza las relaciones padre-hijo, correspondencias 1:N entre tipos de registro (Árboles)[1].

El modelo de red se estandarizó por un informe de CODASYL en 1968, donde se formó un grupo denominado DBTG (*Data Base Task Group*)[2]. Pero con el paso del tiempo en 1971 se creó un nuevo informe con base en muchos comentarios y revisiones de expertos y/o usuarios. El modelo de red se llama así porque representa los datos que contiene en forma de una red de registros y conjuntos (en realidad son listas circulares llamadas "sets") que se relacionan entre sí, formando enlaces; para ello se utilizan registros, tipos de registros y tipos de conjuntos.

Los sistemas jerárquicos y de red constituyen lo que es la primera generación de los *"Sistemas de Gestión de Bases de Datos"*. Pero presentan algunos inconvenientes:

- Es necesario escribir complejos programas de aplicación para responder a cualquier tipo de consulta de datos, por simple que ésta sea.
- La independencia de los datos es mínima.
- No tienen un fundamento teórico.

1.1.4. Nuevos Modelos de Sistemas de Gestión de Bases de Datos

En junio de 1970, Edgar F. Ted Codd, un matemático formado en Oxford de los laboratorios de investigación de IBM, escribió un artículo *"A Relational Model of Data for Large Shared Data Banks"* (Un modelo relacional de datos para grandes bancos de datos compartidos), donde se presenta el modelo relacional, que fue parte de la segunda generación de SGBD.

En él, Codd propuso que los sistemas de bases de datos deberían presentarse a los usuarios con una visión de los datos organizados en estructuras llamadas relaciones. Éstas se definen como conjuntos de tuplas, no como series o secuencias de objetos, con lo que el orden no es importante, por lo que detrás de una relación puede haber cualquier estructura de datos compleja que permita una respuesta rápida a una variedad de consultas.

[1] Todas las relaciones están jerarquizadas en un árbol, por lo que no es capaz de establecer enlaces entre hijos o entre capas, si no es padre-hijo.
[2] Formado por representantes del gobierno de EEUU y representantes del mundo empresarial.

Entonces, se comenzaron a desarrollar muchos sistemas relacionales, apareciendo los primeros a finales de los setenta y a principios de los ochenta. Tal como se ve en las cantidades del cuadro 1.1[3].

	1991	1992	1993	1994	1995	1996	1997	1998	1999
Modelos Anteriores	2000	2090	2109	2050	1866	1721	1701	1689	1638
Modelo Relacional	1844	2502	3328	4435	5925	7652	10513	11685	14254

Cuadro 1.1: Ventas Mundiales Modelo Relacional vs Modelos Anteriores.

Uno de los primeros es *System/R*, de IBM, que se desarrolló para probar la funcionalidad del modelo, proporcionando una implementación de sus estructuras de datos y sus operaciones. Esto condujo a dos grandes desarrollos:

a. El desarrollo de un lenguaje de consultas estructurado SQL (El cual se verá posteriormente).

b. La producción de varios SGDB relacionales durante los ochenta.

A partir del punto dos, aumentaron la producción de varios SGBD relacionales, tanto para microordenadores como para sistemas multiusuario, aunque no son completamente fieles al modelo relacional.

Como sistemas relacionales multiusuario se mencionan DB2 y SQL/DS de IBM, ORACLE de ORACLE Corporation, INGRES de Computer Associates, Informix de Informix Software Inc., y Sybase de Sybase Inc.

En lo que respecta a sistemas relacionales de microordenadores son Paradox, dBase III, y dBase IV de Borland, Access de Microsoft, Foxpro y R:base de Microrim.

Los SGBD relacionales constituyen la segunda generación de los SGBD. Sin embargo, el modelo relacional también tiene sus fallos, siendo uno de ellos su limitada capacidad para modelar los datos. Por esta razón se han realizado diversas investigaciones para resolver el problema. Por ejemplo, en 1976, Chen presentó el modelo entidad-relación, que es la técnica más utilizada en el diseño de bases de datos. En 1979, Codd intentó subsanar algunas de las deficiencias de su modelo relacional con una versión extendida denominada RM/T (1979) y la más reciente RM/V2 (1990). Los intentos de proporcionar un modelo de datos que representen al mundo real de un modo más fiel, han dado lugar a los modelos semánticos.

Como respuesta a la creciente complejidad de las aplicaciones que requieren bases de datos, han surgido dos nuevos modelos: el de datos orientados a objetos y el relacional extendido. Sin embargo, a diferencia de los modelos que los preceden la composición no está clara.

[3] Artículo Hernández Orallo, José. "La Disciplina de los Sistemas de Bases de Datos".

1.2. Lenguaje SQL

Para comprender el funcionamiento de un Gestor de bases de datos, se debe tener en cuenta lo siguiente.

1.2.1. Creación e Implementación

La popularidad de SQL *Structured Query Language* (Lenguaje de Estructuras de Consultas) dentro de las bases de datos es importante en nuestros días, ya que es un lenguaje que se usa para interaccionar con varias tipos de bases de datos en especial con la base de datos relacional.

SQL es una herramienta para organizar, gestionar y recuperar datos almacenados en una base de datos informática. Fue inventado por investigadores de IBM, en especial por concepto de base de datos relacional, publicado por Dr. Codd en la revista científica *Communications of the Association for Computing for computing Machinery*.

El artículo desencadenó una racha de investigaciones en bases de datos relacionales, uno de ellos fue el proyecto de investigación dentro de IBM llamado *System/R*. El objetivo de éste era demostrar la operabilidad del concepto relacional y proporcionar alguna experiencia en la implementación de SGBD, a mediados de los sesenta en los laboratorios de Santa Teresa en San José California.

En 1974 y 1975 la primera fase del proyecto *System/R* produjo un pequeño prototipo en donde se incluían trabajos sobre lenguajes de consulta de base de datos, uno de ellos fue denominado SEQUEL[4].

En 1976 y 1977 el prototipo de investigación del *System/R* fue reescrito desde el principio, ya soportaba consultas multitablas y permitía que varios usuarios compartieran el acceso a datos. Esta nueva implementación fue distribuida a una serie de instalaciones de usuarios de clientes para ser evaluada en 1978 y 1979. El lenguaje de base de datos fue renombrado a SQL o *Structure Query Language* (Lenguaje Estructurada de Consultas), por razones legales. A pesar del cambio de nombre, la pronunciación SQL permaneció. En 1979 el proyecto de investigación *System/R* llegó al final y su lenguaje recibió buenos comentarios en las revistas técnicas.

Pero también el primer proyecto de *System/R* atrajo la atención de un grupo de ingenieros en Menlo Park, quienes formaron una compañía llamada Relational Software, Inc., para SGBD basado en SQL, apareciendo su producto (Oracle) en 1979, éste se convirtió en el primer SGBD relacional comercialmente disponible, mucho mejor que el creó IBM.

Otros profesores de los laboratorios informáticos de la Universidad de California, en Berkeley, también crearon su prototipo de un SGBD relacional, al que llamaron Ingres, que

[4] Se tomó como un sinónimo de Structure English Query Language (Lenguaje Estructurado de Consultas Inglés).

incluía un lenguaje de consulta llamado QUEL[5], después abandonaron Berkeley y fundaron la empresa Relational Technology en 1981.

La fuerza en los mercados comerciales, al tener una base de datos, acrecentaba la popularidad de SQL en los ochenta, en especial a finales cuando IBM lanzó al mercado DB2 como la solución de un SGBD, pero también en varias tipos de computadoras y sistemas operativos del cuadro 1.2.

SGBD	Sistemas Informáticos
DB2	Maxicomputadores IBM bajo MVS
SQL/DS	Maxicomputadores IBM bajo VM y DOS/VSE
Rdb/VMS	Minicomputadores VAX/VMS de Digital
Oracle	Maxicomputadores, minicomputadores y PCs.
Ingres	Minicomputadores y PC's
Sybase	Minicomputadores y LAN
Informix-SQL	Minicomputadores y PC's basados en UNIX
Unify	Minicomputadores basados en UNIX
OS/2 Extended Edition	Sistemas PS/2 de IBM bajo OS/2
SQL Server	PC LAN basados en OS/2
SQLBase	PC LAN basados en DOS y OS/2
dBase IV	PC's y PC LAN.

Cuadro 1.2: Sistemas Gestores de Bases de Datos y los sistemas que se utilizaban.

Uno de los desarrollos más importantes al ser aceptado SQL en los mercados, es que se hizo oficial el hecho de que se había convertido en el estándar para las base de datos relacionales. Esto comenzó en 1982 cuando ANSI encargó al comité X3H2 que seleccionara un lenguaje de consulta de datos, quien eligió SQL y se aplicó a estandarizarlo.

Después de varias revisiones, fue oficialmente adoptado como estándar ANSI X3.135 en 1986 y como estándar ISO en 1987. El estándar ANSI/ISO ha sido adoptado desde entonces como estándar de Federal Information Processing Standard (FIPS) por el gobierno de los Estados Unidos.

También fue tomado en cuenta SQL por los estándares X/OPEN que son comercializados en los mercados europeos, por su portabilidad entre sistemas informáticos, pero por desgracia X/OPEN difiere de ANSI/ISO en varios aspectos.

A pesar de la existencia de un estándar, ningún producto SQL comercial disponible hoy en día se conforma exactamente a él, los vendedores introducen nuevas capacidades y amplían su dialecto en el lenguaje SQL.

[5] Según el Libro "Aplique SQL" era un lenguaje más "estructurado" que SQL, pero sus sentencias no tenían relación con el inglés, lo que lo hizo más complicado, por lo cual posteriormente fue sustituido por el SQL en 1986.

1.2.2 Comandos Básicos

Las sentencias dentro de SQL se clasifican en tres lenguajes o sublenguajes:

1. **Lenguaje de Definición de Datos (Data Description Language (DDL**)): es el conjunto de sentencias que crean o modifican la estructura de la base de datos, lo cual permite crear, modificar, borrar y definir nuevas bases de datos, campos, vistas e índices.

Los comandos básicos del Lenguaje de definición de datos se verán a continuación, pero dependiendo del sistema gestor de base de datos que lo maneje, se van agregan más comandos y más funciones dentro del lenguaje.

CREATE: Se utiliza para crear nuevas bases de datos, tablas, campos, índices o vistas.

Ejemplo:

- Creación de una tabla de "MATERIAS".

```
CREATE TABLA MATERIAS
        (NUMMAT INTEGER NOT NULL,
         CLVMAT INTEGER NOT NULL,
        MATERIA VARCHAR (45) NOT NULL,
         SEMESTRE INTEGER NOT NULL)
```

DROP: Con este comando se eliminan bases de datos, tablas, índices y vistas.

Ejemplo:

- Eliminación de la tabla.

```
DROP TABLE MATERIAS
```

ALTER: Permite modificar las tablas agregando campos o cambiando la definición de los campos.

Ejemplo:

- Agregar una columna a la tabla de 'Materias'.

```
ALTER TABLE MATERIAS
        ADD CREDITOS INTEGER NOT NULL
```

2. **Lenguaje de Control de Datos (Data Control Language (DCL)):** conjunto de comandos que sirve para trabajar en un entorno multiusuario, en donde es muy importante la protección, la seguridad y los permisos a los usuarios.

Los comandos básicos son los siguientes:

GRANT: Se otorgan privilegios a los usuarios en una determinada tabla.

Ejemplo:

- Se da permiso al usuario de "Civil", de usar las instrucciones de 'select' e 'insert' en la tabla de 'Materias'.

```
GRANT SELECT, INSERT
        ON MATERIAS
        TO CIVIL
```

REVOKE: Permite suprimir privilegios previamente definidos en una tabla.

Ejemplo:

- Se revocan los permisos de borrar y actualizar la tabla de 'Materias'

```
REVOKE DELETE, UPDATE
        ON MATERIAS
        TO CIVIL
```

3. **Lenguaje de Manipulación de Datos (Data Manipulation Language (DML)):** es el conjunto de instrucciones que realizan operaciones dentro de una tabla o conjunto de tablas, que permiten generar consultas, modificaciones y eliminación de registros de la base de datos.

SELECT: Se usa para consultar registros de la base de datos que satisfagan un criterio determinado, y es el comando más usado en SQL, ya que con éste se realizan las consultas.

Ejemplo:

- Consulta sencilla de la lista de 'Materias' donde la clave sea mayor a cero y menor a 9999, ordenado por el nombre de materia.

```
SELECT CLVMAT, MATERIA, CREDITOS
WHERE CLVMAT > 0 AND CLVMAT <= 9999
ORDER BY MATERIAS
```

INSERT: Inserta datos en una tabla de la base de datos en una única operación.

Ejemplo:

- Inserción de un registro en la tabla de materias.

```
INSERT MATERIAS
        (nummat, Clvmat, Materia, Semestre, Creditos)
VALUES
        (5, 1254, 'SISTEMAS OPERATIVOS', 1, 5)
```

UPDATE: Modifica los valores de los campos y registros especificados.

Ejemplo:

- Modificación de un registro de materias

UPDATE MATERIAS
SET CREDITOS = CREDITOS + 1
WHERE MATERIA = 'SISTEMAS OPERATIVOS'

DELETE: Elimina los registros de una tabla de una base de datos, bajo un criterio.

Ejemplo:

- Eliminación de un registro donde la clave de materias se igual a '1025'.

DELETE FROM MATERIAS
WHERE CLVMAT=1025

Como se ve en los ejemplos el lenguaje de SQL está conformado de sentencias en inglés que indican una sintaxis, con lo que se desea obtener y de donde obtenerlo, siguiendo una estructura para que se ejecute correctamente, aplicándose la misma regla para cada uno de los lenguajes que lo conforman.

Tal como se muestra con la instrucción Select:

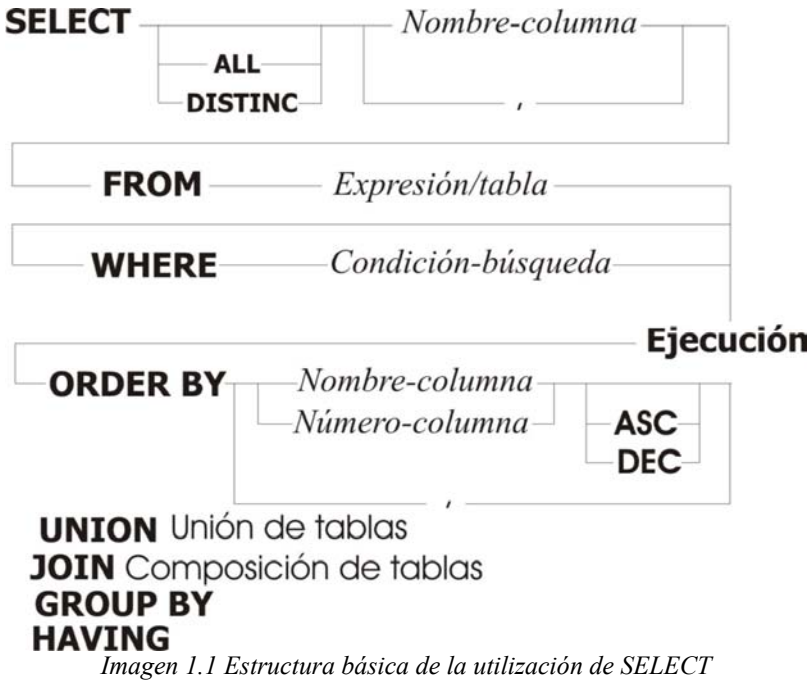


Imagen 1.1 Estructura básica de la utilización de SELECT

Ejemplo sencillo de una consulta:

SELECT CLVMATERIAS, MATERIAS, CREDITOS
FROM MATERIAS
WHERE CREDITOS > 10
ORDER BY MATERIAS

Como se ve en el cuadro de la instrucción 'Select' los últimos componentes son cláusulas, que son las condiciones de selección para definir los datos que se desean.

En las cuales son:

FROM Se utiliza para especificar la tabla a usar.

ORDER BY Ésta se usa para ordenar los registros con un condición o una columna determinada.

GROUP BY Permite agrupar los registros en grupos específicos.

WHERE Es una de de las instrucciones más importantes para definir la condición de búsqueda de los registros.

HAVING Es similar a la instrucción 'Where', pero se usa con 'Group By', sólo que expresa además una condición que debe satisfacer cada grupo.

JOIN Sirve para juntar dos o más tablas.

Las dos últimas instrucciones necesitan parámetros para realizar una o más condiciones en un campo, para ello se deben apoyar en un operador de comparación, que se muestran en el cuadro 1.3.

Comparador	Operación
<	Menor que
>	Mayor que
<>	Distinto de
<=	Menor o igual que
>=	Mayor o igual que
=	Igual que
BETWEEN	Se usa para especificar un intervalo de valores
LIKE	Se usa para comparar en base a un modelo
IN	Para especificar registro de una base de datos.

Cuadro 1.3.: Operadores lógicos.

Por último, falta mencionar lo que son funciones de agregado, pero siempre hay que tomar en cuenta que sólo devolverán un valor por cada función, éstas se muestran en el cuadro 1.4.

Función	Proceso que realiza
AVG	Calcula el promedio de los valores de los registros seleccionados.
COUNT	Suma el número de registros seleccionados
SUM	Suma los valores de los registros seleccionados.
MAX	Devuelve el valor más alto de los registros.
MIN	Devuelve el valor más bajo de los registros.

Cuadro 1.4: Funciones para campos numéricos.

1.3. Modelo Cliente-Servidor.

En la actualidad los sistemas grandes ya no utilizan el modelo de estrella, jerárquico o de árbol, usan el modelo Cliente – Servidor, con el cual se tienen más ventajas que los anteriores.

La ventaja principal es la reducción considerable en el tráfico de la red, en donde el cliente se conecta al servidor cuando es necesario, obtiene los datos que necesita y cierra parcialmente la conexión para que entre otra.

Otra ventaja es que el servidor ya no requiere de tanta potencia de procesamiento, sino que el proceso se puede repartir a cada uno de los clientes, dependiendo del tipo de arquitectura.

1.3.1. Arquitectura Modelo Cliente-Servidor.

La arquitectura cliente-servidor es una forma de dividir programas y equipos de cómputo a fin de que cada parte realice sus tareas en forma eficiente, permitiendo que se simplifiquen para evitar en lo mayor posible un tráfico en la red.

Por lo que se define como el servidor del equipo de cómputo en donde se almacena una aplicación o programa que realiza algunas tareas o procesos para ser distribuido a un grupo de computadoras (clientes).

En el programa se pueden distinguir tres niveles:

1. Manejador de Bases de Datos (Nivel de Datos).
2. Procesador de aplicaciones o reglas del negocio (Nivel de control/procesos).
3. Interfase del usuario (Nivel de presentación).

La arquitectura Cliente servidor se muestra en el siguiente diagrama:

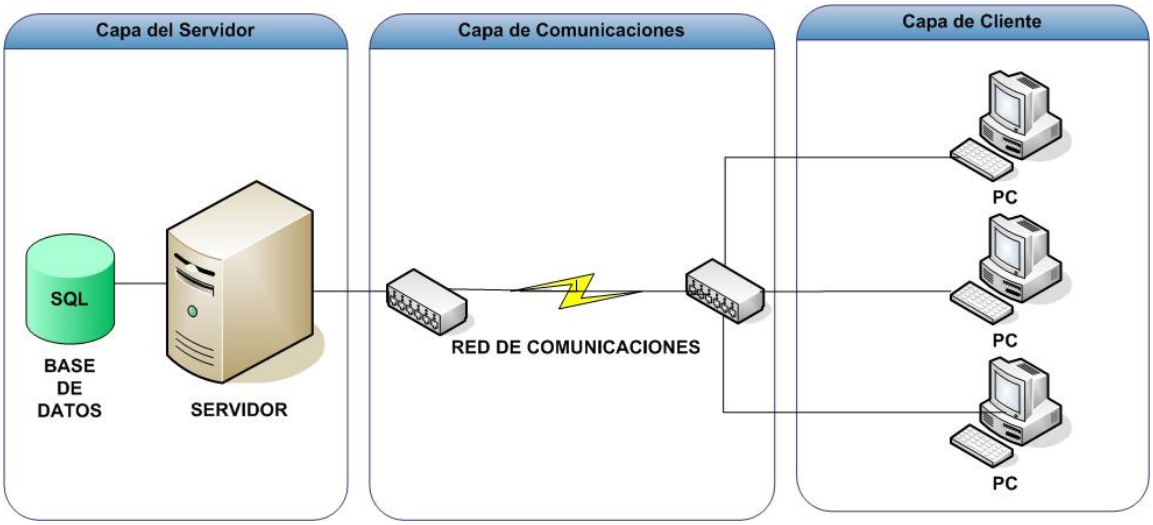


Imagen 1.2 Estructura básica de la arquitectura Cliente-Servidor.

Las aplicaciones cliente/servidor difieren en cómo y en cuántas máquinas se dividan los tres niveles y éstos son los casos:

1) El servidor se encarga de los datos y los procesos, por lo que el cliente sólo trabaja con la interfaz.

2) El servidor se encarga únicamente de los datos, el cliente se encarga de los procesos y la presentación. (A esta arquitectura se le conoce como Cliente/Servidor puro).

3) El servidor se encarga de datos, junto con algunos procesos asociados y el cliente con la presentación y también con procesos. Esta arquitectura es de las más utilizadas ya que se hacen a aplicaciones de Internet con poco tráfico y pocos clientes (menos de 100 clientes).

4) Aquí se tienen dos servidores diferentes, uno como servidor de datos y otro como servidor de aplicaciones, por lo que el cliente sólo se encarga de la interfaz, a esta arquitectura se le conoce como "Arquitectura de tres niveles", que son capaces de soportar mayor cantidad de clientes.

1.3.2. Tipos de Conexión.

Para poder desarrollar un programa uno de los principales puntos en el planteamiento es cómo se puede acceder a la base de datos, en donde debe ser seguro, confiable, conservar la integridad de la información, y tener transferencia rápida.

Los tipos de accesos son los siguientes:

- Para bases de datos locales se puede acceder implementado en Visual Basic con el Jet a través de controles del tipo "Data Control", ésta es la que requiere menos

código, pero cuando se tienen grandes cantidades de datos su rendimiento disminuye.

- Existe otro método en Visual Basic conocido como Data Access Object o DAO (objetos de acceso a datos), su rendimiento es muy bueno al acceder a una base local, pero no es lo mismo cuando el acceso es remoto, ya que requiere una combinación con una interfaz ODBC, lo cual se tiene que configurar en cada una de las máquinas donde se trabaja.

- En la mayoría de lenguajes, a parte de Visual Basic, se pueden utilizar las APIs del driver ODBC para acceder a cualquier base de datos externa mediante driver ODBC, ya sea con la ayuda de un origen de datos. Estos orígenes de datos se almacenan en el CPU cliente con un nombre definido por el usuario, es decir DNS (Domain Name System) o se almacena en el servidor pero se denomina DNS de sistema que puede ser pedido al servidor por cualquier usuario.

- Se pueden utilizar las APIs de la librería OLEDB con el control VBSQL para acceder a bases de datos SQL Server.

- Existe otra forma de conectarse y es por medio de Remote Data Object (**RDO**), esto en Visual Basic.

- Se pueden utilizar programas y controladores de acceso a bases de datos de terceros fabricantes.

Los dos primeros casos no se tomarán en cuenta, ya que se conectan a la base de datos de forma local para que su funcionamiento sea idóneo, por lo que se tomarán en cuenta los siguientes casos.

En el caso de ir por la vía de las API's del driver ODBC, su rendimiento es bueno, pero el problema que tiene es que la programación es complicada, y durante la fase de desarrollo es muy inestable, se debe tener un alto grado de complejidad para determinar los errores, ya que es necesario investigar tanto en el protocolo de comunicación TCP/IP, como en la base de datos.

Si se opta por la vía del uso del control OLE, se requiere declarar ficheros adicionales, drivers y DLL's que no están incluidas en Visual Basic, y que han de adquirirse por separado. Facilita un conjunto de funciones que permiten un alto grado de control sobre el servidor de base de datos externo, y por consiguiente, obtener altos rendimientos, complica mucho la programación.

En el caso de los Remote Data Object (en adelante **RDO**), los rendimientos son perfectamente equiparables al caso de las API's, pero con tiempos de desarrollo mejores y posibilidades más amplias, pero en un caso muy concreto al utilizar una base de datos SQL Server.

En el último caso se depende mucho de los archivos que da el fabricante, por lo que hay que estar consultando cuando se tenga un error en la base de datos y por ello se tiene que checar la página del fabricante en la sección de soporte o unirte a un foro por medio de Internet, para quizás obtener una mejor respuesta al problema planteado.

1.3.3. Interfaz de Comunicación (ODBC).

Para nuestro problema se considera la interfase RDO, pero antes de abordarlo, se tiene que explicar ¿Qué es ODBC?; ya que trabaja en forma conjunta.

ODBC las siglas de Open Data Base Connectivity (conectividad abierta de bases de datos) es un estándar de conexión entre bases de datos desarrollado por Microsoft Corporation, por medio de una interfase de acceso común, usando de apoyo las expresiones básicas de SQL.

El objetivo es hacer posible el acceso a la información desde cualquier aplicación, sin importar que Sistemas Gestor de Bases de Datos almacene los datos, ODBC logra esto al insertar una capa intermedia llamada manejador de Bases de Datos entre ambas partes, ésta tiene como propósito traducir las consultas de datos de la aplicación en comandos compatibles con el SGBD.

Para que funcionen los orígenes de datos de ODBC, tanto la aplicación como el SGBD deben ser programas compatibles, en donde se puede crear un DNS (Servidor de nombres de dominio) que defina los parámetros, ruta y características de una conexión segura.

Una gran parte de las Bases de datos actuales cuentan con una interfaz ODBC, aunque se tienen ciertas limitaciones, tales como la velocidad de transferencia, por ello se usan otras técnicas de programación, tal como RDO, que se verá más adelante.

ODBC provee de características siempre homogéneas, y permite distintos controladores que aseguran la conectividad de la aplicación con diferentes bases de datos, tal como se muestran en la figura 1.3.

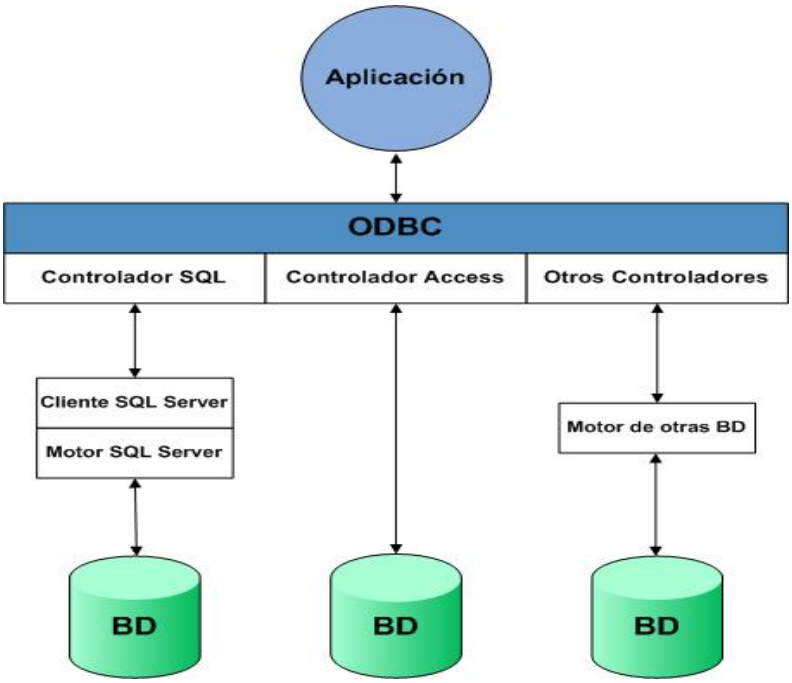


Figura 1.3 Esquema del funcionamiento ODBC.

1.3.4. ¿Qué es una conexión RDO?

RDO (Objetos de Datos Remotos) no es más que la implementación de un conjunto de objetos, consiste en una delgada capa de código sobre las APIs del driver ODBC (de ahí que los rendimientos sean similares) y un control del driver que establece conexiones, en donde se crea un entorno de trabajo para la base, los cursores, y la ejecución de procedimientos complejos usando un mínimo de recursos. Su interconexión con SQL Server 2000 es tan estrecha, que sin duda, la convierte en la herramienta más práctica para acceder a dicha base de datos (equiparable sólo con OLEDB).

En el momento de la conexión, lo que más tarda es el RDO, lo cual es lógico si se piensa que la estructura de objetos que se monta en ese momento es la más complicada. Sin embargo, a partir de este momento, la mejora en tiempos en el caso del RDO respecto a los otros casos es aplastante.

Dentro de las posibilidades que ofrece están:

- Definir consultas múltiples que se procesen en el servidor.
- Uso de índices propios de la base de datos.
- Ejecución síncrona o asíncrona de consultas.
- Ejecución de procedimientos almacenados en la base de datos.
- Consultas parametrizadas.
- Fácil migración desde otros objetos más antiguos (DAO).

Con el RDO se presenta una vía que consigue una mejora en rendimientos de entre un 400% a un 500%, respecto a los rendimientos de los métodos anteriores, y que no requiere de ningún control adicional, se trata de las consultas múltiples. Antes de enviar las SELECT al servidor para que las procese, se agrupan todas ellas en una única sentencia SQL compuesta de todas las sentencias de tipo SELECT necesarias. Esta sentencia se envía por una única conexión realizada contra el servidor, que la procesa en su conjunto, y devuelve un conjunto de registros agrupados en bloques distintos a través de la propiedad 'MoreResults', donde cada bloque representa el conjunto de registros que se obtiene como resultado de cada una de las sentencias SELECT individuales.

Todo este conjunto se procesa para realizar la carga simultánea de las listas, con lo que el rendimiento aumenta espectacularmente por dos razones principalmente, se ha superado en gran parte el cuello de botella de las comunicaciones, y además el servidor de datos es una máquina con un potencial más alto en transferencia de datos que una computadora normal.

Otras posibilidades que sólo se comentarán consisten en la capacidad de integración de las políticas de SQL Server (que ya va integrada con la de Windows NT) dentro de la seguridad de la propia aplicación, y su manejo desde la misma (estas capacidades se mejoran con la versión 2000 de SQL Server); facilidad en la migración del software cuando se evoluciona en el hardware (utilidades de transferencia de las bases de datos entre servidores); posibilidades de publicación y suscripción de tablas entre distintos entornos (SQL Server, AS/400, Access, Oracle, ...); posibilidades de backup/restore fáciles de manejar (integrables con la aplicación partir de la versión 7.0), etc.

La seguridad de una conexión RDO es que no se puede tener control en la estructura de la base de datos, con lo único que se pueden alterar son los datos. Para conocer RDO hay que conocer su modelo que tiene 14 objetos, por lo que sólo se tiene permitido el acceso a datos y no al manejo de la estructura de la base de datos. (Manipulación de tablas, índices y vistas).

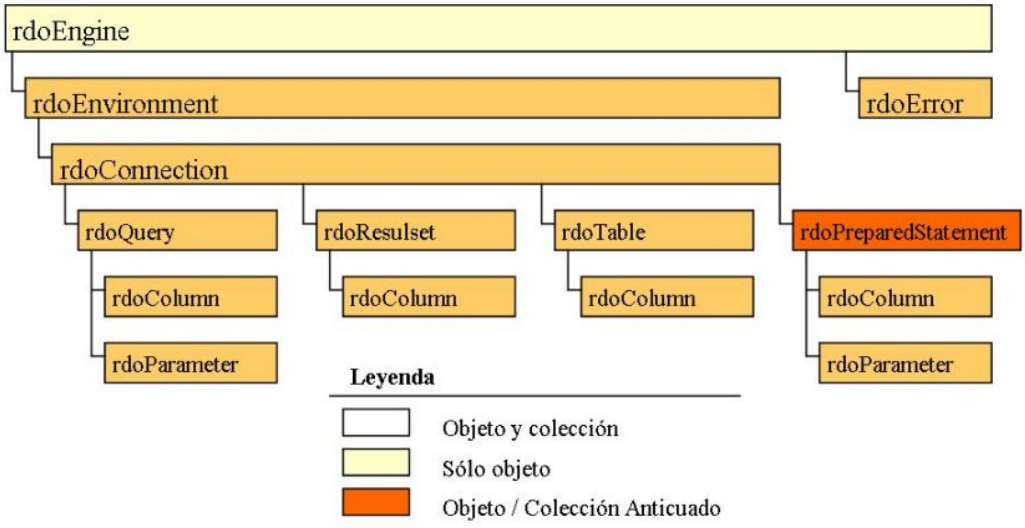


Figura 1.4 Modelo de una conexión RDO.

En el modelo se representan cada uno de los miembros disponibles, a continuación se explican los más comunes.

a) **rdoEngine.** Motor del origen de datos a nivel superior para acceder a los datos.
b) **rdoEnviroment.** Es donde se define un conjunto lógico de conexiones y el alcance de una transacción para un determinado nombre de usuario.
c) **rdoError**. Parte de RDO que contiene una colección de errores.
d) **rdoConnection.** Representa una conexión física con el origen de datos remotos, con un controlador único ODBC.
e) **rdoQuery.** Es una definición de consulta que puede incluir un cero o más parámetros.
f) **rdoResultset.** Representa las filas que devuelve la ejecución de una consulta.
g) **rdoTable**. Representa una definición de una tabla.
h) **rdoColumn.** Representa una columna de datos con un tipo de datos y un conjunto de propiedades comunes.
i) **rdoParameters.** Representa un parámetro asociado aun objeto rdoQuery.

Cada componente tiene una colección de propiedades y métodos que sería muy tedioso explicar pero se puede consultar la fuente, en la bibliografía.

1.3.5. Componentes.

Cuando se trabaja con RDO, se tienen que tener en cuenta conceptos de cierta importancia en la operación, éstos son los controladores de cursores, tipos de conjunto de datos y tipos de bloqueo (tipo de concurrencia).

1.3.5.1. Controladores de Cursores.

Los cursores son una extensión de los conjuntos de resultados que proporcionan un mecanismo donde se trabaja con una fila o un pequeño bloque de filas cada vez en una aplicación en línea (WEB).

El controlador de cursor se usa para definir la manera en que puede desplazarse el cursor de datos. Con RDO el controlador maneja la versión de la computadora local en la administración del cursor.

La administración de cursor es un motor de la base de datos capaz de llevar un control de todos los registros en la colección y de la ubicación en el conjunto de datos.

Se tienen que usar cursores por lotes del cliente con servidores de datos avanzados que permitan diversas peticiones en forma simultánea a una misma conexión, en el cual el servidor administrará las peticiones y le dará resultado conforme haya sido completada.

Aunque también tiene la posibilidad de no usar un cursor, en este caso el servidor sólo traerá un conjunto de datos por cada instrucción SQL sin tener datos anteriores, por lo que se debe reiniciar la consulta; ésta es una ventaja, ya que la conexión es más rápida que usar cursores por lotes.

Los controladores de cursores tienen cinco valores mostrados en la tabla

Opción	Valor	Descripción
rdUseIfNeeded	0	Cursores del lado del servidor
rdUseOdbc	1	Cursores del lado del cliente
rdUseServer	2	Cursor de lado del servidor
rdUseClientBatch	3	Cursor por lotes del lado del servidor
rdUseNone	4	(Sin cursor y sólo envía una petición al servidor para traer los datos de acuerdo a una instrucción SQL).

Cuadro 1.5 Controladores de Cursores

1.3.5.2. Tipo de conjuntos de datos

El "Tipo de conjunto de datos" es la forma en que será devuelta la información del servidor de datos, en el caso del servidor SQL Server 2000, se tienen cuatro opciones:

- **rdOpenFowardOnly.** El servidor envía datos que se recorren hacia adelante (opción predeterminada en caso de no seleccionar ninguna opción).

- **rdOpenKeySet.** El servidor crea un conjunto de datos que puede tener filas actualizables para realizar cualquier proceso, como agregar, consultar, modificar o eliminar filas de una o varias tablas, en donde los miembros de un resultset son fijos.

- **rdOpenDynamic.** Crea un conjunto de filas actualizables al igual que Keyset, la diferencia es que los miembros de un RdoResultset son dinámicos.

- **rdoOpenStatic.** Es una copia estática de los datos que se pueden usar para buscar datos o generar informes, por lo que no detecta cambios en filas actualizables por otros usuarios después de crear los datos.

Un punto importante es que el uso de rdOpenKeyset, no sólo es un conjunto de datos que contienen filas sino también apuntadores o punteros de conjunto de claves a otras filas de los datos requeridos.

El tipo de datos de rdOpenDynamic no soporta claves o marcadores, por que sus datos actúan de manera dinámica, pero el método es poco eficiente si se usan grandes cantidades de datos, ya que en cada conjunto se envían al cliente y cada vez que se actualizan los datos.

1.3.5.3. Tipos de Bloqueo.

El otro punto importante a tratar para el acceso a una base remota es el tipo de bloqueo o concurrencia, éstos se usan para administrar las modificaciones al conjunto de datos (Locktype).

rdConcurReadOnly (Predeterminado). No se establece ningún nivel de bloqueo, al elegir esta opción el cursor es de sólo lectura, pero puede llevar a cabo consultas de acciones para actualizar datos.

rdConcurLock. Permite un bloqueo pesimista a todo el conjunto de filas, el proceso asegura a la aplicación que ningún otro programa que se abra recibirá acceso exclusivo (de lectura y escritura), sobre ninguna de las filas que esté procesando.

rdConcurRowver. Un bloqueo optimista basado en la identificación de las filas, es decir, sólo bloquea las filas necesarias por el valor de identificación de la fila, si hay una modificación marca un error.

rdConcurValues. Realiza un bloqueo optimista basado en una verificación columna por columna de los datos de cada fila, si se hizo marca un error.

rdConcurBatch. Proporciona un valor optimista, en base al valor de la propiedad Update, cuando se utiliza el modo de actualización por lotes, al emplear el método UpdateBatch.

Uniendo cada uno de los componentes que mencionan en este cápitulo, con lo que se crea una base para crear un sistema cliente-servidor de doble capa (servidor con la base de datos y la pc con la aplicación con la interface), en donde se va a requerir un modelo entidad relación para la mejor optimización de los datos.

Por lo que solo falta elegir la interfaz, que depende del lenguaje de programación (Visual Basic), donde RDO tiene una velocidad como OLEDB, cualquier usario que se conecte no sera capaz de alterar la estructura pero si los datos. Y al igual que ADO tendrá un soporte para el manejo de errores, el cual se muestra en inglés, por lo que hay que modificarlo para mostrar mensajes en español.

CAPÍTULO

2. FUNCIONAMIENTO DE CRONOS

2.1. Consideraciones previas

En el mes de septiembre de 1993 fue liberado un programa con el nombre de "CRONOS"; diseñado para el control del banco de horas, extraordinarios, finales y horarios de profesores de las doce carreras de la E. N. E. P. Aragón, actualmente denominada F. E. S. Aragón. Lo realizaron los alumnos: Francisco Xavier Espinosa Madrigal e Ismael Manzo Salazar, quiénes trabajaban en el Departamento de Informática, que estaba adscrito a la Unidad de Planeación.

Con el paso del tiempo el programa, no sólo se utilizó en las doce carreras sino también en las áreas administrativas, presentaba muchas ventajas en aquella época, por lo que se logró su aceptación para formar parte de los procesos administrativos de la escuela.

La figura 2.1 muestra el funcionamiento general por pantalla del programa.

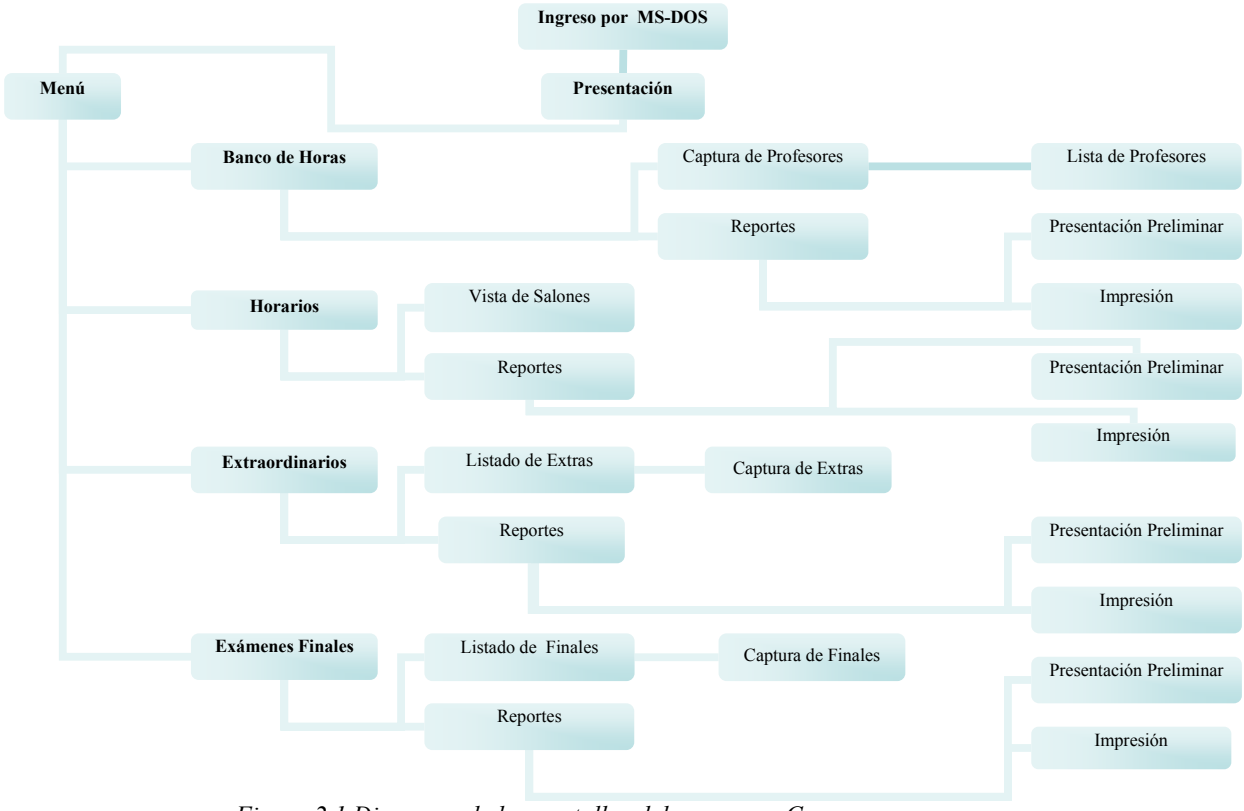


Figura 2.1 Diagrama de las pantallas del programa Cronos

A través de los años, se tuvieron que hacer adaptaciones al programa, con base en la normalización de la UNAM, lo que fue realizado en el Departamento de Informática por el ingeniero Ernesto Manzo Salazar, quien desarrolló la segunda versión.

Esta versión se puede ejecutar en una computadora con un procesador 386, con memoria RAM de 8 Mbytes, utilizando el teclado se realiza el trabajo de una forma rápida (el uso del ratón es opcional, pero poco práctico).

Ahora con las innovaciones que se han presentado en las computadoras y en los sistemas operativos se pone en riesgo su funcionamiento, en primer lugar se menciona la desaparición del sistema operativo MS-DOS que fue sustituido por los últimos sistemas operativos de Windows de Microsoft.

Con las primeras versiones de Windows no se afectó al programa, ya que con Windows versión 3.1, 95 y 98 se utiliza MS-DOS. Con la aparición de Windows 2000, XP y 2003 se creó un emulador para correr este programa, lo que pone en riesgo su duración para las siguientes versiones de los sistemas operativos, ya que no se sabe si todavía van a integrar un simulador de MS-DOS.

La forma de ejecutar el programa es por medio de un acceso directo, en donde se tiene la ruta de un archivo de procesamiento por lotes[6] (con extensión '.bat'), que cuenta con el siguiente contenido:

echo off	' Quita el shell de sistema.
f:	' Unidad donde se encuentra el ejecutable de Cronos (puede ser local o una unidad de red asignada).
cd /CRONOS/CRONOS.INF/	' Ruta donde se localiza el programa.
CRONOS *CAR* ***/LASER***	' Shell para ejecutar el programa, donde CAR es la sigla asignada a la carrera que se va a utilizar (ésta siempre va con mayúscula), y "**/LASER"** indica que es una impresora láser, en caso de omitir es una impresora de matriz o de inyección de tinta.
echo on	' Restaura el shell de sistema al finalizar el programa.

Con los pasos anteriores se ejecuta el programa y se muestra la pantalla de presentación donde aparecen los créditos de los diseñadores, el nombre del programa, la versión (aunque ésta corresponde a cuando fue liberado, ya que con el paso del tiempo se han realizado modificaciones y no se han hecho cambios a la imagen original) y un logotipo emblemático sobre el nombre del programa, en este caso un reloj de arena (figura 2.2.).

[6] Se crea una variable global de las siglas asignadas a la carrera que define la ruta y las bases de datos que se van a abrir.

Figura 2.2 Pantalla de Inicio

Al terminar de pasar la pantalla de presentación aparece un cuadro[7], en el que se debe escribir el período escolar que se va a usar (figura 2.3).

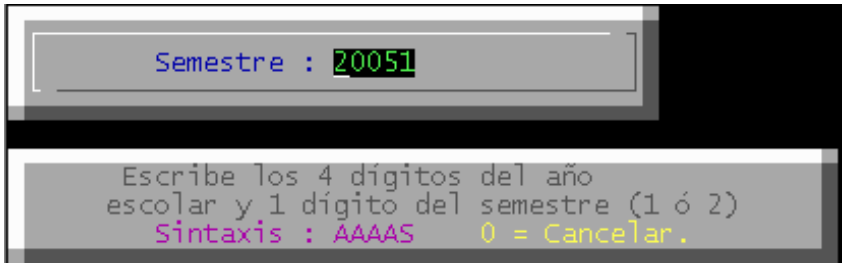


Figura 2.3 Petición del semestre a utilizar

Dependiendo del período se seleccionarán las bases de acuerdo con los últimos 3 dígitos; en caso de no existir se terminará el programa, las bases principales son:

- horas.*** ‘ Base del banco de horas.
- hordat.*** ‘ Base de horarios.
- extra_?.*** ‘ Base de extraordinarios.
- final.*** ‘ Base de final.
- materias.cro ‘ Base de materias.
- horocp.*** ‘ Base de permisos.

En donde los tres dígitos mencionados van en lugar de los asteriscos en cada base de datos, y el signo de interrogación corresponde al jurado de extraordinarios que corresponde al semestre. Cabe recordar que todas estas bases se tienen en una carpeta del área correspondiente.

Al escribir un período escolar válido se procede a mostrar el menú general (figura 2.4), en donde se muestra la sigla de cada opción o un botón con el nombre completo de la opción,

[7] Un problema que se muestra en esta pantalla es que no existe validación de usuario, ya que la única validación depende de la clave de acceso al servidor NOVELL o la contraseña de la propia computadora si el programa se trabaja localmente.

éste puede ser seleccionado con ayuda del ratón; en caso de seleccionar una sigla con el teclado se presiona la letra indicada.

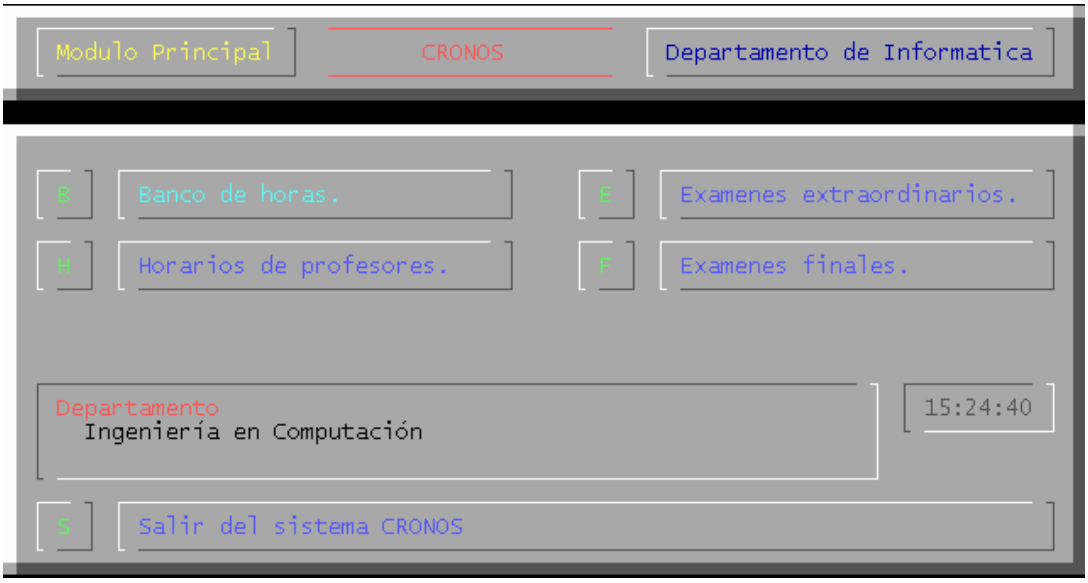


Figura 2.4 Menú Principal

En esta pantalla existe un subrutina escondida para definir un superusuario que va a tener posibilidad de realizar ajustes dentro de las bases de datos, por seguridad no se mostrará su funcionamiento.

Dentro del programa existen bases específicas que solamente pueden ser modificadas en el archivo *modcomun.pas* de las que se pueden mencionar con especial interés: las bases de categorías y salones[8].

2.2. Módulo de Banco de Horas

El módulo de banco de horas es muy importante, porque con ello el Departamento de Personal notifica a la UNAM de las horas que tiene el profesor, y se define el sueldo que va a percibir.

También se utiliza para obtener el número de horas que tiene asignada cada área y que deberá reportar a la División o Unidad de Área correspondiente con los profesores, ayudantes de profesor o técnicos académico que van a integrar el banco.

Al teclear la letra 'B' o seleccionar la leyenda banco de horas con el ratón se cambia de pantalla y se muestra el menú principal del banco de horas (figura 2.5).

[8] Para agregar una categoría o un salón se tiene que modificar el código del archivo, y volver a compilar el programa "Cronos".

Figura 2.5 Menú principal del Banco de Horas

El primer botón manda a la pantalla de captura de banco de horas y el segundo permite visualizar o imprimir los reportes.

2.2.1. Estructura de la pantalla de captura del banco de horas

Al seleccionar captura del banco de horas se muestra una nueva pantalla (figura 2.6), dividida en 3 partes:

I. Datos del Profesor: Nombre del profesor, registro federal de causantes y un indicador si es o no funcionario.

II. Registro del Banco de Horas: Número total del registro del banco de horas y posición del registro actual de banco de horas.

III. Datos del Registro del Banco de Horas: Clave de materia, nombre de materia, categoría y las horas separadas por rubros (horas anteriores y horas actuales).

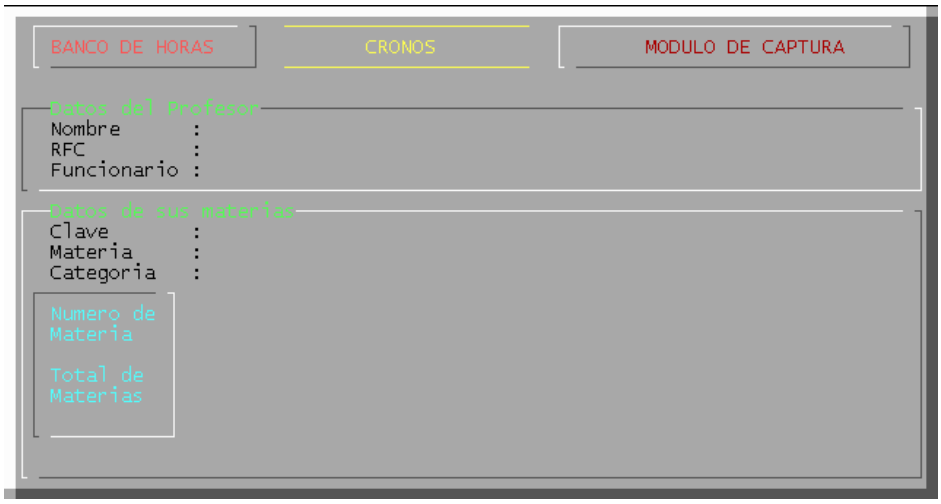


Figura 2.6 Pantalla de captura del Banco de Horas

El cursor se va a posicionar en el nombre del profesor, si se quiere ingresar un nuevo registro el proceso se realiza en forma automática, ya que al momento de ingresar el nombre por medio del teclado, a través de una subrutina, se agrega una estructura del profesor en memoria, aunque después se incorpore el R.F.C. y se defina si el profesor es funcionario con 'S' o 'N'. La estructura de la base del profesor es la siguiente:

```
Profesor = Record
       Claveprof : Word;       { 2 bytes Clave única profesor}
       RFC       : String[15];{16 bytes Registro Federal de Causantes}
       Nombre    : String[40];{41 bytes Nombre del Profesor}
       Funciona  : Boolean;    { 1 byte  Bandera de Funcionario}
     end;                      {60 total }
```

En caso de consultar un profesor se da un 'Enter' dentro del espacio vacío donde está el nombre, se activa un módulo dentro del programa donde carga una pantalla con todos los profesores existentes en la base (figura 2.7), entonces se ejecutan varios ciclos para revisar dentro de la base de horas si tiene asignados registros, esto es para que en la pantalla asigne un '*' si es que no tiene algún registro asignado, en caso contrario no lo muestra.

Figura 2.7 Pantalla con el listado de los profesores ingresados al Banco de Horas

Todos los profesores son guardados en un espacio de memoria, son seleccionados por medio de un puntero y de una rutina, dentro de la lista de profesores se ve el puntero, ya que es el reglón resaltando al profesor que está seleccionado, el usuario lo va cambiando utilizando las teclas de dirección o con el ratón.

En esta pantalla se tienen los siguientes procesos:

- Primero, si se presiona la tecla 'Supr' y el registro no tiene asignado un '*' muestra un mensaje de error (un profesor no puede ser eliminado si tiene asignado un registro de banco de horas).

- Segundo, si se oprime la tecla ‘Supr’ en un registro que si tenga un ‘*’ éste se elimina de la lista y de la posición de la memoria, para después actualizar la base de profesores.
- En el tercero, al presionar ‘Enter’ se envía el profesor que se ha seleccionado y sus datos a la pantalla anterior, limpiando la pantalla de profesores.
- El último proceso se utiliza en caso de que no se tenga ningún registro almacenado, presionando ‘Enter’ se limpia la pantalla y se posiciona en el nombre del profesor.

Al momento de cargar los datos del profesor, se activa una subrutina para cargar en memoria cada uno de los registros del banco de horas que tiene asignado; esto lo muestra en el cuadro donde está el número de registro de banco de horas (figura 2.8), si tiene registros se posiciona en el primero, de lo contrario muestra un cero.

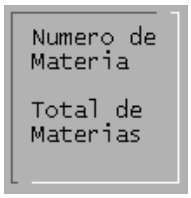


Figura 2.8 Cuadro donde se muestra la materia que está cargada y el número total de ellas

Para ingresar un nuevo registro se tiene un cuadro en la parte inferior derecha para ayuda (presionar la tecla ‘Ins’ para agregar un registro), se presiona la tecla indicada y aparece una leyenda “Se insertará la materia o actividad número:” (figura 2.9), se teclea ‘Enter’ para crear un nuevo registro en el banco de horas en memoria o se oprime ‘Esc’ para cancelar el proceso y no asignar un espacio en memoria (en caso de querer borrar se explicará más adelante ampliamente).

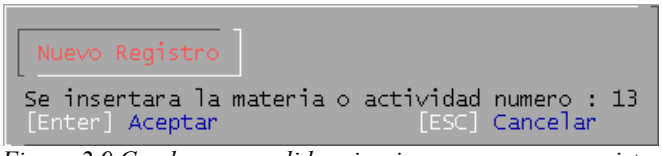


Figura 2.9 Cuadro para validar si se ingresa un nuevo registro

Después de este proceso el cursor es enviado a la opción ‘Clave de Materia’, entonces cargará en memoria la base de materias[9] la cual tiene la siguiente estructura:

```
Materias = Record
       Clave   : word;          { 2 bytes Clave de Materia}
       Nommate : String[50];    {51 bytes Nombre de la Materia}
       Sem     : Byte;          { 1 byte Bandera para el semestre}
     end;
```

Se da ‘Enter’ para visualizar la lista de materias (se muestra la clave y el nombre) como lo muestra la figura 2.10, esto se da por medio de una subrutina que simula el proceso de un

[9] La insercción de materias se hace por medio de un programa externo (insmat.exe), en el cual se agregan los registros a un archivo de base de datos llamado materias.cro, el mismo proceso es para borrar un registro de la estructura y esto se realiza con otro programa externo (delmat.exe).

combo, entonces se selecciona el registro y se cargan en memoria las opciones correspondientes de la materia (figura 2.10).

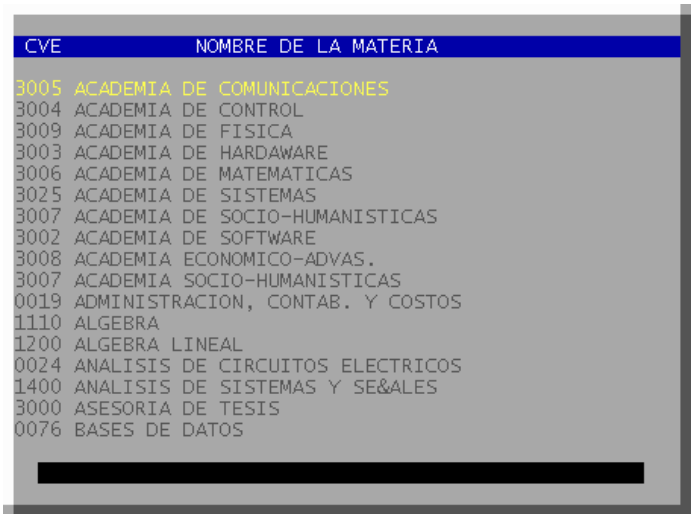


Figura 2.10 Cuadro de selección con las materia.

Al momento de presionar 'Enter' el cursor se ubica ahora en la posición de categoría, donde se carga la que fue previamente almacenada en un archivo temporal, esto es para facilitar su captura, ya que se repiten con mucha frecuencia. Para modificar esto, se presiona la tecla '→', donde aparece la lista de todas las categorías cargadas dentro del programa[10] (también tiene una función parecida a la de un combo) (Figura 2.11).

```
catego : array[1..maxcatego] of string[32] {Nombre de la categoría}
catego2 : array[1..maxcatego] of string[15] {Categoría forma abreviada}
```

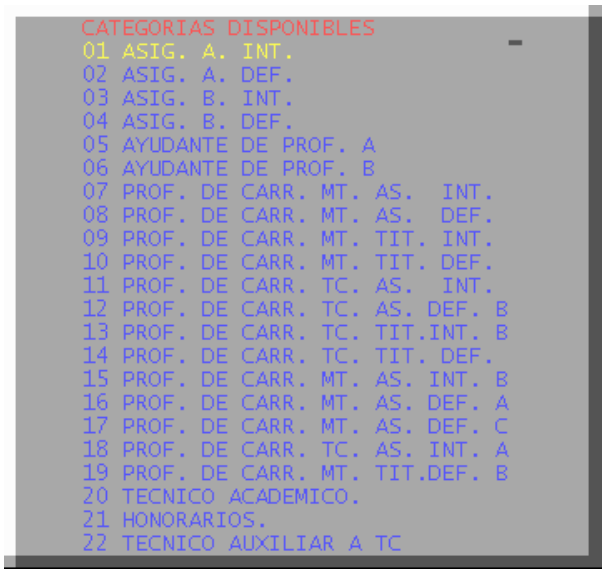


Figura 2.11 Cuadro de selección con las categorías disponibles

[10] Las categorías están escritas en un array del archivo modcomun.pas.

Luego de seleccionar una categoría se sigue un proceso para definir que tipo de hora tiene:

- **Horas interinas** (0 - interino), que cuenta con 12 categorías de horas (figura 2.12), éstas se pueden ver al teclear 'F1' cuando el cursor se encuentra dentro de alguna de ellas (figura 2.13).

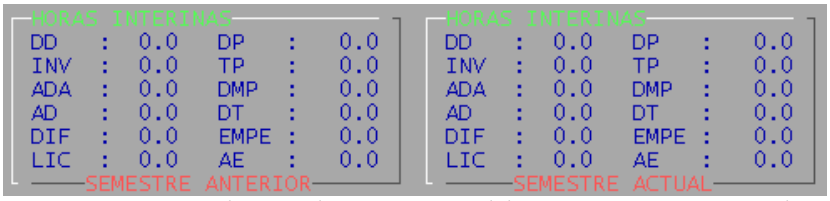


Figura 2.12 Cuadros con horas interinas del semestre anterior y actual

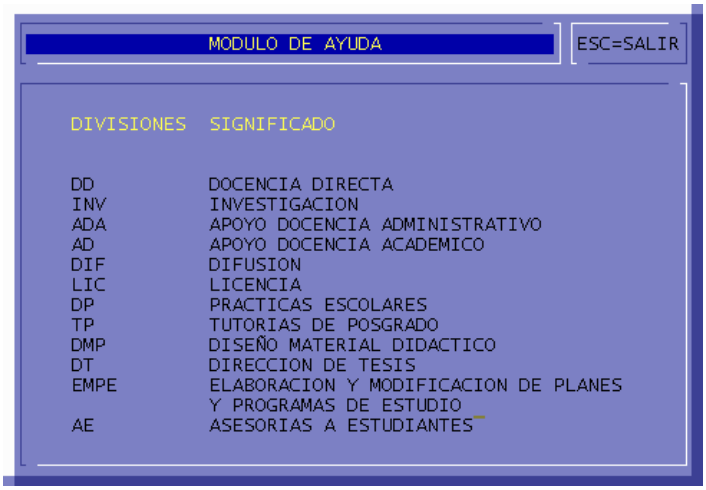


Figura 2.13 Cuadro de ayuda sobre la información de las horas

- **Horas definitivas** (1 - definitivo), son las horas que manejan solamente 3 categorías, de las cuales se tienen Docencia Directa (DD), Investigación (INV) y Licencia (LIC) (figura 2.14).

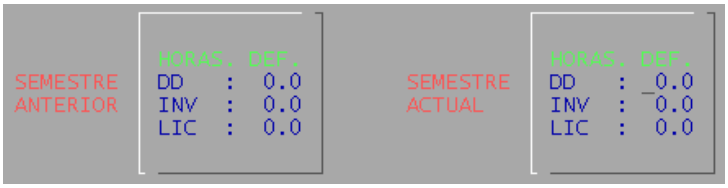


Figura 2.14 Cuadros con horas definitivas en el semestre anterior y actual

La estructura de las horas en el banco dentro del programa se integra de la siguiente manera:

```
Horas = Record
        Claveprof : Word;        { 2 bytes Clave del profesor}
        Categoria : byte;        { 1 byte Lugar del array de la categoría}
        Clavemat  : word;        { 2 bytes Clave de materia}
        Horas     : array[1..15] of integer; {30 bytes Horas}
      end;                       {35 total}
```

Tanto en las horas interinas como en las definitivas sólo el rubro de Docencia Directa maneja horas teóricas, todos los demás manejan horas prácticas, lo cual será contabilizado al mostrar el reporte.

Las horas se parten en dos cuadros, el cuadro con título de horas anteriores, que es una referencia de las horas de un período anterior (este proceso lo realiza un programa de 'utileria.exe', ubicado en la carpeta de ejecutable de Cronos), y las horas actuales donde el usuario ingresa, modifica o borra las horas según lo requiera.

Por último para salir del cuadro de horas se debe presionar 'Enter' y ubicarse en el cuadro donde indica el número de registro y después oprimir 'Esc' para salir del banco.

Para el banco de horas hay algunas consideraciones:

- Para borrar un registro de banco de horas se tiene una condición muy importante: "No se puede borrar un registro con horas anteriores", por lo que tiene que aparecer un mensaje para borrarlo (figura 2.15).

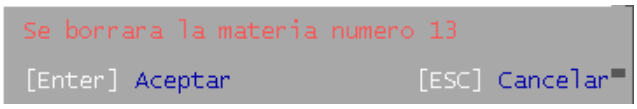


Figura 2.15 Mensaje de confirmación para borrar un registro

- Cuando se genera un nuevo banco, las horas actuales pasan a ser horas anteriores, y también se copian en horas actuales.

2.2.2. Pantalla de reportes del banco de horas

Al presionar el siguiente botón dentro del menú principal del banco, aparece la pantalla del "Módulo de Reportes" (figura 2.16), toda esta rutina y procesos se encuentran en modbanco.pas.

Figura 2.16 Pantalla para realizar reportes del Banco de Horas

Esta pantalla se divide en varias partes, la primera corresponde a los rubros que están agrupados de acuerdo a las categorías en las que fue asignada:

- Profesores de Asignatura
- Profesores de Carrera
- Técnicos Académicos
- Ayudantes de Profesor
- Funcionarios
- Personal por Honorarios
- Horas de Apoyo Extracurricular

El último rubro “Justificación por Horas”, es un módulo que se usaba los primeros años, pero fue desapareciendo para ser sustituido por las propuestas de los profesores, que también han cambiado su formato semestre por semestre.

Existen otras opciones importantes, se puede destacar “Sólo Totales”, que permite imprimir de los rubros seleccionados los resultados generales de cada banco de horas.

Las formas de salida del programa son dos:

Por pantalla: Se cambia el modo texto, que se ha venido utilizando, por un modo gráfico tipo “CGA” diseñado por píxeles a una resolución de 640 x 480 (figura 2.17).

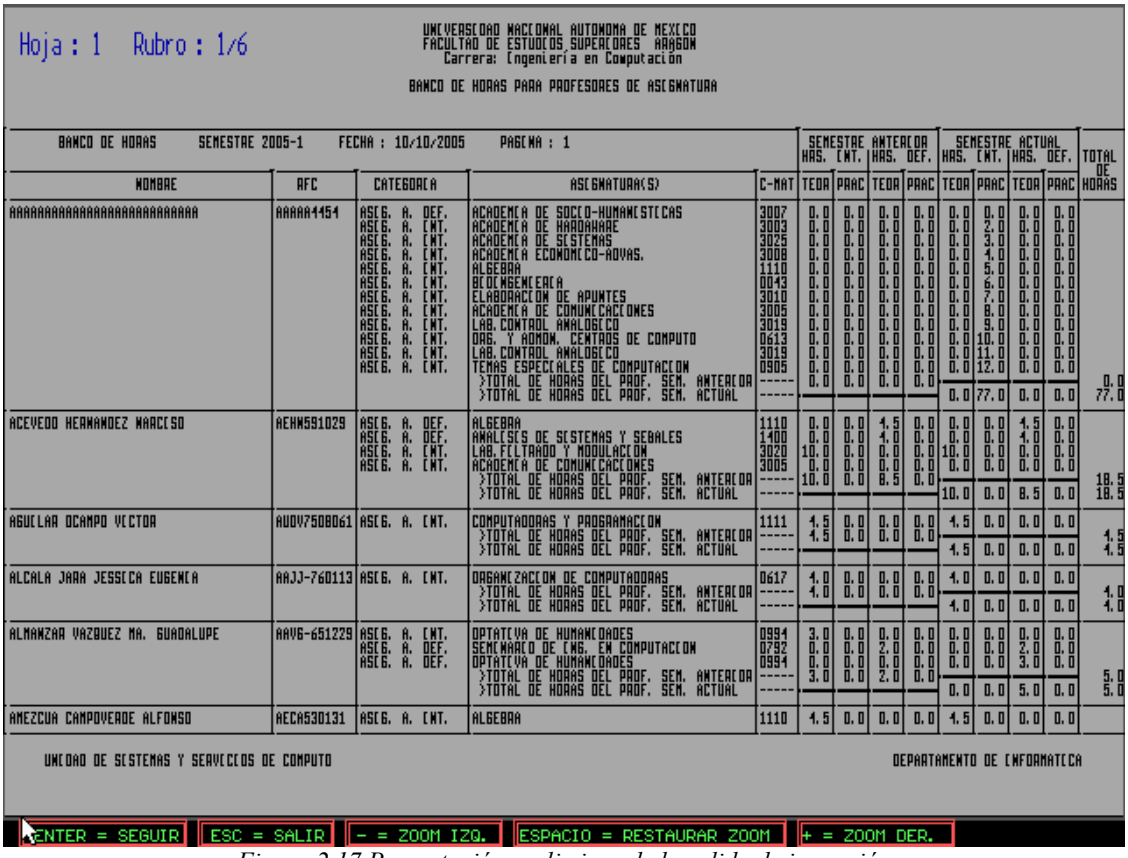


Figura 2.17 Presentación preliminar de la salida de impresión

Por impresora: Permite imprimir la información en una impresora que previamente se definió al momento de ejecutarse el programa, donde una pantalla muestra la evolución de la impresión (figura 2.18).

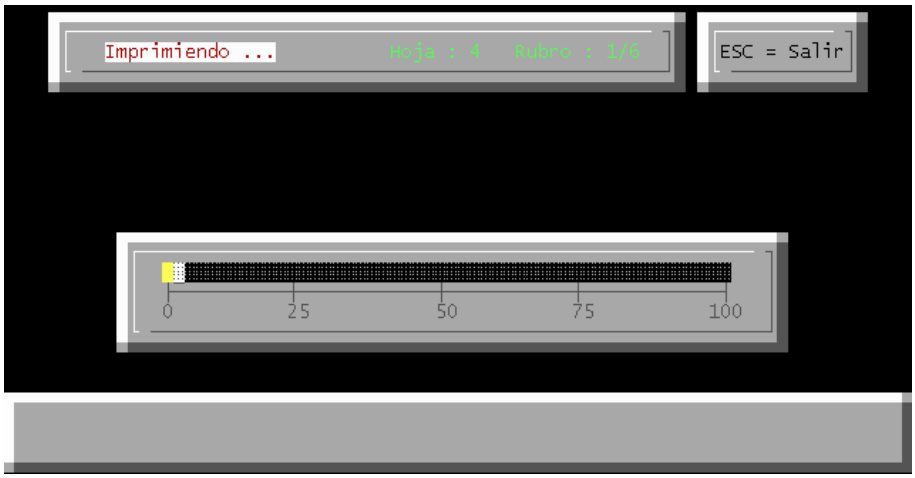


Figura 2.18 Evolución del proceso de impresión

La última opción “Hojas: de 1 hasta 9999”, indica el rango de las hojas que se van a imprimir, aunque está un poco limitada, pues es difícil cambiar el rango, ya que en ocasiones no imprime el establecido por el usuario.

2.3. Módulo de Horarios

El módulo de horarios es donde se van a capturar cada uno de los horarios que va tener determinada carrera, no se tiene una pantalla inicial como el banco de horas, si no que entra primero a la pantalla de captura y ésta tiene la opción de ‘F4’ para llamar a la pantalla de reportes (figura 2.19).

Figura 2.19 Módulo de captura de horarios

2.3.1. Funcionamiento de la pantalla de captura de horarios

El módulo de horarios es el más importante y complejo para el programador, ya que es el que tiene más detalles y el que maneja más bases de datos al mismo tiempo (profesor, materias, permisos, horarios y salones).

Para insertar un registro se presiona 'INS', esto permite insertar un espacio en memoria indicando la dirección con un puntero; en donde se agregan las siguientes estructuras:

```
  HoraComp = Array[1..6] Of Word;   {Alto=Ini; Bajo=Fin Horas Encriptadas}

Datos = Record                      'Estructura del registro
        RegOrd,                     'Número de registro
        Salon,                      'Clave del Salón
        Grupo,                      'Grupo
        CveMat,                     'Clave de la Materia
        CveProf : Word;             'Clave del Profesor
        Hora    : HoraComp;         'Subestructura de horas
      end;
```

También se abren las estructuras de salones, profesores y materias, pero lo que el usuario ve en pantalla es la inserción de un renglón (figura 2.20).

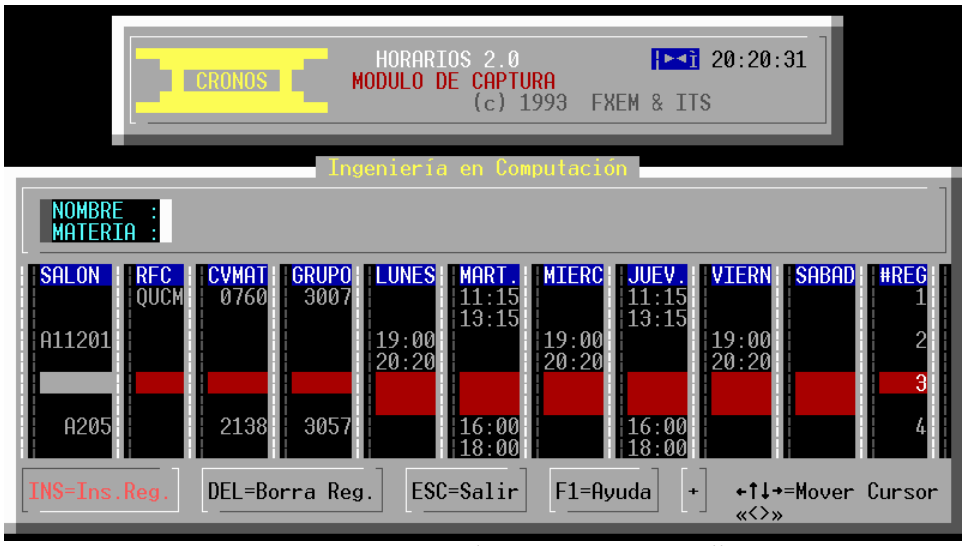


Figura 2.20 Creación de un registro en pantalla

Como es una pantalla de texto MS-DOS, la pantalla de captura tiene que ser muy compacta, para que sea rápida, aunque tiene un inconveniente, ya que no está ligada con el banco de horas y el usuario puede ingresar el número de registros que quiera, sin tener un control absoluto.

En el momento de capturar un salón, se deben tener asignados los permisos para su uso, ya que de lo contrario no se ven en la lista de salones. Estos permisos sólo los puede asignar el superusuario de Cronos, que captura previamente dentro de la misma pantalla, pero por razones de seguridad no se mostrará su funcionamiento.

Los salones cargados en la base de permisos son los que se mostrarán en la lista, para seleccionar un salón sólo se utilizan las flechas de arriba o abajo (figura 2.21).

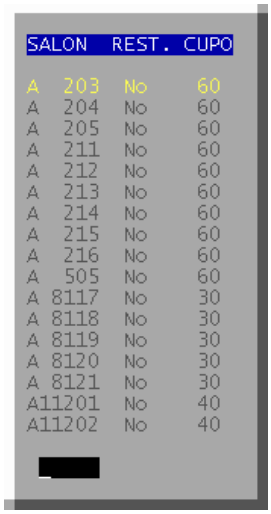


Figura 2.21 Listado de los salones

Como se vio con anterioridad en la estructura de horarios, lo que se guarda es el campo de 'SALON', no el número de salón, sino el índice para indicar la posición donde se encuentra, a través de una subrutina muestra el valor del salón y el cupo, tal como en el siguiente ejemplo:

```
   SalonFisico : Array[1..MaxSalon*2] Of Word =
{224}              (111,60,112,60,113,60,114,60,115,60,116,60,...
```

Con ayuda de las flechas se selecciona el salón, pero en caso de requerir más información sobre los permisos de ocupación o si existe otro horario, se presiona la "Barra Espaciadora", y se visualizará un gráfico de apoyo (figura 2.22).

Figura 2.22 Gráfico con la ocupación de un salón

En caso de dejar la casilla sin salón, a ésta se le asignará un valor predeterminado ('A0'), así se guardará en memoria y en la estructura de horarios.

Después viene la columna correspondiente a los profesores, pero sólo tiene la opción para ver el 'RFC' (también tiene por predeterminado el campo nulo), al presionar 'Enter'

aparece la lista de la base de profesores[11], de la cual se puede seleccionar uno con las flechas de arriba o la de abajo, al elegir un profesor se presiona 'Enter'.

De la misma forma se cargan las materias, con apoyo en la clave de las mismas, también tiene un valor por defecto '0', pero éste no es mostrado en pantalla, se realiza por código. Con base en la clave de materia lo busca en la base de datos o al presionar 'Enter' se despliega la pantalla de materias[12].

Ahora se posiciona en la columna de grupo, que al igual que la clave de materia tiene un valor predeterminado '0', con la diferencia de que no despliega una lista como la de profesores o materias, sólo valida que sean ingresados 4 dígitos de tipo numérico.

El siguiente paso es asignar el horario, lo cual es un poco complejo, ya que los campos no son de tipo date, sino word (es la mitad de bytes que una variable tipo integer) en un formato de encriptar para transformar un horario a un número.

Esto se hace dividiendo las horas en dos partes: en horas y minutos, en donde previamente se validan los dígitos, las horas van de 7 hasta las 23 horas; en cuanto a los minutos van de 0 a 60 en intervalos de 5 minutos.[13]

La fórmula[14] para encriptar un horario es la siguiente:

$V = ((H - 7) * 12 + (M / 5)) + 1$
V = Valor, H = horas, M = minutos.

Para desencriptar se realiza el siguiente proceso donde primero $V=V-1$ y después:

$H = (V \bmod 12) + 7$
$M = (V \operatorname{div} 12) * 5$
V = Valor, H = horas, M = minutos.

Pero eso no es todo, ya que para cada asignación se tiene que realizar un proceso completo.

a) Estar dentro del rango de permisos asignados a un salón.
b) Si no tiene salón asignado no se realiza más que la validación del ingreso de horas y minutos que sean válidos.
c) Si se tiene un permiso y un salón, se debe verificar que no exista un traslape con los otros horarios, por lo que realiza un ciclo para que no se encime con otro horario (figura 2.23).

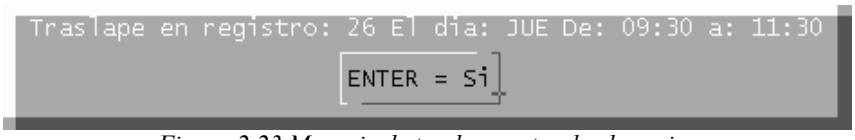


Figura 2.23 Mensaje de traslape entre dos horarios

[11] Ver la figura 2.7, en la página 29.
[12] Ver la figura 2.10 en la página 31.
[13] Se tiene como máximo a las 23 horas, que es cuando se cierra por completo la escuela y en algunas clases que se imparten en la carrera de Arquitectura.
[14] Ver tesis de licenciatura de Manzo Salazar, Ernesto.

d) Validar si se ingresó la hora (7 a 23) y los minutos (00 a 55 con intervalos de 5 minutos), tanto de inicio como de término (figura 2.24).

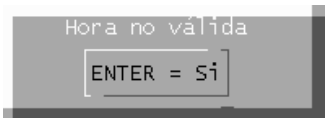


Figura 2.24 Mensaje de error de una hora no válida

e) Asignar directamente un número de registro.

Para borrar un registro, sólo se tiene que presionar la tecla 'Suprimir' o 'Del'[15] como se muestra en la pantalla, pero se debe tener cuidado porque lo hace de manera directa sin preguntar al usuario.

También tiene otras funciones incluidas dentro del programa, distribuidas en dos menús, para cambiar de menú hay que presionar '+', en la pantalla de inicio se ve el menú 1 y a continuación se muestra el menú (figura 2.25).

Figura 2.25 Segundo menú de la barra de captura de horarios

Las funciones restantes son las siguientes:

I) Pantalla de Ayuda: Muestra una pantalla de ayuda muy básica para apoyar a los usuarios a manejar los Horarios (figura 2.26).

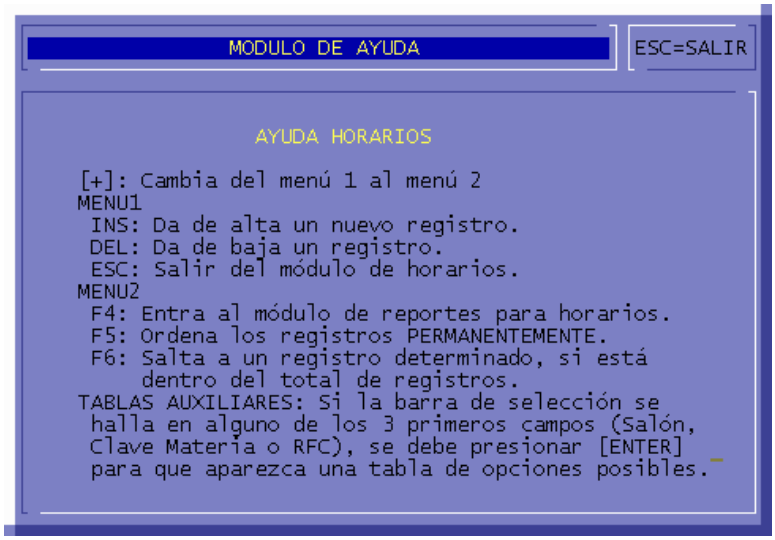


Figura 2.26 Módulo de ayuda

[15] Esto es 'Delete', que en español significa suprimir. Ya que cuando se creó el programa la mayoría de las computadoras de la facultad contaban con teclado en inglés.

II) Saltar: permite moverse rápidamente de un registro a otro, únicamente tecleando el número de registro en donde se va a posicionar (figura 2.27).

Figura 2.27 Pantalla para ir a otro registro

III) Ordenar: reacomoda la estructura de horas en tres fases de ordenamiento, tal como se muestra en la figura 2.28.

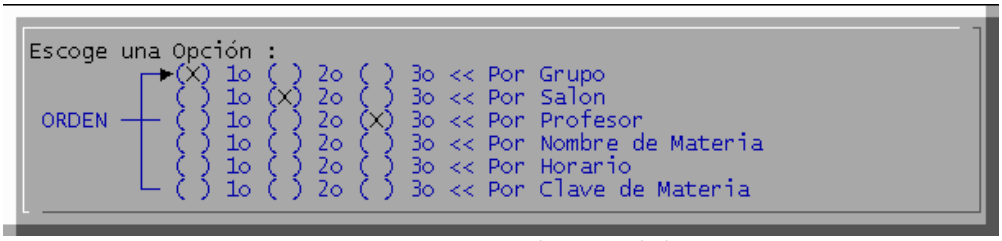


Figura 2.28 Opciones para ordenación de los registros

2.3.2. Módulo de reportes de horarios

IV) Imprimir (F4), al presionar el botón o la tecla antes mencionada se pasa directamente a la pantalla de impresión[16], en donde se pueden imprimir los registros con un ordenamiento con referencia a un campo, el cual puede ser salón, grupo, profesor o materia y además con RFC impreso o sin él (figura 2.29).

Figura 2.29 Pantalla de impresión de horarios

[16] Al igual que en el banco, se puede visualizar antes de imprimir o cambiar la opción para imprimir en la impresora. Ésta sólo se realiza con la impresión de la cabecera y posteriormente la impresión de acuerdo con el orden establecido.

2.4. Módulo de Extraordinarios

Al presionar el botón del módulo de captura de extraordinarios no se pasa directamente a la captura de exámenes extraordinarios, sino que se muestra una ventana con el número de vuelta de extraordinario que se va a utilizar (figura 2.30), ya que los semestres comprenden 2 vueltas, pero en ocasiones se llega a tener hasta una tercera, como sucedió en el regreso a clases después de la huelga de la UNAM en el 2000.

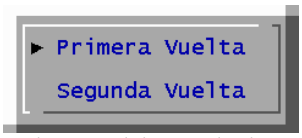


Figura 2.30 Selección del período de extraordinarios

A partir de esto se crea o se edita un archivo que va a contener los extraordinarios, según el período escolar y la selección del archivo tal como se muestra en el cuadro 2.1.

Semestre	Primera vuelta	Segundo vuelta	Período Extraordinario
Primer Semestre	Período EA	Período EB	Período EC
Segundo Semestre	Período EC o ED	Período ED o EF	Período EF o EG

Cuadro 2.1 Tabla de asignación de jurados de extraordinarios.

Al ingresar un nuevo período extraordinario en un semestre, sólo se debe seguir la letra siguiente en orden alfabético. Por ejemplo en el primer período la letra correspondería a la letra 'C' y en el siguiente semestre escolar se recorren las letras. Para el segundo semestre sería la letra 'F', a menos que se recorriera por existir otra vuelta en el primer semestre, tal como se menciona, al nuevo semestre le corresponde una 'G'.

Al presionar la primera o la segunda vuelta, si no encuentra el archivo de extraordinarios, crea uno nuevo de forma automática, como por ejemplo 'EXTRAS_A.051', que significa extraordinarios de primera vuelta con jurado EA del semestre 2005-1. En caso contrario abre el archivo.

La pantalla principal de extraordinarios (figura 2.31), tiene la misma forma que la del banco de horas, con un botón para entrar a la pantalla de captura y otro para ir a la de reportes.

Figura 2.31 Pantalla principal de extraordinarios

2.4.1. Estructura de Captura de Extraordinarios

Después de pasar este proceso se visualiza la pantalla de captura de extraordinarios (figura 2.32). Como el tamaño de la pantalla de texto es limitado, ésta se maneja en dos pantallas, una donde muestra la lista y otra donde se capturan los datos.

Figura 2.32 Módulo de Captura (Listado de Extraordinarios Capturados)

Cuando se presiona la tecla para ingresar un extraordinario se crea un espacio en memoria con la siguiente estructura:

```
Examen = record
       Prof1    : word;       { 2 bytes}
       Prof2    : word;       { 2 bytes}
       ClaveMat : word;       { 2 bytes}
       salon    : word;       { 2 bytes}
       Fecha    : longint;    { 4 bytes}
       salon2   : word;       { 2 bytes}
       Fecha2   : longint;    { 4 bytes}
       Turno    : Byte;       { 1 byte }
     end;                     {19 total}
```

Al mostrar la pantalla de captura de extraordinarios se ingresan los datos que se muestran en la estructura, agregando el nombre completo del sínodo 1 o sínodo 2 y el nombre completo de la materia, todo esto tomado de sus respectivas bases (figura 2.33).

Al capturar el sínodo 1 y el sínodo 2, lo único que carga es el número de profesor (no está a la vista), el RFC y el nombre. Todo funciona como un combo, como se ha explicado anteriormente en el banco de horas.

De igual forma se tienen que seleccionar las materias (junto con la clave de materia), como lo que se vio en banco de horas y horarios. Los siguientes campos en conectar son los del día y la hora, para ello se presiona 'Enter' y con las fechas de izquierda o derecha se selecciona el campo y las flechas de arriba-abajo para cambiar los valores.

Figura 2.33 Módulo de Captura (Captura de Extraordinarios)

A continuación viene un proceso sencillo para el usuario, pero difícil para el programador, ya que se tiene que validar cada componente de lo que es la fecha y la hora, a diferencia de los horarios, ahora se maneja la fecha por lo que no se utiliza el proceso de encriptación que se usa en los horarios, sino una función que maneja el lenguaje de pascal para encriptar (packtime) y desencriptar (unpacktime)[17].

```
packtime(desfe,fecha);  {Encriptar fecha y hora en Pascal}

Unpacktime(fecha,Desfe);      {Desencriptar fecha y hora en Pascal}
```

Ahora la selección de salón se apega al proceso de permiso en los horarios. Sólo se visualizan en la pantalla (figura 2.21), los salones donde se tenga permiso asignado y no se muestra el gráfico de ocupación de salón, tal como se hace en los horarios.

Después viene otro punto el cual no parece tener mucha trascendencia para el usuario, pero si para el área de servicios escolares, ésta es la asignación de turno (matutino, vespertino o ambos), la importancia es que algunas carreras tienen materias compartidas y para poder diferenciarlas se le asigna un jurado distinto.

Turno	Jurado	Turno	Jurado
Matutino 1	01	Vespertino 1	51
Matutino 2	02	Vespertino 2	52
Matutino 3	03	Vespertino 3	53
Matutino 4	04	Vespertino 4	54
Matutino 5	05	Vespertino 5	55
Ambos	01		

Cuadro 2.2 Listado de los jurados disponibles en Cronos

[17] Esta función también se puede apreciar perfectamente en la tesis del Ing. Ernesto Manzo Salazar.

La última opción ‘ACEPTAR EL REGISTRO’ es para guardar el registro (figura 2.34), grabarlo en memoria y en el disco, para posteriormente pasar a la pantalla anterior y mostrarlo en la lista de extraordinarios.

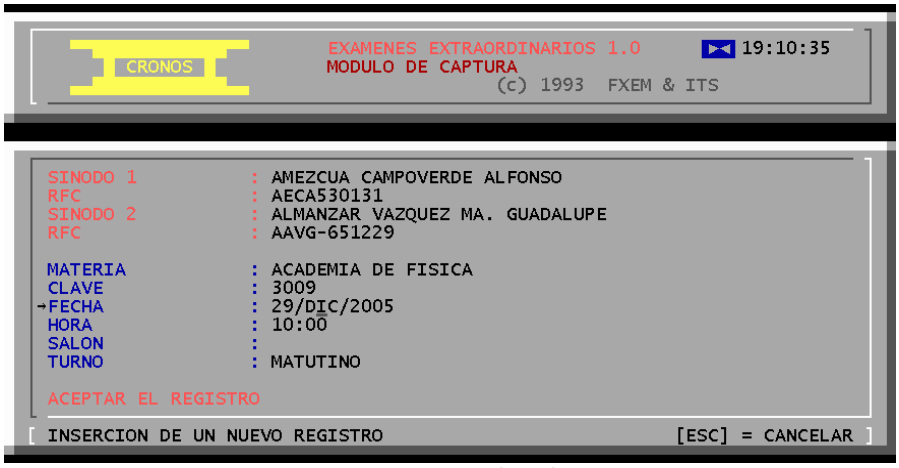


Figura 2.34 Captura completa de un registro

Así como la función de un nuevo registro, y la de editar, existen otras de más fácil comprensión:

- Borrar registro (DEL): Se selecciona un registro, éste es buscado y se muestra un mensaje para prevenir al usuario de que se va a borrar el mensaje (Figura 2.35).

Figura 2.35 Pantalla de verificación para la eliminación de un registro

- Ordenar registro (F2): Esta función es muy parecida a la de horarios y sirve para ordenar cada uno de los registros, pero a diferencia ésta sólo puede ordenar por materia (figura 2.36), aparece un mensaje donde indica que se está ordenando el registro.

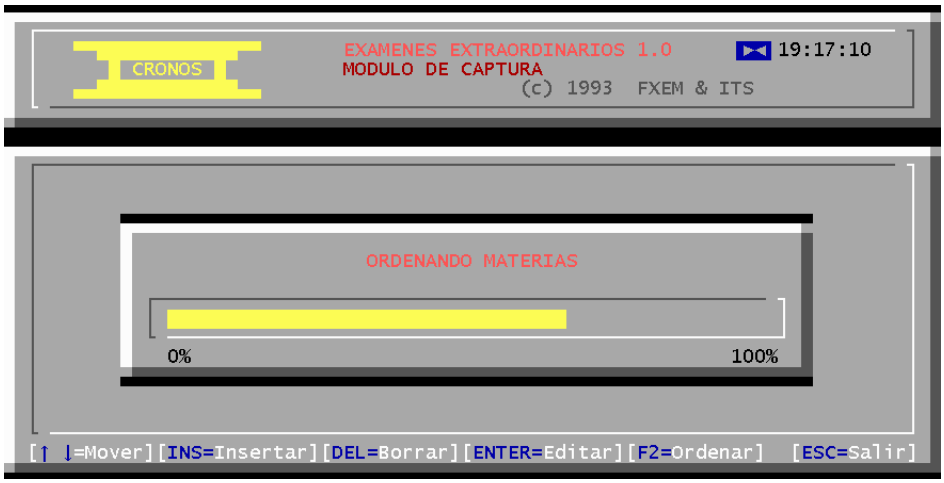


Figura 2.36 Avance en el ordenamiento por materias

- Salir (ESC): guarda la estructura, cierra la ventana y retorna a la pantalla principal de extraordinarios.

2.4.2. Pantalla de Reportes de Extraordinarios

En el menú principal de extras se selecciona el botón de reportes para mostrar la pantalla correspondiente a los reportes (figura 2.37).

Figura 2.37 Módulo de reportes de exámenes extraordinarios

La pantalla de reportes es muy parecida a las que se han mencionado, pero se divide en dos secciones, la primera se compone de un listado de extraordinarios por materia (con RFC y otra sin RFC)[18] y la segunda sólo está integrada por un elemento: los recordatorios, los cuales son impresos para ser entregados tanto al titular como al suplente, para recordarles el día, lugar y hora en que van a ser aplicados los extraordinarios.

Al igual que los horarios y el banco tiene la opción de verse en pantalla o manda directamente a impresión.

[18] No hay que olvidar verificar si se publica o no el RFC del profesor, ya que esta clave es muy importante.

2.5. Módulo de Finales

Del Menú principal se tiene otra opción 'Exámenes Finales'. Para entrar hay que seleccionar el botón 'Finales' o presionar la letra "F", lo que despliega el menú principal de finales[19], con dos botones uno para el 'Módulo de Captura' y otro para el 'Módulo de Reportes' (figura 2.38).

Figura 2.38 Pantalla principal de los exámenes finales

2.5.1. Funcionamiento de la captura de Finales

Al presionar 'Módulo de Captura', se abre o se crea el archivo FINALES.XXX, y se carga las bases en memoria por medio de punteros, las bases de materias, profesores, salones y la de finales. Las bases de materias, profesores y salones ya se conocen, pero la estructura de finales no:

```
Final = record
        ClaveMat : word;       { 2 bytes}
        Prof1    : word;       { 2 bytes}
        Prof2    : word;       { 2 bytes}
        Prof3    : word;       { 2 bytes}
        Grupo    : word;       { 2 bytes}
        FechaI1  : longint;    { 4 bytes}
        FechaF1  : longint;    { 4 bytes}
        salon1   : word;       { 2 bytes}
        FechaI2  : longint;    { 4 bytes}
        FechaF2  : longint;    { 4 bytes}
        salon2   : word;       { 2 bytes}
      end;
```

Como puede verse cuenta con tres profesores, para la evaluación, pero en la actualidad sólo se maneja el profesor titular en su materia, con la excepción de las carreras del SUA, Arquitectura y Diseño Industrial.

[19] Tiene el mismo formato que la pantalla del banco de horas y los extraordinarios.

Realizado esto se genera la pantalla de captura de exámenes finales (figura 2.39), la cual cuenta con dos vueltas, que a diferencia de los horarios, se capturan juntas, tal como lo muestra la pantalla de captura.

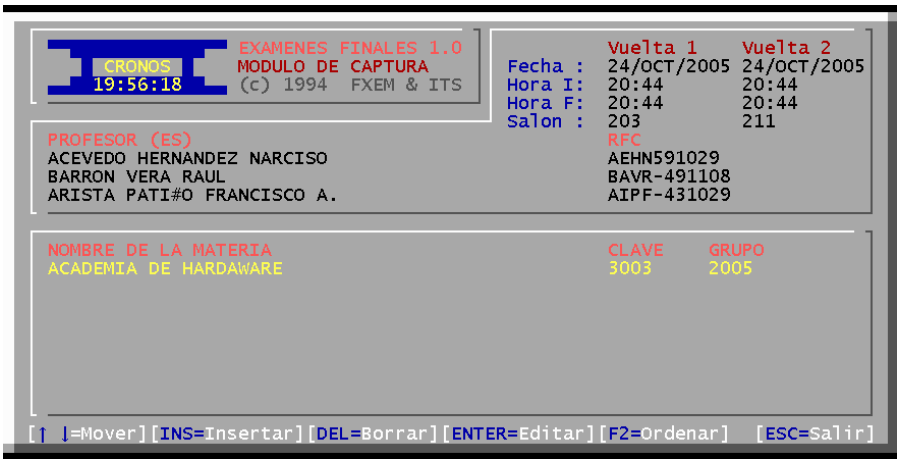


Figura 2.39 Módulo de Captura (lista de exámenes finales)

En esta pantalla, se tienen las mismas opciones que la pantalla con la lista de los exámenes extraordinarios, se muestran la lista de los exámenes finales capturados. Para insertar un examen final se presiona 'INS', cambia la pantalla de captura (figura 2.40).

Figura 2.40 Módulo de Captura (captura de exámenes finales)

Al igual que los horarios, con apoyo de las fechas se selecciona el profesor 1 y se presiona 'Enter', aparece la pantalla con la lista de profesores (pantalla 2.7).

El mismo proceso se debe seguir, si se capturan el profesor 2 y 3, para darlos de alta, con apoyo de la misma pantalla de profesores y la misma base de datos, lo único diferente que toma en la base de datos es el número de profesor que es lo que guarda en la estructura de extraordinarios.

A continuación se captura la materia, que al presionar 'Enter' se muestra la pantalla con cada una de las materias de la carrera (figura 2.8).

Al capturar la materia, el programa muestra su nombre y realiza una búsqueda en el array de materias, para encontrar la clave que le corresponde y posteriormente mostrarlo en pantalla.

El siguiente paso es capturar la información de cada una de las vueltas que comprenden los finales, como lo es el día, la hora de inicio, la hora de término y por último el salón donde se va a aplicar, para ello se muestra la pantalla con los salones que se pueden usar[20] (figura 2.21).

Ya capturados los datos se presiona 'ACEPTAR EL REGISTRO', para que se guarde toda la información en el array de la estructura 'final', guardando sólo los datos necesarios y codificando los horarios, como se realizó en los horarios. Por lo que el registro se agrega y se muestra en la pantalla donde está el listado.

Para borrar un registro, primero se selecciona con apoyo de la flechas de arriba o de abajo, se presiona 'DEL' y aparece una pantalla para rectificar si se va a eliminar (figura 2.41).

Figura 2.41 Pantalla de verificación para eliminar un registro

Para editar algún registro, sólo se selecciona y se presiona 'ENTER', aparece la pantalla de captura de finales, pero con la información que tiene el examen final seleccionado, por lo que se realiza lo mismo que al ingresar un final.

La última opción es simplemente para ordenar la lista y esto se realiza al presionar 'F2', organizando la lista en orden alfabético con respecto a la materia que se esté impartiendo. Para salir de la aplicación sólo se presiona 'ESC'

2.5.2. Pantalla de reportes de Exámenes Finales

La pantalla de reportes para finales tiene algunas opciones (figura 2.42), de las que se pueden destacar:

[20] Por lo regular se utiliza el salón donde se imparte la materia.

- Se pueden imprimir en orden alfabético por materia o si se presiona 'barra espaciadora' en 'Grupo' se imprime con respecto a los grupos.
- Puede incluir el RFC o no incluirlo.
- Otro tipo de reporte, es el de los recordatorios para los maestros.
- Impresión por pantalla o directa a la impresora.
- Determinación del rango de hojas a imprimir.

Figura 2.42 Módulo de reportes de exámenes finales

2.6 Otros programas a considerar

Uno de los puntos a considerar es que el programa tiene que ser compatible con otras áreas, para evitar la doble captura de datos, realizar los procesos administrativos con una base de datos, estas áreas deben ser compatibles, tanto en sus campos como en la longitud de ellos. De éstos se tienen dos programas, NEW y KARDEX.

2.6.1. Programa de captura de ordinarios y extraordinarios de servicios escolares (NEW)

El programa NEW fue creado para que las carreras de licenciatura puedan ingresar los horarios del período ordinario y los exámenes extraordinarios, para ser usado por Servicios Escolares en las inscripciones, fue diseñado para correr en el sistema operativo MS-DOS, y trabaja con bases de datos de dbase (Figura 2.43).

Figura 2.43 Pantalla principal de New

Tanto los exámenes ordinarios cómo los extraordinarios se componen de tres bases de datos, y de cuatro módulos, que comprenden captura, consulta, mantenimiento o reportes (figura 2.44).

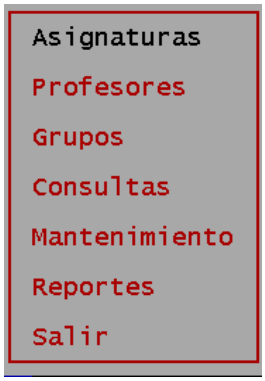


Figura 2.44 Menú de opciones tanto ordinarios como extraordinarios

Base de Asignaturas: es donde se ingresan cada una de las materias que se van a utilizar para capturar en el sistema de NEW, el usuario no puede modificar los registros, sólo consultarlos.

La estructura de asignatura se muestra en el cuadro 2.3.

Nombre	Campo	Tipo	Longitud
Nombre de la materia	NOMBRE	Texto	36
Carrera	PL	Texto	3
Clave de la materia	CL	Texto	4
Créditos	CR	Texto	2
Semestre	SE	Texto	2

Cuadro 2.3 Estructura de la base de Asignaturas.

Tal como se ve en el cuadro hay una diferencia entre los campos de Cronos, en donde hay más campo como lo son carrera, créditos y el semestre.

La pantalla 2.45 muestra cómo funciona la consulta de asignaturas.

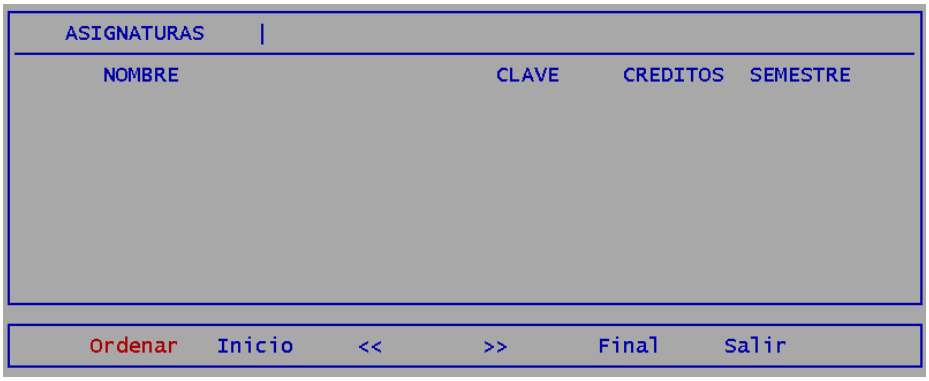


Figura 2.45 Pantalla de consulta de las Asignaturas

Base de Profesores: Es muy importante para inscribir al profesor y que pueda impartir clases en alguna asignatura, al igual que las materias, en NEW se tienen más campos, tal como se puede ver en el cuadro 2.4.

Nombre	Campo	Tipo	Longitud
Clave del Profesor	CLPR	Texto	13
Nombre del Profesor	NOPR	Texto	45
Grado Escolar	GEPR	Texto	1
Nacionalidad	NAPR	Texto	1
Sexo	SEPR	Texto	1
Número de Trabajador	NTPR	Texto	7
	USED	Texto	1

Cuadro 2.4 Estructura de la base de Profesores

En la base de Cronos sólo se manejan dos datos: la clave de profesor y el nombre, lo que hace incompatible con esta base de datos, porque no se podía exportar de Cronos a New.

Por lo tanto se tenía que capturar la información en New con una sola pantalla para dar de alta, baja, cambio o consulta; lo cual se ve en la figura 2.46.

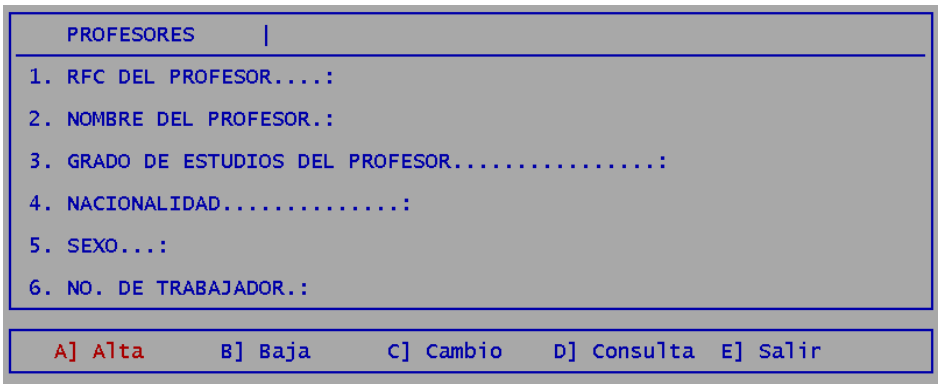


Figura 2.46 Pantalla de captura de profesores

Base de Grupos: Es en esta base donde se arman cada uno de los grupos para dar de alta las inscripciones que hay en la escuela, los campos que componen esta tabla son los que se muestran en el Cuadro 2.5.

Nombre	Campo	Tipo	Longitud
Clave del Grupo	CLGR	Texto	4
Clave de Asignatura	CAGR	Texto	4
Turno	TUGR	Texto	1
RFC del primer Profesor	CPGR	Texto	13
RFC del segundo Profesor	CSGR	Texto	13
Cupo	CUGR	Texto	1
Control	CONTROL	Texto	1

Cuadro 2.5 Estructura de la base de Profesores

La diferencia con las tablas de Cronos, son cuatro:

- Primera, en Cronos se captura en un registro sólo un profesor, y no dos.
- Segunda, el cupo, que en la base de Cronos se encuentra de tamaño predeterminado, por lo que el área de Servicios Escolares tenía que realizar las correcciones a mano.
- Tercera, es que no se tiene un campo de Control que es ajeno a la base de horarios ("Cronos"), por lo que se tenía que ingresar aparte.
- Cuarta no se tiene un campo de turno, por lo que con programación se tenía que determinar el turno: matutino o vespertino, si era diurno o mixto se realizaba a mano.

La pantalla de captura de Grupos tanto en ordinarios como extraordinarios es la figura 2.47.

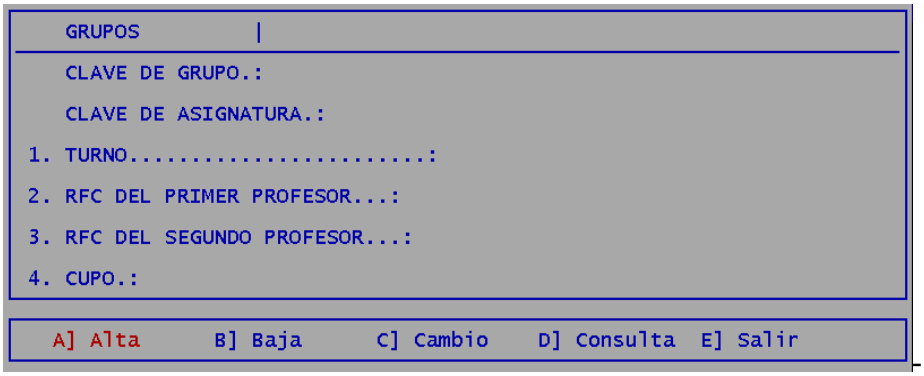


Figura 2.47 Pantalla de captura de Grupos

2.6.2. Programa de Secretaría Académica para firma de asistencia (Kardex)

Kardex es un programa realizado en Visual Basic, para elaborar las tarjetas de asistencias del personal docente en la escuela, diseñado y creado por el pasante Eric Rubén Adauto Gómez en el año de 2002, que trabajaba en el Departamento de Informática. Ver figura 2.48.

Figura 2.48. Pantalla de presentación de Kardex

El programa es muy sencillo, usa una base creada por el programa UTILERIA.EXE, para su ejecución, el cual genera una base en modo texto con el siguiente formato:

UAFIR (cadena inicio) + CAR (carrera) + "." + 999 (Semestre con 3 dígitos)[21]

Dentro de este archivo se introduce la información de profesores, materias y horarios, de forma secuencial con una separación del retorno de carro, como en el cuadro 2.6.

Campo	Tipo	Longitud
RFC	Texto	14
Nombre	Texto	32
Horario Lunes	Texto	8
Horario Martes	Texto	8
Horario Miércoles	Texto	8
Horario Jueves	Texto	8
Horario Viernes	Texto	8
Horario Sábado	Texto	8
Grupo	Texto	4
Clave Materia	Texto	4
Nombre Materia	Texto	45
Salón	Texto	5

Cuadro 2.6 Estructura del archivo de texto UAFIR…

Mediante un procedimiento de apertura del archivo, traslada los datos a una base temporal en "DATOS.MDB" (Base realizada en Access), conforme a las especificaciones que introduzca el usuario en la pantalla 2.49.

[21] Se maneja el semestre como extensión del archivo, para facilitar el uso del programa.

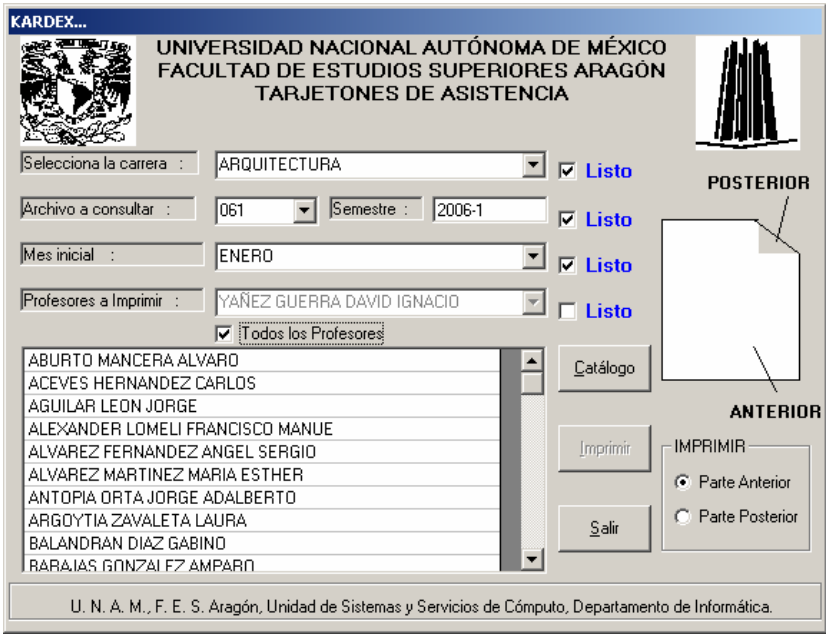


Figura 2.49 Pantalla del programa Kardex

En la base "Datos" hay dos tablas, la primera es adicional, con información de las carreras que puede usar el programa, y se definen como en el cuadro 2.7.

Campo	Tipo	Longitud
Clave de la Carrera	Texto	3
Nombre de la Carrera	Texto	50

Cuadro 2.7 Estructura de la tabla Carreras

La segunda tabla se llama "Temporal", con la misma estructura que el cuadro 2.5, donde la única diferencia es que en los campos de los horarios, cambia su longitud de 8 a 13 caracteres para darle formato al campo antes de la impresión, tal como se muestra en el cuadro 2.8.

Origen	Formato	Tamaño
Horario en archivo UAFIR	07000900	8 caracteres
Horario en tabla Temporal	07:00 A 09:00	13 caracteres

Cuadro 2.8 Diferencias entre dos bases

El programa es muy fácil de usar, primero se selecciona la carrera, además de presionar el botón de verificación que está a la derecha, ésta busca en todos los archivos con la carrera especificada utilizando la clave de carrera, cada uno de los semestres que se tienen, y carga el combo de semestres.

Posteriormente se selecciona el semestre y se da el título del Período Escolar, cuando está listo, se presiona el cuadro de verificación y se pasa a la siguiente opción, que permite seleccionar en que mes comienza a funcionar el semestre para que el programa considere los seis meses correspondientes para imprimir el kardex, tres semestres en la parte frontal y otros tres en la parte posterior de la tarjeta. Por último se seleccionan los profesores, que pueden ser todos o sólo un rango de ellos.

CAPÍTULO

3. DESARROLLO DE WCRONOS.

3.1. Resultados del Estudio Previo.

Para la creación del programa se planteó el objetivo principal: Desarrollar un programa para sustituir a Cronos, que facilite a los usuarios el trabajo con una interfase amigable y que sea capaz de compartir la información con otras áreas administrativas que lo requieran.

El programa ha funcionado de 1993 a 2006 (Trece años), está realizado en el lenguaje de Pascal, con bases de datos encriptadas en un archivo, y se tiene sólo el antecedente de que se crearon dos programas para sustituirlo, sin éxito alguno.

Los problemas a resolver del programa Cronos, considerando a partir de lo mostrado en el capítulo II son:

a) La interfaz de MS-DOS ha sido sustituida y está siendo obsoleta.
b) Facilitar al usuario el manejo y funcionamiento de programa.
c) Tener una base única de permisos y planta física, para la asignación correcta de salones, sin traslapes.
d) Fácil comprensión del código para modificaciones futuras.
e) Desarrollo de una base de datos compatibles con otras áreas administrativas.
f) Mejorar presentación de reportes.
g) Buscar otro sistema operativo para el nuevo sistema, con una arquitectura del sistema Cliente-Servidor.
h) Implementar un SGBD a cambio de la base de datos encriptada en un archivo.
i) Considerar las estructuras de cada uno de los programas y las áreas para realizar una base compatible.

3.2. Análisis del Proyecto.

La idea de acuerdo al resultado preliminar es crear un programa parecido a Cronos en una plataforma distinta, que permita tener un mejor desempeño, en donde se cambie la estructura de la base se datos (archivos en MS-DOS) a una estructura relacional en un sistema gestor de base de datos.

El sistema se enfoca a usuarios locales por lo que no es necesario diseñar un sistema WEB, por lo cual se tendrá más seguridad al desarrollar una aplicación personalizada por máquina, la cual se conectará por medio de unos archivos instalados, para que se pueda realizar la comunicación mediante un entorno RDO.

Tomando en cuenta que las computadoras en donde se va a implementar el sistema, tienen las siguientes características:

- Procesadores AMD con modelos Athlon, Intel modelos Celeron Pentium II, III y IV,
- Memoria que va de 64 a 1024 Megabytes.
- Discos duros de 1.2 a 80 Gigabytes.
- Sistema Operativo Microsoft Windows 98 SE, Windows 2000 y Windows XP.
- Tarjetas de Red
- Impresora, ya sea en modo local o a través de la red.
- Internet.
- Impresora local o por red.

Con estos datos y con base en los conocimientos descritos en lo que es una arquitectura Cliente-Servidor, primero se debe implementar el servidor.

3.2.1. Implementación en el Servidor

Como se vio en el capítulo I, se toma la teoría para implantar un sistema Cliente-Servidor, en éste se tendrá la capa de datos, mientras que en la aplicación se tiene la etapa de control y la de presentación.

Para ello se cuenta con servidor con las siguientes características:

- Motherboard Intel 82801.
- Dos procesadores Intel Xeon a 2.66 GHz.
- Memoria RAM DDR 1 Gigabyte.
- Tarjeta de vídeo nVIDIA Geforce 4MX
- Disco Duro de 140 Gigabytes.
- 2 tarjetas de red, una a velocidad de 100 Megabits

3.2.1.1.Programas a utilizar

Se puede contar con los programas (Software), que son parte de un proyecto anterior, y que serán base para este sistema, lo que permitirá un ahorro de recursos, ya que son paquetes originales con licencias de usuarios para funcionamiento del servidor, los cuales se mencionan posteriormente.

a) Sistema Operativo Microsoft Windows 2000 Server Original
b) Gestor de Base de Datos Microsoft SQL 2000 Server Enterprise Original
c) Lenguaje Microsoft Visual Basic 6.0 Original

3.2.1.2.Microsoft Windows 2000 Server

Microsoft Windows 2000 Server es un sistema operativo de 32 bits, se eligió este Sistema Operativo por varias características; la principal de ellas que es un software original con una licencia de 50 usuarios.

La seguridad de Windows 2000 depende mucho de la configuración del sistema en cada uno de los servicios de archivo y de impresión, implantaciones de seguridad, herramientas de sistema y muy en especial las tareas administrativas automatizadas.

Para montar el servidor se activó el servicio de directorios (Active Directory), además del último Service Pack 5 o 6, que contienen soluciones para tapar problemas o huecos de seguridad y además activar las actualizaciones automáticas para cualquier otra actualización que se tenga.

También se revisaron las directivas de seguridad como la activación de contraseñas codificadas, o como la activación de las directivas de auditoría con que cuenta para la revisión de ingreso de usuarios, procesos que se están trabajando. También considerado los lineamientos básicos de seguridad que se tienen actualmente:

- Contraseñas con un mínimo de 8 caracteres.
- Las contraseñas deben tener letras, números y signos.
- Cada contraseña debe ser asignada por usuario, en donde se hace responsable a quien maneje la información.
- El administrador debe tener una lista de todas las contraseñas.

Se bloquearon los servicios que no se utilizan hasta saber correctamente la configuración, éstos son:

- ISS (Servicio WEB)
- WINS (Servicio de nombres de Internet de Windows)
- DHCP (Estándar de TCP/IP para asignación de IP o información)
- Servicio de Impresión

Por lo que quedan en especial

- Active Directory

- DNS

Para evitar cualquier contagio de virus, se prefirió bloquear el sistema de archivos, ya que con un virus o troyano se pone en riesgo la integridad del sistema, por lo que por ahora sólo funcionará como servidor de base de datos.

3.2.1.3. Microsoft SQL Server 2000

Tal como se menciona en el capítulo I y en el análisis con que se cuenta, se usará la base de datos Microsoft SQL Server 2000, ya que es un gestor de base de datos relacionales, la cual permite organizar la información en tablas y ver sus relacione entre sí.

Se implementará un sistema Cliente–Servidor de dos capas (como se vio en el capítulo I), los usuarios accederán a la base de datos por medio de aplicaciones en sus equipos cliente.

Una de las ventajas que ofrece SQL Server 2000 es permitir el almacenamiento de gran tamaño, que no podría implementase en una base de datos como es Access, también considerada la implementación de un administrador de las bases de un modo gráfico (SQL Server Enterprise Manager), cosa que todavía no se implementaba en las bases de datos, como ahora.

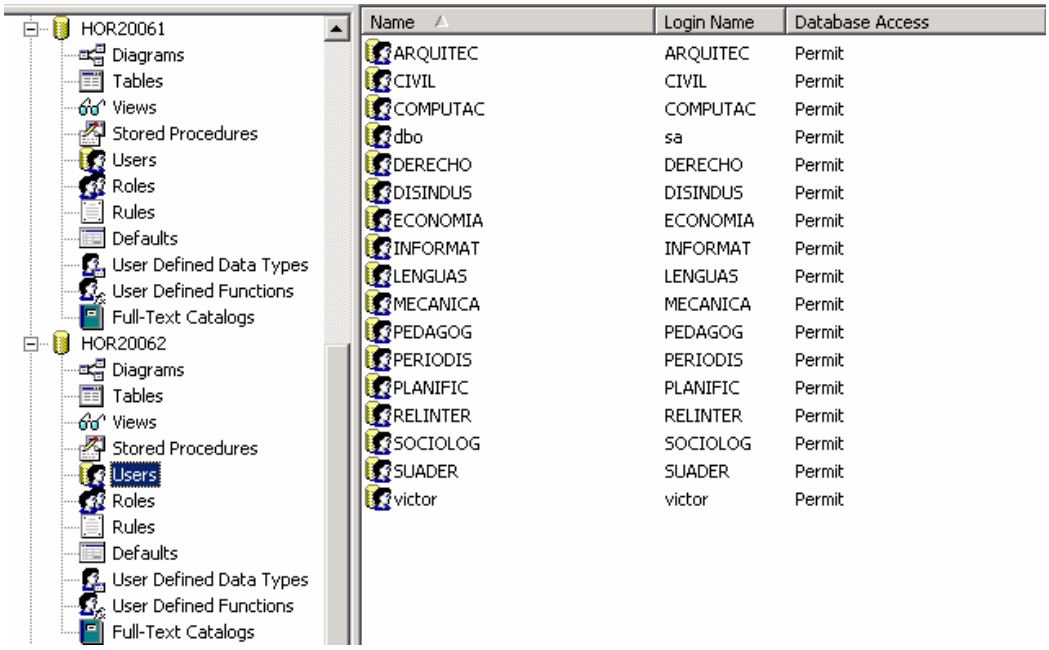


Figura 3.1 Imagen Enterprise Manager

El cual tiene que considerar también que tiene huecos de seguridad, en especial un gusano que hizo que el servidor y la red se cayeran por lo que se tienen que revisar periódicamente los parches y el service pack 3, los cuales se tienen que leer con mucha atención o se pueden dañar los datos o el accesos a datos, tal como la versión 4 que ya no soporta sistemas operativos con Windows 98.

Se asignaron una contraseña al administrador por seguridad, y una contraseña a cada una de las áreas, para poder monitorear los accesos a las bases de datos, donde por norma sólo el área puede ver su información y no puede accesar a ninguna otra, en caso de que se dañe la base de datos, es responsabilidad tanto del administrador como del área correspondiente.

También se cuenta con otro programa, el cual es muy importante para las consultas de las bases o para realizar operaciones en la base, llamado SQL Query Analyzer.

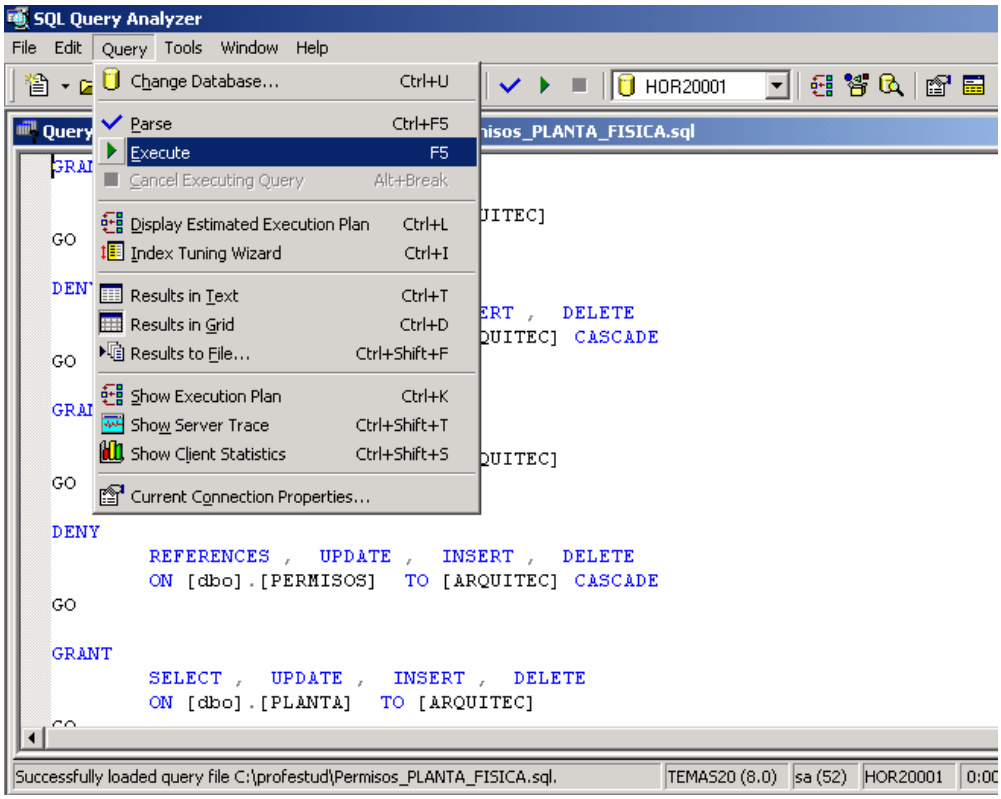


Figura 3.2 Pantalla SQL Query Analyzer

3.2.1.4. Microsoft Visual Basic

Visual Basic creado por Microsoft en 1985, desarrollado para crear aplicaciones gráficas que contienen objetos (pero no hay que confundirlo con un lenguaje orientado a objetos, aunque su funcionamiento interno sea muy parecido), métodos y propiedades.

Donde una propiedad según F. J. Ceballos "Representa todos los datos que por definición están asociados con ese objeto", lo cual es un concepto claro de una propiedad, el usuario puede definirlo porque son propiedades de tipo públicas, por ejemplo el nombre de un objeto.

Los métodos son todos los posibles eventos que se pueden desarrollar u ocurrir en un objeto, donde el programador puede ingresar el código para que funcionen cuando el método o evento se active, el más común es cuando se da un clic con el ratón.

Por cada aplicación se desarrolla un proyecto donde se almacenarán todos los objetos que participarán en la construcción del programa. Los objetos más usados se mencionarán al momento de construir el programa.

3.2.1.5. Programa Administrador

El programa administrador no se menciona en los programas a utilizar, ya que fue creado en Visual Basic, pero no fue desarrollado por RDO, por razones de seguridad no se mostrará el código de programación, pero sí el funcionamiento del mismo.

La pantalla principal del programa administrador es muy sencilla, ya que sólo se ven los botones para acceder a cada una de las bases globales con que cuenta el programa, como lo muestra la figura 3.3.

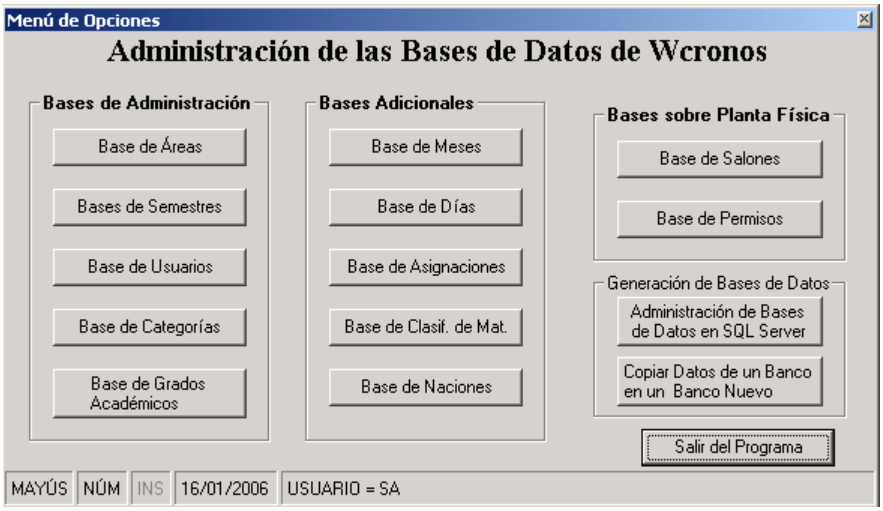


Figura 3.3 Pantalla Principal del Programa Administrador

El programa administrador realiza las siguientes funciones:

- Administrar las bases principales (Áreas, usuarios, permisos y otras).
- Creación de las bases de datos por área y semestre.
- Copiar la información de un semestre a otro (como el programa de utilería de Cronos, el cual generaba las bases).
- Administración de cuentas.

Éste trabaja directamente en el servidor y no puede ser ejecutado por otra computadora, para tener un control estricto en los accesos a las bases de datos.

El programa ayuda a validar la información ingresada, otra opción sería ingresarla por medio de SQL o directamente por el programa visual de SQL Server, lo cual puede dañar la integridad de la información y causar la pérdida de tiempo al reparar los errores.

Las partes de mayor importancia del programa son la administración de áreas, usuarios y bases para Wcronos, por lo que sólo se muestra la administración de las bases de datos como se ve en la pantalla 3.4.

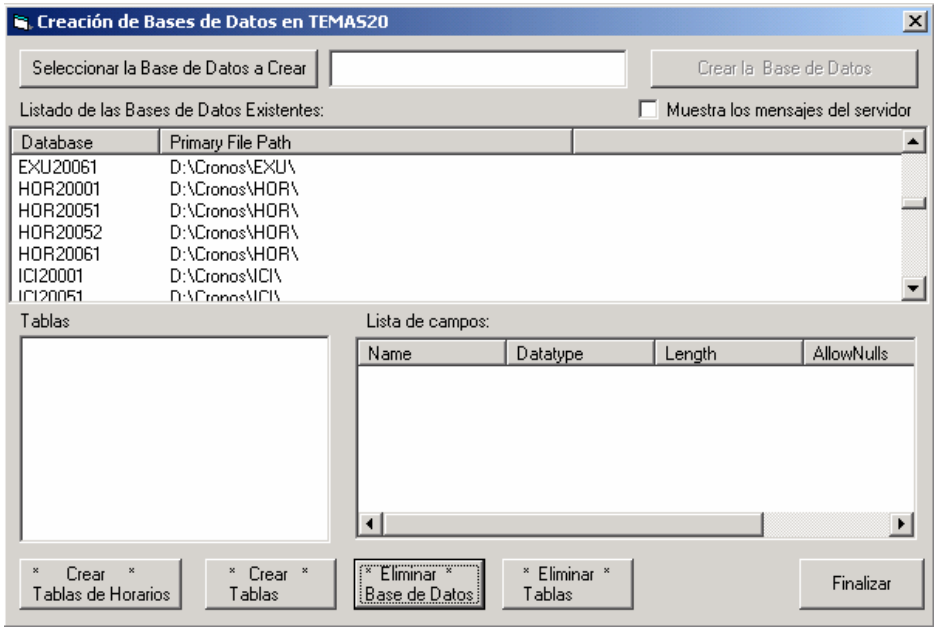


Figura 3.4 Pantalla donde se administran las bases de datos

La pantalla comprende tres ventanas, en donde se mostrarán las bases creadas, las tablas y los campos.

También cuenta con dos botones en la parte superior, uno para seleccionar el área y el semestre (pantalla 3.5), que son dos combos donde se cargan las áreas y todos los semestres dados de alta, al presionar aceptar busca el registro del área y trae la abreviatura de tres letras para juntarla con el semestre y dar el nombre de la base de datos a crear.

Figura 3.5 Pantalla para seleccionar la bases de datos

Ya con la base creada, se envía el nombre al cuadro de texto de la pantalla de administración de bases, se presiona el botón "Crear Base", se crea la base vacía y se agrega a la lista de bases de datos.

Ya creada la base se va al menú principal (figura 3.3), para crear la estructura o modificarla mediante intrucciones SQL, y posteriormente ingresar los permisos necesarios. Para realizar un proceso de exportación entre bases de datos, hay que seleccionar el botón “Copiar Datos de un Banco Nuevo”, mostrará otra pantalla para seleccionar las base origen y la base destino como lo muestra la figura 3.6.

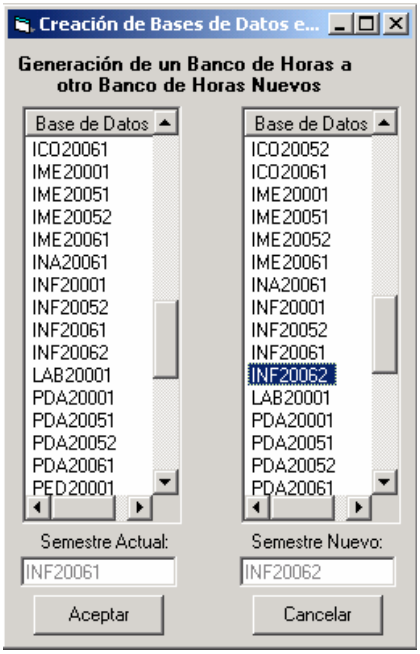


Figura 3.6 Pantalla para seleccionar las bases a copiar.

Ya seleccionadas las dos bases se presiona el botón "Aceptar", se va a la siguiente pantalla (figura 3.7.), donde se copia el contenido de cada una de las tablas de la base origen a la base recién generada que son los tres primeros botones, al copiar estos tres rubros uno se da cuenta porque se bloquean y se activa el botón "Generación de Banco".

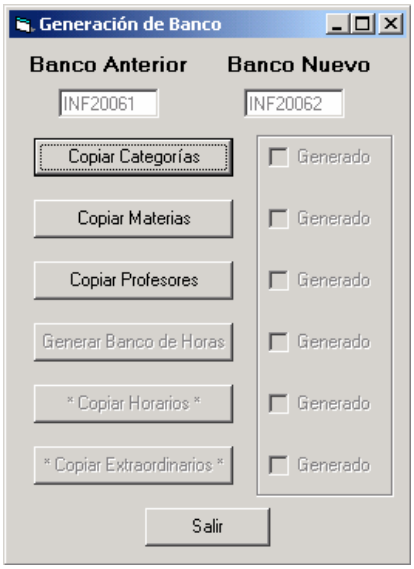


Figura 3.7 Pantalla para copiar la información de una base a otra.

Al presionar se copia la información que tiene horas asignadas y se elimina en el banco de horas la información que no es útil, y todos los grupos se ponen en 0, para no tener problemas con la nueva distribución en los horarios.

3.3. Bases de Datos

Con base en el desarrollo del capítulo I y análisis del programa Cronos en el capítulo II se desea tener un modelo Entidad-Relación, que responde a las características y necesidades de cada área, lo cual es más difícil ya que esto no sólo contempla 12 carreras, 4 posgrados, un Centro de Lenguas Extranjeras y un Sistema de Universidad Abierta (Carrera de Derecho), sino también 9 áreas administrativas que tienen diversas modalidades de personal contratado.

3.3.1. Modelo Entidad-Relación

El modelo usado por el programa Cronos, conforme a necesidades y requerimientos se puede simplificar sin alterar la estructura básica por una más práctica, usando reglas de normalización.

Normalización de datos en la enciclopedia se define como *"Fijación de determinadas dimensiones y calidades a un producto"* (Enciclopedia Salvat-Diccionario), refiriéndose el término a lo que es un producto físico y se puede interpretar que un producto puede ser un programa o hasta una base de datos, no es físico pero sí es un producto lógico, el cual se puede adecuar el concepto para una base de datos aplicándose a la fijación de determinadas reglas y procesos para estos fines:

- Mejorar velocidad de acceso.
- Minimizar la cantidad de veces que se accede a una pieza de información.
- Mantener la integridad de la base de datos.

Las reglas de normalización como se describió en el capítulo I fueron presentadas por Edgar F. Ted Codd, éstas son:

I. Eliminar grupos repetitivos.

II. Eliminar los datos redundantes.

III. Eliminar las claves que no se relacionen con la clave principal.

IV. No almacenar datos calculados en las tablas.

V. Aislar las múltiples relaciones conexas

Al realizar la normalización de la base de datos no se realizaron las dos últimas reglas de normalización, ya que se afectaba el proceso de los reportes o durante el programa se generaba inestabilidad al mismo.

El modelo final de la base es la siguiente figura 3.8:

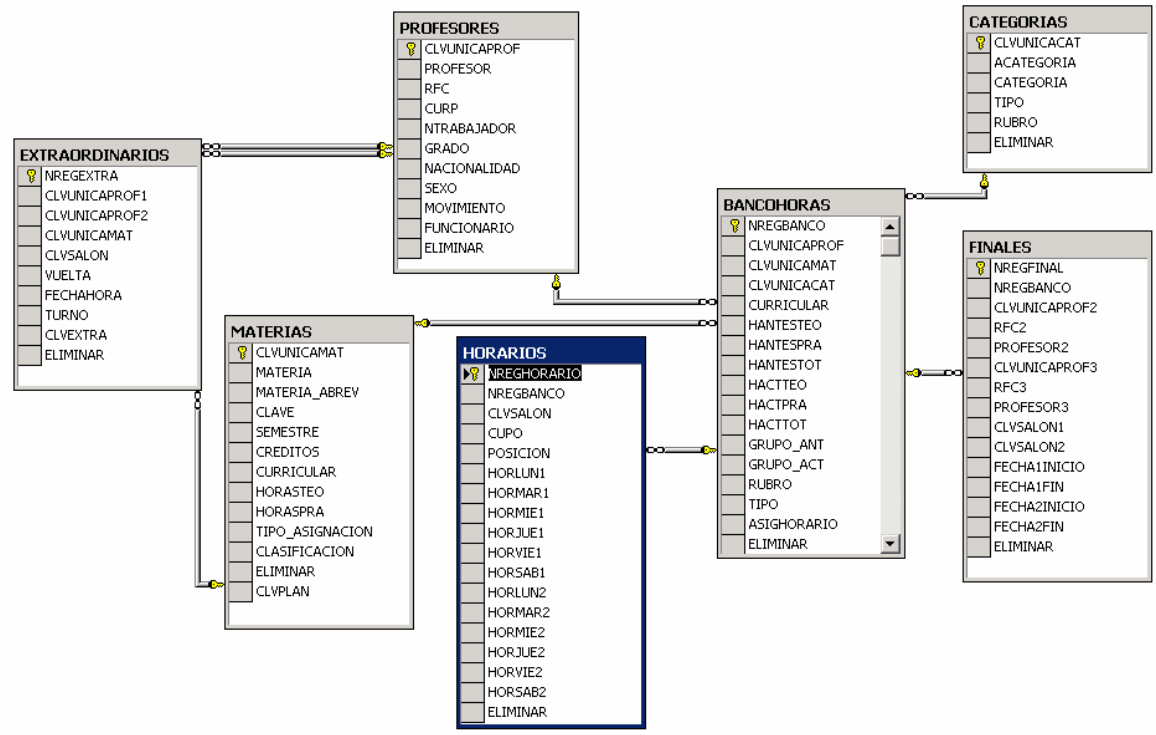


Figura 3.8 Modelo relacional de las tablas para el banco de horas

Cada parte de este modelo se explica conforme se construye el programa, tomando como base las estructura de las bases de los programas “Cronos”, “New” y “Kardex”.

Cabe resaltar que en cada área administrativa en la UNAM, se dan varios cambios y por ello se requieren modificaciones, ya sea a la base de datos o a los reportes, como ejemplo de ello fue la asignación del plan de estudios a las materias en 2004, por lo que se asignaron en 2006, por lo que tuvo que ser alterado en su estructura, mas no en los reportes.

Otro punto importante en este diseño era el ancho de banda sobre el que corría la aplicación, ya que al principio del banco de horas se cargan los datos necesarios en el CPU, lo cual hace el programa muy pesado, pero aumenta la velocidad por tráfico de red.

3.4. Construcción del Programa

Dentro del análisis previo se tiene considerada la estructura del programa, en la cual se tiene que diseñar la forma de capturar la información según el análisis que se dio anteriormente y junto con la base de datos, se empieza a construir el programa de Wcronos en plataforma Visual Basic 6.0 , abarcando todos los Módulos que se usaron en Cronos.

Antes de empezar a diseñar el programa, como se va a utilizar una base de datos remota es necesario implementar un proceso de validación, por lo que se crea un formulario para dar acceso a dicha base; el formulario propuesto se muestra en la figura 3.9.

Figura 3.9 Validación de Usuario

La estructura de programación de este formulario no será mencionada por cuestiones de seguridad del programa y de la base de datos, ya que se encuentra operando actualmente, por lo que se pasará al siguiente formulario que es el menú principal, el cual se muestra en la figura 3.10.

El menú principal es un formulario de tipo MDI, al que se le agregó un menú amigable como existe en la mayoría de los programas, una barra de herramientas donde se tienen los principales íconos con sus respectivas imágenes tomadas del objeto "ImageList", además de una barra de estado en la parte inferior "StatusBar"; aquí contiene la información básica de referencias como existe en los principales editores de texto, agregando usuario y carrera.

Figura 3.10 Menú principal de Wcronos

Cada uno de los menús e íconos que llevan a su formulario correspondiente un modelo de código, donde el único cambio es el nombre de menú y el nombre del formulario, como lo muestra el siguiente ejemplo:

```
Private Sub conshorsalon_Click()
   MousePointer = 11                    'Pongo el ratón con el reloj de arena
   fconssalon.Show                      'Activo la pantalla a utilizar
   MousePointer = 0                     'Pongo el ratón en su forma normal
End Sub
```

La única excepción es la opción "Salir del Programa" o el ícono para terminar el programa, ya que desconecta cualquier conexión o entorno RDO, que no se ha cerrado por lo que se aplica tanto al dar un clic o al cerrar el formulario con la 'X', ejecutando el siguiente programa:

```
Private Sub msalir_Click()
   desconectarEntorno                   'Desconecto el entorno RDO
   End                                  'Finalizo el programa
End Sub
```

Otro aspecto importante en este formulario es que se tiene que guardar en variables globales los siguientes campos que se utilizarán a lo largo del programa:

```
Global usuario As String                'Usuario
Global clave As String                  'Password (Clave codificada)
Global clave_carrera  As String         'Carrera
Global conexionodbc As String           'Conexión
Global Período_escolar As String        'Semestre
```

Y por último se realizó un formulario para mostrar el menú (Fig. 3.11) como lo mostraría Cronos, para que el usuario se adapte al cambio de un programa de MS-DOS a un programa de Windows, que a la fecha ya es obsoleto, porque los usuarios prefieren manejar iconos o menús.

Figura 3.11 Menú de inicio rápido.

3.4.1. Materias

El primer formulario que se va a tratar para cada una de las áreas es el de Materias, que es una base esencial para realizar movimientos y procesos en el banco de horas, la estructura de la base de materias es la siguiente:

MATERIAS	TIPO	LONG	LLAVE	NOMBRE
CLVUNICAMAT	INT	4	*	Clave Única de Materia
MATERIA	VARCHAR	254		Nombre de la Materia
MATERIA ABREV	VARCHAR	50		Abreviatura de la Materia
CLAVE	VARCHAR	7		Clave de la Materia
SEMESTRE	VARCHAR	6		Semestre (Opcional)
CRÉDITOS	FLOAT	8		Créditos que tiene la Materia
CURRICULAR	VARCHAR	1		Si es curricular la Materia
HORASTEO	FLOAT	8		Horas Teóricas de la Materia
HORASPRA	FLOAT	8		Horas Prácticas de la Materia
TIPO ASIGNACION	VARCHAR	10		Tipo de Asignatura
CLASIFICACION	VARCHAR	20		Clasificación
ELIMINAR	VARCHAR	1		N no eliminado, otro carácter se elimina
CLVPLAN	INT	5		Clave del plan de estudios

Con base en esta estructura se creó un formulario donde se manejan en cuadros de texto[22], o un 'ComboBox' que permite seleccionar opciones predeterminadas tomadas de la base de datos maestra. Además del objeto 'OptionButton' para la opción Sí o No.

Se insertaron los botones para realizar las funciones básicas que se realizan en una base de datos: agregar, eliminar, cambiar y terminar la sesión. Sólo faltaría el proceso de consulta, por lo que se usa una cuadrícula que se llama 'MSFlexGrid', con la cual se puede consultar e intercambiar datos con los cuadros de texto y los Combobox's, lo cual ahorra mucho tiempo, considerando el diagrama de programación por módulos:

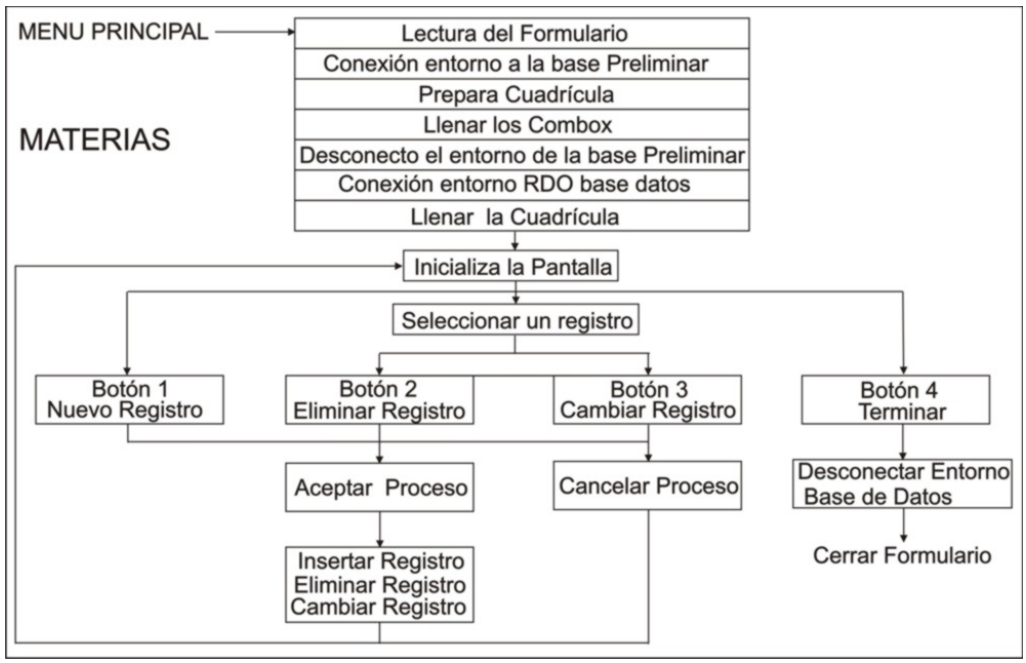


Diagrama 3.1 Módulos de programación para el formulario de Materias

[22] En donde se escriben letras o números dependiendo la validación con que se programe.

En la parte inferior del formulario se agregó un 'StatusBar' (Barra de Estado), para poner mensajes de ayuda al usuario. De acuerdo con lo anterior se diseñó el siguiente formulario, ver la figura 3.12.

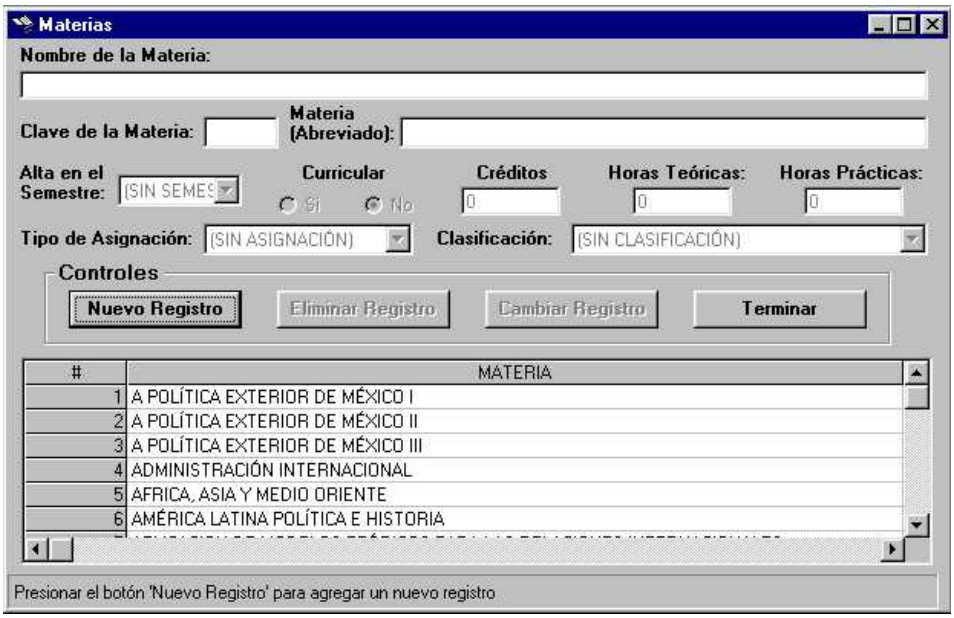


Figura 3.12 Formulario de Materias.

3.4.1.1. Módulo base de inicialización del formulario

El primer paso para poder usar el formulario es diseñar un modelo para cuando se lea, por lo que se creó el siguiente modelo, con base en los comentarios marcados con negritas:

```
Private Sub Form_Load()
   Dim nombre, acceso, base, lectura As Variant      'Variables locales
   On Error GoTo MENSAJE                             'Activación del control de errores RDO
  'Leo los valores de la base predeterminada (nombre, acceso y base)
  'Conectar Entorno RDO para llenar desde el inicio
  Call ModulosGlobales.conectarBD(nombre, acceso, base)
   fmaterias.WindowState = 2                         'Establecer el tamaño del formulario
   PROCESO = ""                                      'Vacío la variable PROCESO para no dañar el programa
   'Prepara el cuadrícula donde llevarán los datos para la consulta
   Prepara_Cuadro
   llena_semestres                                   'Lleno los combos o listas
   Llena_Asignaciones
   Llena_Clasif
   Call desconectarBD                                'Cierro la base de datos predeterminada
   'Establezco los nuevos valores de la conexión a la base que se va a usar
   nombre = usuario                                  'Usuario
   acceso = clave                                    'Contraseña
   base = clave_carrera + Período_escolar            'Base de Datos a utilizar
   lectura = "S"                                     'Consulta de solo lectura
   'Conecto a la base de datos a usar
   Call ModulosGlobales.conectarBD(nombre, acceso, base)
   'Pongo las materia o los datos para llenar  los cuadricula de acuerdo al formulario
   LlenaCuadro
```

```
   Inicializa                                    'Inicializa la pantalla del formulario
   'Bloqueo los menús para que no seleccionen otro programa
   MDIPrincipal.obancohoras.Enabled = False
   'La Barra de herramientas
   MDIPrincipal.BHerramientas.Buttons.Item(1).Enabled = False
   Exit Sub                                      'Termino la fase de lectura del formulario

MENSAJE:  ++++++ Módulo para mostrar errores +++++++
   'Si existe un error en la conexión lo muestro en pantalla
   Dim Errorbase As rdoError
   Dim aviso As String
   'Grabo el mensaje en una variable
   aviso = "Se ha producido un error al abrir la conexión:" & Err & " - " & Error & vbCr
   MsgBox aviso                                  'Muestro el error cometido
   For Each Errorbase In rdoErrors               'Si hay más errores los muestra en pantalla
                                                 'Grabo el mensaje en una variable
      aviso = aviso & Errorbase.Description  & ":" & Errorbase.Number & vbCr
       MsgBox aviso                              'Muestro cada uno de los mensajes
   Next Errorbase
   Exit Sub                                      'Termino la lectura
```

3.4.1.2. Módulo de conexión y desconexión del entorno RDO

La parte principal del anterior código no se muestra, sino que es llamado al módulo global con la instrucción "Call", en donde se establece previamente el entorno y la máquina de RDO, lo cual facilita el proceso de conexión a la base de datos por medio del siguiente módulo:

```
Sub conectarBD(usuario1 As Variant, clave1 As Variant, basedatos As Variant)
   Dim Datosconexion As String                   'Cadena de conexión
   On Error Resume Next                          'Activo el controlador de errores
   'Agrego a la variable los datos necesarios para realizar la conexión
   Datosconexion$ = "UID=" & usuario1 & ";" _
            & "PWD=" & clave1 & ";" _
            & "DATABASE=" & basedatos & ";" _
            & "SERVER=10.10.10.1" & ";" _
            & "Driver={SQL SERVER}" & ";" _
            & "DNS=";"
   'segun como lo detecte el ODBC, Ahora establezco la conexión a abrir
   Set rdoconexion = rdoentorno.OpenConnection(dsName:="", Prompt:=rdDriverNoPrompt,
ReadOnly:=False, Connect:=Datosconexion)
   If Err.Number > 0 Then                        'Si existe error
      Basura = Aviso_Error(Err.Description, Err.Number) 'Muestro el error
      Exit Sub                                   'Salgo del ciclo, sin escribir nada
   End If
End Sub
```

Pero este código no funciona si no se establece contacto con el servidor generando un entorno de tipo RDO que se activa al inicio del programa, tal como se describe en el Capítulo 1, sólo escribiendo el código para establecer el entorno RDO:

```
Function conectarEntorno()
   Set rdoentorno = rdoEngine.rdoEnvironments(0)  'Establezco un entorno de trabajo
   rdoentorno.CursorDriver = rdUseOdbc            'Establezco el cursor ODBC
End Function
```

Por supuesto se tiene que mostrar el código para desconectar la base de datos:

```
Function desconectarBD()
   If Not rdoconexion Is Nothing Then          'Si no está conectada la conexión continua
      rdoconexion.Close                        'Cierro la conexión
      Set rdoconexion = Nothing                'Borro de memoria la conexión hecha
   End If
End Function
```

Pero también al finalizar se tiene que desconectar el entorno del programa, por lo que se verá el código para quitar el entorno:

```
Function desconectarEntorno()
   If Not rdoentorno Is Nothing Then           'Si no existe el entorno virtual continuo
      rdoentorno.Close                         'Cierro el entorno virtual
      Set rdoentorno = Nothing                 'Borro de memoria el entorno virtual creado
   End If
End Function
```

3.4.1.3. Módulo base de llenado de un *Combobox*.

Como se ve, en el módulo siguiente se deben llenar todos los ComboBox's que tiene el formulario, para ello se genera un módulo por cada uno, el cual será llamado cada vez que se requiera y no sólo al inicio de la lectura de cada formulario, a continuación se muestra el módulo de llenado de semestres:

```
Private Sub Llena_Semestres()
   Dim CADENA As String                        'Variables locales
   On Error GoTo MENSAJE                       'Activo el control de errores
   ComboSemestre.AddItem "(SIN SEMESTRE)"      'Valor  predeterminado del ComboBox
   'Pido al servidor la sentencia de lectura en SQL
   Set rdotabla1 = rdoconexion.OpenResultset("SELECT ANIO, PERÍODO FROM SEMESTRES")
   With rdotabla1                  'Establezco un entorno para usar Resultset o las tablas pedidas
      If Not .EOF Then                         'Si no tengo registros continúo y no entro al IF
         While Not .EOF                        'Realízalo hasta fin de archivo
            'Guarda el registro en una cadena en caso de que sea una cadena compuesta
            CADENA = !ANO + !PERÍODO
            ComboSemestre.AddItem CADENA       'La cadena se  anexa  al combo
            .MoveNext                          'Me muevo al siguiente registro
         Wend
      End If
   End With
   rdotabla1.Close                             'Cierro el Resultset o la consulta
   Set rdotabla1 = Nothing                     'Limpio registros de la tabla1
   ComboSemestre.ListIndex = 0                 'Inicializo el ComboBox, dejándolo sin valor
   Exit Sub                                    'Termino
         'Módulo para mostrar errores
End Sub
```

El mismo proceso se llevará a cabo para cada uno de los 'ComboBox' que se tienen en el programa en los formularios de captura.

3.4.1.4. Módulo para dar formato y el llenado de la cuadrícula

Una de las ventajas que tiene un programa como el anterior es que se pueden consultar los datos dentro de la misma pantalla, para ello se tiene que definir qué datos se van a poner en la cuadrícula, cuáles serán visibles y cuáles ocultos, para ello se muestra cómo se define los anchos de la cuadrícula, como se observa en el siguiente código:

```
Private Sub Prepara_Cuadro()
   Cmaterias.Cols = 12                          'Determino el número de columnas
   Cmaterias.Rows = 1                           'Determino el número de renglones
   'Establezco el tamaño de cada una de las columnas
   Cmaterias.ColWidth(0) = 1000 'CLVUNICAMAT
   Cmaterias.ColWidth(1) = 7600 'MATERIA
   Cmaterias.ColWidth(2) = 700 'CLAVE DE MATERIA
   Cmaterias.ColWidth(3) = 10 'MATERIA ABREVIADA      'Columna oculta
   Cmaterias.ColWidth(4) = 600 'SEMESTRE
   Cmaterias.ColWidth(5) = 600 'CURRICULAR
   Cmaterias.ColWidth(6) = 600 'CREDITOS
   Cmaterias.ColWidth(7) = 800 'HORAS TEORICAS
   Cmaterias.ColWidth(8) = 800 'HORAS PRACTICAS
   Cmaterias.ColWidth(9) = 2000 'TIPO DE ASIGNACION
   Cmaterias.ColWidth(10) = 2500 'CLASIFICACION DE LA MATERIA
   Cmaterias.ColWidth(11) = 700 'CLAVE DEL CLVPLAN   'Columna anexada 2006
End Sub
```

Antes de continuar con el siguiente código, se tiene que explicar un cambio con respecto al Cronos anterior, aquí se manejan dos tipos de claves de materias, por un conflicto entre los planes de estudios otorgados por DGAE (Dirección General de Asuntos Estudiantiles), ya que se repiten las claves o nombres en algunas materias, en el programa se implementó una clave única para no repetir o confundir materias, ya que para Servicios Escolares se tiene un arreglo con una combinación con el plan de estudios y la carrera que se implementa para generar un índice para ciertas carreras, que para el programa no es un proceso adecuado, en lo que respecta a horarios.

Sólo se tienen que consultar por Internet en la página de DGAE, los planes de estudios correspondientes a las carreras de Ingeniería Mecánica Eléctrica y Economía, ya que existen variaciones en los planes de estudios en algunas de las últimas materias.

Haciendo esta aclaración, se muestra el modelo de código para llenar la cuadrícula que se tomó para todos los módulos de captura:

```
Private Sub LlenaCuadro()

   Dim CADENA As String                         'Doy de alta las variables a usar en este módulo
   On Error GoTo MENSAJE                        'Activo el control de errores
   Cmaterias.Clear                              'Inicializo la cuadrícula para no repetir registros
   Cmaterias.Rows = 1                           'Reduzco el número de renglones
   'Pongo los títulos de la cuadrícula
   Cmaterias.Row = 0                            'Me ubico en el renglón 0
   Cmaterias.Col = 0                            'Me ubico en la columna 0
```

```
   Cmaterias.CellAlignment = 4                    'Alineación de la columna
   Cmaterias.Text = "#"                           'Nombre de la celda
…….. 'Repito por todos los campos dados de alta al preparar la cuadrícula.
   Cmaterias.Col = 11                             'Me ubico en la columna 11
   Cmaterias.CellAlignment = 4                    'Alineación al centro
   Cmaterias.Text = "CLVPLN"                      'Nombre de la celda
   'Realizo una llamada al servidor para traer sólo los datos necesarios
   Set rdotabla1 = rdoconexion.OpenResultset("SELECT ELIMINAR, MATERIA, CLVUNICAMAT, CLAVE,
MATERIA_ABREV, SEMESTRE, CURRICULAR, CREDITOS, HORASTEO, HORASPRA,
TIPO_ASIGNACION, CLASIFICACION, CLVPLAN FROM MATERIAS WHERE ELIMINAR = 'N' ORDER
BY MATERIA", rdOpenKeyset, rdConcurReadOnly, rdAsyncEnable + rdExecDirect)
   With rdotabla1                                 'Trabajo con la tabla 1
      If Not .EOF Then                            'Si no tengo registros continúo
         While Not .EOF                           'Realízalo hasta fin de archivo
            Dim cadena3 As String
            Cmaterias.Col = 0                     'Me ubico en la 1 celda
            Cmaterias.Row = Cmaterias.Rows - 1    'Me ubico en el renglón actual
            CADENA = !CLVUNICAMAT                 'Guarda el registro en una cadena
            Cmaterias.AddItem CADENA              'La cadena se anexa al combo, agregando un renglón
            Cmaterias.Row = Cmaterias.Rows - 1    'Me ubico en el renglón nuevo
            Cmaterias.Col = 1                     'Me ubico en la 2 celda
            CADENA = !MATERIA       'Guarda el registro en una cadena (para cadena o número)
            Cmaterias.Text = CADENA               'La cadena se anexa a la cuadrícula
            … 'Inserto las demás columnas
            Cmaterias.Col = 11                    'Me ubico en la 12 celda
          'En caso de que el campo acepte valores nulos se tiene que validar la entrada
            If !CLVPLAN <> "" Then
               CADENA = !CLVPLAN                  'Guarda el registro en una cadena
               Cmaterias.Text = CADENA            'La cadena se anexa a la cuadrícula
            End If
            .MoveNext
         Wend
      End If
   End With
   rdotabla1.Close                                'Cierro el Resultset
   Set rdotabla1 = Nothing                        'Limpio registros de la tabla1
   Exit Sub                                       'Termino
'Módulo para mostrar errores
End Sub
```

Pero hay que definir que la cuadrícula se debe ajustar al tamaño de la pantalla desde 640x480 hasta la máxima resolución que pueda soportar la computadora, lo cual entre más grande sea la resolución, el tamaño de la cuadrícula y del formulario aumentará, sólo al incorporar el siguiente código en el evento 'Resize' del formulario:

```
Private Sub Form_Resize()
   Dim ANCHO, LARGO As Integer                    'Variables locales
   ANCHO = fmaterias.Width                        'Obtengo el ancho y el largo del formulario
   LARGO = fmaterias.Height
  If ANCHO > 9299 Then                  'Si no rebasa el tamaño mínimo del ancho del formulario
     Cmaterias.Width = ANCHO – 350               'Ajusta el ancho de la cuadrícula de materias
   End If
   If LARGO > 5880 Then                  'Si no rebasa el tamaño máximo del ancho del formulario
     Cmaterias.Height = LARGO – 3900             'Ajusta el largo de la cuadrícula de materias
   End If
```

```
End Sub
```

3.4.1.5. Validación de datos al insertar registro en el formulario

Al momento de ingresar información al formulario y colocarse en el segundo cuadro de texto[23], en donde lo primero es presentar una referencia de la ayuda en el panel, al momento de estar en el objeto (cuadro de texto, Combobox, etc.), en el evento 'GotFocus' como lo muestra el código:

```
Private Sub Tmateria_GotFocus()
   BMateria.Panels(1) = "Ingresar el 'Nombre de la Materias'"    'Muestro el mensaje
End Sub
```

Si es necesario al momento de escribir algún dato se puede validar en este caso sería para números:

```
Private Sub Tclvmateria_KeyPress(KeyAscii As Integer)
   If KeyAscii = 13 Then                          'Verifico si es ENTER
      Tabrevmateria.SetFocus                      'Me voy al siguiente cuadro de texto
      'Tabrevmateria.SelText                      'Selecciono el texto
   Else
      If KeyAscii = 8 Then                        'Si presiono el backspace
         Exit Sub                                 'Termino el módulo
      End If
      If KeyAscii < 48 Or KeyAscii > 57 Then      'Sólo acepto caracteres numéricos (0-9)
         KeyAscii = 0                             'En caso contrario anulo la tecla tecleada
      End If
   End If
End Sub
```

Y el caso de cantidades, se debe verificar que sólo exista un punto en la cantidad insertada, esto se hace antes de verificar si son caracteres numéricos, entonces se anexa al código anterior lo siguiente:

```
      If KeyAscii = 46 Then                       'Si el caracter es un punto
         If InStr(1, Tcreditos.Text, ".", 1) = 0 Then   'Si no encuentro un punto en la caja de texto
            Exit Sub                              'Termino el módulo
         Else
            KeyAscii = 0                          'Anulo el caracter al encontrar otro punto
            Exit Sub                              'Termino el módulo
         End If
      End If
```

Otro punto de validación es definir el número de caracteres que tendrá la caja de texto, es muy importante poner un límite de caracteres, ya que puede dañar la base de datos, por ello sólo se utiliza la propiedad 'MaxLength', y se le asigna el tamaño correspondiente.

[23] El primer cuadro siempre será la clave única del formulario, por lo que el usuario no podrá realizar ninguna modificación y se asignará por el servidor cuando se ingrese un nuevo registro.

Para el caso de los Combobox, en la propiedad de 'Style' se selecciona la opción 'DropDown List' para sólo seleccionar las opciones que se insertaron.

3.4.1.6. Módulo de inicialización de la pantalla del formulario

Al validar cada uno de los elementos del formulario hay que establecer un módulo para que tenga los valores predeterminados del formulario por ello, hasta ahora mostramos cómo funciona el módulo "inicializa", que viene incluido en el código de inicio del formulario, el cual es el siguiente:

```
Private Sub Inicializa()
   'Pongo la información predeterminada del formulario
   tclvunicamateria.Text = ""
   Tmateria.Text = ""
   Tclvmateria.Text = ""
   Tabrevmateria.Text = ""
   Tplan.Text = ""
   ComboSemestre.ListIndex = 0
   OSI.Value = False
   ONO.Value = True
   Tcreditos = "0"
   Thteoricas = "0"
   thpracticas = "0"
   ComboAsignacion.ListIndex = 0
   ComboClasif.ListIndex = 0
   'Desactivo las cajas de texto para que no las modifiquen
   Tmateria.Enabled = False
   Tclvmateria.Enabled = False
   Tabrevmateria.Enabled = False
   Tplan.Enabled = False
   ComboSemestre.Enabled = False
   OSI.Enabled = False
   ONO.Enabled = False
   Tcreditos.Enabled = False
   Thteoricas.Enabled = False
   thpracticas.Enabled = False
   ComboAsignacion.Enabled = False
   ComboClasif.Enabled = False
   'Activo las opciones que requiero
   MarcoControles.Enabled = True
   'Muestro todos los botones
   BOTON1.Visible = True
   BOTON2.Visible = True
   BOTON3.Visible = True
   BOTON4.Visible = True
   'Habilito los botones necesarios
   BOTON1.Enabled = True
   BOTON4.Enabled = True
   'No activo las opciones de eliminar y cambiar
   BOTON2.Enabled = False
   BOTON3.Enabled = False
   'Pongo los títulos originales de los botones
   BOTON1.Caption = "Nuevo Registro"
   BOTON2.Caption = "Eliminar Registro"
```

```
   BOTON3.Caption = "Cambiar Registro"
   BOTON4.Caption = "Terminar"
   Cmaterias.Enabled = True                    'Activo el cuadro donde tengo mis materias
   PROCESO = "INICIO"                          'El proceso a seguir es de INICIO
End Sub
```

El módulo de inicialización se puso completo para poner la nomenclatura que se utilizó en el programa, de acuerdo al siguiente cuadro:

Objeto	Nomenclatura	Significado
Textbox	T	Cuadro de texto
Combobox	Combo	Cuadro de selección
OptionButton	O	Círculo de opción
MSFlexGrid	C	Cuadrícula
CommandButton	Boton	Botón de procesos
Frame	Marco	Marco
Form	F	Formulario
Label	Label	Etiqueta

3.4.1.7. Módulos de los procesos por botón

Para evitar crear botones para cada proceso se diseñó un formato usando el comando de selección de Visual Basic: 'Select Case', donde el proceso es sencillo al seleccionar los tres primeros botones del formulario de materias (3.12), se ocultan los laterales, a los centrales se les cambia el título para realizar o cancelar el proceso, cambiar el nombre del proceso y se activan los objetos que puede manejar el usuario.

Para ingresar un registro se realizan varios pasos, primero hay que darle formato a los botones, luego ocultar los que no se van a utilizar, para posteriormente activar todos los objetos a utilizar, pero se tiene que inicializar cada uno de ellos porque se va a insertar un nuevo registro. Posteriormente se asigna el proceso de "NUEVO", para ser usados en el botón 2 ó 3, dependiendo de la opción del usuario; el código en el botón 1, es el siguiente:

```
Private Sub Boton1_Click()
   Select Case PROCESO                         'Primero veo que proceso voy a utilizar
      Case "INICIO"                            '* Deseo un Nuevo Registro *
         'Desactivo los botones que no utilizo
         BOTON1.Enabled = False
         BOTON2.Enabled = False  'El botón sólo se activará al dar un ENTER al término de la captura
         BOTON4.Enabled = False
         BOTON3.Enabled = Trae                 'Activo sólo el botón 3
         'Oculto los botones que no utilizo
         BOTON1.Visible = False
         BOTON4.Visible = False
         'Pongo los nuevos títulos
         BOTON2.Caption = "Guardar"
         BOTON3.Caption = "Cancelar"
         'Activo los objetos a modificar
         Tmateria.Enabled = True
         Tclvmateria.Enabled = True
         Tabrevmateria.Enabled = True
```

```
    Tabrevmateria.Enabled = True
    Tplan.Enabled = True
    ComboSemestre.Enabled = True
    OSI.Enabled = True
    ONO.Enabled = True
    Tcreditos.Enabled = True
    Thteoricas.Enabled = True
    thpracticas.Enabled = True
    ComboAsignacion.Enabled = True
    ComboClasif.Enabled = True
    'Vacío todos los cuadros de texto
    Tmateria.Text = ""
    Tclvmateria.Text = ""
    Tabrevmateria.Text = ""
    Tplan.Text = ""
    'Se agrega un cero a los cuadros que trabajan con números o cantidades
    Tcreditos.Text = "0"
    Thteoricas.Text = "0"
    thpracticas.Text = "0"
    'Vacío los combos
    ComboSemestre.ListIndex = 0
    ComboAsignacion.ListIndex = 0
    ComboClasif.ListIndex = 0
    'Pongo el valor por defecto en los círculos de selección
    OSI.Value = True
    ONO.Value = False
    PROCESO = "NUEVO"                    '#####Cambio el proceso######
    Tmateria.SetFocus                    'Me ubico en el nombre de la materia
  Case Else
    Inicializa                           'Inicializo la pantalla en caso de no encontrar el proceso
End Select
```

En caso de borrar un registro (presionar el botón 2), es un proceso que se realiza similar al anterior, donde se omite inicializar y activar los objetos del formulario, sólo que se cambia el nombre del proceso "BORRAR" y el de los títulos

```
    BOTON2.Caption = "Borrar"            'Pongo los nuevos títulos
    BOTON3.Caption = "Cancelar"
    PROCESO = "BORRAR"                   'Cambio el nombre del proceso
```

El proceso "CAMBIAR" (presionar el botón 3), se omite al inicializar los objetos, pero si se activan los objetos del formulario para que el usuario pueda realizar cambios y como en el proceso anterior sólo se cambian los títulos y el nombre del proceso de la siguiente forma:

```
    BOTON2.Caption = "Actualizar"        'Pongo los nuevos títulos
     BOTON3.Caption = "Cancelar"
    PROCESO = "CAMBIAR"                  'Cambio el proceso
```

Por ello, se va a explicar el código del botón 2 en donde se omite el proceso de inicialización, muestra de errores y validación en el proceso "CAMBIAR", ya que se repite en el proceso "NUEVO"; no ser tan repetitivos en el código, el cual se muestra a continuación:

```
Private Sub Boton2_Click()
```

```
'Variables locales
Dim CADENASQL, Vclvunicamateria, Vmateria, Vclvmateria, Vabrevmateria, Vsemestre As Variant
Dim Vcurricular, Vcreditos, Vhteoricas, Vhpracticas, Vasignacion, Vclasificacion, Vplan As Variant
Dim CADENA As String
On Error GoTo MENSAJE                    'Activo el control de errores
Select Case PROCESO                      'Primero veo que proceso voy a utilizar
Case "INICIO"
    'Proceso inicializar antes de borrar un registro.
Case "NUEVO"
     'Validación de cada una de las variables
    If Tmateria.Text = "" Then           'Si no tiene materia
       'Muestro un mensaje
       MsgBox "Falta ingresar el 'Nombre de la Materia'", vbInformation, "Falta Información"
       Tmateria.SetFocus                 'Me ubico en el lugar de la materia
       Exit Sub                          'Termino el módulo
    End If
    If Tclvmateria.Text = "" Then
       MsgBox "Falta ingresar la clave de la Materia", vbInformation, "Falta Información"
       Tclvmateria.SetFocus
       Exit Sub
    End If
    If Tabrevmateria.Text = "" Then
       MsgBox "Falta ingresar la Materia de forma 'Abreviada'", vbInformation, "Falta Información"
       Tabrevmateria.SetFocus
       Exit Sub
    End If
    If ComboSemestre.ListIndex < 1 Then
       MsgBox "No se selecciono ningún 'Semestre'", vbInformation, "Error"
       Exit Sub
    End If
    If Tcreditos.Text = "" Then
       MsgBox "Falta asignar los 'Créditos de la Materia'", vbCritical, "Error"
       Tcreditos.SetFocus
       Exit Sub
    End If
    If Thteoricas.Text = "" Then
       MsgBox "Falta asignar las 'Horas Teóricas de la Materia'", vbCritical, "Error"
       Thteoricas.SetFocus
       Exit Sub
    End If
    If thpracticas.Text = "" Then
       MsgBox "Falta asignar las 'Horas Prácticas de la Materia'", vbCritical, "Error"
       thpracticas.SetFocus
       Exit Sub
    End If
    If ComboAsignacion.ListIndex < 0 Then    'Validación para los Combobox
       MsgBox "En la lista no hay ninguna asignación, reiniciar el programa", vbCritical, "Error"
       Exit Sub                          'Termino el módulo
    End If
    If ComboClasif.ListIndex < 0 Then
       MsgBox "En la lista no hay ninguna clasificación, reiniciar el programa", vbCritical, "Error"
       Exit Sub          'Termino el módulo
    End If
    '#############################################################
    '#VERIFICO SI EXISTE UNA MATERIA CON EL MISMO NOMBRE#
    '#############################################################
```

```
'Realizo una llamada al servidor
Set rdotabla1 = rdoconexion.OpenResultset("SELECT MATERIA, ELIMINAR FROM MATERIAS
WHERE ELIMINAR = 'N'")
With rdotabla1                          'Establezco un entorno para usar rdotabla1
  If Not .EOF Then                      'Si no tengo registros continuo
    While Not .EOF                      'Realízalo hasta fin de archivo
      CADENA = !MATERIA                 'Guarda el registro en una cadena
      If Tmateria.Text = CADENA Then      'Si existe una materia igual
        'Existe una materia repetido
        MsgBox "Existe una 'Materia' con el mismo nombre", vbInformation, "Materia Repetida"
        rdotabla1.Close                 'Cierro la tabla
        Set rdotabla1 = Nothing
        Exit Sub                        'Término el módulo
      End If
      .MoveNext                         'Me muevo al siguiente registro
    Wend
  End If
End With
rdotabla1.Close                          'Cierro la tabla
Set rdotabla1 = Nothing                 'Limpio registros de la tabla1
'################################
'#INSERTO EL NUEVO REGISTRO#
'################################
'Verifico si no son nulos y Guardo la información en cadenas
If Tmateria.Text <> "" Then
  Vmateria = "'" + Tmateria.Text + "'"
Else
  Vmateria = "''"
End If
If Tabrevmateria.Text <> "" Then
  Vabrevmateria = "'" + Tabrevmateria.Text + "'"
Else
  Vabrevmateria = "''"
End If
If Tplan.Text <> "" Then
  Vplan = "'" + Tplan.Text + "'"
Else
  Vplan = "''"
End If
If Tclvmateria.Text <> "" Then
  Vclvmateria = "'" + Tclvmateria.Text + "'"
Else
  Vclvmateria = "''"
End If
'Semestre
Vsemestre = "'" + RTrim(ComboSemestre.Text) + "'"
'Si se tiene alguna opción y se guarda en una cadena
If OSI.Value = True Then
  Vcurricular = "'S'"
Else
  Vcurricular = "'N'"
End If
'Veo si tiene valores nulos en cadenas numéricas o asigno el valor a una variable
If Tcreditos.Text <> "" Then
  Vcreditos = Tcreditos.Text
Else
```

```
        Vcreditos = "0"
      End If
      If Thteoricas.Text <> "" Then
        Vhteoricas = Thteoricas.Text
      Else
        Vhteoricas = "0"
      End If
      If thpracticas.Text <> "" Then
        Vhpracticas = thpracticas.Text
      Else
        Vhpracticas = "0"
      End If
      'Asignación de valores en combobox
      Vasignacion = "'" + RTrim(ComboAsignacion.Text) + "'"
      Vclasificacion = "'" + ComboClasif.Text + "'"
      'El campo Eliminar se asigna  el valor predeterminado  "N", que significa no borrado.
      'Guardo todo en una cadena SQL
      CADENASQL = "INSERT INTO MATERIAS (MATERIA, MATERIA_ABREV, CLAVE, SEMESTRE,
CURRICULAR, CREDITOS, HORASTEO, HORASPRA, TIPO_ASIGNACION, CLASIFICACION,
CLVPLAN, ELIMINAR) Values (" + Vmateria + ", " + Vabrevmateria + ", " +          Vclvmateria
+ ", " + Vsemestre + ", " + Vcurricular + ", " + Vcreditos + ", " + Vhteoricas + ", " +
Vhpracticas + ", " + Vasignacion + ", " + Vclasificacion + ", " + Vplan + ", " + "'N' )"
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)        'Ahora ejecuto el proceso
      rdotabla1.Close                                  'Cierro el Resulset
      Set rdotabla1 = Nothing                          'Limpio registros de la tabla1
      PROCESO = "INICIO"                               'Vuelvo todo a la normalidad
      Inicializa                                       'Inicializo la pantalla
      LlenaCuadro                                      'Actualizo el cuadro
    Case "CAMBIAR"
      'Validación de cada una de las variables
      If tclvunicamateria.Text = "" Then
      ……..
      End If
      '##########################################
      '#CAMBIO LA INFORMACIÓN DEL REGISTRO#
      '##########################################
      'Verifico si no son nulos y Guardo la información en cadenas
      'Clave única de la materia
      Vclvunicamateria = "'" + tclvunicamateria.Text + "'"
      ……..
      'El cambo Eliminar, se queda sin alteraciones
      'Guardo todo en una cadena SQL
      CADENASQL = "UPDATE MATERIAS SET " + "MATERIA=" + Vmateria + ", " +
"MATERIA_ABREV=" + Vabrevmateria + ", " + "CLAVE=" + Vclvmateria + ", " +  "SEMESTRE=" +
Vsemestre + ", " + "CURRICULAR=" + Vcurricular + ", " +  "CREDITOS=" + Vcreditos + ", " +
"HORASTEO=" + Vhteoricas + ", " +  "HORASPRA=" + Vhpracticas + ", " +  "TIPO_ASIGNACION=" +
Vasignacion + ", " +  "CLASIFICACION=" + Vclasificacion + ", " +  "CLVPLAN=" + Vplan +  " WHERE
CLVUNICAMAT=" + Vclvunicamateria
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)        'Ahora ejecuto el proceso
      rdotabla1.Close
      Set rdotabla1 = Nothing
      PROCESO = "INICIO"                               'Vuelvo todo a la normalidad
      Inicializa                                       'Inicializo la pantalla
      LlenaCuadro                                      'Actualizo el cuadro
    Case "BORRAR"
      If MsgBox("¿Desea eliminar el registro?", vbYesNo + vbQuestion, "Eliminar") = vbYes Then
```

```
        '##############################################
        '#BORRO EL REGISTRO MOSTRADO EN PANTALLA#
        '##############################################
        'Verifico si no es nula la clave principal y Guardo la información en cadenas
        Vclvunicamateria = tclvunicamateria.Text
        'Guardo todo en una cadena SQL
        CADENASQL = "UPDATE MATERIAS SET " + "ELIMINAR= 'S'" +  " WHERE
CLVUNICAMAT=" + Vclvunicamateria        'Para eliminara el registro se cambia la letra por 'S'
        Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)        'Ejecuto el proceso
        rdotabla1.Close                 'Cierro el Resultset
        Set rdotabla1 = Nothing         'Limpio registros de la tabla1
        PROCESO = "INICIO"              'Vuelvo todo a la normalidad
        Inicializa                      'Inicializo la pantalla
        LlenaCuadro                     'Actualizo el cuadro
      End If
    Case Else
      Inicializa                        'Inicializo la pantalla
    End Select
  Exit Sub                              'Termino
'Proceso para mostrar errores
End Sub
```

Al revisar el código, se puede definir que para trabajar con la base de datos sólo se están utilizando instrucciones SQL, tal como se definió en el capítulo 1, "Select" para realizar una consulta de datos, "Insert" para agregar un registro, "Update" para cambiar o para borrar un registro.

"Delete" no se usó para borrar, simplemente para que el usuario no borre datos accidentalmente, ya que como es una base de datos relacional se perderían millones de datos, para ello se tiene un campo llamado "ELIMINAR" donde se asigna un valor de "N" y si se cambia ese valor, esta acción solo la puede realizar el administrador de la base de datos.

En lo que respecta al botón 3 es menos complicado, ya que simplemente es la función de cancelación de cada proceso, volviendo al estado original de consulta por medio del módulo inicializa, por lo que el código es el mismo en los tres casos, cambiando sólo el nombre del proceso, como lo muestra el siguiente ejemplo:

```
    Case "NUEVO"
      PROCESO = "INICIO"            'Vuelvo todo a la normalidad
      Inicializa                    'Inicializo la pantalla
```

3.4.1.8. Módulo de intercambio de la cuadrícula con los demás objetos

Es el último módulo que se requiere para que funcione la captura de materias, ya que para realizar un cambio o eliminar un registro, primero se tiene que seleccionar y que el usuario lo vea en pantalla, por lo que hay que validar los datos o usar ciertos métodos para que el objeto tenga la información correcta a la hora de intercambiar datos, tal como lo muestra el siguiente código:

```
Private Sub Cmaterias_Click()
```

```
'Necesito dos variables para obtener los renglones y las columnas
Dim RENGLON, COLUMNA, CADENA As Variant
Dim i As Integer
If Cmaterias.Row = 0 Then                    'Si es el primer renglón
   Exit Sub                                  'Termino el proceso
End If
RENGLON = Cmaterias.Row                      'Me ubico en el registro donde se dio un clic
COLUMNA = Cmaterias.Col
Cmaterias.Row = RENGLON
Cmaterias.Col = 0
tclvunicamateria.Text = Cmaterias.Text       'Ahora cargo los datos con la clave única de materia
Cmaterias.Row = RENGLON
Cmaterias.Col = 1
Tmateria.Text = Cmaterias.Text               'Ahora cargo los datos en el nombre de la materia
Cmaterias.Row = RENGLON
Cmaterias.Col = 2
Tclvmateria.Text = Cmaterias.Text            'Pongo los datos completos de la materia
Cmaterias.Row = RENGLON
Cmaterias.Col = 3
Tabrevmateria.Text = Cmaterias.Text          'Abreviatura de la materia
Cmaterias.Row = RENGLON
Cmaterias.Col = 4
Basura = Cmaterias.Text    'Semestre
For i = 0 To ComboSemestre.ListCount         'Busco el semestre que está dentro del cuadro
   CADENA = ComboSemestre.List(i)    ¡       'Obtengo el texto del combo
   If (Trim(CADENA) = Trim(Basura)) Then
      ComboSemestre.Text = ComboSemestre.List(i)        'Encontré el valor
      Exit For                               'Salgo del ciclo de for
   Else
      ComboSemestre.ListIndex = 0            'Vacío el ComboBox
   End If
Next i
Cmaterias.Row = RENGLON
Cmaterias.Col = 5
If Cmaterias.Text = "S" Then                 'La Materia es Curricular "S" o "N"
   ONO.Value = False
   OSI.Value = True
Else
   OSI.Value = False
   ONO.Value = True
End If
Cmaterias.Row = RENGLON
Cmaterias.Col = 6
Tcreditos.Text = Cmaterias.Text              'Créditos
Cmaterias.Row = RENGLON
Cmaterias.Col = 7
Thteoricas.Text = Cmaterias.Text             'Horas Teóricas
Cmaterias.Row = RENGLON
Cmaterias.Col = 8
thpracticas.Text = Cmaterias.Text            'Horas Prácticas
Cmaterias.Row = RENGLON
Cmaterias.Col = 9                            'Asignación de materias
Basura = Cmaterias.Text                      'Guardo la información en una variable
For i = 0 To ComboAsignacion.ListCount       'Busco el grado que está dentro del cuadro
   CADENA = ComboAsignacion.List(i)          'Obtengo el texto del combo
   If (Trim(CADENA) = Trim(Basura)) Then
```

```
        ComboAsignacion.Text = ComboAsignacion.List(i)     'Encontré el valor
        Exit For                                           'Salgo del ciclo de for
      Else
        ComboAsignacion.ListIndex = 0                      'Vacío el ComboBox
      End If
    Next i
    Cmaterias.Row = RENGLON
    Cmaterias.Col = 10                                     'Clasificación de materias
    Basura = Cmaterias.Text                                'Guardo la información en una variable
    For i = 0 To ComboClasif.ListCount                     'Busco el grado que está dentro del cuadro
      CADENA = ComboClasif.List(i)                         'Obtengo el texto del combo
      If (Trim(CADENA) = Trim(Basura)) Then
        ComboClasif.Text = ComboClasif.List(i)             'Encontré el valor
        Exit For          'Salgo del ciclo de for
      Else
        ComboClasif.ListIndex = 0                          'Vacío el ComboBox
      End If
    Next i
    Cmaterias.Row = RENGLON
    Cmaterias.Col = 11
    Tplan.Text = Cmaterias.Text                            'Plan de estudios
    BOTON3.Enabled = True                                  'Activo los botones de eliminar y modificar
    BOTON2.Enabled = True
End Sub
```

3.4.2. Profesores

A diferencia de Cronos cada parte tiene un código distinto para realizar una captura, por lo que el diseño del programa se realizó para que fuera lo más parecido, ahora se van a analizar las diferencias para no repetir todos los procesos. La siguiente etapa la compone la captura de profesores de asignaturas, primero se verá la estructura de la base de profesores:

PROFESORES	TIPO	LONG	LLAVE	NOMBRE
CLVUNICAPROF	INT	4	*	Clave única del profesor
PROFESOR	VARCHAR	45		Profesor empezando por el apellido
RFC	VARCHAR	15		Registro federal de causantes
CURP	VARCHAR	18		Clave única del registro personal
NTRABAJADOR	VARCHAR	8		Número de trabajador de la UNAM
GRADO	VARCHAR	14		Grado Académico (1-Postdoctorado, 2-doctorado,
				3-maestría, 4-diplomado, 5-licenciatura,
				6-técnico y 7-técnico)
NACIONALIDAD	VARCHAR	50		Nacionalidad
SEXO	VARCHAR	1		Sexo (M-Masculino, F-Femenino y S-Sin sexo)
MOVIMIENTO	VARCHAR	1		Tipo de movim. (A-alta, B-baja y C-cambio)
FUNCIONARIO	VARCHAR	1		Tiene cargo de Funcionario (S-si, N-no)
ELIMINAR	VARCHAR	1		N' NO ELIMINADO , otro caracter ELIMINADO

A diferencia de la tabla de profesores de Cronos se tiene una clave única de profesores, que sustituye al campo de RFC como llave única, ya que al estar como administrador del

programa se puede repetir el campo en sus 13 dígitos o por errores al momento de teclear el registro, lo cual hace más difícil su modificación.

Se puede pensar que se usarán los campos de CURP o el número de trabajador como un campo llave, pero el proceso para obtener estos campos ya sea por Gobernación o por el Departamento de Personal, es un procedimiento que lleva mucho tiempo, por ello se decidió tener como el campo llave la "Clave Única de Profesor".

El campo de nombre es muy importante, pero se tienen muchos casos especiales en cuanto a su estructura, por lo que no se dividió como otros programas que lo separan en tres campos, los cuales son apellido paterno, apellido materno y nombres.

El campo de Funcionario se retomó de Cronos, ya que es una bandera para distinguirlos de los demás profesores. En caso de los demás campos son solicitados por Servicios Escolares para inscribirlos en DGAE, donde eran tomados del programa de NEW.

Ya con la tabla, se tiene que pensar en el diagrama por módulos donde se demuestra la semejanza con el diagrama de materias, por lo que el diagrama para los profesores es el siguiente:

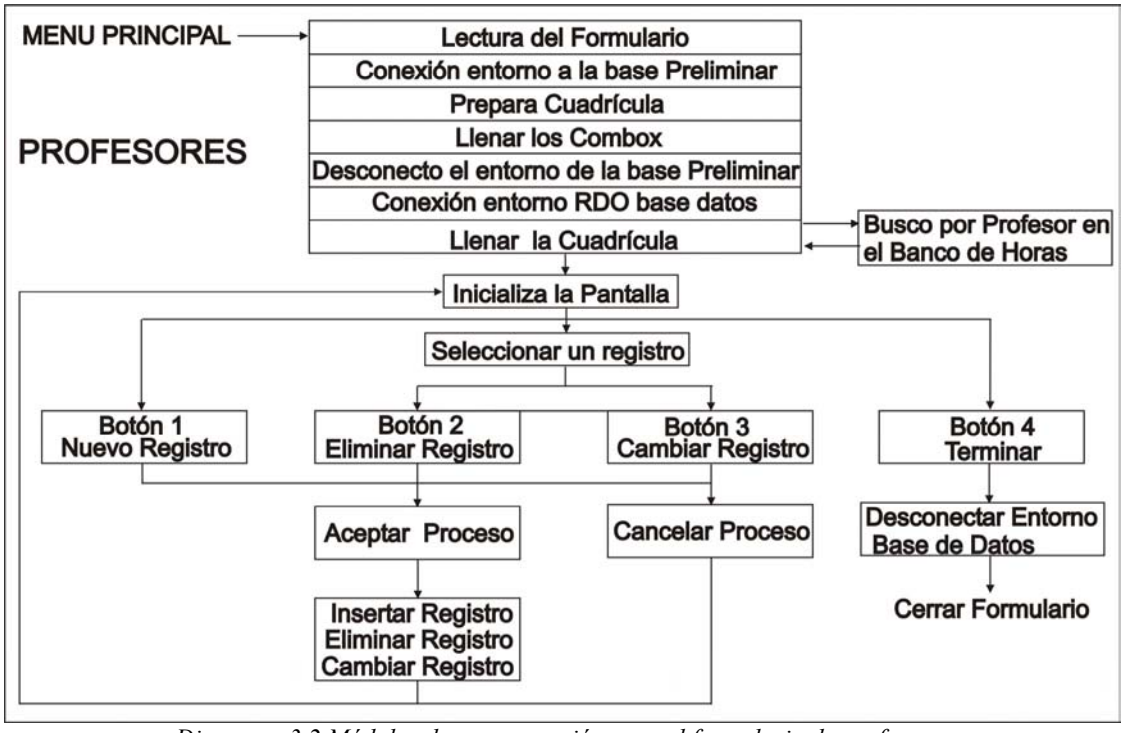


Diagrama 3.2 Módulos de programación para el formulario de profesores

En base a todos los campos se crea un formulario muy semejante al que se crea en materias, agrupado en varios grupos, como son los objetos de captura, los botones y el cuadro de consulta, como lo muestra la figura 3.13.

Figura 3.13 Formulario de profesores

En lo que respecta al código del formulario, el modelo es el mismo que se tiene en materias, donde los cambios se hacen con base en los campos, en apoyo de la figura 3.13 se puede distinguir cuáles usan cuadros de texto, ComboBox o Círculos de opción. De acuerdo con el tamaño de cada campo, se asigna un espacio en el cuadro de consulta.

Por cada ComboBox se tiene un proceso de llenado, por lo que se tiene un módulo para llenar Grado, Nacionalidad y Sexo, en cuanto a los siguientes módulos se realiza un proceso parecido, pero se tienen una instrucción que no se utilizó en materias en el módulo 'LlenaCuadro':

```
'Busco el profesor para ver si tiene horas en el banco de horas
temp = Buscabancoprofesor(CADENA, Cprofesores.Row)
```

Donde se llama al módulo: 'Buscabandoprofesor'; para saber si un profesor no tiene asignadas horas, se le cambia el color a rojo, en caso contrario se muestra con el color negro. El código para buscar un profesor en el banco de horas es el siguiente:

```
Function Buscabancoprofesor(Vclvunicaprof As String, RENGLON As Integer)
   Dim CADENASQL As Variant                    'Variables Locales
   On Error GoTo MENSAJE                       'Activo el control de errores
   'Escribo la cadena SQL donde voy a ver si hay registros de profesores en el banco de horas
   CADENASQL = "SELECT NREGBANCO, CLVUNICAPROF, ELIMINAR FROM BANCOHORAS
WHERE ELIMINAR = 'N' AND CLVUNICAPROF = " + Vclvunicaprof
   Set rdotabla2 = rdoconexion.OpenResultset(CADENASQL, rdOpenKeyset, rdConcurReadOnly,
rdAsyncEnable + rdExecDirect)               'Ejecuto la instrucción SQL
   If rdotabla2.EOF Then                       'Si no tiene ningun registro en banco de horas
      'La clave y el nombre de profesor se pone en rojo
      Cprofesores.Row = RENGLON                'Clave única del profesor
      Cprofesores.Col = 0
      Cprofesores.CellForeColor = vbRed        'Cambio el color del tipo de letra
      Cprofesores.Row = RENGLON                'Nombre del Profesor
      Cprofesores.Col = 1
```

```
    Cprofesores.CellForeColor = vbRed          'Cambio el color del tipo de letra
  Else
    'La clave y el nombre de profesor se pone en negro
    Cprofesores.Row = RENGLON                  'Clave única del profesor
    Cprofesores.Col = 0
    Cprofesores.CellForeColor = vbBlack        'Cambio el color del tipo de letra
    Cprofesores.Row = RENGLON                  'Nombre del profesor
    Cprofesores.Col = 1
    Cprofesores.CellForeColor = vbBlack        'Cambio el color del tipo de letra
  End If
  rdotabla2.Close                              'Cierro el Resultset
  Set rdotabla2 = Nothing                      'Limpio registros de la tabla1
'Proceso para mostrar los mensajes de errores
End Function
```

Con esto, el módulo de la cuadrícula valida el color, en caso de seleccionar un profesor en negro se desactiva el botón de borrar, en caso contrario se activa, siguiendo lo que se hizo en Cronos donde se puede eliminar el profesor si y sólo si no tiene registros de banco de horas, con el siguiente código:

```
  If Cprofesores.CellForeColor = vbRed Then
    BOTON2.Enabled = True                      'Habilito el botón de eliminar
  Else
    BOTON2.Enabled = False                     'Deshabilito el botón de eliminar
  End If
```

3.4.3. Banco de Horas

Al tener la captura de materias y profesores, junto con otras bases ya se puede trabajar con el banco de horas, que es la parte fundamental del programa, como se mostró al realizar el análisis de "Cronos", donde se usarán las estructuras anteriores, pero considerando que las carreras o áreas tienen distintas necesidades.

Por ello, se requirió hacer un prototipo de la base de datos, a la que le realizaron las modificaciones necesarias conforme se iba programando, para que fuera adaptando a los reportes y a los procesos principales, se incorporaron banderas o al anexar el campo de grupo que se tenía en los horarios, esto fue una decisión estratégica para validar y tener una base de tipo relacional.

La base del banco de horas queda de la siguiente forma:

BANCO DE HORAS	TIPO	LONG	LLAVE	NOMBRE
NREGBANCO	INT	4	*	Número de registro del banco
CLVUNICAPROF	INT	4		Clave única del profesor
CLVUNICAMAT	INT	4		Clave única de la materia
CLVUNICACAT	INT	4		Clave única de la categoría
RUBRO	INT	4		Rubro de la categoría(manejo de impresión)
TIPO	VARCHAR	1		D-DEFINITIVO I-INTERINO
CURRICULAR	VARCHAR	1		Indicación si es curricular (0- NO, 1-SI)

HANTESTEO	FLOAT	8		Horas anteriores teóricas
HANTESPRA	FLOAT	8		Horas anteriores prácticas
HANTESTOT	FLOAT	8		Horas anteriores totales
HACTTEO	FLOAT	8		Horas actuales teóricas
HACTPRA	FLOAT	8		Horas actuales prácticas
HACTTOT	FLOAT	8		Horas actuales totales
GRUPO_ACT	VARCHAR	6		Grupo actual (Opcional)
GRUPO_ANT	VARCHAR	6		Grupo anterior (Opcional)
ELIMINAR	VARCHAR	1		N' NO ELIMINADO , otro caracter ELIMINADO
ASIGHORARIO	VARCHAR	1		S' Sin horario, H' con horario

Con apoyo de la tabla de banco de horas, los diagrama por módulos de programación anteriores, y considerar la interfaz del usuario – programador de Cronos, se modifico la estructura del diagrama, para poder a cada uno de los profesores individualmente con sus horas correspondientes:

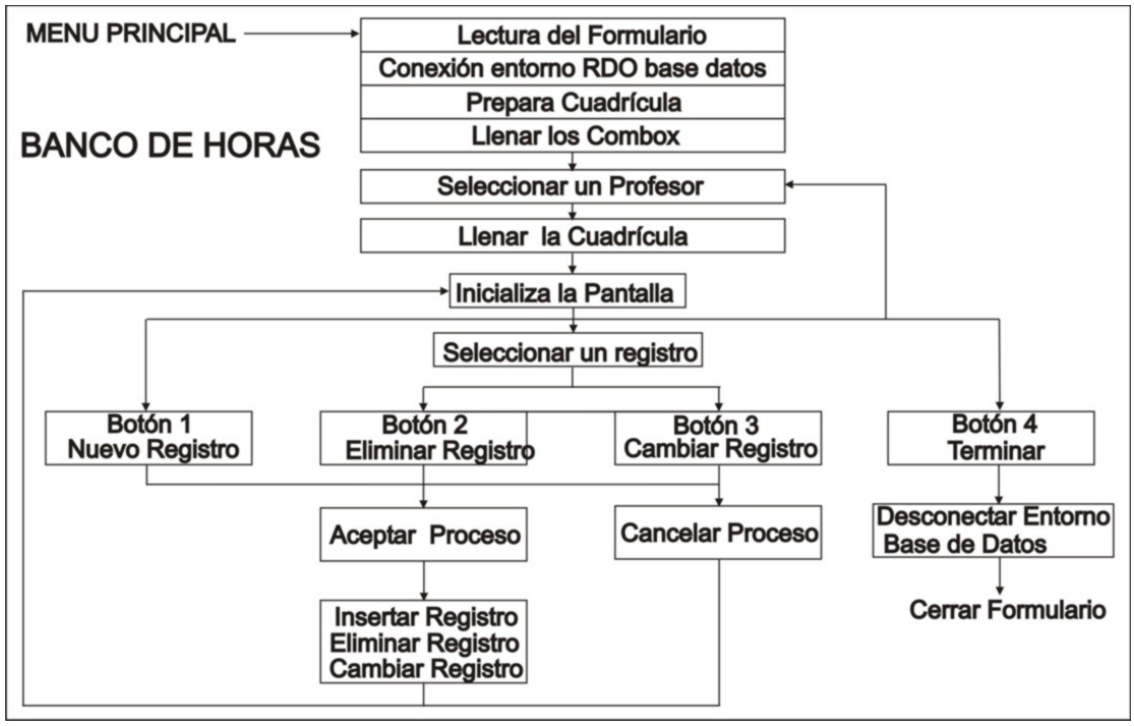


Diagrama 3.3 Módulos de programación para el formulario de Banco de Horas

Es en este módulo donde se va a integrar lo que son relaciones, el usuario no se va a dar cuenta de que existen, por lo que a diferencia de los otros formularios se tienen que manejar las relaciones de forma oculta con sólo usar la clave única, tal como lo muestra la tabla anterior, así como el formulario que se diseñó en la figura 3.14.

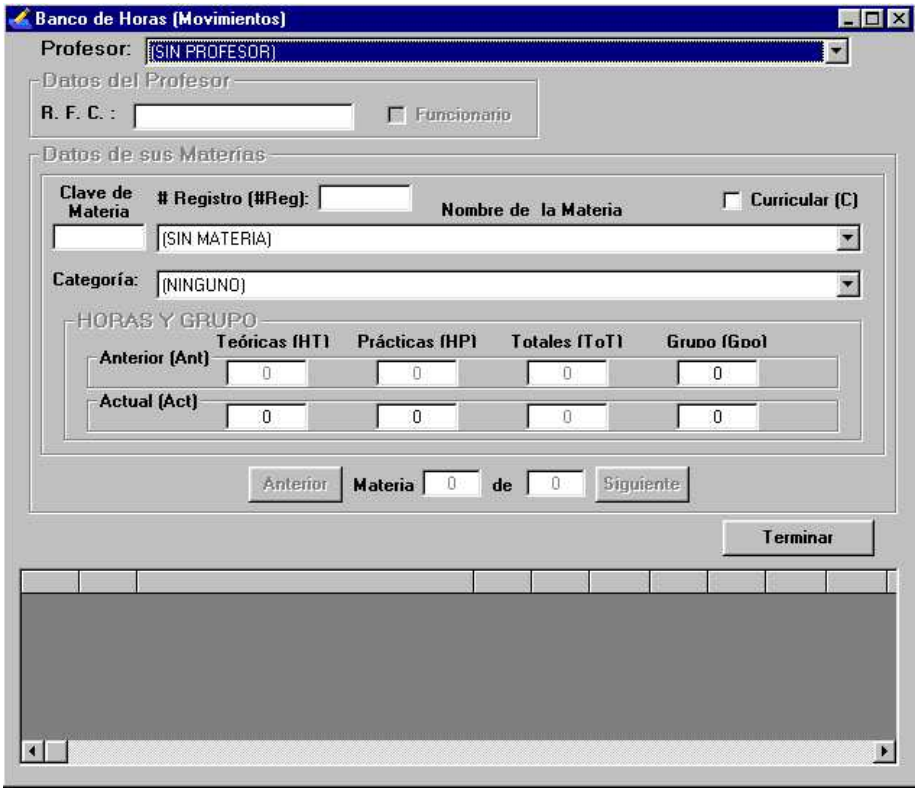


Figura 3.14 Formulario de Banco Horas

Aunque se repite el módulo de inicio del formulario existe un cambio, ya que ahora no se va a utilizar información de una base general, sino todo se va a usar de la base utilizada, tal como se muestra en parte del código de inicialización del formulario:

```
Call ModulosGlobales.conectarBD(nombre, acceso, base)
Prepara_Cuadro                    'Preparo el cuadro
'Lleno los combos
LlenaProfesores                   'Lleno el combo de profesores
LlenaMaterias                     'Lleno el combo de materias
LlenaCategorias                   'Lleno el combo de categorías
InicializaCompleta                'Inicializo la pantalla
```

Los módulos llenados de los combos tienen cambios en su estructura, el primero que es llamado de la base que se utiliza en el semestre y por la carrera, en vez de llamar la base general. Otro es que no sólo se llena el combo sino que se requiere más información de la que se agrega, por lo que hay que tener más información almacenada en la memoria; en primera instancia se guardaron en cuadrículas y después se consideró generar el "array"[24], donde se almacenan los datos.

Al estar programando se encontraron muchas más ventajas a la cuadrícula, que en al array, ya que se podían visualizar los datos y al mismo tiempo se podían corregir errores para el funcionamiento del programa, donde no se pierde funcionalidad, y puede ser un indicativo

[24]Array esta es una instrucción de Visual Basic para almacenar datos en memoria.

de si la memoria física está en buen estado. En cuanto a la programación se utiliza el módulo de “Llenacuadro”, para no perder el modelo de programación.

Donde sí se tiene un cambio es en el módulo de “Inicializa”, ya que se tiene que manejar una misma interfaz que Cronos, y no se ven todos los registros de golpe, sino conforme al profesor que se va seleccionar, por ello se crearon dos módulos de inicialización, uno completo y uno parcial.

El módulo completo se llama “InicializaCompleta”, que es el mismo procedimiento completo que se usa en el módulo de “Inicializa”, para ser usado al cambiar de profesor; pero el módulo parcial, el cual se nombró “InicializaBanco” es un procedimiento donde no toca los objetos que se relacionan con el profesor (clave única del profesor, nombre del profesor, RFC, funcionario), el cual se va a usar para los movimientos de la base de datos (altas, bajas, cambios o eliminación).

Por último, el módulo de “Llenacuadro”, no se encuentra al leer el formulario, esto es porque se tienen que considerar todas las fallas dentro de la base, incluyendo que no se tenga registro de ningún profesor, materia o categoría; también considerando lo que se mencionó al momento de inicializar.

Antes de llamar a este módulo, se tiene que seleccionar el profesor dando un clic al combo, donde se registra el profesor en pantalla con base en el siguiente código:

```
Private Sub Comboprofesor_Click()
   'Utilizo este proceso cuando los cuadros han sido cargados
   Select Case PROCESO
     Case Is = "INICIO"
       Dim CADENA, cuenta As Variant
       CADENA = ""                                  'Inicializo la cadena
       'Si selecciono el profesor por default
       If Comboprofesor.Text = "(SIN PROFESOR)" Then
          InicializaCompleta                        '*Inicializo la pantalla completa*
          Exit Sub                                  'Termino
       End If
       '#Cargo los datos del profesor#
       CADENA = Trim(Comboprofesor.Text)            'Guardo el nombre del profesor en una cadena
       For cuenta = 1 To (Cprofesores.Rows - 1)     'Busco nombre correspondiente en el cuadro
          Cprofesores.Row = cuenta                  'Realizo el ciclo en la cadena de for
          Cprofesores.Col = 1
          If CADENA = Trim(Cprofesores.Text) Then   'Comparo las cadenas
             Cprofesores.Col = 0
             Tclvprofesor.Text = Cprofesores.Text   'Pongo la clave del profesor
             Cprofesores.Col = 2
             Trfc.Text = Cprofesores.Text           'Pongo el RFC
             Cprofesores.Col = 3
             If Cprofesores.Text = "S" Then         'Activo si es Funcionario
                cfuncionario.Value = 1
             Else
                cfuncionario.Value = 0
             End If
             FRAMEprofesor.Enabled = True           'Activo el Frame
```

```
        Exit For                                    'Termino el ciclo for y continúo
      End If
    Next cuenta
    If Tclvprofesor.Text = "" Then Exit Sub         'Si no cargo la clave de profesor termino
    InicializaBanco                                 '*Inicializo la pantalla del banco de horas*
  End Select
End Sub
```

Al llamar "InicializaBanco"se llena la información en la cuadrícula pero con un ajuste, ya que se tiene que llamar a un solo profesor y no todos los registros, por lo que se usa la clave de profesor como apoyo, como se indica a continuación:

```
  LlenaCuadro (Tclvprofesor.Text)
```

Ahora en "Llena Cuadro" también se tiene que limitar a no llamar todos los registros, sino que se delimita por la instrucción SQL, que es la siguiente:

```
  CADENASQL = "SELECT NREGBANCO, CLAVE, MATERIA_ABREV, HANTESTEO, HANTESPRA,
HANTESTOT, HACTTEO, HACTPRA, HACTTOT, GRUPO_ANT, GRUPO_ACT, CLVUNICACAT,
CURRICULAR, CLVUNICAPROF, CLVUNICAMAT, RUBRO, TIPO, ASIGHORARIO, ELIMINAR,
CLVPLAN FROM VBANCO WHERE ELIMINAR = 'N' AND CLVUNICAPROF=" & Vclvunicaprof & "
ORDER BY MATERIA_ABREV"
```

Aquí no se llama a la tabla Banco de Horas, sino a una Vista llamada VBanco, que no es más que una tabla virtual generada por una instrucción SQL en el servidor, que se explicará en el siguiente capítulo.

La tabla virtual hace que no se modifique el formato de programación, por lo que el módulo siguiente es muy parecido a los que se tiene en materias y profesores.

El siguiente paso es la validación de objetos, pero hay que resaltar el código del combo de materias, su funcionamiento es el siguiente: al seleccionar la materia se tienen que llenar los campos que se relacionan con materia, los que no se activan son asignados directamente, pero hay opcionales como lo son la asignación de horas actuales, donde el usuario puede modificar la cantidad.

```
Private Sub Combomateria_Click()
  'Utilizo este proceso cuando los cuadros han sido cargados
  Dim CADENA, cuenta As Variant
  Select Case PROCESO
   Case Is = "NUEVO"
    CADENA = ""                                 'Inicializo la cadena
    If Combomateria.Text = "(SIN MATERIA)" Then         'Si selecciono la materia por default
      InicializaMateria                         'Inicializo la pantalla completa
      Exit Sub                                  'Termino
    End If
    '*Cargo los datos de la materia*
    CADENA = Trim(Combomateria.Text)            'Guardo el nombre de la materia en una cadena
    For cuenta = 1 To (Cmaterias.Rows - 1)      'Busco nombre correspondiente en el cuadro
      Cmaterias.Row = cuenta                    'Realizo el ciclo en la cadena de for
      Cmaterias.Col = 2
```

```
        If CADENA = Trim(Cmaterias.Text) Then     'Comparo las cadenas
           Cmaterias.Col = 0                      'Pongo la clave única de la materia
           Tclvunicamateria.Text = Cmaterias.Text
           Cmaterias.Col = 1                      'Pongo la clave de la materia
           tclvmateria.Text = Cmaterias.Text
           Cmaterias.Col = 3                      'Activo si es Curricular
           If Cmaterias.Text = "S" Then
              ccurricular.Value = 1
           Else
              ccurricular.Value = 0
           End If
           Cmaterias.Col = 4                      'Pongo las horas teóricas
           Thorateoricaactual = Cmaterias.Text
           Cmaterias.Col = 5                      'Pongo las horas prácticas
           Thorapracticaactual = Cmaterias.Text
           Cmaterias.Col = 6                      'Pongo el plan de estudios
           Tplan.Text = Cmaterias.Text
           'Combocategoria.SetFocus               'Me ubico en el combo de categorías
           'Si todos los datos están listo activo aceptar
           If Combomateria.ListIndex > 0 And Combocategoria.ListIndex > 0 And Tgrupoactual.Text <> ""
Then
              Boton2.Enabled = True              'Activo el botón de aceptar
           End If
           Exit For                              'Termino el ciclo for y continúo
        End If
     Next cuenta
     If tclvmateria.Text = "" Then Exit Sub     'Si no cargo la clave de materia termino
   Case Is = "CAMBIAR"
     'Repito el proceso tal como se escribió durante el proceso de Nuevo para evitar que se realice en otro
momento que no sea para agregar o cambiar un registro
  End Select
End Sub
```

También al seleccionar el combo de categorías se realiza un proceso similar, donde se cargan los siguientes rubros:

a. La clave única de la categoría
b. El rubro de la categoría
c. El tipo de categoría

En lo que respecta a los controles se da la validación de los datos ya sea cuando se agrega o cambia un registro, tal como se describe en materias, pero se tiene un nuevo agregado, ya que se debe validar que no se repita el mismo registro, con excepción de ciertas carreras, por lo que sólo se mostrará esta parte del código:

```
'*ANTES DE GUARDAR VERIFICO SI SE REPITE CON UN REGISTRO*
'Llamo a la base de datos
Set rdotabla1 = rdoconexion.OpenResultset("SELECT CLVUNICAPROF, CLVUNICAMAT, GRUPO_ACT,
ELIMINAR FROM BANCOHORAS WHERE ELIMINAR = 'N'")
With rdotabla1                                  'Trabajo con rdotabla1
    Dim temp1, temp2, temp3 As Variant
    If Not .EOF Then                            'Si no es fin de archivo continúo
```

```
        Do While Not .EOF          'Realizo un ciclo para ver que no exista un registro idéntico
           temp1 = Trim(Str(!CLVUNICAPROF))     'Guardo los registros importantes
           temp2 = Trim(Str(!CLVUNICAMAT))
           temp3 = !grupo_act
           'Comparo los registros importantes
           If Vclvunicaprof = temp1 And Vclvunicamat = temp2 And Tgrupoactual.Text = temp3 Then
             'Muestro un mensaje de error
             MsgBox "Registro Repetido, verificar profesor, materia y grupo", vbExclamation, "Error: No se
puede guardar"
              rdotabla1.Close                    'Cierro el Resutlset
              Set rdotabla1 = Nothing            'Limpio registros de la tabla1
               Exit Sub                          'Termino el módulo
           End If
           'Este error se marca para todas las carreras con excepción de la que manejan
           'más de un profesor en un salón, como arquitectura y diseño.
           If clave_carrera <> "ARQ" And clave_carrera <> "DIS" And Tgrupoactual.Text <> 0 Then
             'Comparo los registros de materia y grupo no sean idénticos
             If Vclvunicamat = temp2 And Tgrupoactual.Text = temp3 Then     'Muestro un mensaje de error
                MsgBox "Esta Materia tiene un grupo ya asignado, seleccionar otro", vbExclamation, "Error:
Materia y Grupo repetido"
                rdotabla1.Close                  'Cierro el Resutlset
                Set rdotabla1 = Nothing          'Limpio registros de la tabla1
                Exit Sub                         'Termino el módulo
             End If
          End If
          .MoveNext
          Loop
      End If
  End With
```

Para terminar este módulo se considera una interfase parecida a la de cronos para identificar cuántos registros se tienen y en qué registro se encuentra uno, dónde se agregó código al módulo de “LlenaCuadro”, lo cual fue muy sencillo y se llena con el siguiente código:

```
Ttotalmaterias.Text = Cbanco.Rows - 1          'Pongo el número total de materias
```

En cuanto al número de materias se determina por la posición del registro:

```
Tmaterianumero.Text = Cbanco.Row               'Ahora guardo el número de renglón
```

Para los botones de ‘Siguiente’ y ‘Anterior’, se retoma el código del llenado de la cuadrícula, sólo se agrega el siguiente código al finalizar:

Botón Anterior	Botón Siguiente
If RENGLON > 1 Then ***'Lo dejo habilitado*** *Banterior.Enabled = True* *Else* ***'Desactivo el botón*** *Banterior.Enabled = False* *End If*	*If RENGLON < (Cbanco.Rows - 1) Then* ***'Lo dejo habilitado*** *Bsiguiente.Enabled = True* *Else* ***'Desactivo el botón*** *Bsiguiente.Enabled = False* *End If*

3.4.4. Horarios.

Un problema del programa anterior, es que los horarios se desligan totalmente del banco de horas, por lo que en el programa se tiene una relación para no tener que capturar dos veces, y que la base sea compatible con la de Servicios Escolares.

La estructura de la tabla de horarios es la siguiente:

HORARIOS	TIPO	LONG	LLAVE	NULO	NOMBRE
NREGHORARIO	INT	4	*	NO	Número de Registro del Horario
NREGBANCO	INT	4		NO	Número de registro del banco
CLVSALON	VARCHAR	6	*	NO	Clave única de los salones
CUPO	INT	4		NO	CUPO DEL SALÓN
POSICION	VARCHAR	1		NO	"A", "B", "C", "D" profesor según prioridad
HORLUN1	DATETIME	8		SI	Hora de inicio del lunes
HORMAR1	DATETIME	8		SI	Hora de término del lunes
HORMIE1	DATETIME	8		SI	Hora de inicio del martes
HORJUE1	DATETIME	8		SI	Hora de término del martes
HORVIE1	DATETIME	8		SI	Hora de inicio del miércoles
HORSAB1	DATETIME	8		SI	Hora de término del miércoles
HORLUN2	DATETIME	8		SI	Hora de inicio del jueves
HORMAR2	DATETIME	8		SI	Hora de término del jueves
HORMIE2	DATETIME	8		SI	Hora de inicio del viernes
HORJUE2	DATETIME	8		SI	Hora de término del viernes
HORVIE2	DATETIME	8		SI	Hora de inicio del sábado
HORSAB2	DATETIME	8		SI	Hora de término del sábado
ELIMINAR	VARCHAR	1		NO	N' NO ELIMINADO , otro carácter ELIMINADO

Las diferencias que se ven en la tabla comparada con la de Cronos es que se toman los campos de profesor, materia y grupo de lo que es la referencia del campo de la clave del banco.

Otra diferencia se establece cuando se asigna el cupo, a pesar de que se tiene en otra tabla, una de las modificaciones es que el usuario puede asignar el número de personas que se aceptarán en la clase y el último campo es el de posición, que no es más que una simple bandera para aquellas carreras que tienen más de un profesor en un salón con el mismo horario[25].

El formulario que se diseñó para capturar los horarios, con base en la tabla de horarios es el que se muestra en la figura 3.15.

[25] Las carreras que tienen este lineamiento son Arquitectura y Diseño Industrial.

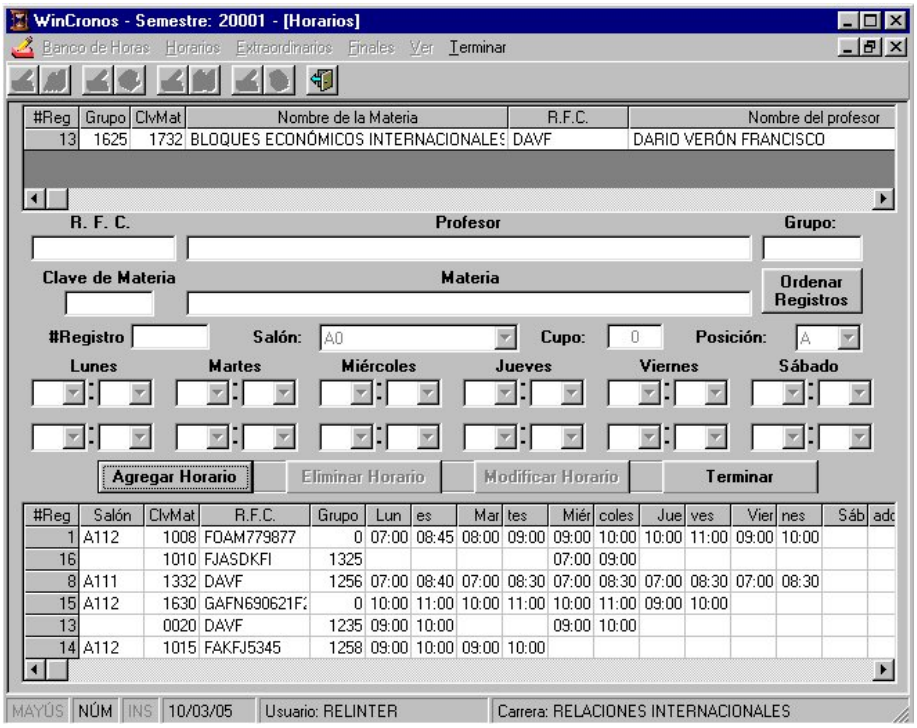


Figura 3.15 Formulario de Horarios

Aunque el formulario es distinto a los otros, no se modifican mucho los módulos, tal como se puede ver en siguiente diagrama:

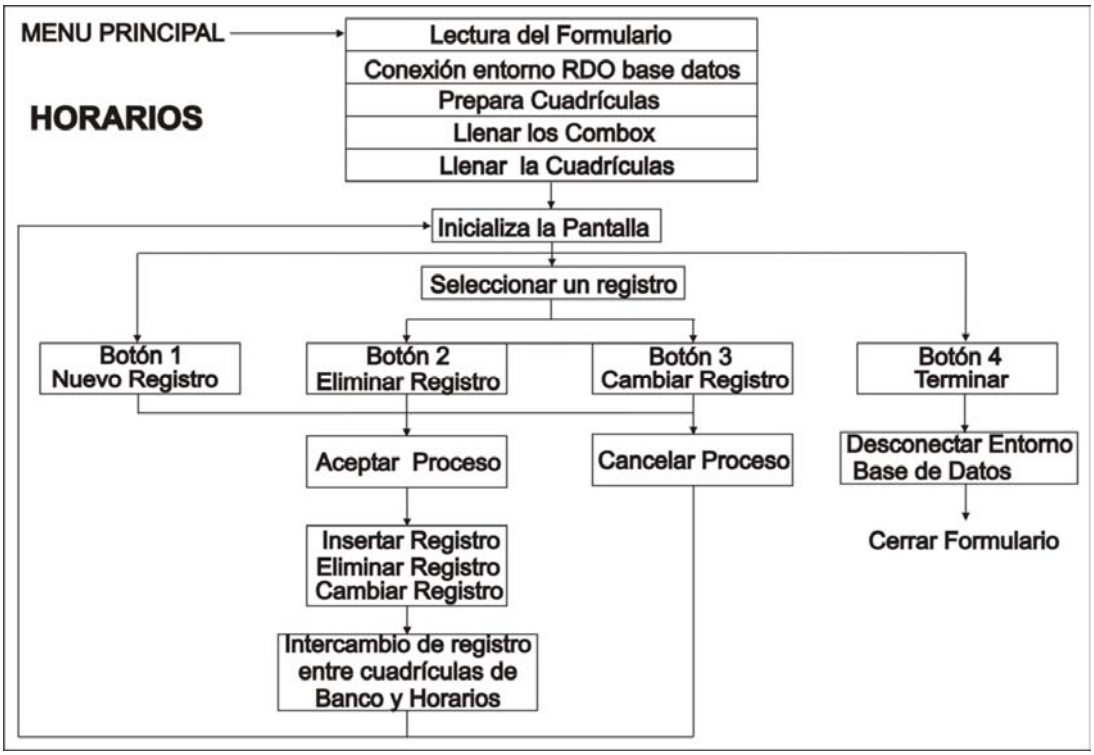


Diagrama 3.4 Módulos de programación para el formulario de horarios

Con base en el formulario se tiene algo que no se había manejado en los otros: el manejo de horarios. Por lo que para capturar, ya sea con teclado o el ratón, se analizaron distintos programas, y una opción era usar combos; donde al realizar una estadística lo ideal era establecer rangos de 5 en los minutos y manejar las horas de las 7 a las 23.

Por lo que se usa el siguiente código para el llenado de horas y minutos cuando se activa el formulario, pero como son seis días, que se dividen en hora de inicio y hora de término, se hizo un código único, por lo que se crea un índice en los combos para no repetir código, el cual queda así:

```
Private Sub Llena_combox()
  Dim i As Integer
  'Lleno los combobox #1 con las horas válidas
  For i = 0 To 11                          'Realizo un ciclo FOR
    Combo1.Item(i).AddItem "07"            'Agrego las horas de 7 a 23.
    Combo1.Item(i).AddItem "08"
    Combo1.Item(i).AddItem "09"
    Combo1.Item(i).AddItem "10"
    Combo1.Item(i).AddItem "11"
    Combo1.Item(i).AddItem "12"
    Combo1.Item(i).AddItem "13"
    Combo1.Item(i).AddItem "14"
    Combo1.Item(i).AddItem "15"
    Combo1.Item(i).AddItem "16"
    Combo1.Item(i).AddItem "17"
    Combo1.Item(i).AddItem "18"
    Combo1.Item(i).AddItem "19"
    Combo1.Item(i).AddItem "20"
    Combo1.Item(i).AddItem "21"
    Combo1.Item(i).AddItem "22"
    Combo1.Item(i).AddItem "23"
  Next i
  'Lleno los combobox #2 con los minutos válidos
  For i = 0 To 11                          'Realizo un ciclo FOR
    Combo2.Item(i).AddItem "00"            'Agrego los minutos en rangos de 5 minutos
    Combo2.Item(i).AddItem "05"
    Combo2.Item(i).AddItem "10"
    Combo2.Item(i).AddItem "15"
    Combo2.Item(i).AddItem "20"
    Combo2.Item(i).AddItem "25"
    Combo2.Item(i).AddItem "30"
    Combo2.Item(i).AddItem "35"
    Combo2.Item(i).AddItem "40"
    Combo2.Item(i).AddItem "45"
    Combo2.Item(i).AddItem "50"
    Combo2.Item(i).AddItem "55"
  Next i
End Sub
```

Este paso es sencillo, pero como el usuario puede teclear el horario lo complica, ya que la propiedad “Style” tiene que estar en “Dropdown Combo”, lo que hace que se tenga que validar las horas y los minutos, esto se hace con el evento “KeyPress”, como lo muestra el siguiente código:

```
Private Sub Combo1_KeyPress(Index As Integer, KeyAscii As Integer)
   Dim STRValido, valor As String                    'Establezco las variables a utilizar
   STRValido = "0123456789"                          'Cadena de caracteres válidos
   If KeyAscii = 13 Then                             'Al teclear "ENTER" cambio de posición
      If Len(Combo1.Item(Index).Text) = 0 Then       'Cambio de posición
         If Index < 11 Then                         'Si es menor al número del combo
            Combo1.Item(Index + 1).SetFocus         'Me voy al siguiente combo de horas
         Else
            If BOTON2.Enabled = True Then           'Si el botón 2 está activado
               BOTON2.SetFocus                      'Me ubico en el primer botón
            End If
         End If
      Else
         Combo2.Item(Index).SetFocus                'Me voy al siguiente combo de minutos
      End If
   Else
      If KeyAscii = 8 Then                          'Valido si es la barra de retroceso
         If Len(Combo1.Item(Index).Text) = 0 Then   'Si no tengo caracteres dentro del combo
            If Index > 0 And Index < 12 Then        'Si el índice es mayor a 0
               Combo2.Item(Index - 1).SetFocus      'Cambia de cuadro
            End If
         End If
         Exit Sub                                   'Salgo del programa
      End If
      If InStr(STRValido, Chr(KeyAscii)) = 0 Then   'Valido si es un número
         KeyAscii = 0                               'Anulo el caracter tecleado
         Exit Sub                                   'Salgo del módulo
      End If
      If Len(Combo1.Item(Index).Text) = 0 Then      'Si es el primer caracter realizo esta rutina
         'Si es 7,8,9 agrego un "0" y me voy al siguiente cuadro
         If Chr(KeyAscii) = "7" Or Chr(KeyAscii) = "8" Or Chr(KeyAscii) = "9" Then
            valor = Chr(KeyAscii)                   'Guardo el valor tecleado
            KeyAscii = 0                            'Anulo el caracter tecleado para inserta una cadena
            Combo1.Item(Index).Text = "0" + valor   'Agrego una nueva cadena con el valor tecleado
            Combo2.Item(Index).SetFocus             'Me voy a la siguiente posición
            Exit Sub                                'Salgo del módulo
         End If
         Exit Sub                                   'Salgo del módulo
      End If
      If Len(Combo1.Item(Index).Text) = 1 Then      'Rutina para aceptar sólo 2 caracteres
         valor = Chr(KeyAscii)                      'Guardo el valor tecleado
         KeyAscii = 0       'Anulo el valor tecleado para evitar un BUCLE que cambia la información
         Combo1.Item(Index).Text = Combo1.Item(Index).Text + valor  'Guardo la cadena en el combo
         valor = Combo1.Item(Index).Text            'Válido que se una entrada válida
         Select Case valor                          'Veo los valores mediante un "select"
            Case "07"
               Combo2.Item(Index).SetFocus          'Me voy a la siguiente posición
               Exit Sub                             'Salgo del módulo
          .........
            Case "22"
               Combo2.Item(Index).SetFocus          'Me voy a la siguiente posición
               Exit Sub                             'Salgo del módulo
            Case "23"
               Combo2.Item(Index).Text = "00"       'Este valor sólo acepta en minutos "00"
               If Index < 11 Then                   'Si es menor al número del combo
```

```
            Combo1.Item(Index + 1).SetFocus     'Me voy al siguiente combo de horas
          Else
            If BOTON2.Enabled = True Then       'Si el botón 2 está activado
              BOTON2.SetFocus                   'Me ubico en el primer botón
            End If
          End If
          Exit Sub                              'Salgo del módulo
        Case Else
          Basura = MsgBox("Hora no válida", vbCritical, Error)    'mensaje de error
          KeyAscii = 0  'Anulo el valor tecleado para evitar un BUCLE que cambia la información
          Combo1.Item(Index).Text = ""          'Vacío el combo
          Exit Sub                              'Salgo del módulo
      End Select
      Combo2.Item(Index).SetFocus               'Me voy a la siguiente posición
    Else
      'Muestro un mensaje de Error si son mas de 2 caracteres
      Basura = MsgBox("El cuadro de selección, sólo acepta 2 caracteres", vbCritical, Error)
      KeyAscii = 0        'Anulo el valor tecleado para evitar un BUCLE que cambia la información
    End If
  End If
End Sub
```

En cuanto a los minutos se realiza un proceso parecido, pero con modificaciones, por lo que se escribe a continuación:

```
Private Sub Combo2_KeyPress(Index As Integer, KeyAscii As Integer)
  Dim STRValido2, valor2 As String              'Establezco las variables a utilizar
  STRValido2 = "012345"                         'Cadena de caracteres válidos
  If KeyAscii = 13 Then                         'Al teclear "ENTER" cambio de posición
    If Index < 11 Then                          'Si es menor al número del combo
      Combo1.Item(Index + 1).SetFocus           'Me voy al siguiente combo de horas
    Else
      If BOTON2.Enabled = True Then             'Si el botón está activado
        BOTON2.SetFocus                         'Me ubico en el primer botón
      Else
        Exit Sub                                'Termino
      End If
    End If
  Else
    If KeyAscii = 8 Then                        'Valido si es la barra de retroceso
      If Len(Combo2.Item(Index).Text) = 0 Then  'Si no tengo caracteres dentro del combo
        If Index >= 0 And Index < 12 Then       'Si el índice es mayor a 0
          Combo1.Item(Index).SetFocus           'Cambia de cuadro
        End If
      End If
      Exit Sub          'Salgo del programa
    End If
    If InStr(STRValido2, Chr(KeyAscii)) = 0 Then  'Valido  si es un número
      KeyAscii = 0          'Anulo el caracter tecleado
      Exit Sub          'Salgo del módulo
    End If
    valor2 = Combo2.Item(Index).Text
    If Combo2.Item(Index).Text = "" Then        'Rutina para aceptar un caracter
      Exit Sub                                  'Salgo del módulo
    End If
    If Len(Combo2.Item(Index).Text) = 1 Then    'Rutina para aceptar sólo 2 caracteres y no más
```

```
        valor2 = Chr(KeyAscii)                          'Guardo el valor tecleado
        KeyAscii = 0        'Anulo el valor tecleado para evitar un BUCLE que cambia la información
        Combo2.Item(Index).Text = Combo2.Item(Index).Text + valor2   'Guardo la cadena en el combo
        valor2 = Combo2.Item(Index).Text                'Valido que se una entrada válida
        Select Case valor2                              'Veo los valores mediante un select
          Case "00"
            If Index < 11 Then                          'Si es menor al número del combo
              Combo1.Item(Index + 1).SetFocus           'Me voy al siguiente combo de horas
            Else
              If BOTON2.Enabled = True Then             'Si el botón está activado
                BOTON2.SetFocus                         'Me ubico en el primer botón
              Else
                Exit Sub                                'Termino
              End If
            End If
            Exit Sub                                    'Salgo del módulo
         …….                                            'Repito el código en rangos de 5 minutos
          Case "55"
            If Index < 11 Then                          'Si es menor al número del combo
              Combo1.Item(Index + 1).SetFocus           'Me voy al siguiente combo de horas
            Else
              If BOTON2.Enabled = True Then
                BOTON2.SetFocus                         'Me ubico en el primer botón
              Else
                Exit Sub                                'Termino
              End If
            End If
            Exit Sub                                    'Salgo del módulo
          Case Else
            Basura = MsgBox("Minuto no válido", vbCritical, Error)          'mensaje de error
            KeyAscii = 0  'Anulo el valor tecleado para evitar un BUCLE que cambia la información
            Combo2.Item(Index).Text = ""                'Vacío el combo
            Exit Sub                                    'Salgo del módulo
        End Select
        If Index > 11 Then                              'Si es menor al número del combo
          Combo1.Item(Index + 1).SetFocus               'Me voy al siguiente combo de horas
        Else
          If BOTON2.Enabled = True Then                 'Si el botón está activado
            BOTON2.SetFocus                             'Me ubico en el primer botón
          Else
            Exit Sub                                    'Termino
          End If
        End If
      Else
        'Muestro un mensaje de Error si son más de 2 caracteres
        Basura = MsgBox("El cuadro de selección, sólo acepta 2 caracteres", vbCritical, Error)
        KeyAscii = 0        'Anulo el valor tecleado para evitar un BUCLE que cambia la información
      End If
    End If
End Sub
```

Uno de los combos con mayor importancia es el de salones, pero este no se llena como los anteriores, sino los toma de la tabla de permiso[26] de la base de datos de planta física de

[26] La tabla se actualiza por el formulario de permisos, no se mostrará funcionamiento del formulario por seguridad del sistema.

acuerdo al área que use el programa, en donde sólo el administrador los puede asignar o modificar, por ello sólo se mostrará la parte principal del código (por cuestiones de seguridad):

```
  oracionSQL = "SELECT DISTINCT CLVSALON FROM PERMISOS WHERE CLVAREA ='" +
clave_carrera + "' AND ELIMINAR='N'"          'Guardo la sentencia SQL con el salón en una variable
  Set rdotabla1 = rdoConexion2.OpenResultset(oracionSQL, rdOpenKeyset, rdConcurReadOnly,
rdAsyncEnable + rdExecDirect)                 'Genero la tabla
  ComboSalon.Clear                            'Vacío el combo
  ComboSalon.AddItem "A0"                     'Valor por default de la caja de verificación
  With rdotabla1
    If Not .EOF Then                          'Si no tengo registros continúo
      While Not .EOF                          'Realízalo mientras no sea fin de archivo
        CADENA = !CLVSALON                    'Guarda el registro en una cadena
        ComboSalon.AddItem CADENA             'La cadena anéxala al combo
        .MoveNext                             'Me muevo al siguiente registro
      Wend
    End If
  End With
  rdotabla1.Close                             'Cierro el Resultset
  Set rdotabla1 = Nothing                     'Limpio registros de la tabla1
```

La planta física es una base de datos aparte, donde se guarda la información de todos los horarios y la ocupación de salones; contiene las siguientes tablas:

- Planta Física
- Salones
- Permisos

No se mostrarán las tablas por cuestiones de seguridad, pero si se mencionarán, ya que son muy importantes para los horarios.

Después de llenar los combos, se preparan y llenan las dos cuadrículas, no una como se había manejado en los módulos anteriores, en este caso una es para el banco de horas y otra para los horarios, donde se sigue el módulo de preparación o llenado de cuadrícula como se mostró con anterioridad, para no cargar todos los datos se utiliza la bandera mencionada en banco de horas, para el cual se utiliza la instrucción:

```
oracionSQL = "SELECT NREGBANCO, GRUPO_ACT, CLAVE, MATERIA_ABREV, RFC, PROFESOR,
ELIMINAR, ASIGHORARIO FROM VBANCO WHERE ELIMINAR='N' AND ASIGHORARIO='N' ORDER
BY " + Vcampo
```

En cambio para el llenado de la cuadrícula de horarios, se agregaron todos los registros:

```
 oracionSQL = "SELECT ELIMINAR, MATERIA_ABREV, NREGHORARIO, CLAVE, RFC, PROFESOR,
GRUPO_ACT, CLVSALON, HORLUN1, HORLUN2, HORMAR1, HORMAR2, HORMIE1, HORMIE2,
HORJUE1, HORJUE2, HORVIE1, HORVIE2, HORSAB1, HORSAB2, NREGBANCO, CUPO, POSICION,
ASIGHORARIO FROM VHORARIOS WHERE ELIMINAR='N' ORDER BY " + Vcampo
```

Y en el último módulo de inicializa también se tiene un ajuste, y esto es con los combos de horarios o salones, por lo que se anexa la parte del código faltante:

```
  For j = 0 To 11                     'Desactivo los combos de horas y minutos y a la vez los vacío
    Combo1.Item(j).ListIndex = -1             'Combo horas
```

```
    Combo1.Item(j).Enabled = False
    Combo1.Item(j).Text = ""
    Combo2.Item(j).ListIndex = -1                    'Combo minutos
    Combo2.Item(j).Enabled = False
    Combo2.Item(j).Text = ""
  Next j
  If ComboSalon.ListCount = 0 Then                   'Inicializo el combo de salón
    ComboSalon.ListIndex = -1
  Else
    ComboSalon.ListIndex = 0
  End If
  If Comboposicion.ListCount = 0 Then                'Inicializo el combo de salón
    Comboposicion.ListIndex = -1
  Else
    Comboposicion.ListIndex = 0
  End If
```

La parte correspondiente a las acciones de los botones es la misma, excepto al momento de grabar o modificar un registro, ya que hay que validar cada horario, verificar que no existan traslapes con los mismos registros y que tenga permiso para ser asignados, por lo que se mostrará el código para guardar un registro nuevo (lo mismo es para cambiar, sólo cambiando la instrucción de "insert" por "update"):

```
Case "NUEVO"
      'Validamos las entradas
      If Tregbanco.Text = "" Then                    'Si no tengo el registro único de  banco de horas
        MsgBox "No se tiene asignado Profesor, Materia y Grupo para dar de alta una Horario",
vbCritical, "Error"                                  'Muestro un mensaje
        BOTON3.SetFocus                              'Me ubico en el boton de cancelar
        Exit Sub                                     'Termino el módulo
      End If
      'Verifico si no son nulos y Guardo la información en cadenas
      If Tregbanco.Text <> "" Then
        Vnregbanco = Tregbanco.Text                  'Registro del banco
      Else
        Vnregbanco = "0"
      End If
      CADENASQL = "SELECT NREGBANCO, ELIMINAR FROM VHORARIOS WHERE ELIMINAR='N'
AND NREGBANCO=" + Vnregbanco                         'Veo si ya tiene asignado un horario
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)      'Me conecto a la base de datos
      If Not rdotabla1.EOF Then                      'Existe ya un registro de horarios
        MsgBox "Existe un registro de 'Horarios' relacionado con el 'Banco de Horas' asignado ",
vbInformation, "Error"                               'Muestro un mensaje
        BOTON3.SetFocus                              'Me ubico en el boton de cancelar
        rdotabla1.Close                              'Cierro el Resulset
        Set rdotabla1 = Nothing                      'Limpio registros de la tabla1
        Exit Sub                                     'Termino el módulo
      End If
      'Guardo el contenido del salón del combo del salón
      If ComboSalon.Text = "" Or ComboSalon.ListIndex = -1 Or ComboSalon.ListIndex = 0 Then
        Vsalon = "SIN SALÓN"
      Else
        Vsalon = ComboSalon.Text
      End If
      'Valido la entrada de datos para ver sin son horas para ser ingresadas en un campo de horas
```

```
        For i = 0 To 10                    'Primero valido que la información sea válida por ciclos for
          'Si la longitud de horas tiene pero la de minutos no es un
          If Len(Combo1.Item(i).Text) <> 2 Then error
            If Len(Combo1.Item(i).Text) <> 0 Then
              'Muestro el mensaje de error
              Basura = MsgBox("Una casilla de horas está incompleta (" + Str(i) + ")", vbCritical, "Error")
              Exit Sub                     'Temino el proceso
            End If
          End If
          'Si existen horas pero no tengo minutos es un error
          If Len(Combo2.Item(i).Text) <> 2 Then
            If Len(Combo2.Item(i).Text) <> 0 Then
              Basura = MsgBox("Una casilla de minutos está incompleta (" + Str(i) + ")", vbCritical,
"Error")                                   'Muestro el mensaje de error
              Exit Sub                     'Termino el proceso
            End If
          End If
        Next i
          'Los datos están validados y empieza porque estén completos los cuatro combos
        '"Lunes"
        If Combo1.Item(0).Text = "" And Combo2.Item(0).Text = "" And Combo1.Item(1).Text = "" And
Combo2.Item(1).Text = "" Then              'Si no hay datos me voy al siguiente día
          'Continúo
        Else
          'Si los combos  de horas está vacío
          If (Combo1.Item(0).Text = "" Or Combo1.Item(1).Text = "") Then
            'Muestro el mensaje de error
            Basura = MsgBox("Falta llenar un cuadro de horas del día lunes", vbCritical, "Error")
            Exit Sub                       'Termino el proceso
          End If
          'Si los combos de minutos está vacío
          If (Combo2.Item(0).Text = "" Or Combo2.Item(1).Text = "") Then
            'Muestro el mensaje de error
            Basura = MsgBox("Falta llenar un cuadro de minutos del día lunes", vbCritical, "Error")
            Exit Sub                       'Termino el proceso
          End If
          'Rutina para juntar las horas con los minutos y formar los dos horarios
          If Combo1.Item(0).Text <> "" And Combo2.Item(0).Text <> "" Then
            Vlunesinicio = Combo1.Item(0).Text + ":" + Combo2.Item(0).Text
          End If
          If Combo1.Item(1).Text <> "" And Combo2.Item(1).Text <> "" Then
            Vlunestermino = Combo1.Item(1).Text + ":" + Combo2.Item(1).Text
          End If
          'Valido si la segunda hora es menor o igual a la primera hora
          If TimeValue(Vlunestermino) <= TimeValue(Vlunesinicio) Then
            Basura = MsgBox("El segundo horario del lunes es menor o igual al primero horario",
vbCritical, "Error")                       'Muestro el mensaje de error
            Exit Sub                       'Termino el proceso
          End If
        End If
        '"Martes" …  '"Miercoles" …  '"Jueves" …  '"Viernes" …  '"Sábado" …
        '*Proceso para validación con o sin salón*
        If ComboSalon.Text = "A0" Or ComboSalon.Text = "" Then     'Si no tiene asignado salón
          'Guardo el registro normal sin guardar en la base de horarios de la base pricipal de planta física
          'No se revisan translapes, pero si valido entradas
          If Tregbanco.Text <> "" Then     'Registro de la Categoria
```

```
        Vnregbanco = Tregbanco.Text
      Else
        Vnregbanco = "0"
      End If
     …..
      If Vsabadotermino <> "" Then              'Horario de termino del sábado
        Vsabadotermino = "'" + Vsabadotermino + "'"   'Horario correcto, pongo los apostrofes
      Else
        Vsabadotermino = "''"
      End If
      'Guardo todo en una cadena SQL
      CADENA1 = "INSERT INTO HORARIOS (NREGBANCO, CLVSALON, CUPO, POSICION,
HORLUN1, HORLUN2, HORMAR1, HORMAR2, HORMIE1, HORMIE2, HORJUE1, HORJUE2, HORVIE1,
HORVIE2, HORSAB1, HORSAB2, ELIMINAR) Values ("
      CADENA2 = Vnregbanco + ", " + Vsalon + ", " + Vcupo + ", " + vposicion + ", " + Vlunesinicio
+ ", " + Vlunestermino + ", " + Vmartesinicio + ", " + Vmartestermino + ", " + Vmiercolesinicio + ", " +
Vmiercolestermino + ", " + Vjuevesinicio + ", " + Vjuevestermino + ", " + Vviernesinicio + ", " +
Vviernestermino + ", " + Vsabadoinicio + ", " + Vsabadotermino + ", " + "'N' )"
      CADENASQL = CADENA1 + CADENA2
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)  'Llamo a la base de datos
      CADENASQL = "UPDATE BANCOHORAS SET " +  "ASIGHORARIO='H'" + _
      " WHERE NREGBANCO=" + Vnregbanco                'Guardo todo en una cadena SQL
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)
      'Busco el registro para ver que número de registro tiene asignado
      CADENASQL = "SELECT NREGHORARIO, NREGBANCO, CLVSALON, ELIMINAR FROM
HORARIOS WHERE ELIMINAR ='N' AND NREGBANCO=" + Vnregbanco
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)  'Llamo a la base de datos
      With rdotabla1                            'Utilizo  rdotabla 1
        If Not .EOF Then
          Vnreghorario = ""
          Vnreghorario = !NREGHORARIO     'Guardo en una variable
        End If
      End With                                  'Termino este módulo
    Else
      '*Primero que el horario esté dentro de los permisos válidos*
      Vsalon = ComboSalon.Text                  'Válido con la base de permisos de acuerdo al salón
      CADENASQL = "SELECT CLVAREA, CLVSALON, ELIMINAR, LUNHORA1, LUNHORA2,
MARHORA1, MARHORA2, MIEHORA1, MIEHORA2, JUEHORA1, JUEHORA2, VIEHORA1, VIEHORA2,
SABHORA1, SABHORA2 FROM PERMISOS WHERE CLVAREA ='" + clave_carrera + "' AND
CLVSALON='" + Vsalon + "' AND ELIMINAR='N'" 'Guardo la sentencia SQL con el salon en una variable
      Set rdotabla2 = rdoConexion2.OpenResultset(CADENASQL, rdOpenKeyset, rdConcurReadOnly,
rdAsyncEnable + rdExecDirect)               'Genero la tabla
      With rdotabla2                            'Trabajo con la tabla de permisos
        If .EOF Then                            'Si no tengo registros marco error
          MsgBox "'NO' se tiene permiso asignado", vbCritical, "Error"   'Marco un mensaje de error
          rdotabla2.Close                       'Cierro la tabla
          Set rdotabla2 = Nothing               'Limpio registros de la tabla2
          BOTON3.SetFocus                       'Me ubico en el botón de cancelar
          Exit Sub                              'Termino
        End If
        'Válido para cada día 'Lunes
        'Si tengo un horario dentro del día específico continúo
        If Vlunesinicio <> "" And Vlunestermino <> "" Then
          .MoveFirst                            'Me voy al primer registro
          Do While Not .EOF
            'Obtengo la hora de cada registro obtenido
```

```
                CADENA1 = Format(!LUNHORA1, "SHORT TIME")
                CADENA2 = Format(!LUNHORA2, "SHORT TIME")
                If CDate(Vlunesinicio) >= CDate(CADENA1) And CDate(Vlunestermino) <=
CDate(CADENA2) Then                         'Ahora si el horario está dentro de un permiso
                    Exit Do                 'Cumple Continúo y salgo del ciclo
                End If
                .MoveNext                   'Me cambio de registro
                If .EOF Then                'Si termino de revisar todos los registros
                  'Muestro un mensaje de que no tiene permiso asignado
                  MsgBox "No se tiene permiso asignado (Lunes)", vbCritical, "Error"
                  BOTON3.SetFocus           'Me ubico en el botón de cancelar
                  rdotabla2.Close           'Cierro la tabla
                  Set rdotabla2 = Nothing   'Limpio registros de la tabla2
                  Exit Sub                  'Termino
                End If
              Loop
            End If
            'Martes…'Miercoles…'Jueves…'Viernes… 'Sábado
            'Si tengo un horario dentro del día específico continúo
          End With
          rdotabla2.Close                   'Cierro el Resulset
          Set rdotabla2 = Nothing           'Limpio registros de la tabla2
         '*Ahora buscamos translapes en los horarios registrados del salón correspondiente*
          'Guardo la sentencia SQL con el salón en una variable
          CADENASQL = "SELECT CLVSALON, ELIMINAR, NREGHORARIO, POSICION, HORLUN1,
HORLUN2, HORMAR1, HORMAR2, HORMIE1, HORMIE2, HORJUE1, HORJUE2, HORVIE1, HORVIE2,
HORSAB1, HORSAB2 FROM HORARIOS WHERE CLVSALON='" + Vsalon + "' AND ELIMINAR='N'"
          Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL, rdOpenKeyset, rdConcurReadOnly,
rdAsyncEnable + rdExecDirect)             'Genero la tabla
          With rdotabla1                    'Trabajo con la tabla de horarios de la carrera
            'Válido para cada día para que no tenga translapes con otro registro
            If Not .EOF Then                'Si tengo registro realizo este proceso
              'Lunes
              'Si tengo un horario dentro del día específico continúo
              If Vlunesinicio <> "" And Vlunestermino <> "" Then
                .MoveFirst                  'Me voy al primer registro
                Do While Not .EOF           'Inicializo la variable
                  tmpregistro = ""
                  tmpregistro = Str(!NREGHORARIO)       'Cambio el tipo de dato de numerico a cadena
                  'Obtengo la hora de cada registro obtenido
                  CADENA1 = Format(!HORLUN1, "SHORT TIME")
                  CADENA2 = Format(!HORLUN2, "SHORT TIME")
                  'Veo que existan registros del día correspondiente
                  If CADENA1 = "" Or CADENA1 = "00:00" Then
                    .MoveNext               'No hay registro que comparar y continúo con el siguiente
                  Else
                    If CDate(Vlunesinicio) >= CDate(CADENA1) And CDate(Vlunesinicio) <
CDate(CADENA2) Then                         'Ahora comparo la hora inicial si no hay translape
                  'Si pertenece a Arquitectura, Diseño Industrial o SUA comparo que la posición sea distinta
                    If clave_carrera = "ARQ" Or clave_carrera = "DIS" Or clave_carrera = "SUA" Then
                        If Trim(Comboposicion.Text) = !posicion Then
                          MsgBox "Translape en el salón " + Vsalon + " con el horario(Lunes):" +
CADENA1 + " a " + CADENA2 + " del registro: " + tmpregistro + ", cambiar la posición del profesor",
vbInformation, "Translape de Horario"       'Muestro un mensaje de que tiene un translape
                          rdotabla1.Close   'Cierro el Resulset
                          Set rdotabla1 = Nothing 'Limpio registros de la tabla1
```

```
                    BOTON3.SetFocus     'Me ubico en el botón de cancelar
                    Exit Sub            'Termino
                  Else                  'Continúo
                  End If
                Else
                  'Muestro un mensaje de que tiene un translape
                  MsgBox "Translape en el salón " + Vsalon + " con el horario(Lunes):" + CADENA1 + " a " + CADENA2 + " del registro: " + tmpregistro, vbCritical, "Translape de Horario"
                  BOTON3.SetFocus       'Me ubico en el botón de cancelar
                  rdotabla1.Close       'Cierro el Resutlset
                  Set rdotabla1 = Nothing 'Limpio registros de la tabla1
                  Exit Sub              'Termino
                End If
              End If

              If CDate(Vlunestermino) > CDate(CADENA1) And CDate(Vlunestermino) <= CDate(CADENA2) Then   'Ahora comparo la hora final si no hay translape
              'Si pertenece a Arquitectura, Diseño Industrial o SUA comparo la posición sea distinta
              If clave_carrera = "ARQ" Or clave_carrera = "DIS" Or clave_carrera = "SUA" Then
                  If Trim(Comboposicion.Text) = !posicion Then

                    MsgBox "Translape en el salón " + Vsalon + " con el horario(Lunes):" + CADENA1 + " a " + CADENA2 + " del registro: " + tmpregistro + ", cambiar la posición del profesor", vbInformation, "Translape de Horario"   'Muestro un mensaje de que tiene un translape
                    BOTON3.SetFocus     'Me ubico en el boton de cancelar
                    rdotabla1.Close     'Cierro el Resulset
                    Set rdotabla1 = Nothing 'Limpio registros de la tabla1
                    Exit Sub            'Termino
                  Else                  'Continúo
                  End If
                Else
                  'Muestro un mensaje de que tiene un translape
                  MsgBox "Translape en el salón " + Vsalon + " con el horario(Lunes):" + CADENA1 + " a " + CADENA2 + " del registro: " + tmpregistro, vbCritical, "Translape de Horario"
                  BOTON3.SetFocus       'Me ubico en el botón de cancelar
                  rdotabla1.Close       'Cierro el Resulset
                  Set rdotabla1 = Nothing 'Limpio registros de la tabla1
                  Exit Sub              'Termino
                End If
              End If
              'Verifico que el horario no se encuentre encapsulado dentro del horario escrito
              If CDate(CADENA1) > CDate(Vlunesinicio) And CDate(CADENA2) < CDate(Vlunestermino) Then
                MsgBox "Translape en el salón " + Vsalon + " con el horario(Lunes):" + CADENA1 + " a " + CADENA2 + " del registro: " + tmpregistro + ", cambiar la asignación del horario", vbInformation, "Translape de Horario"   'Muestro un mensaje de que tiene un translape
                BOTON3.SetFocus         'Me ubico en el botón de cancelar
                rdotabla1.Close         'Cierro el Resulset
                Set rdotabla1 = Nothing 'Limpio registros de la tabla1
                Exit Sub                'Termino
              Else                      'Continúo
              End If
            .MoveNext                   'Me cambio de registro
            End If
          Loop
        End If                          'Si paso reviso todos y no hay error continúo
```

```
        'Martes… 'Miercoles… 'Jueves… 'Viernes…'Sábado…
      rdotabla1.Close                    'Cierro el Resulset
      Set rdotabla1 = Nothing            'Limpio registros de la tabla1
      'Preparo los registros para grabar los datos
      If Tregbanco.Text <> "" Then       'Registro de la Categoría
         Vnregbanco = Tregbanco.Text
      Else
         Vnregbanco = "0"
      End If
     ……..
      'Guardo todo en una cadena SQL
      CADENA1 = "INSERT INTO HORARIOS (NREGBANCO, CLVSALON, CUPO, POSICION,
HORLUN1, HORLUN2, HORMAR1, HORMAR2, HORMIE1, HORMIE2, HORJUE1, HORJUE2, HORVIE1,
HORVIE2, HORSAB1, HORSAB2, ELIMINAR) Values ("
      CADENA2 = Vnregbanco + ", " + Vsalon + ", " + Vcupo + ", " + vposicion + ", " + Vlunesinicio
+ ", " + Vlunestermino + ", " + Vmartesinicio + ", " + Vmartestermino + ", " + Vmiercolesinicio + ", " +
Vmiercolestermino + ", " + Vjuevesinicio + ", " + Vjuevestermino + ", " + Vviernesinicio + ", " +
Vviernestermino + ", " + Vsabadoinicio + ", " + Vsabadotermino + ", " + "'N' )"
      CADENASQL = CADENA1 + CADENA2
      'Llamo a la base de datos de la carrera
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)
      'Busco el registro para ver que número de registro tiene asignado
      CADENASQL = "SELECT NREGHORARIO, NREGBANCO, CLVSALON, ELIMINAR FROM
HORARIOS WHERE ELIMINAR ='N' AND NREGBANCO=" + Vnregbanco + " AND CLVSALON=" +
Vsalon
      'Llamo a la base de datos
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)
      'Guardo todo en una cadena SQL
      CADENASQL = "UPDATE BANCOHORAS SET " +  "ASIGHORARIO='H'" + _
       " WHERE NREGBANCO=" + Vnregbanco
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)
      rdotabla1.Close                    'Cierro el Resultset
      Set rdotabla1 = Nothing            'Limpio registros de la tabla1
      'Busco el registro para ver que número de registro tiene asignado
      CADENASQL = "SELECT NREGHORARIO, NREGBANCO, CLVSALON, ELIMINAR FROM
HORARIOS WHERE ELIMINAR ='N' AND NREGBANCO=" + Vnregbanco + " AND CLVSALON=" +
Vsalon
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL)  'Llamo a la base de datos
      With rdotabla1                     'Utilizo a rdotabla1
        If Not .EOF Then
          Vnreghorario = ""
          Vnreghorario = !NREGHORARIO     'Guardo en una variable
          If Vnreghorario <> "" Then  'Si existe, guardo el registro en la base de planta física
            'Guardo todo en una cadena SQL
            CADENA1 = "INSERT INTO PLANTA (NREGHORARIO, CLVAREA, CLVSALON,
HORLUN1, HORLUN2, HORMAR1, HORMAR2, HORMIE1, HORMIE2, HORJUE1, HORJUE2, HORVIE1,
HORVIE2, HORSAB1, HORSAB2, ELIMINAR) Values ("
            CADENA2 = Str(Vnreghorario) + ", '" + clave_carrera + "', " + Vsalon + ", " +
Vlunesinicio + ", " + Vlunestermino + ", " + Vmartesinicio + ", " + Vmartestermino + ", " +
Vmiercolesinicio + ", " + Vmiercolestermino + ", " + Vjuevesinicio + ", " + Vjuevestermino + ", " +
Vviernesinicio + ", " + Vviernestermino + ", " + Vsabadoinicio + ", " + Vsabadotermino + ", " + "'N' )"
            'OJO LA CLAVE DE LA CARRERA DEBE IR CON '' SINO MARCA ERRROR
            CADENASQL = CADENA1 + CADENA2
            'Llamo a la base de datos de la planta física
            Set rdotabla4 = rdoConexion2.OpenResultset(CADENASQL)
            rdotabla4.Close              'Cierro el Resultset
```

```
            Set rdotabla4 = Nothing          'Limpio registros de la tabla1
          End If
        End If
      End With
    End If
    rdotabla1.Close                          'Cierro el Resultset
    Set rdotabla1 = Nothing                  'Limpio registros de la tabla1
    PROCESO = "INICIO"                       'Vuelvo todo a la normalidad
    Inicializa                               'Inicializo la pantalla
    'Oculto el registro del banco o actualizo por completo el cuadro
    If Vnregbanco <> "" And Vnregbanco <> 0 And Cbanco.Rows <> 2 And Cbanco.Rows <> 1 And
Cbanco.Rows = 0 Then
      Basura = Oculta_un_Banco(Vnregbanco)   'Inserto el registro al final de la lista
    Else
      Llena_Banco                            'Actualizo el cuadro
    End If
    'Inserto el registro nuevo o actualizo por completo el cuadro
    If Vnreghorario <> "" Or Vnreghorario <> 0 Then
      Basura = Inserta_un_Horario(Vnreghorario) 'Inserto el registro al final de la lista
    Else
      Llena_Horarios                         'Actualizo el cuadro
    End If
```

Ésta es una parte muy pesada y tediosa del programa, ya que requiere de muchas validaciones, se pudo recortar el código, pero se perdía funcionalidad al definir día en mensaje o el código, en especial cuando existía un error o falla, por lo que se decidió poner las validaciones individuales, lo que hizo que se aumentara el tamaño del código.

3.4.5. Extraordinarios.

La parte de captura de extraordinarios no se modificó mucho como lo trabaja Cronos, se mantiene la estructura general, la diferencia es en la base de datos, ya que sólo se maneja una sola base para primera, segunda y tercera vuelta.

La estructura de la tabla es la siguiente:

EXTRAORDINARIOS	TIPO	LONG	LLAVE	NOMBRE
NREGEXTRA	INT	4	*	Número de Registro del extraordinario
CLVUNICAPROF1	INT	4		Clave única del profesor propietario
CLVUNICAPROF2	INT	4		Clave única del profesor suplente
CLVUNICAMAT	INT	4		Clave única de la materia
VUELTA	VARCHAR	1		Clave única de la materia
FECHAHORA	DATETIME	8		Fecha y hora de realización de examen extra
CLVSALON	VARCHAR	6	*	Clave única de los salones
TURNO	VARCHAR	10		Turno para especificar una clave de extra
CLVEXTRA	VARCHAR	2		Clave del período del extra
ELIMINAR	VARCHAR	1		N' NO ELIMINADO , otro carácter ELIMINADO

Para lograr que se tenga una sola base, se agrega un campo extra, el cual se llama Vuelta, en donde se podran definir 1 a N vueltas, por lo que para poder asignar este valor, se crea una variable general y se requiere apoyo de otro formulario, como lo muestra la pantalla 3.16.

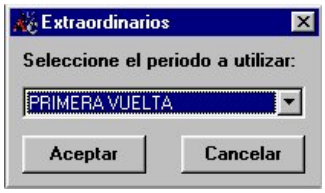


Figura 3.16. Formulario de selección del número de vuelta.

Por lo que tambien se tiene que anexar esta parte a diagrama por módulo, como se muestra a continuación:

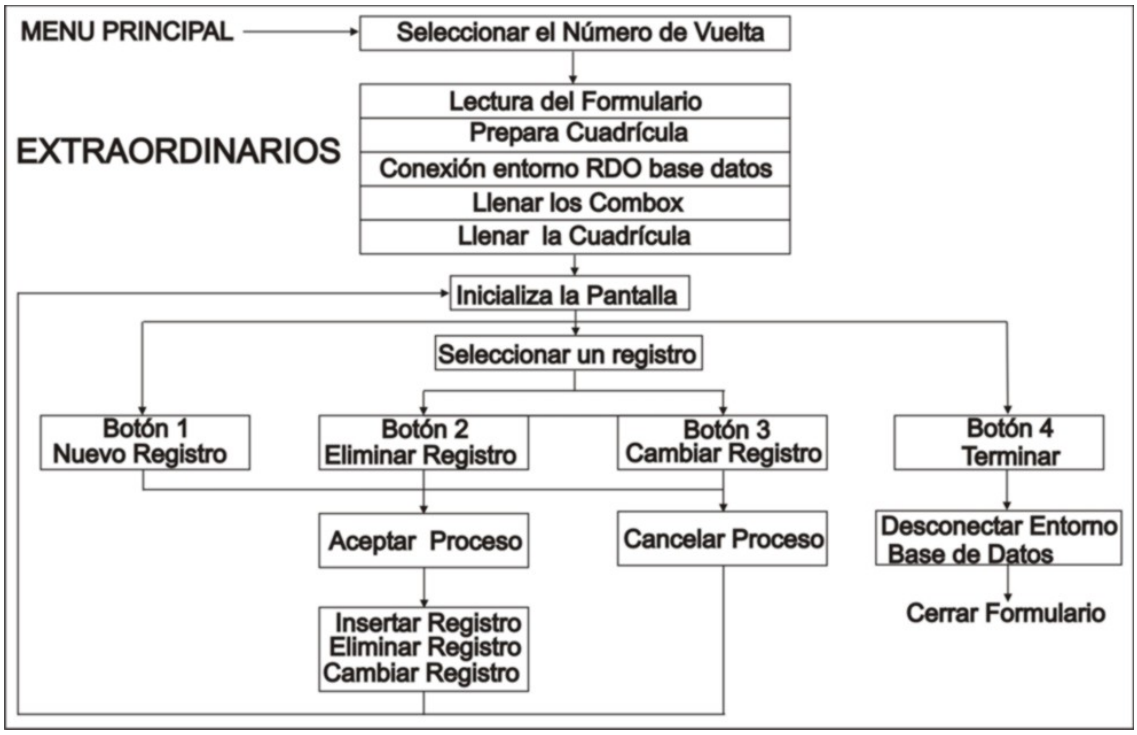


Diagrama 3.5 Módulos de programación para el formulario de Extraordinarios

El formulario es muy simple, ya que sólo sirve para definir una variable, por lo no se mostrará el código; la variable pasa al siguiente formulario de Extraordinarios, el cual se muestra en la pantalla 3.17.

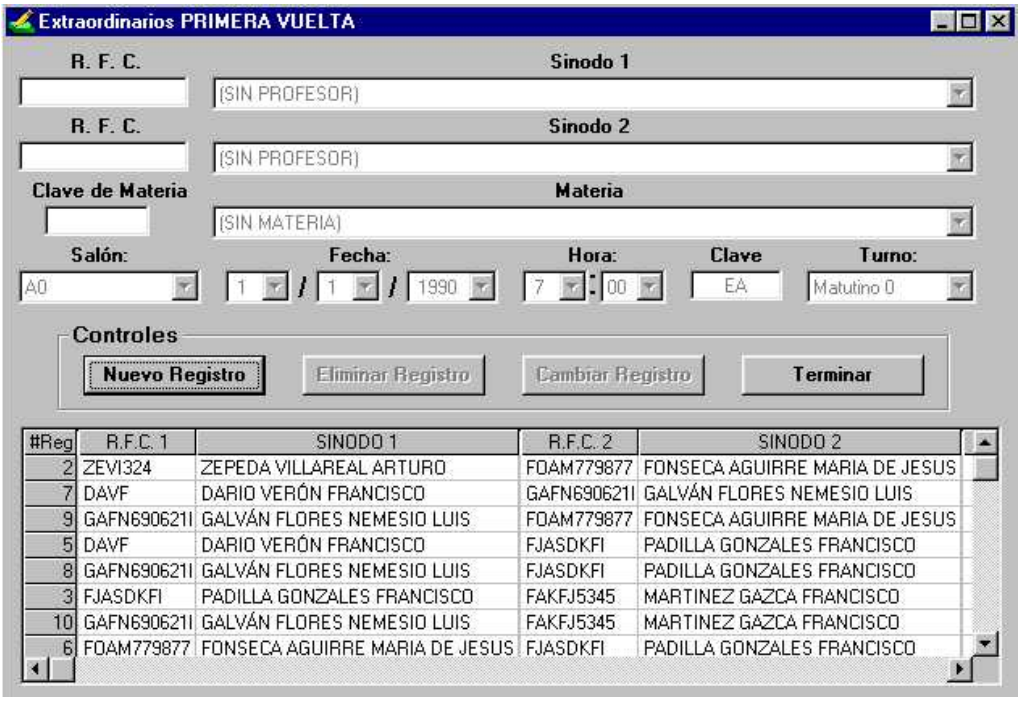


Figura 3.17 Formulario de Extraordinarios.

Al comenzar al leer, el código obtiene el valor de la variable del formulario anterior, aparte lo asigna al título del formulario actual y se cierra, tal como se muestra a continuación:

```
vuelta = Trim(fvuelta.ComboVuelta.Text)        'Guardo la vuelta en un variable
Select Case vuelta                             'Escribo en el texto el número de vuelta
   Case "PRIMERA VUELTA"
      Tnvuelta.Text = ""                       'Inicializo
      Tnvuelta.Text = 1                        'Indico a una variable el número de vuelta "tnvuelta"
   Case "SEGUNDA VUELTA"
      Tnvuelta.Text = ""                       'Inicializo
      Tnvuelta.Text = 2                        'Indico a una variable el número de vuelta "tnvuelta"
        ……
   Case Else
      vuelta = "SIN VUELTA"                    'Me coloco en la pantalla sin asignar vuelta
      Tnvuelta.Text = ""                       'Inicializo
      Tnvuelta.Text = 0                        'Indico a una variable el numero de vuelta "tnvuelta"
End Select
Unload fvuelta                                 'Desactivo el formulario de vuelta
fextras.Caption = "Extraordinarios " + vuelta  'Pongo el Título del formulario
```

Ya después se debe llenar cada uno de los combos que hay en el formulario, pero es el mismo formato para profesor, materia, salón; en cuanto a las horas, se llenan como se muestra en los horarios, por lo que sólo falta ver como se llena la fecha y el turno.

El código para llenar la fecha es el siguiente:

```
For i = 1 To 31                                'La cifras de los días básicos 1 - 31
   Combodia.AddItem Str(i)
Next i
```

```
Combomes.AddItem "ENE"                    'Pongo los MESES(0-11)
Combomes.AddItem "FEB"
Combomes.AddItem "MAR"
Combomes.AddItem "ABR"
Combomes.AddItem "MAY"
Combomes.AddItem "JUN"
Combomes.AddItem "JUL"
Combomes.AddItem "AGO"
Combomes.AddItem "SEP"
Combomes.AddItem "OCT"
Combomes.AddItem "NOV"
Combomes.AddItem "DIC"
For i = 1990 To 2999                      'La cifras de los años 1990 - 2990
   Comboanio.AddItem Str(i)
Next i
```

Y para los turnos se sigue el siguiente código:

```
Comboturno.AddItem "Matutino 1" ……..      'Agrego los turnos permitidos
Comboturno.AddItem "Matutino 9"
Comboturno.AddItem "Vespertino 1"
Comboturno.AddItem "Vespertino 9"……..
Comboturno.AddItem "Sin Turno 1"
```

Otro módulo a tratar, es generar la clave diseñada para la vuelta de extraordinarios, tal como se explicó en el capítulo anterior, el código es el siguiente:

```
Private Sub CrearClaveExtra()
   Dim Vnumsem As String                  'Defino mis variables
   Dim Vnumvue As String
   If Len(Período_escolar) = 5 Then       'Primero Obtengo que período de semestre
      Vnumsem = Trim(Right(Período_escolar, 1))  'Sólo obtengo el número de semestre
   Else
      Exit Sub
   End If
   If Tnvuelta.Text <> "" Then            'Ahora obtengo el número de vuelta
      Vnumvue = Trim(Str(Tnvuelta.Text))
   Else
      Exit Sub
   End If
   'Busco a que período corresponde
   If Vnumsem = "1" And Vnumvue = "1" Then   'Semestre uno, Vuelta uno
      Tclvextra.Text = "EA"
   End If
   If Vnumsem = "1" And Vnumvue = "2" Then   'Semestre uno, Vuelta dos
      Tclvextra.Text = "EB"
   End If
   If Vnumsem = "2" And Vnumvue = "1" Then   'Semestre dos, Vuelta uno
      Tclvextra.Text = "EC"
   End If
   If Vnumsem = "2" And Vnumvue = "2" Then   'Semestre dos, Vuelta dos
      Tclvextra.Text = "ED"
   End If
   If Vnumsem = "3" And Vnumvue = "1" Then   'Semestre tres, Vuelta uno
      Tclvextra.Text = "EE"
   End If
```

```
  If Vnumsem = "3" And Vnumvue = "2" Then        'Semestre tres, Vuelta dos
     Tclvextra.Text = "EF"
  End If
End Sub
```

Los demás módulos son similares a los que se explicaron en materias, tal vez la única diferencia es la fecha y la hora, por lo que esto puede explicar la diferencia que se tiene en cada módulo, con el siguiente código de validación del botón 2:

```
'Validamos que la fecha sea válida
Vfecha = Combodia.Text + "/" + Combomes.Text + "/" + Comboanio.Text
If Not IsDate(Vfecha) Then                    'Validamos si es válida
   'Muestro un mensaje
   MsgBox "No tiene una fecha válida, rectificar que la fecha sea existente", vbCritical, "Error"
   BOTON3.SetFocus                            'Me ubico en el boton de cancelar
   i = Combodia.ListIndex                     'Vemos la posición
   If i >= 28 Or i <= 31 Then                 'Si está el índice dentro de los parámetros erróneos
      Combodia.ListIndex = Combodia.ListIndex – 1        'Tomamos un valor anterior
   End If
   Exit Sub                                   'Termino el módulo
End If
'Válido la entrada de horas
If Len(Trim(Combohoras.Text)) <= 0 Or Len(Trim(Combohoras.Text)) > 2 Then
      'Muestro el mensaje de error
      Basura = MsgBox("La casilla de horas esta incompleta", vbCritical, "Error")
      Exit Sub                                'Temino el proceso
End If
'Si la longitud de horas tiene pero la de minutos no es un error
If Len(Trim(Combominutos.Text)) <= 0 Or Len(Trim(Combominutos.Text)) > 2 Then
     'Muestro el mensaje de error
     Basura = MsgBox("La casilla de minutos esta incompleta", vbCritical, "Error")
     Exit Sub                                 'Termino el proceso
End If
```

Y para guardar o cambiar el registro se tienen que juntar las variables, en el módulo del botón 2, como se muestra a continuación:

```
 'Fecha
  If Combodia.Text = "" Or Combodia.ListIndex = -1 Then        'día
     vdia = "1"
  Else
     vdia = Combodia.Text
  End If
  If Combomes.Text = "" Or Combomes.ListIndex = -1 Then        'mes
     vmes = "1"
  Else
     vmes = Str(Combomes.ListIndex + 1)
  End If
  If Comboanio.Text = "" Or Comboanio.ListIndex = -1 Then      'año
     vanio = "1990"
  Else
     vanio = Comboanio.Text
  End If
  Vfecha = Trim(vdia) + "/" + Trim(vmes) + "/" + Trim(vanio)   'Guardo la fecha
'Hora
  If Combohoras.Text = "" Or Combohoras.ListIndex = -1 Then    'hora
```

```
    vhora = "01"
  Else
    vhora = Combohoras.Text
  End If
  If Combominutos.Text = "" Or Combominutos.ListIndex = -1 Then     'minuto
    vminuto = "00"
  Else
    vminuto = Combominutos.Text
  End If
  Vhorario = Trim(vhora) + ":" + Trim(vminuto)      'Guardo el horario de inicio
  Vfechahora = "'" + Vfecha + " " + Vhorario + "'"  'Uno la fecha más la hora
```

En cambio, para obtenerlo de la base y colocarlo en la cuadrícula, se realiza el proceso inverso del código anterior, pero para que se explique mejor se muestra el código que está dentro del módulo de "Llenarcuadro":

```
  Cextras.Col = 11                          'Me ubico en la 12 celda
  CADENA = !FECHAHORA                       'Guarda el registro en una cadena
  CADENA = Format(CADENA, "d/m/yyyy") 'Cambio el formato para solo mostrar la fecha
  If CADENA <> "" Then                      'Si tengo horario lo pongo en el cuadro
    Cextras.Text = CADENA                   'La cadena anexala al combo
  End If
  Cextras.Col = 12                          'Me ubico en la 13 celda
  CADENA = !FECHAHORA                       'Guarda el registro en una cadena
  CADENA = Format(CADENA, "Short Time")  'Cambio el formato para sólo mostrar la hora
  If CADENA <> "00:00" And CADENA <> "" Then  'Si tengo horario lo pongo en el cuadro
    Cextras.Text = CADENA                   'La cadena anexada al combo
  End If
```

Por último, se explica cómo se pasa la información de la cuadrícula a cada uno de los combos, ya sea de fecha o de hora, en parte del código correspondiente a la cuadrícula:

```
Cextras.Row = RENGLON                       'Fecha
Cextras.Col = 11
Basura = Cextras.Text                       'Guardo la informacion en una variable
'Desgloso la cadena para obtener
  vdia = Str(Day(Basura))       'dia
  For i = 0 To Combodia.ListCount           'Realizo un ciclo para buscar el salón
    CADENA = Combodia.List(i)               'Obtengo el texto del combo
    If (Trim(CADENA) = Trim(vdia)) Then
      Combodia.Text = CADENA                'Encontre el valor
      Exit For                              'Salgo del ciclo de for
    Else
      Combodia.ListIndex = 0                'Pongo el valor por default
    End If
  Next i
  vmes = Str(Month(Basura))                 'mes
  For i = 0 To Combomes.ListCount           'Realizo un ciclo para buscar el salón
    CADENA = Combomes.List(i)               'Obtengo el texto del combo
    If Val(Trim(vmes)) - 1 = i Then
      Combomes.Text = CADENA                'Encontre el valor
      Exit For                              'Salgo del ciclo de for
    Else
      Combomes.ListIndex = 0                'Pongo el valor por default
    End If
  Next i
```

```
    vanio = Str(Year(Basura))                'anio
    For i = 0 To Comboanio.ListCount         'Realizo un ciclo para buscar el salón
       CADENA = Comboanio.List(i)            'Obtengo el texto del combo
       If (Trim(CADENA) = Trim(vanio)) Then
          Comboanio.Text = CADENA            'Encontre el valor
          Exit For                           'Salgo del ciclo de for
       Else
          Comboanio.ListIndex = 0            'Pongo el valor por default
       End If
    Next i
 Cextras.Row = RENGLON                       'Horario
 Cextras.Col = 12
 Basura = Cextras.Text                       'Guardo la información en una variable
 'Desgloso la cadena para obtener
    vhora = Str(Hour(Basura))                'Hora
    For i = 0 To Combohoras.ListCount        'Realizo un ciclo para buscar el salón
       CADENA = Combohoras.List(i)           'Obtengo el texto del combo
       If (Trim(CADENA) = Trim(vhora)) Then
          Combohoras.Text = CADENA           'Encontre el valor
          Exit For                           'Salgo del ciclo de for
       Else
          Combohoras.ListIndex = 0           'Pongo el valor por default
       End If
    Next i
    vminuto = Str(Minute(Basura))            'Minutos
    For i = 0 To Combominutos.ListCount      'Realizo un ciclo para buscar el salón
       CADENA = Combominutos.List(i)         'Obtengo el texto del combo
       If (Trim(CADENA) = Trim(vminuto)) Then
          Combominutos.Text = CADENA         'Encontre el valor
          Exit For                           'Salgo del ciclo de for
       Else
          Combominutos.ListIndex = 0         'Pongo el valor por default
       End If
    Next i
```

Los códigos que no se mencionan son porque, ya como se explico anteriormente, considerando cada uno de los campos de la tabla de extraordinarios.

3.4.6. Finales

Lo que corresponde a los finales es muy similar a lo que es la unión en horarios y los extraordinarios, en donde no se manejan vueltas, sino hasta la captura, por lo que en la tabla de finales se repiten los campos de Cronos, tal como se ve en la estructura de la tabla:

FINALES	TIPO	LONG	LLAVE	NOMBRE
NREGFINAL	INT	4	*	Número de Registro del Final
NREGBANCO	INT	4		Número de registro del banco
CLVUNICAPROF2	INT	4		Clave única del profesor suplente
PROFESOR2	VARCHAR	45		Profesor empezando por el apellido.
RFC2	VARCHAR	15		Registro federal de causantes
CLVUNICAPROF3	INT	4		Clave única del profesor suplente 2
PROFESOR3	VARCHAR	45		Profesor empezando por el apellido.

RFC3	VARCHAR	15		Registro federal de causantes
CLVSALON1	VARCHAR	6		Clave única de los salones uno
CLVSALON2	VARCHAR	6		Clave única de los salones dos
FECHA1INICIO	DATETIME	8		Horario de inicio de 1 vuelta
FECHA1FIN	DATETIME	8		Horario de término de 1 vuelta
FECHA2INICIO	DATETIME	8		Horario de inicio de 1 vuelta
FECHA2FIN	DATETIME	8		Horario de término de 1 vuelta
ELIMINAR	VARCHAR	1		N' NO ELIMINADO, otro caracter ELIMINADO

Esta tabla se familiariza mucho con la combinación de horarios y de extraordinarios, pero se puede ver fácilmente por el diagrama de módulos de programación:

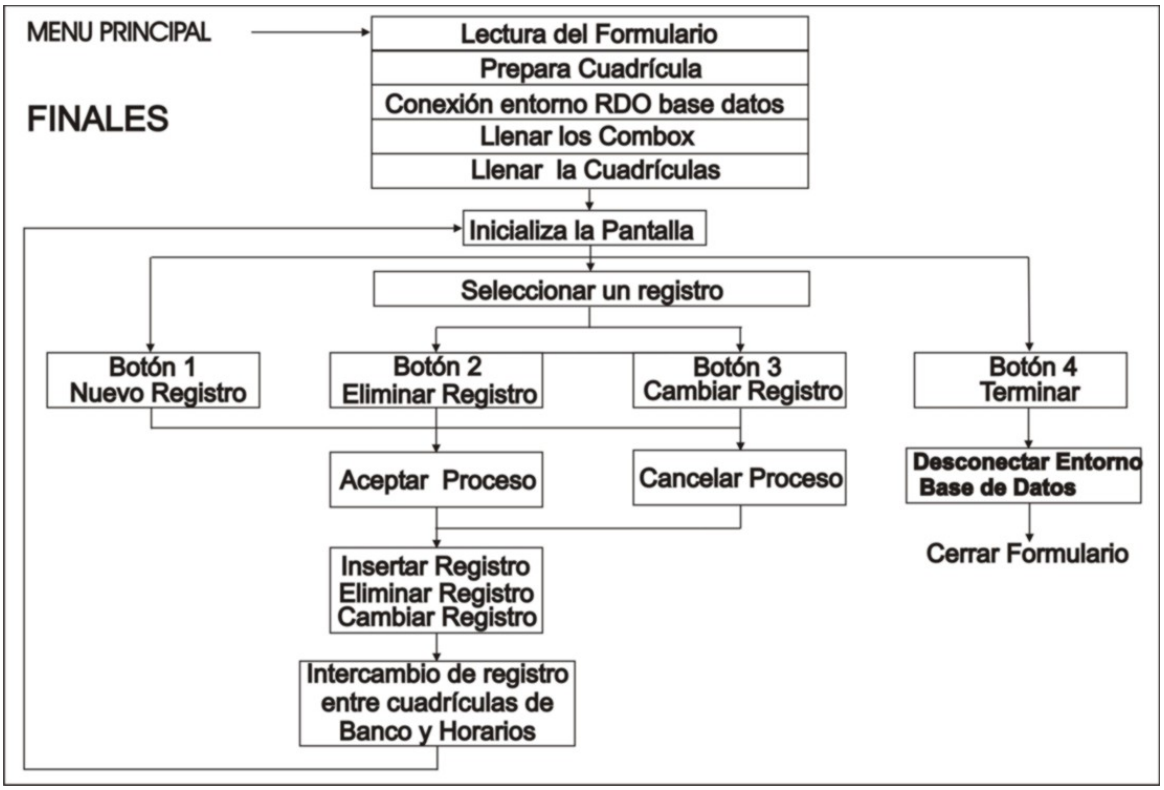


Diagrama 3.6 Módulos de programación para el formulario de Finales

Otro detalle se marca en la fecha, ya que se manejan dos campos, uno para cuando comienza el final y el otro para cuando termina. La hora viene incluida en dos campos, aunque no se vea en la tabla.

El diseño del formulario es muy semejante al que se mostró en horarios, ya que se consideran por cada conjunto de horarios con la misma materia o por cada horario se genera un registro de un final, por ello puede llevar de uno a tres profesores, tal como lo muestra la pantalla 2.18.

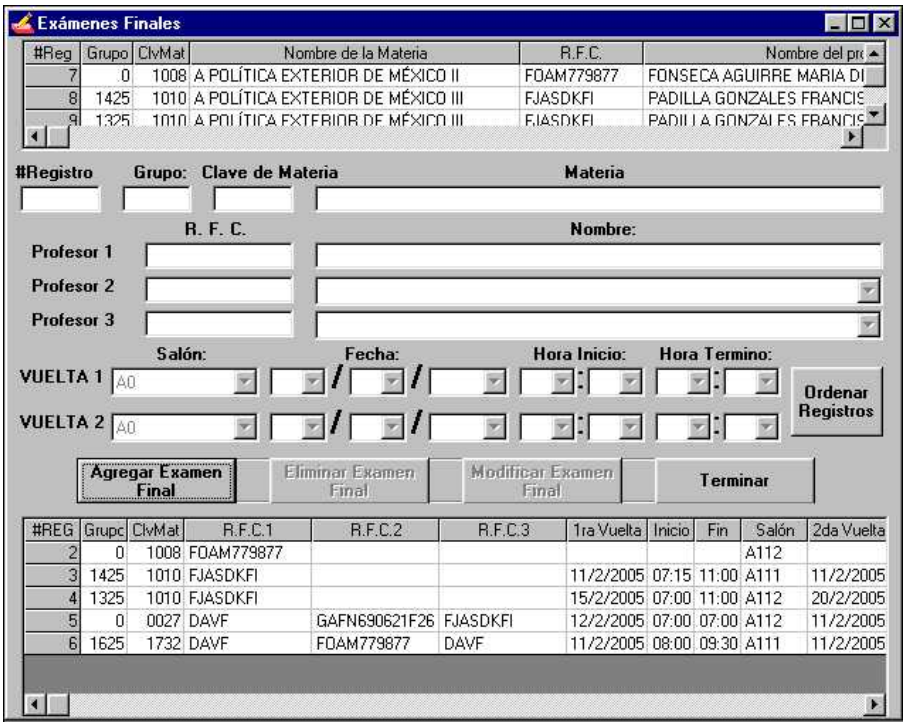


Figura 2.18. Formulario de Finales.

Éste es un formulario muy saturado porque contiene muchos campos, pero en realidad con base en la tabla son pocos, sólo es un apoyo para el usuario, en cuanto al código de cada uno de los módulos ya se explicó anteriormente, pero con los ajustes necesarios para usar los campos de finales.

3.5. Programa para Exportar a las diferentes áreas

Para que el programa sustituya por completo al programa de Cronos, falta por crear una interfaz u otro programa para exportar los campos necesarios a cada una de las áreas que requieren información del programa, tal como lo hacía el programa "Utileria.exe".

Por lo que se crea un programa en donde se exporta la información a las áreas de Servicios Escolares, Secretaría Académica, Departamento de Personal y para la página de Horarios en Internet.

Sólo se menciona este programa porque es una parte esencial de Wcronos, pero por cuestiones de seguridad no se dará ningún detalle de este programa, pero sí se puede mencionar que se manejan todas las tablas que se mencionaron con anterioridad.

CAPÍTULO

4. GENERACIÓN DE REPORTES CON CRYSTAL REPORTS.

4.1. Crystal Reports.

Cystal Reports es un programa que está diseñado para trabajar con bases de datos, su función principal es ayudar a mostrar, analizar e interpretar la información importante para el usuario, facilitando la creación de diferentes tipos de reportes, además de una gran variedad de herramientas poderosas necesarias para generar informes complejos o especializados.

El diseño para generar los informes que se desean se puede obtener prácticamente de cualquier fuente de datos, ya sea a través de los asistentes incorporados o por medio de una interfaz que se puede integrar al programa, en especial en Visual Basic. Todo esto ofrece ayuda para entender mejor los datos y descubrir relaciones importantes que están ocultas para los usuarios.

La conexión que se ofrece con los datos es mucho más rápida, ya que tiene una interfaz tipo Explorer del Explorador de datos para seleccionar las fuentes de datos; además, nos ofrece la posibilidad de elegir entre los controladores disponibles, en especial la de manejar una conexión vía ODBC.

La interfaz para el diseño de informes, ofrece un mayor control del posicionamiento de objetos de los informes con la capacidad de mover, alinear, ajustar de tamaño, copiar y pegar múltiples objetos, en vez de programar el reporte, lo cual es muy tedioso y pesado al momento de diseñar informes.

Los archivos que maneja Crystal Reports tienen extensión ".rpt", para crear archivos más pequeños con mayor rapidez de compresión y un almacenamiento más eficiente de informes complejos que contengan datos guardados.

La sintaxis de Crystal es el lenguaje de fórmulas usado en versiones anteriores, mientras que la sintaxis Basic es nueva para Seagate Crystal Reports. Se pueden crear fórmulas usando sintaxis Crystal o Basic. Casi cualquier fórmula escrita con una se puede crear con la otra. Como en el caso de la sintaxis Crystal, la sintaxis Basic no requiere ninguna DLL adicional para su ejecución.

Para la interfaz desde visual Basic sólo tienen que dar clic en proyecto, y a su vez en referencias, sólo basta con activar la casilla donde se encuentra el componente de: Crystal Reports Engine 8 Object Library.

Crystal Reports es el generador de reporte por excelencia de Visual Basic desde versiones anteriores a .NET. Esta no es la primera vez que se distribuye una versión de este generador de reporte junto a una versión de Visual Studio; en la versión 4 de Visual Basic se incluía una versión de Crystal Reports.

Al igual que desde el ambiente de desarrollo de Visual Studio, también se puede crear reportes desde el ambiente de desarrollo de Crystal Reports y luego ejecutarlo desde el proyecto Visual Basic. Los Reportes desarrollados bajo el ambiente de Crystal Reports pueden ser compilados a .EXE para poder ser ejecutados *stand-alone (Por si solo)*.

Entre las características principales que debemos conocer en Crystal Reports para poder crear una buena solución de reportes están los "***fields"***, que son los componentes que nos permiten mostrar información sobre el ambiente de desarrollo, y se clasifican en:

Formula Fields: Campos donde el valor puede ser obtenido de fórmulas de cálculo y/o operaciones sobre otros campos.

Summary Fields: Campos utilizados para acumular y/o promediar valores según las operaciones que se deseen.

Parameters Fields: Campos que permiten enviar valores al reporte desde una aplicación o entrada del usuario.

SQL Expresión Fields: Campos para ejecutar funciones propias del motor de base de datos que se esté utilizando.

Group Name Fields: Campos utilizados para agrupar la salida del reporte.

Running Total Fields: Similares a los *Summary Fields*, pero éstos pueden ser condicionados.

Special Fields: Conjunto de campos preestablecidos que se utilizan en los reportes, tales como: número de página, total de páginas, fecha de impresión, etc.

4.2. Vistas en la base de datos en SQL Server 2000.

Una parte esencial para trabajar con los reportes de Crystal Reports, es obtener los datos de la base de datos, por lo que en primera instancia se pensaba en ejecutar una sentencia SQL, pero se encontró otra instancia que nos ayuda en gran medida, la cual es una vista.

"Una vista es una tabla virtual que está definida por una consulta que consiste en una instrucción SELECT"[27].

[27] Concepto tomado de "Guía completa de Microsoft SQL Server 2000", Marci Frohock, pp. 380.

Es de gran apoyo, ya que la base de datos es de tipo relacional, donde la vista puede unirlas con apoyo de una instrucción SQL que es muy compleja, para llamarla desde el programa como si fuera una simple tabla.

En la vista se pueden realizar todas las operaciones de una tabla (insert, select, update, delete), esto es de mucha ayuda, pero para el programa sólo se utilizará la instrucción Select, por lo que se bloquearon las demás operaciones para cada uno de los usuarios.

Por lo que al crear la base de datos, se tiene que crear paralelamente la vista, con sus correspondientes restricciones; todo con apoyo de CREATE VIEW para crear tablas virtuales para banco de horas, horarios, extraordinarios y finales.

Veamos el ejemplo de la vista del Bando de Horas:

```
/##### Vista para el Banco de Horas: dbo.VBANCO #####/
CREATE VIEW
        dbo.VBANCO
AS      SELECT
                dbo.BANCOHORAS.NREGBANCO, dbo.BANCOHORAS.CLVUNICAPROF,
                dbo.PROFESORES.PROFESOR, dbo.PROFESORES.RFC,
                dbo.BANCOHORAS.CLVUNICAMAT, dbo.MATERIAS.MATERIA_ABREV,
                dbo.MATERIAS.CLAVE, dbo.BANCOHORAS.CLVUNICACAT,
                dbo.CATEGORIAS.ACATEGORIA, dbo.BANCOHORAS.CURRICULAR,
                dbo.BANCOHORAS.HANTESTEO, dbo.BANCOHORAS.HANTESPRA,
                 dbo.BANCOHORAS.HANTESTOT, dbo.BANCOHORAS.HACTTEO,
                dbo.BANCOHORAS.HACTPRA, dbo.BANCOHORAS.HACTTOT,
                dbo.BANCOHORAS.GRUPO_ANT, dbo.BANCOHORAS.GRUPO_ACT,
                dbo.BANCOHORAS.ELIMINAR, dbo.PROFESORES.FUNCIONARIO,
                 dbo.BANCOHORAS.RUBRO, dbo.BANCOHORAS.TIPO,
                dbo.BANCOHORAS.ASIGHORARIO
        FROM
                dbo.BANCOHORAS
                INNER JOIN              dbo.CATEGORIAS
                ON      dbo.BANCOHORAS.CLVUNICACAT = dbo.CATEGORIAS.CLVUNICACAT
                INNER JOIN              dbo.MATERIAS
                ON      dbo.BANCOHORAS.CLVUNICAMAT = dbo.MATERIAS.CLVUNICAMAT
                INNER JOIN              dbo.PROFESORES
                ON      dbo.BANCOHORAS.CLVUNICAPROF = dbo.PROFESORES.CLVUNICAPROF
GO
```

La cual será llamada cada vez que se requiera como VBanco, donde “V” es el significado de que es una tabla virtual.

4.3. Evaluación y mejoras de los reportes del programa.

La versión anterior del programa Cronos como ya se explicó, está realizada para correr en un ambiente MS-DOS, un sistema operativo que se manejaba para los sistemas de Windows 3.1, 95 y 98, pero con la incorporación de XP se ha ido eliminando poco a poco.

Los reportes que es al que nos vamos a enfocar son los de Cronos, los cuales tienen sus ventajas y desventajas, ya que lograba conjuntar datos de profesores, materias, salones y horarios, pero el manejo de los mismos era complicado, ya que todo se realiza por programación para que usuario vea los reportes sean impresos en pantalla o en impresora según se requiriera, tal como se explicó en el capítulo II.

Lo único que no sabía el usuario era que para agregar un campo, o solo mover partes del reporte, se llevaban días para realizarlo, ya que habría que llamar al campo o ubicarlo en una posición estática que le diera forma.

Otro problema era la inserción de una imagen, donde el principal problema es que se tenía que reconstruir todo el reporte para poder ingresar una imagen, ya que todo lo que se maneja se basa en líneas y texto, tal como se tiene en la figura 4.1.

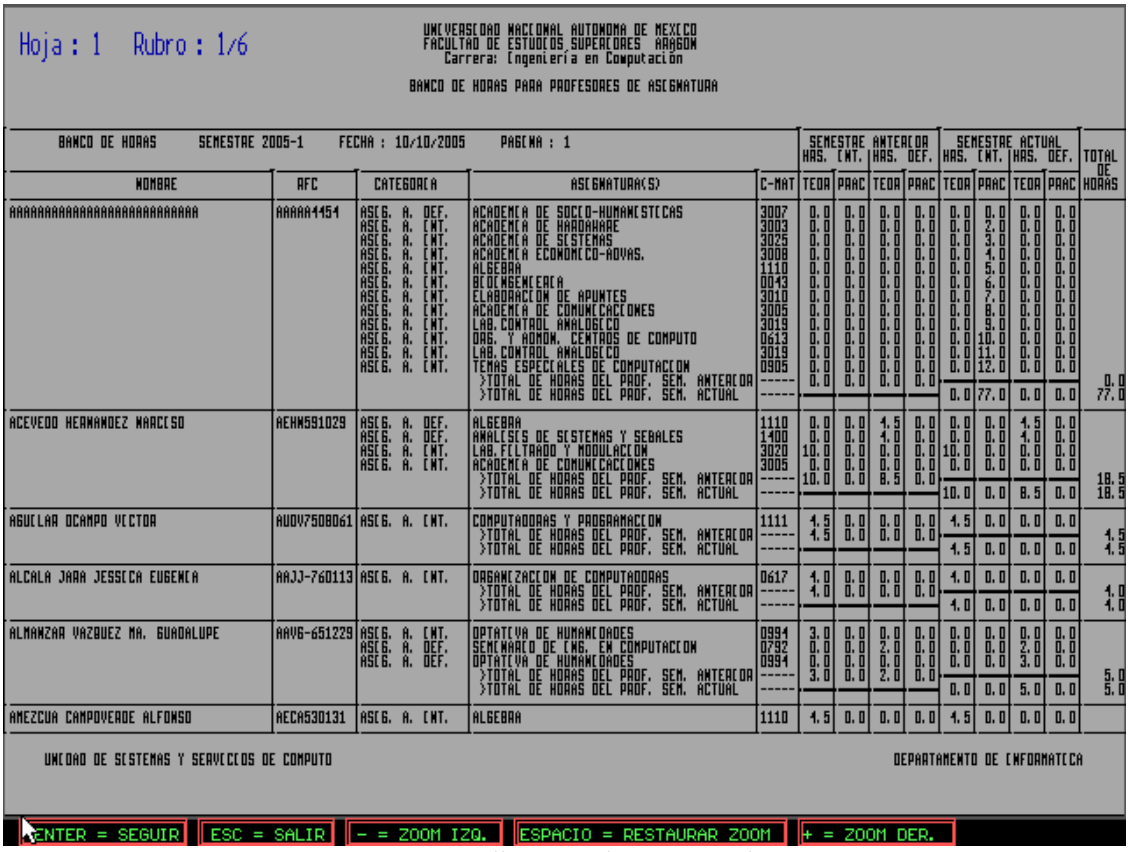


Figura 4.1 Pantalla previa de impresión de Cronos

Al considerar el programa de Wcronos se tenía que crear una interfaz parecida o mejor que la que tenía Cronos, lo cual era un reto importante en especial por que los programas que se manejan en Windows tienen una vista preliminar, por lo que no se podía crear un reporte solamente para imprimir.

Por lo que se tomó la decisión de buscar una interfaz que se incorporara a Visual Basic, la cual fue Crystal Reports versión 8, y ésta no fue la única herramienta considerada para

generar los reportes; sino que se evaluaron otras antes de seleccionar la interfaz; de las otras que existen se pueden mencionar:

- Active Reports de Data Dynamics
- SQL Reporting Services
- Microsoft Offices Automation
- Otros métodos de programación de reportes.

4.3.1. Mejoras en cada uno de los reportes.

El sistema fue mejorado gracias a su programación en Visual Basic con interfaz en Crystal Reports, ya que se da una mejor presentación a los reportes y en la forma de imprimir, además de otras ventajas.

En donde hay que mencionar que la interfaz, supera las ventajas del programa anterior, donde se presenta un menú más amplio de opciones, para cada uno de los reportes a mostrar. Además de que se pueden modificar de forma fácil y rápida en el programa de Crystal Report, tal como se muestra en la figura 4.2.

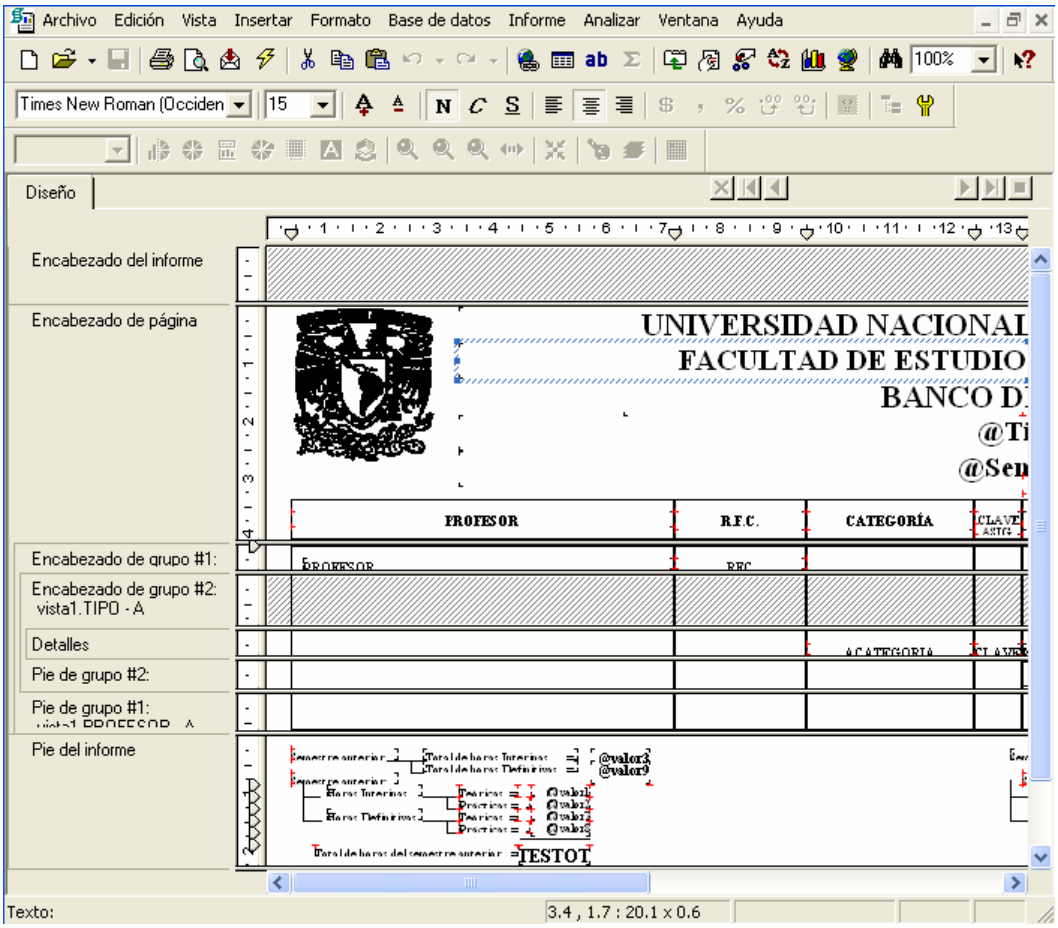


Figura 4.2 Pantalla de diseño de Crystal Reports

4.3.2. Construcción de los reportes.

Para empezar a trabajar con Crystal Reports es fácil insertar los campos para ir generando los reportes, primero hay que definir todos los campos a utilizar, y considerar el tipo y el tamaño del campo, en un archivo *.ttx.

Ya en el programa, se pide la opción de crear un nuevo reporte, por lo que va a pedir la fuente de datos, donde se va a seleccionar la opción ***"Active Data (Field Definitions Only)"***, que quiere decir información activa (definición de datos únicamente). En donde se tiene que indicar donde está el archivo, al seleccionarlo se verá como la pantalla 4.3.

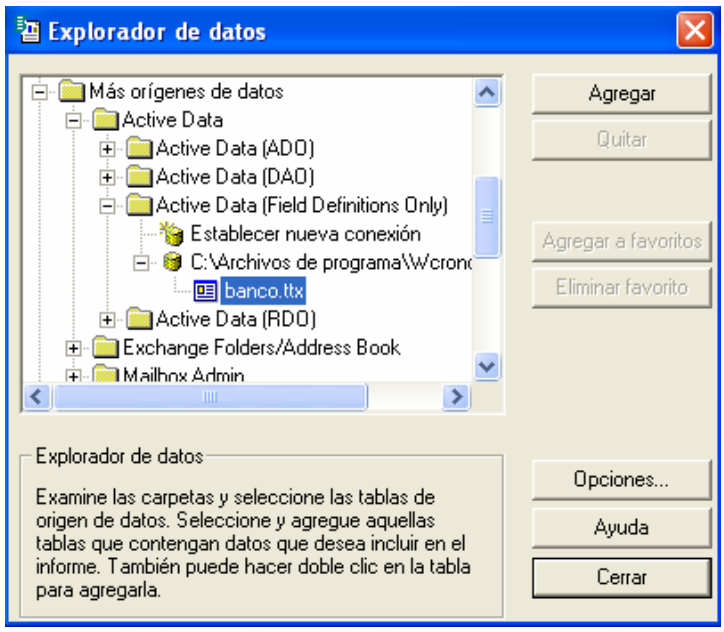


Figura 4.3 Exporador de Fuentes de Datos de Crystal Reports.

Y el programa muestra la pantalla de diseño con un explorador de datos (pantalla 4.4), se pueden seleccionar cada uno de los campos, con lo que ya se puede empezar a generar, uno por uno, los reportes dándole el formato que desee a cada uno de ellos.

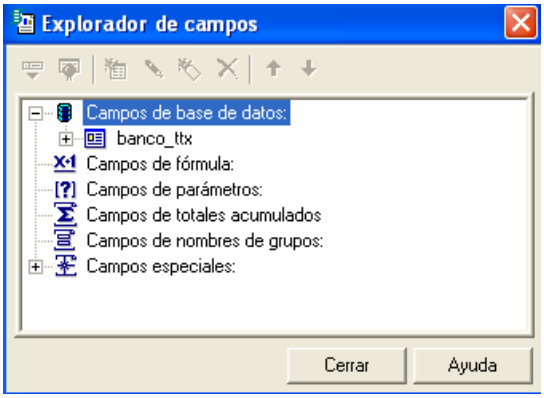


Figura 4.4 Exporador de Datos de Crystal Reports.

Cabe destacar que en cada informe se tiene el mismo formato de encabezado (exceptuando recordatorios), contiene el logo de la UNAM y el de la FES Aragón, con su correspondiente título, el nombre del tipo de reporte, la forma de ordenación de los registros y el semestre correspondiente.

También se considera que en la mayoría de reportes, en el pie de página contendrá "Unidad de Sistemas y Servicios de Cómputo", Página n de m, donde n es el número de página y m el número de páginas totales, además el nombre de "Departamento de Informática".

En los siguientes puntos, se dirá como se construirá cada uno de los reportes del cual se compone Wcronos.

4.3.2.1. Banco de horas.

Como se vió en el capítulo II, los rubros que se manejan en el banco de horas son los siguientes:

- Profesores de Asignatura
- Profesores de Carrera
- Técnicos Académicos
- Ayudantes de Profesor
- Honorarios
- Funcionarios
- Horas de Apoyo Extracurricular
- Resumen General del Banco

Dependiendo la opción elegida, será el tipo de reporte, tal como lo muestra la pantalla 4.5, aunque cabe destacar que se agregó una opción donde se pueden imprimir únicamente los totales de la opción deseada, es decir, sólo la sumatoria de horas totales.

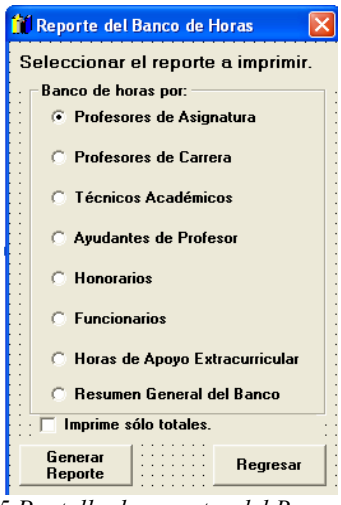


Figura 4.5 Pantalla de reportes del Banco de Horas

En el diseño del reporte, ya creado primeramente, se va a dar una definición por grupos, en el primer grupo se toma en cuenta el campo del profesor para ordenarlo alfabéticamente, posteriormente se tomó un segundo grupo considerando el campo tipo, en donde se van agrupar por dos tipos de horas: definitivas e interinas.

También hay que considerar otro detalle, se van a manejar dos tipos de reportes, con RFC o sin él, por lo que se crean dos formatos de reportes. Con base en esto, se tiene el siguiente reporte como la figura 4.6.

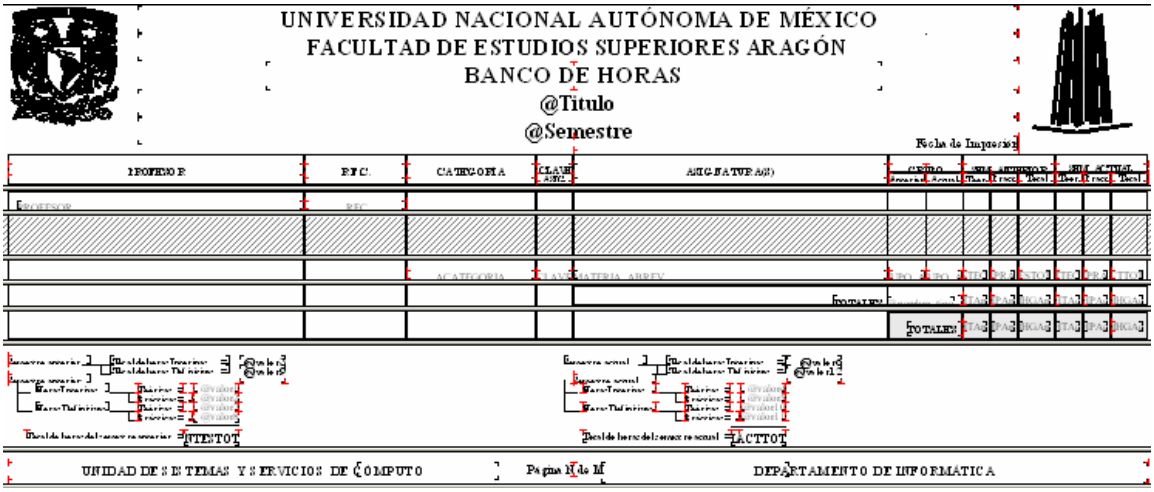


Figura 4.6 Diseño del reporte del Banco de Horas

La parte correspondiente a los totales se compone de sumatorias, que se obtienen antes de desplegar el reporte o se toman de él, de acuerdo a los campos que se manejan.

Lo principal en este reporte es definir la categoría de acuerdo a las que se mencionaron anteriormente, por lo que se usa el siguiente modelo código, donde se elige una parte que es seleccionada por la sentencia CASE:

```
Case Is = "ProfAsig"                    'Profesor de Asignatura (RUBRO 1)
   'Realizo las operaciones para profesores interinos
   CADENASQL = "SELECT Sum(HANTESTEO) AS VALOR1, Sum(HANTESPRA) AS VALOR2,
Sum(HANTESTOT) AS VALOR3, Sum(HACTTEO) AS VALOR4, Sum(HACTPRA) AS VALOR5,
Sum(HACTTOT) AS VALOR6, ELIMINAR, RUBRO, TIPO, FUNCIONARIO FROM VRBANCO GROUP BY
TIPO, RUBRO, FUNCIONARIO, ELIMINAR HAVING ELIMINAR='N' AND RUBRO=1 AND TIPO='I' AND
FUNCIONARIO = 'N'"
   'Call ModulosGlobales.conectarBD2(nombre, acceso, base)
   Set rdotabla2 = rdoconexion.OpenResultset(CADENASQL, rdOpenFowardOnly, rdConcurReadOnly)
   With rdotabla2                       'Establezco un entorno para usar Resulset
      If Not .EOF Then                  'Si no tengo registros continúo
         valorA = !valor1               'Guardo los valores en cadenas
         valorB = !valor2
         valorC = !valor3
         valorD = !valor4
         valorE = !valor5
         valorF = !valor6
      End If
   End With
```

```
        rdotabla2.Close                         'Cierro la base de datos
        Set rdotabla2 = Nothing                 'Conecto la base de datos
        'Realizo las operaciones para profesores interinos
        CADENASQL = "SELECT Sum(HANTESTEO) AS VALOR7, Sum(HANTESPRA) AS VALOR8,
Sum(HANTESTOT) AS VALOR9, Sum(HACTTEO) AS VALOR10, Sum(HACTPRA) AS VALOR11,
Sum(HACTTOT) AS VALOR12, ELIMINAR, RUBRO, TIPO, FUNCIONARIO FROM VRBANCO GROUP BY
TIPO, RUBRO, FUNCIONARIO, ELIMINAR HAVING ELIMINAR='N' AND RUBRO=1 AND TIPO='D'
AND FUNCIONARIO = 'N'"
        Set rdotabla3 = rdoconexion.OpenResultset(CADENASQL, rdOpenFowardOnly, rdConcurReadOnly)
        With rdotabla3                          'Establezco un entorno para usar Resultset
          If Not .EOF Then                      'Si no tengo registros continúo
            valorG = !valor7                    'Guardo los valores en cadenas
            valorH = !valor8
            valorI = !valor9
            valorJ = !valor10
            valorK = !valor11
            valorL = !valor12
          End If
        End With
        rdotabla3.Close                         'Cierro la base de datos
        Set rdotabla3 = Nothing                 'Conecto la base de datos
        If Checkreptotal.Value = 1 Then         'Valido si imprimo con RFC
          'Imprimo el reporte del banco de horas, pero que no tiene RFC
          Set CrystalReport = CrystalApplication.OpenReport("C:\Archivos de
programa\Wcronos\crbanco2.rpt")  'Establezco el reporte variable con el método 'Openrecord'
        Else
          'Imprimo el reporte del banco de horas con todos los campos
          Set CrystalReport = CrystalApplication.OpenReport("C:\Archivos de
programa\Wcronos\crbanco.rpt")  'Establezco el reporte variable con el método 'Openrecord'
        End If
        'Busco los datos para el informe global
        CADENASQL = "SELECT PROFESOR, RFC, ACATEGORIA, CLAVE, MATERIA_ABREV,
GRUPO_ANT, GRUPO_ACT, HANTESTEO, HANTESPRA, HANTESTOT, HACTTEO, HACTPRA,
HACTTOT, TIPO, RUBRO, FUNCIONARIO, ELIMINAR FROM VRBANCO WHERE ELIMINAR = 'N' AND
RUBRO = 1 AND FUNCIONARIO = 'N' AND (HANTESTOT + HACTTOT) <> 0 ORDER BY PROFESOR"
        'Ejecuto la sentencia para llamar a la base de datos
        Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL, rdOpenKeyset, rdConcurReadOnly,
rdAsyncEnable + rdExecDirect)
        If Not rdotabla1.EOF Then               'Si tiene datos
          rdotabla1.MoveFirst                   'Nos ubicamos en el primer registro
          'Cargo la base de datos, Tablas y Objetos de las tablas
          CrystalReport.DiscardSavedData
          Set CrystalDatabase = CrystalReport.Database  'Establezco las constantes
          Set CrystalDatabaseTables = CrystalDatabase.Tables
          Set CrystalDatabaseTable = CrystalDatabaseTables.Item(1)
          'Establezco la primera fórmula
          Set CrystalFormulaField = CrystalReport.FormulaFields.Item(14)
          CrystalFormulaField.Text = "'" & carrera & "'"                'Defino su valor
          'Establezco la segunda fórmula
          Set CrystalFormulaField = CrystalReport.FormulaFields.Item(15)        'Defino su valor
          CrystalFormulaField.Text = "'PROFESORES DE ASIGNATURA - SEMESTRE " & Vsemestre &
"'"
          'Establezco los valores totales de las sumatorias recabadas antes de llamar al reporte
          '**********************CRÉDITOS INTERINO**************************
          Set CrystalFormulaField = CrystalReport.FormulaFields.Item(16)
          CrystalFormulaField.Text = "' " & Format(valorA, "###0.00") & "'"
```

```
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(17)
    CrystalFormulaField.Text = "' " & Format(valorB, "###0.00") & "'"
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(18)
    CrystalFormulaField.Text = "' " & Format(valorC, "###0.00") & "'"
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(19)
    CrystalFormulaField.Text = "' " & Format(valorD, "###0.00") & "'"
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(20)
    CrystalFormulaField.Text = "' " & Format(valorE, "###0.00") & "'"
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(21)
    CrystalFormulaField.Text = "' " & Format(valorF, "###0.00") & "'"
    '**********CRÉDITOS DEFINITIVOS**********
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(22)
    CrystalFormulaField.Text = "' " & Format(valorG, "###0.00") & "'"
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(23)
    CrystalFormulaField.Text = "' " & Format(valorH, "###0.00") & "'"
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(24)
    CrystalFormulaField.Text = "' " & Format(valorI, "###0.00") & "'"
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(25)
    CrystalFormulaField.Text = "' " & Format(valorJ, "###0.00") & "'"
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(26)
    CrystalFormulaField.Text = "' " & Format(valorK, "###0.00") & "'"
    Set CrystalFormulaField = CrystalReport.FormulaFields.Item(27)
    CrystalFormulaField.Text = "' " & Format(valorL, "###0.00") & "'"
    'SetPrivateData establece los datos del reporte de Horarios a un recordset en memoria.
    CrystalDatabaseTable.SetPrivateData 3, rdotabla1
    CrystalReport.Preview                    'Llamo el método para ver el reporte
Else
    Basura = MsgBox("No se tienen datos para imprimir", vbInformation, "Mensaje")
End If
```

Al final, el reporte se mostrará en pantalla y quedará como se muestra en la siguiente figura 4.7., en donde se podrá consultar, imprimir, exportar o enviar por correo.

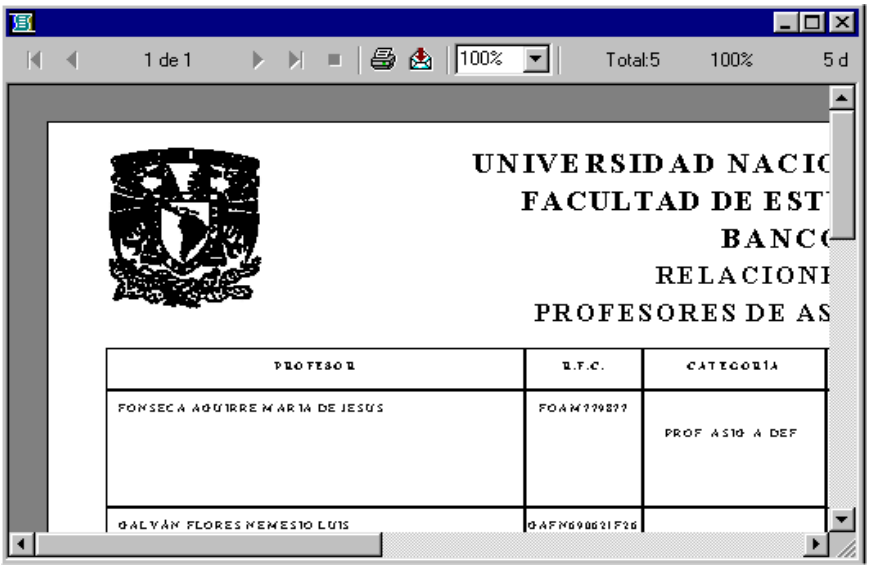


Figura 4.7 Pantalla de reportes del banco de horas

Pero falta considerar un rubro: “Reporte General del Banco”, donde se agrupan todos las sumatorias de acuerdo a las categorías que se tienen en el menú para generar los reportes y otra parte donde se conjuntan todos los totales, para generar un total general, y así definir el total de horas, el cual será práctico para los usuarios.

4.3.2.2. Horarios

El reporte por horarios cambia porque aquí no se manejan categorías, si no que se ordena por grupo, materia, salón y profesor, por lo que se tiene que crear un reporte por cada rubro y se crea el reporte genérico (figura 4.8.), lo único que cambia es la forma de ordenarlo.

Figura 4.8 Diseño del reporte de horarios

En este reporte sobresale una fórmula para darle forma a la hora, en donde sólo se cambia el campo de la hora, esta fórmula se escribe en el diseñador de reportes, la cual se muestra a continuación:

```
if tonumber(Hour({horarios_ttx.HORLUN1})) = 0 then
''
else if minute({horarios_ttx.HORLUN1}) = 0 then
      totext(Hour ({horarios_ttx.HORLUN1}),0) + ":00"
   else if minute({horarios_ttx.HORLUN1}) < 10 then
      totext(Hour ({horarios_ttx.HORLUN1}),0) + ":0"  + totext(Minute({horarios_ttx.HORLUN1}),0)
      else
        totext(Hour ({horarios_ttx.HORLUN1}),0) + ":"  + totext(Minute({horarios_ttx.HORLUN1}),0)
```

Cabe mencionar que así como el anterior reporte, puede imprimirse con o sin RFC, tal como se muestra en el formulario previo a mostrar los reportes, en la figura 4.9.

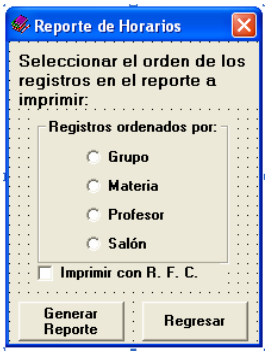


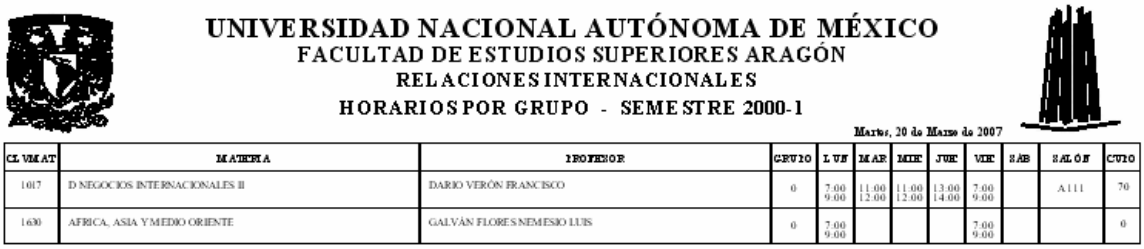
Figura 4.9 Pantalla de selección de reportes de horarios

Por ejemplo, para la opción de “Grupo” se va a ver el código modelo de esta opción, donde se considera si se imprime o no con RFC, como se muestra a continación:

```
Select Case OPCION
    Case Is = "GRUPOS"                          'Valido si imprimo con RFC
      If CheckRFC.Value = 1 Then                'Imprimo con RFC
        'Establezco el reporte variable con el método 'Openrecord'
        Set CrystalReport = CrystalApplication.OpenReport("C:\Archivos de programa\Wcronos\
crhorariosg.rpt")
      Else                                      'No imprimo con RFC
        'Establezco el reporte variable con el método 'Openrecord'
        Set CrystalReport = CrystalApplication.OpenReport("C:\Archivos de programa\Wcronos\
crhorariosg2.rpt")
      End If                                    'Conecto la base de datos
      CADENASQL = "SELECT CLAVE, MATERIA_ABREV, PROFESOR, RFC, GRUPO_ACT,
CLVSALON, HORLUN1, HORLUN2, HORMAR1, HORMAR2, HORMIE1, HORMIE2, HORJUE1,
HORJUE2, HORVIE1, HORVIE2, HORSAB1, HORSAB2, CUPO, POSICION, ELIMINAR FROM
VRHORARIOS WHERE ELIMINAR = 'N' ORDER BY GRUPO_ACT"
      Set rdotabla1 = rdoconexion.OpenResultset(CADENASQL, rdOpenKeyset, rdConcurReadOnly,
rdAsyncEnable + rdExecDirect)               'Si tiene datos
      If Not rdotabla1.EOF Then             'Nos ubicamos en el primer registro
        rdotabla1.MoveFirst             'Cargo la base de datos, Tablas y Objetos de las tablas
                                        'Así que nosotros obtenemos el método: SetPrivateData
        CrystalReport.DiscardSavedData  'Establezco las constantes
        Set CrystalDatabase = CrystalReport.Database
        Set CrystalDatabaseTables = CrystalDatabase.Tables
        Set CrystalDatabaseTable = CrystalDatabaseTables.Item(1)    'Establezco la primera fórmula
        Set CrystalFormulaField = CrystalReport.FormulaFields.Item(1)   'Defino su valor
        CrystalFormulaField.Text = """ & carrera & ""'Establezco la segunda fórmula
        Set CrystalFormulaField = CrystalReport.FormulaFields.Item(2)   'Defino su valor
        CrystalFormulaField.Text = "'HORARIOS POR GRUPO  -  SEMESTRE " & Vsemestre & """
        'SetPrivateData establece los datos del reporte de Horarios a un recordset en memoria.
        CrystalDatabaseTable.SetPrivateData 3, rdotabla1
        'Llamo el método para ver el reporte
        CrystalReport.Preview
      Else
        Basura = MsgBox("No se tienen datos para imprimir", vbInformation, "Mensaje")
      End If
```

Así, para cada una de las otras opciones de ordenamiento, se van a generar códigos parecidos como en el ejemplo anterior. Además de que en el código que se genera, también se presentan las fórmulas para generar el reporte de acuerdo a los datos pedidos; es importante destacar que estas fórmulas son para definir los datos que se van a importar desde la base de datos para mostrar los títulos al reporte final.

Cabe destacar que por cada opción, se generan dos reportes distintos, uno con RFC y otro sin él, por lo que para cada ruta de acceso se van a cambiar, ya que el ordenamiento de los datos lo va a realizar Crystal Reports, el resultado del reporte se muestra en la figura 4.10.



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN
RELACIONES INTERNACIONALES
HORARIOS POR GRUPO - SEMESTRE 2000-1

Martes, 20 de Marzo de 2007

CLVMAT	MATERIA	PROFESOR	GRUPO	LUN	MAR	MIE	JUE	VIE	SAB	SALÓN	CUPO
1017	D NEGOCIOS INTERNACIONALES II	DARIO VERÓN FRANCISCO	0	7:00 9:00	11:00 12:00	11:00 12:00	13:00 14:00	7:00 9:00		A111	70
1630	AFRICA, ASIA Y MEDIO ORIENTE	GALVÁN FLORES NEMESIO LUIS	0	7:00 9:00				7:00 9:00			0

Figura 4.10 Reporte de Horarios ordenados por Grupo

4.3.2.3. Extraordinarios

Para presentar el formulario de selección, se tiene que seleccionar la vuelta, tal como se vió en el capítulo III, en la parte correspondiente a extraordinarios, donde sólo va a tomar en cuenta el campo donde se define la vuelta.

Para el formulario de reportes se tienen que considerar los dos tipos de reportes separados, tal como se muestra en la figura 4.11.

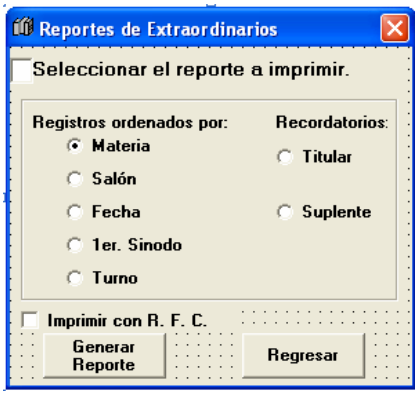


Figura 4.11 Formulario de selección de reportes de Extraordinario

El primer tipo de formato de reporte es como se ordenan los registros en el reporte, de los cuales se tienen las siguientes opciones:

- Materia
- Salón
- Fecha
- 1er sínodo
- Turno

En cuanto al código, es muy similar al que se mostró en los reportes de horarios, lo que cambia es el nombre del reporte con o sin RFC y la instrucción SQL que se muestra a continuación:

CADENASQL = "SELECT CLAVE, MATERIA_ABREV, SINODO1, RFC1, SINODO2, RFC2, VUELTA, CLVSALON, FECHAHORA, TURNO, CLVEXTRA, ELIMINAR FROM VREXTRAS WHERE ELIMINAR = 'N' AND VUELTA = " + Trim(Tnvuelta.Text) + " ORDER BY MATERIA_ABREV"

El resultado del reporte se muestra en la figura 4.12.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN
RELACIONES INTERNACIONALES
EXTRAORDINARIOS SEMESTRE 2000-1 1a Vuelta (Materia)

Martes, 20 de Marzo de 2007

CLAVE	MATERIA	HORA	SALÓN	DIA	FECHA
1007	A POLÍTICA EXTERIOR DE MÉXICO I EA91 ZEPEDA VILLAREAL ARTURO EA91 MARTINEZ GAZCA FRANCISCO	21:00	A112	Domingo	6-Jun-2004

Figura 4.12 Formulario de selección de reportes de Extraordinario

En el formato de extras, lo único que sobresale es la fórmula para mostrar el día que se realizará el examen extraordinario:

WeekdayName (DayOfWeek ({extras_ttx.FECHA}))

El otro formato es distinto a como se había manejado, y éstos son los recordatorios para el profesor titular y para el profesor suplente, en donde se sigue manejando el modelo de reporte de horarios, pero sí cambia, ya que no conservan el encabezado ni el pie de página, sólo se imprime un cuadro donde contiene los extraordinarios que van aplicar, como el que se muestra en la figura 4.13.

EXTRAORDINARIOS SEMESTRE 2000-1 1a Vuelta (TITULAR)
RELACIONES INTERNACIONALES
PROFESOR: DARIO VERON FRANCISCO

MATERIA	FECHA	HORA	SALÓN
AFRICA, ASIA Y MEDIO ORIENTE	11-Jul-2004	11:00	A111
A POLÍTICA EXTERIOR DE MÉXICO I	1-Jun-2005	7:00	A111

SE LES RECUERDA QUE TIENE CINCO DÍAS HÁBILES PARA ENTREGAR CALIFICACIONES Y LLENAR ACTAS.

Figura 4.13 Recordatorios de un extraordinario

Los recordarios van a constar del nombre del profesor, la materia, la fecha, la hora y el salón, además que se les da una nota del recordatorio para entregar las calificaciones y llenado de actas.

4.3.2.4. Finales

La parte de reportes de exámenes finales tiene los mismos tipos que los que se mostraron en los extraordinarios, se componen de dos grupos, el primero es donde se muestran los finales, donde se elige la forma de ordenamiento y el segundo corresponde a los recordatorios que se entregarán a los profesores, tal como se ve en el formulario en la pantalla 4.14.

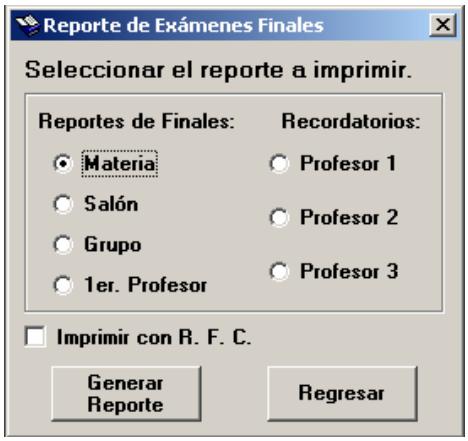
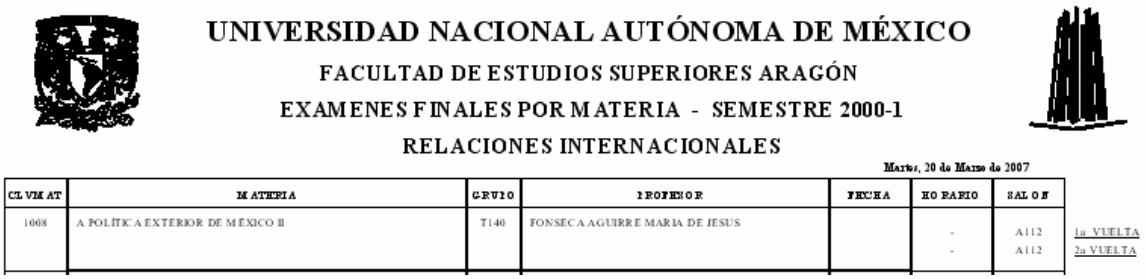


Figura 4.14 Formulario de reportes de finales

El primer tipo de reporte de finales se muestra ordenado por el campo de materia, salón, grupo y el primer profesor, la única fórmula importante es la de la hora, también se ven las dos vueltas que quedan como la pantalla 4.15.



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

EXAMENES FINALES POR MATERIA - SEMESTRE 2000-1

RELACIONES INTERNACIONALES

Martes, 20 de Marzo de 2007

CLAVE MAT	MATERIA	GRUPO	PROFESOR	FECHA	HORARIO	SALON	
1008	A POLITICA EXTERIOR DE MÉXICO II	T140	FONSECA AGUIRRE MARIA DE JESUS		-	A112	1a VUELTA
					-	A112	2a VUELTA

Figura 4.15 Reporte de Finales

En cuanto a los recordatorios de los profesores, su estructura sólo cambiaría de acuerdo a los campos que tienen los finales; se muestra un recordatorio para finales en la pantalla 4.16.

PROFESOR	DARIO VERÓN FRANCISCO		1a VUELTA			2a VUELTA		
CLAVE MAT	MATERIA	GRUPO	FECHA	HORARIO	SALON	FECHA	HORARIO	SALON
0027	APOYO ACADÉMICO (D Y E MAT VIDEO,INFO,IMP Y RADIO)	0	12-Feb-2005	7:00 --7:00	A112	11-Feb-2005	7:00 --	A112
1732	BLOQUES ECONÓMICOS INTERNACIONALES	1625	11-Feb-2005	8:00 --9:30	A111	11-Feb-2005	9:00 --11:00	A111

Figura 4.16 Recordatorios de Finales

4.3.3. Interfase entre el programa de Crystal Reports y Visual Basic.

Todo lo mostrado en este capítulo no sirve hasta que Visual Basic se pueda comunicar con Crystal Reports simplemente por código, sino que requiere apoyo de el sistema de programación en RDO, usando de apoyo ODBC.

Ya que el proceso por el cual Crystal Reports accede a los datos por una instancia ODBC consta de cinco capas (figura 4.17):

CAPA DE BASE DE DATOS: Es la ubicación física donde se encuentra la fuente de datos, en este caso va a estar en el servidor de datos de SQL Server, donde se tienen las tablas y los archivos.

CAPA DE TRADUCCIÓN: Es la capa compuesta de uno o más controladores proporcionados por SGBD (SQL Server y Windows 2000 Server), los cuales permiten que ODBC se comunique con la base de datos.

CAPA DE ODBC (RDO): Es la capa compuesta del conjunto de varios archivos DLL y INI incorporados en el entorno Windows, que actúan como puerta de enlace mediante la cual la base de datos solicita y transmite datos.

CAPA DE TRADUCCIÓN ODBC (RDO): Para ello Crystal Reports usa un archivo para tal fin, éste es P2sodbc.dll (archivo con la biblioteca de vínculos dinámicos para comunicarse con ODBC), el cual es el controlador que transmite los datos desde y hacia ODBC, es decir, sirve de intérprete para obtener la información.

CAPA DE CRYSTAL REPORTS (SQL): Esta parte es la que corresponde a la interacción de el lenguaje SQL que solicita los datos apropiados para el informe y Crystal Reports para poder mostrar al usuario los datos en pantalla. Esto se puede realizar desde Crystal o dentro del programa para facilitar las cosas.

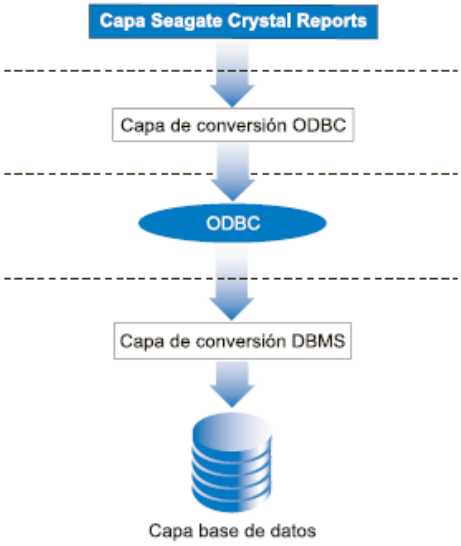


Figura 4.17 Modelo de Acceso de la base de datos de Crystal Reports

La comunicación con Visual Basic, se tiene que realizar manualmente y dar de alta el ActiveX correspondiente a Crystal Reports, para ello se tiene que ir al menú de "Proyecto", posteriormente ir a "Referencias" y seleccionar los siguientes opciones:

- Crystal Report Engine 8 Object 2.5 Library
- Crystal Report 8 Library

- Crystal Report Export
- Crystal Report Control Viewer

Estas opciones se pueden ver en la pantalla 4.18.

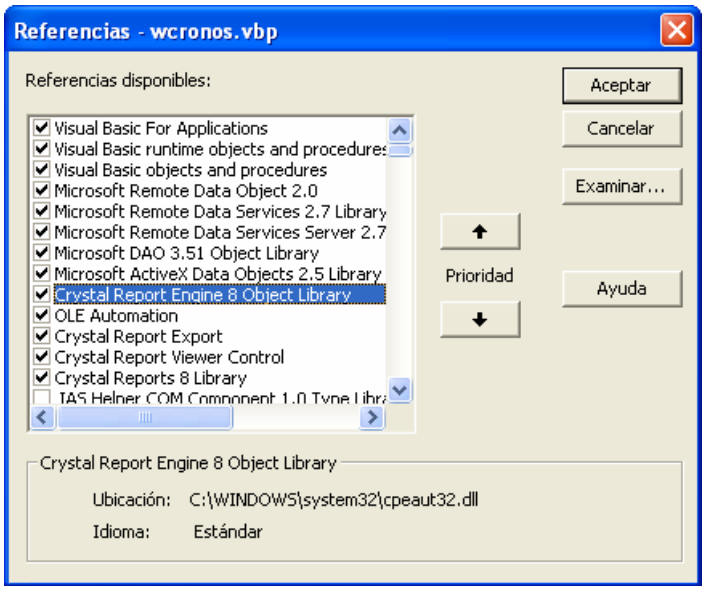


Figura 4.18 Pantalla de Referencias de Visual Basic

4.3.4. Funcionamiento del visualizador de Crystal Reports.

Crystal Reports tiene la ventaja de que se pueden utilizar diversos visores, existen seis tipos de visores de Crystal Report:

- Visor de informes para ActiveX
- Visor de informes para Java mediante Explorador JVM
- Visor de informes para Java mediante complementos Java
- Visor de informes para complementos Netscape
- Visor de informes para HTML estándar con tramas
- Visor de informes para HTML estándar

Pero el visor que se utiliza para el programa está dirigido específicamente para el entorno Windows, en donde se explicará el visor de informes para ActiveX. Las funciones que existen en la barra de herramientas estándar de este visor, tienen las siguientes funciones:

- Ir a primera página
- Ir a página anterior
- Número de página y total de páginas
- Ir a página siguiente
- Ir a última página
- Imprimir

- Exportar
- Porcentaje del tamaño del reporte en pantalla
- Controles del formulario (minimizar, restaurar, maximizar y cerrar)

Pero para usar el visor se tienen que definir las siguientes variables, las cuales son las siguientes:

```
'Pongo los objetos variables los cuales voy a usar en la aplicación
Dim CrystalApplication As CRPEAuto.Application
Dim CrystalReport As CRPEAuto.Report
Dim CrystalDatabase As CRPEAuto.Database
Dim CrystalDatabaseTables As CRPEAuto.DatabaseTables
Dim CrystalDatabaseTable As CRPEAuto.DatabaseTable
Dim CrystalSubreport As CRPEAuto.Report
Dim CrystalFormulaField As CRPEAuto.FormulaFieldDefinition
```

Y para llamarlo sólo se tiene que realizar el siguente proceso, tal como se mostró anteriormente:

```
'Cargo la base de datos, Tablas y Objetos de las tablas, así que nosotros obtenemos el método:
SetPrivateData
CrystalReport.DiscardSavedData
'Establezco las constantes
Set CrystalDatabase = CrystalReport.Database
Set CrystalDatabaseTables = CrystalDatabase.Tables
Set CrystalDatabaseTable = CrystalDatabaseTables.Item(1)
'Establezco la primera fórmula
Set CrystalFormulaField = CrystalReport.FormulaFields.Item(4)
'Defino su valor
CrystalFormulaField.Text = """ & carrera & """
'Establezco la segunda fórmula
Set CrystalFormulaField = CrystalReport.FormulaFields.Item(5)
'Defino su valor
CrystalFormulaField.Text = "'EXTRAORDINARIOS  SEMESTRE " & Vsemestre & "  " +
Trim(Tnvuelta.Text) + "a Vuelta (Fecha)'"
'SetPrivateData establece los datos del reporte de Horarios a un recordset en memoria.
CrystalDatabaseTable.SetPrivateData 3, rdotabla1
'Llamo el método para ver el reporte
CrystalReport.Preview
```

CAPÍTULO

5. PERSPECTIVA DE EVOLUCIÓN DEL SISTEMA.

5.1. Implementaciones en el Programa en un futuro.

El programa cumple con la función que realiza el programa Cronos, pero se pueden implementar más funciones y operaciones para que tenga un mejor rendimiento, facilitando las tareas admisnistrativas de las áreas o carreras que hay en la escuela.

Estas mejoras pueden ser tomadas en cuenta para que integren el programa, o sean tomadas para diseñar otro programa que utilice la base de datos, todo depende la jerarquía de los datos y los usuarios que lo utilicen.

5.1.1. Generación de Estadísticas

El principal fin para desarrollar el programa Wcronos es integrar a todas las áreas y carreras que se imparten en el plantel en una misma base de datos, con varias finalidades, en la que destaca facilitar el uso y servir de apoyo en áreas administrativas que los requieran.

Se desarrolló un programa parecido al programa "Utileria.exe", para enviar la información a las áreas que requieren el apoyo de la información que genera el programa, de las cuales se puede destacar:

- Exportación de datos para Servicios Escolares (Horarios y Extraordinarios).
- Exportación de datos para Secretaría Académica (Kardex).
- Consulta de datos para Departamento de Personal (Banco de Horas).
- Obtención de información para publicar horarios y extraordinarios en Internet.
- Apoyo para la designación de salones (Planta Física).

Estos son puntos que tenía contemplado Cronos, pero es necesario considerar otros procesos que se pueden incorporar posteriormente para mejorarlos.

- Generación de estadísticas solicitadas por la Unidad de Planeación.
- Unificar la base de evaluaciones de profesores.
- Implantación de un nuevo reporte para generar las propuestas de profesores.

Las cuales se mencionan más adelante.

5.1.2. Planta Física

El desarrollo de la Planta Física, es uno de los puntos con mayor relevancia, ya que con el programa anterior se venían arrastrando una serie de problemas, que con Wcronos los están superando, por ejemplo:

- Una tabla única donde se almacena salones, permisos y la planta física.
- Interacción entre permisos y horarios, lo que permita reducir los traslapes entre horarios, en especial entre las carreras.
- Una interfaz sencilla para que el administrador asigne los permisos de manera eficiente.
- Un proceso para generar un reporte general por salón de la ocupación de los salones, ya sea visual o escrito.
- El reporte de planta física sea capaz de aprovechar al máximo la ocupación.
- La información se actualice al momento de obtener los horarios.

Aunque parece muy fácil, se invirtió mucho tiempo en la logística del desarrollo de este punto, desde la integración de permisos y horarios al mismo tiempo, hasta la creación del reporte de planta física (figura 5.1), en donde se muestra traslapes o invasión de horarios.

UNIDAD DE SISTEMAS Y SERVICIOS DE CÓMPUTO
DEPARTAMENTO DE INFORMÁTICA

Página 1 de 59
Semestre 2007-1
20/03/2007

Edificio A01
Salón A111 Nombre: EDIFICIO A1 NIVEL 1 SALON 1

Día\Hora	07:00		08:00		09:00		10:00		11:00		12:00		13:00		14:00		15:00		16:00		17:00		18:00		19:00		20:00		21:00		22:00	
Lunes	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL							CLE	CLE	CLE	CLE	REL	REL	REL	REL	REL	REL	REL	REL		
Martes	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL			CLE	CLE	CLE	CLE	REL	REL	REL	REL	REL	REL	REL	REL		
Miércoles	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL							CLE	CLE	CLE	CLE	REL	REL	REL	REL	REL	REL	REL	REL		
Jueves	REL	REL	REL	REL															CLE	CLE	CLE	CLE	REL	REL	REL	REL	REL	REL	REL	REL		
Viernes					REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL	REL			CLE	CLE	CLE	CLE	REL	REL	REL	REL	REL	REL	REL	REL		
Sábado																																

Figura 5.1 Reporte de la Planta Física.

Pero con el paso del tiempo las necesidades han cambiado, por lo que genera ciertos problemas que no se consideraron al momento de desarrollar el programa, uno de ellos es que hay horarios que se dividen en dos partes, ya que no se tienen los espacios necesarios o porque en ciertas materias tienen horas teóricas en un salón y horas prácticas en un laboratorio.

Otro fue que la carrera de Arquitectura tenía que ocupar sus horarios después de las 22:00 hrs., lo que se arregló fácilmente al aumentar a 23:00 hrs., los permisos y los horarios, pero hay que mencionar que se consideró este caso.

5.1.2.1. Visualizador de espacios ocupados

Su importancia radica en que no sólo la Unidad de Sistemas y Sistemas de Cómputo vea la planta física sino que también las carreras que tengan horarios, para que puedan tener una mejor administración de los espacios en los salones asignados, tal como se tiene en Cronos.

Por lo que se enfocó a desarrollar un formulario para mostrar la ocupación por salón donde se muestra la información de los horarios propios de la carrera, cuando sea ocupado por otra área y cuando no tenga permiso para ingresar un horario a un salón.

Todo se realizó por colores:

COLOR	SIGNIFICADO
Negro.	Sin permiso para ingresar un horario a un salón.
Blanco.	Con permiso de ingresar un horario y con espacio libre.
Azul.	Horario asignado y con permiso.
Amarillo.	Horario con traslape (esto se puede dar en carreras que tengan a más de un profesor en un mismo salón como Diseño Industrial y Arquitectura).
Rojo.	Horario asignado a otra área.

Toda esta información la toma de la tabla de la planta física y de los permisos, se solicita por salón, para evitar que se genere mucho tráfico en la red. El resultado de esto se muestra en la figura 5.2.

Figura 5.2 Formulario de consulta gráfica por salón.

5.1.3. Evaluaciones de Profesores.

Otra parte que se puede integrar a la base de datos y no al programa Wcronos, serían las evaluaciones de profesor, pero la información que integra Wcronos es considerada administrativa, solamente se autoriza a los jefes de carrera o los secretarios técnicos, por que en la evaluación de profesores la captura de cuestionarios la realizan los ayudantes de profesor y los prestadores de servicio social.

Aunque sea otro programa, la información dependerá de los horarios, lo cual permitirá escribir dos veces la misma información, una para horarios y otra para las evaluaciones de profesor, en donde se tendrá una dependencia.

El programa se puede encausar a dos caminos, el primero para la captura de cuestionarios realizados por los alumnos o el segundo donde los alumnos pueden ingresar la información directamente y se almacena en un servidor, cualquier método tiene que tener el apoyo de las bases de Wcronos, directa o indirectamente.

5.1.3.1. Captura de los cuestionarios de las evaluaciones de profesor

Este punto sólo se tomará si se van a capturar los cuestionarios, se puede considerar el programa que se usa para ello, Captura 3.0, en donde se tienen que llenar las bases de evaluaciones manualmente, unidos por otras bases, la de materias, grupos, y profesores, tal como se muestra en la figura 5.3.

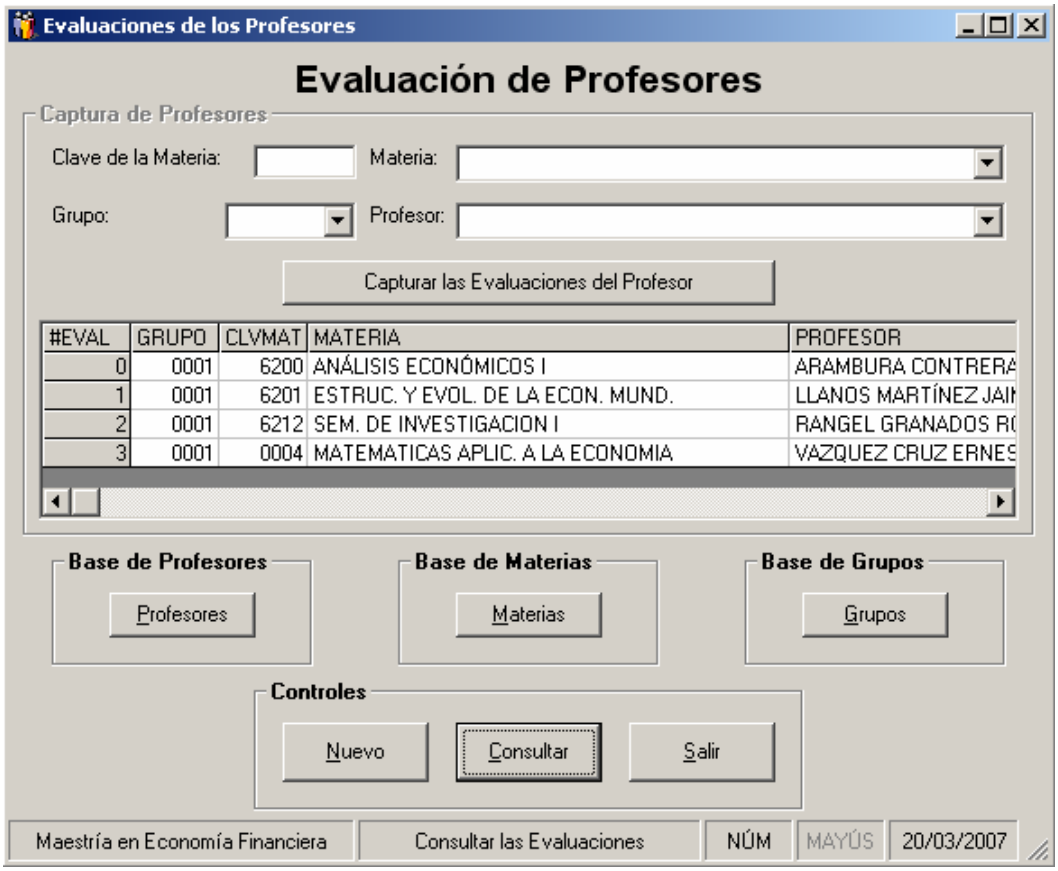


Figura 5.3 Pantalla de captura de evaluaciones de profesores

Toda la información generada puede omitirse, ya que puede ser tomada de las base de datos de los horarios, siempre y cuando los usuarios actualicen la información.

Los usuarios sólo seleccionarán qué evaluaciones se van a usar y con base en las evaluaciones, en donde se van a capturar las respuestas que escriben en los cuestionarios, tal como se muestra en la figura 5.4.

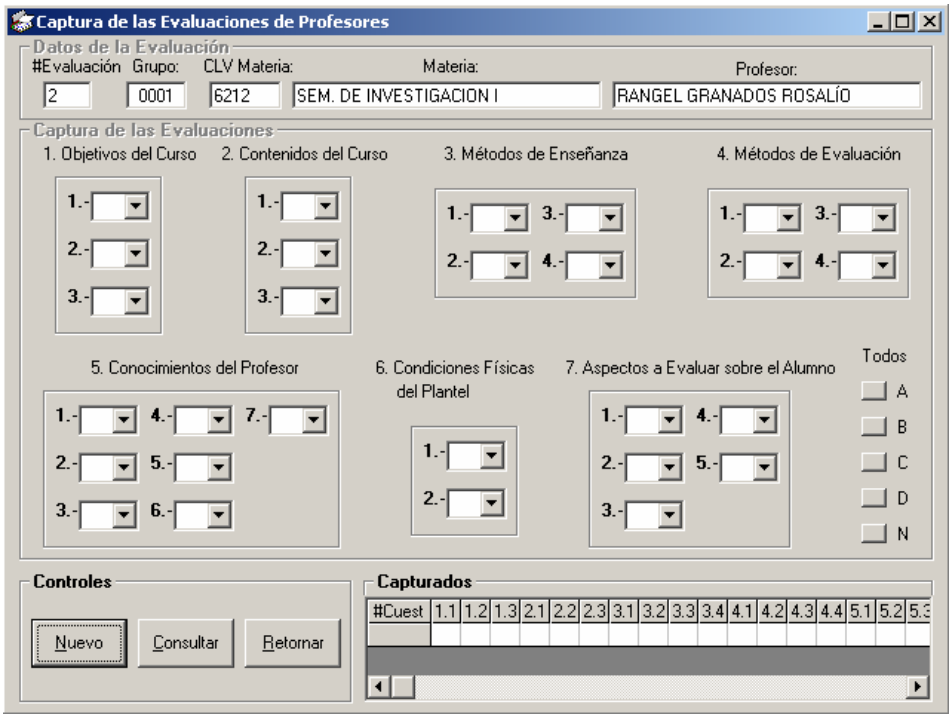


Figura 5.4 Pantalla de captura de cuestionarios de evaluaciones de profesor

5.1.3.2. Captura en línea de los cuestionarios de las evaluaciones de profesor

Una nueva modalidad que se plantea implantar en la Facultad, es que los alumnos puedan evaluar a cada uno de sus profesores a través de Internet y con ello se omite el uso del papel y de forma automática se introduce la información a una base de datos.

La forma de validación para que los alumnos accedan al programa, es por medio de su número de cuenta y el número de folio de su tira de materias como se ve en la pantalla de inicio:

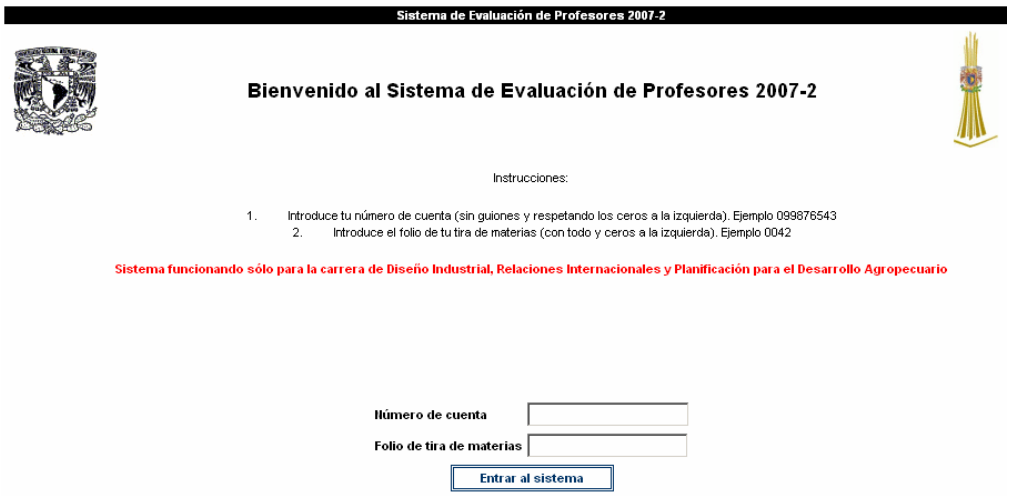


Figura 5.5 Pantalla de inicio de evaluaciones en línea de profesores.

Al momento de ser autentificado aparece otra pantalla en donde muestra una lista, con cada una de las materias inscritas por el alumno, como se ve en la figura 5.6.

Sistema de Evaluación de Profesores 2007-2

Bienvenido
TORRES FERNANDEZ SARAHYD DEL CARMEN
Planificación para el Desarrollo Agropecuario

Menú
Evaluar Cursos

Salir del sistema

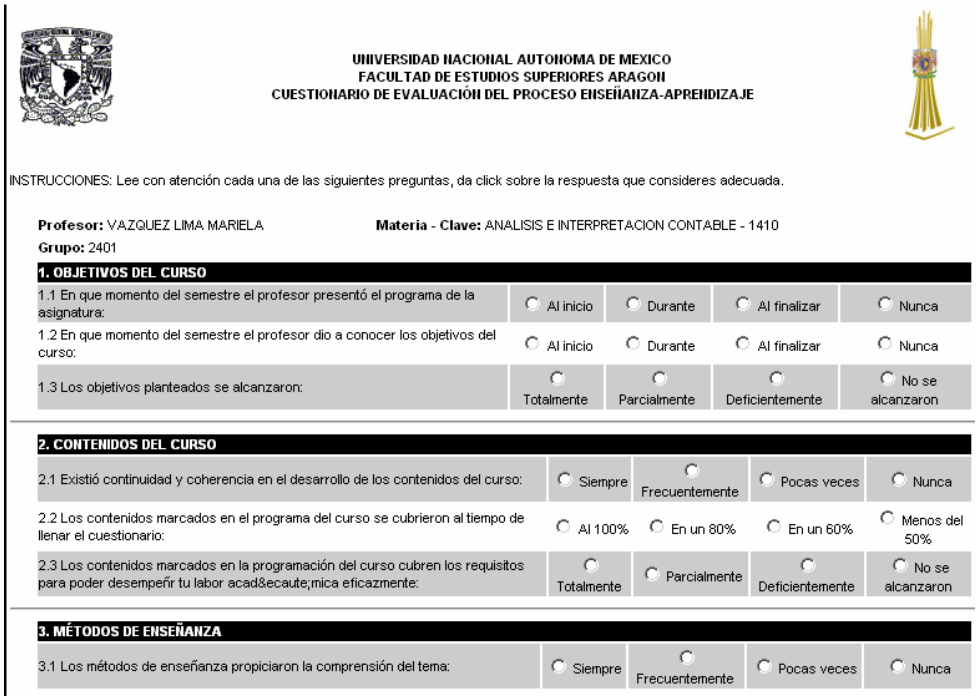
Cursos inscritos

De click en el botón "evaluar" para llevar a cabo la evaluaión del curso

Grupo	Clave	Nombre de materia	Profesor	Evaluar
2401	1410	ANALISIS E INTERPRETACION CONTABLE	VAZQUEZ LIMA MARIELA	Evaluar
2401	1411	DISEO DE LA INVESTIGACION	REYES COUTURIER TEOFILO	Evaluar
2401	1412	ECONOMIA REGIONAL	ANTOPIA ORTA JORGE	Evaluar
2401	1413	PLANEACION ESPACIAL Y FISICA	LOERA ESPARZA JOSÉ ALFREDO	Evaluar
2401	1414	PLANEACION ESTRATEGICA	SOLIS SANCHEZ BERNARDO	Evaluar
2401	1415	PRODUCCION AGRICOLA	CEDILLO PORTUGAL EUGENIO	Evaluar
2401	1416	PRODUCCION ANIMAL	FLORES MORENO PEDRO	Evaluar
2401	1417	SOCIEDAD Y POLITICA DEL MEXICO ACT.	ROMERO CERVANTES MARÍA DEL ROSARIO	Evaluar

Figura 5.6 Lista de materias de evaluaciones en línea de profesores.

Al seleccionar cualquier elemento de la lista se mostrará el cuestionario de evaluación de profesores (figura 5.7), el cual debe ser llenado en su totalidad para ser grabado, de caso contrario aparecerá la primera pregunta que no se haya evaluado.



UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
FACULTAD DE ESTUDIOS SUPERIORES ARAGON
CUESTIONARIO DE EVALUACIÓN DEL PROCESO ENSEÑANZA-APRENDIZAJE

INSTRUCCIONES: Lee con atención cada una de las siguientes preguntas, da click sobre la respuesta que consideres adecuada.

Profesor: VAZQUEZ LIMA MARIELA **Materia - Clave:** ANALISIS E INTERPRETACION CONTABLE - 1410
Grupo: 2401

1. OBJETIVOS DEL CURSO				
1.1 En que momento del semestre el profesor presentó el programa de la asignatura:	Al inicio	Durante	Al finalizar	Nunca
1.2 En que momento del semestre el profesor dio a conocer los objetivos del curso:	Al inicio	Durante	Al finalizar	Nunca
1.3 Los objetivos planteados se alcanzaron:	Totalmente	Parcialmente	Deficientemente	No se alcanzaron

2. CONTENIDOS DEL CURSO				
2.1 Existió continuidad y coherencia en el desarrollo de los contenidos del curso:	Siempre	Frecuentemente	Pocas veces	Nunca
2.2 Los contenidos marcados en el programa del curso se cubrieron al tiempo de llenar el cuestionario:	Al 100%	En un 80%	En un 60%	Menos del 50%
2.3 Los contenidos marcados en la programación del curso cubren los requisitos para poder desempeñr tu labor acad&ecaute;mica eficazmente:	Totalmente	Parcialmente	Deficientemente	No se alcanzaron

3. MÉTODOS DE ENSEÑANZA				
3.1 Los métodos de enseñanza propiciaron la comprensión del tema:	Siempre	Frecuentemente	Pocas veces	Nunca

Figura 5.7 Cuestionario de evaluaciones en línea de profesores.

Para el funcionamiento de éste proceso se tuvo que unificar el contenido de dos bases, la base de inscripciones de Servicios Escolares y la base de Wcronos.

Para lograr esto se debe que realizar un proceso donde se tienen que actualizadar ambas bases antes poner en funcionamiento el programa en línea, para evitar que se generen errores en la captura de la base de datos.

5.1.3.3. Generación de reportes y estadísticas.

Si se desarrolla el programa de captura, paralelamente se tiene que crear el programa para generar los reportes y estadísticas, donde se tiene que considerar que los procesos de calificación sean los adecuados, a menos que los modifiquen los miembros del consejo técnico de la escuela.

Los reportes a considerar son tres tipos:

- Calificaciones globales ordenadas por profesor alfabéticamente.
- Calificaciones globales ordenadas por promedio.
- Evaluaciones individuales de los profesores donde se muestren los promedios de los cuestionarios.

5.1.4. Propuestas de Profesores

Un proyecto para el mejoramiento de Wcronos y que ayudaría a reducir el tiempo a las áreas o carreras es la generación de propuestas de contratación, en un principio no se consideró, ya que requiere agregar campos a la base de datos, además de varios detalles que tiene que valorar con el Departamento de Personal.

Debido a los tiempos, esta parte no se desarrolló pero se tiene considerada, esto hasta que se tengan integrado el programa en cada área, el principal problema es que no se tienen contemplados todos los horarios (sólo se consideró a las licenciaturas).

Este es un proyecto donde se debe trabajar con cuidado, ya que tiene muchos requerimientos, como son los datos de los profesores, que pueden variar constantemente, además de que la información se establezca con claridad, ya que la propuesta será firmada por el profesor, el jefe de carrera y la dirección. Será enviada al Departamento de Personal como apoyo.

5.2. Beneficios Generados

Al momento de realizar el análisis en el capítulo II, no sólo se pensó en cambiar de sistema operativo MS-DOS a Windows, y cambiar la base de datos codificada a un Sistema Gestor de Bases de Datos (SQL Server), si no implementar nuevas funciones y tratar de que el sistema tenga una interfaz amigable para el usuario.

Se mostrarán los alcances logrados a lo largo del período de instauración del sistema, la estandarización de la base de datos y la exportación de datos.

5.2.1. Beneficios al integrar Wcronos

Los beneficios alcanzados al integrar el programa en cada una de las áreas que hay en la Facultad de Estudios Superiores Aragón, son:

- Se mejoró el sistema de seguridad que tiene el programa, desde el servidor hasta la autentificación de los usuarios.
- Se agregó la captura de materias al programa, la cual puede ser modificada al asignar permisos de escritura.
- Se mejoró la captura de los profesores, al agregar más campos, para poder ser exportada la información a servicios escolares.
- Mejoramiento para el proceso de captura del banco de horas.
- Se tiene un reporte general por cada rubro del banco de horas, donde al final se tiene la suma total.
- Unión entre el banco de horas y los horarios, evitando la doble captura de información.
- Se ingresaron gráficos y se pueden utilizar las impresoras que usa Windows.
- Se tiene un control más estricto en la asignación de salón, evitando traslapes.

5.2.2. Estandarización de las bases de datos.

El principal causante de la generación del programa, es la forma de exportar la información a las diversas áreas, ya que cada una depende de los sistemas que tiene cada dependencia que hay en la UNAM, los cuales van actualizando la información.

Por ejemplo, se tenía que captura la información de los horarios en dos programas en Cronos para publicar la información en Internet y enviarla a personal, aparte de realizarlo en NEW para transferir a Servicios Escolares.

Por lo que se puede establecer que hubo análisis con cada base de datos y a partir de esto se diseñó la base, considerando cada uno de los campos. Se tienen las siguientes relaciones mostradas en la siguiente tabla 5.1.

	Servicios Escolares	Departamento de Personal	Unidad de Planeación	Informática	Secretaría Académica
Materias	*	*		*	*
Profesores	*	*	*	*	*
Banco de horas		*			
Horarios	*		*	*	*
Extraordinarios	*			*	
Finales				*	

Tabla 5.1 Relación entre áreas administrativas con la base de datos

Por respeto a las áreas administrativas no se mostrará un desglose más completo de la información.

5.3. Bases para crear una aplicación dentro del servidor.

La plataforma utilizada para Windows 98, 2000 y XP no puede ser eterna, al igual de Cronos, tiene cierto tiempo de vida, por lo cual se realizará un cambio cuando se tenga otro sistema operativo, que pueda alterar el funcionamiento del mismo.

Por ejemplo, Cronos es un programa que se creó con un lenguaje de programación de 16 bits, por lo que para que compilara a 32 bits dañaría el funcionamiento de las bases, pero con WCronos no se afectará a menos que se cambie el Sistema Gestor de Base de Datos, ya que el programa administrador tendría que ser compatible con las características mostradas.

Se puede retomar el camino de tener de ya tener el programa en un servidor (SGBD de tres capas), tomando como base las características y detalles que se tiene en esta tesis, para la posible construcción de otro programa, cuando éste ya se absoleto, aunque parece muy sencillo realizarlo existen ciertas variables o políticas que pueden generar dificultades.

Hay que tener una buena planeación, análisis, consulta de las áreas y considerar el material con que se cuenta, para aprovecharlo al máximo.

Tal vez el costo de los programas originales y el diseño del programa Wcronos, llevaron a que el material que se tenía se aprovechara, esto se ve reflejado en su buen funcionamiento, tal vez el programa no cumplió al 100% las necesidades, pero ha facilitado el trabajo de muchas áreas.

Diseñar un sistema actualmente es muy complejo, pero no imposible, se tiene que realizar una buena planeación para llevarlo a cabo, elaborarlo es difícil, ya que requiere apoyo de un equipo:

1. Analista para analizar y establecer los tiempos de cada parte.
2. Informes y estadísticas de lo que se va a realizar en el programa.
3. Obtención de comentarios, opiniones de los usuarios que van a utilizar el programa.
4. Instalación del Servidor.
5. Instalación del Servidor de bases de datos.
6. Administrador de sistema operativo del servidor.
7. Administrador de la base de datos.
8. Programador o programadores que realicen los módulos de programas.
9. Establecer un tiempo de pruebas.
10. Establecer las nuevas modificaciones.
11. Análisis de otros procesos que se puedan integrar.
12. Evitar que al realizar un proceso de captura se den muchos clic, ya que pueden confundir al usuario.

El contenido de esta tesis, se puede considerar para desarrollar e implantar un sistema dentro de un servidor, teniendo así, tanto la base de datos como la aplicación para ser usada vía WEB, pero en especial hay que tomar en cuenta todos los elementos de seguridad con que se cuenta para evitar que existan intrusos en el sistema.

Existe una especial observación para el siguiente sistema, ya que se presentó un inconveniente durante el proceso de instalación; se tiene que considerar que en un horario se puedan manejar hasta dos espacios de trabajo, uno para la teoría y otro para la práctica, en especial para aquellas carreras que llevan talleres.

CONCLUSIONES

El proceso de investigación, análisis y desarrollo del programa de administración académica “Wcronos” para la Facultad de Estudios Superiores Aragón, ha cumplido con el objetivo de sustituir a su programa antecesor, mejorar en cada una de sus funciones y de integrarse de forma sencilla a las áreas del plantel.

Ya está implantado en un sistema cliente-servidor de dos capas, la primera está en un servidor de datos que funciona como un Sistema Gestor de Base de Datos donde se almacenará toda la información, la segunda se refiere a la capa de aplicación que es el programa “Wcronos”, se instalará en cada uno de los usuarios que utilicen el programa.

Desde el momento de la planeación, se tuvieron que decidir cada uno de los componentes para poder implantarlo, qué camino se tenía que seguir, y considerar las etapas que conllevan a implantar un sistema que está compuesto de cuatro partes.

La primera es el servidor, el cual se tiene que instalar y administrar lo mejor posible, considerando cada uno de los posibles problemas de seguridad que tiene un servidor con un sistema operativo Windows. La segunda comprende la implantación del Sistema Gestor de Base de Datos, donde se tiene el modelo de la base de datos creada.

La tercera es la aplicación que tiene las bases del programa Cronos para su funcionamiento e integración con cada una de las áreas que lo venían trabajando, además de la integración de otras nuevas áreas.

En la aplicación no se creó cada módulo de una forma distinta, sino que se estableció un modelo de programación para captura y otro para los reportes de cada módulo que compone el programa (Banco de Horas, Horarios, Extraordinarios, Finales). Las diferencias se mostraron en la tesis.

En lo que respecta a la cuarta etapa es un programa de exportación para enviar a las diversas áreas administrativas, pero por razones de seguridad no se muestran en esta tesis.

La meta que implica desarrollar este proyecto se cumplió, pero los tiempos cambian, por lo que cada día surgen nuevas y mejores tecnologías, no sólo hay que pensar en que todo termina aquí, sino que con el tiempo todo esto será sustituido por otro proyecto que supere los alcances de este sistema.

BIBLIOGRAFÍA

Libros

LAWRENCES., Orilia. **Las Computadoras y la Información: Introducción.** Escalona G. Roberto (Trad.), 2da. Edición. México: McGraw-Hill, 1987. 772 págs.

COHEN Karen, Daniel. **Sistemas de Información para la Toma de Decisiones.** 2da. Edición. México: McGraw-Hill, 1996. 266 págs.

MURDICK, Robert G. y ROSS, Joel E Coaut. **Sistemas de Información basados en Computadoras para la Administración Moderna.** Meza Nieto, J. (Trad.). México: Diana, 1974. 640 págs.

CASTAÑO, Adoración de Miguel, **Fundamentos y Modelos de Bases de Datos.** España: RA-MA. 1997. 360 págs.

SIMMONS, Curt. **Guía Avanzada Microsoft Windows 2000 Server Configuración.** 1ra. Edición, España: Prentice Hall Iberia, 2000. 442 Págs.

SMITH, Curtis y AMUNDSEN, Michael. **Aprendiendo Programación de base de datos con Visual Basic 6 en 21 días.** Garza M., A. David, México: Prentice Hall Hispanoamericana, 1999. 912 págs.

FROHOCK G. Marci, et al. **Guía Completa de Microsoft SQL Server 2000.** Barco M. Angel, et al. (Trad.). España: McGraw-Hill, 2000. 1020 págs.

CEBALLO S., Fco. Javier, **Enciclopledia de Microsoft Visual Basic 6.** 1ra Edición, México: Alfaomega Grupo Editor, 2000. Edición original España: RA-MA Editorial. 1060 págs.

Tesis

MANZO Salazar, Ernesto
Sistema automático de consulta de horarios vía Internet para la UNAM Campus Aragón
Tesis de Licenciatura
ENEP ARAGON, 2000, UNAM, México

Manual

GARCÍA Hernández, Ing. Pedro Gabriel
Diplomado en Diseño e Implantación de Bases de Datos en Microsoft SQL Server
Junio 2003, México.

Manual Técnico de Cronos
Departamento de Informática
ENEP ARAGON, 1997, UNAM, México

Manual de Usuario en Español
Seagate Crystal Reports™ 8
Seagate Inc., 2000, EEUU

Link de Internet (Estos pueden quedar fuera de servicio)

Manuales obtenidos a través de:

www.programacion.net

- RDO (Remote Data Objects)

- L.I. Jesús Ramón López Domínguez
  Por que usar Microsoft Remote Data Object 2.0 y no DAO o cualquier otro
  Sistemas Culiacán - Redes y Comunicaciones, Cervecería Cuauhtemoc Moctezuma
  Febrero 2001, Culiacán Sinaloa, México

- José Hernández Orallo, (jorallo@dsic.upv.es)
  La Disciplina de los Sistemas de Bases de Datos.
  Historia, Situación Actual y Perspectivas.
  Dep. De Sistemas Informaticos y Computación
  Universidad Politécnica de Valencia, mayo 2002, España

GLOSARIO

Acceso Aleatorio: Se llama de acceso aleatorio porque el procesador accede a la información que está en la memoria en cualquier punto sin tener que acceder a la información anterior y posterior.

Acceso Secuencial: Método usado para acceso a datos, moviéndose a través de los mismos en el orden en que fueron almacenados.

AMD (Advanced Micro Devices, Inc.): Segunda compañía mundial productora de microprocesadores y circuitos electrónicos.

ANSI (American National Standards Institute): El Instituto Nacional Estadounidense de Estándares es una organización sin ánimo de lucro que supervisa el desarrollo de estándares para productos, servicios, procesos y sistemas en los Estados Unidos.

API (Programming Interface - Interfaz de Programación de Aplicaciones): Conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta librería para ser utilizado por otro software como una capa de abstracción.

COBOL (COmmon Business Oriented Language, Lenguaje Común Orientado a Negocios): Fue creado con el objetivo de generar un lenguaje de programación universal que pudiera ser usado en cualquier computadora, es decir, a la llamada informática de gestión.

CODASYL (Conference on Data Systems Languages): Consorcio de industrias informáticas con el objeto de regular el desarrollo de un lenguaje de programación estándar utilizada en multitud de computadoras, resultando de aqui el lenguaje COBOL.

DB2 (Database 2): Marca comercial, propiedad de IBM, bajo la cual se comercializa el sistema de gestión de base de datos. El modelo que utiliza es el jerárquico en lugar del modelo relacional que otros manejadores (Sistemas de Gestión) utilizan.

Desencriptar: Proceso inverso de encriptar se denomina decrypt (desencriptar).

DS (Data System): Sistema de datos.

DNS (Domain Name System): Base de datos distribuida y jerárquica que almacena información asociada a nombres de dominio en redes como Internet. Los usos más comunes son la asignación de nombres de dominio a direcciones IP y la localización de los servidores de correo electrónico de cada dominio.

Encriptar: Proteger archivos expresando su contenido en un lenguaje cifrado. Los lenguajes cifrados simples consisten en la sustitución de letras por números mediante complejos algoritmos. Los datos encriptados suelen llamarse "texto cifrado".

EOF (End-Of-File, fin de fichero en inglés): Condición en un sistema operativo por la cual ninguna otra información puede ser leida de la fuente de datos. Tal fuente de datos es normalmente un fichero o stream.

Field: Espacio reservado para un dato a ocupar, campo.

IBM (International Business Machines): Empresa que fabrica y comercializa hardware, software y servicios relacionados con la informática.

ISO (International Organization for Standardization o Organización Internacional para la Estandarización): Organización internacional no gubernamental, compuesta por representantes de los organismos de normalización (ONs) nacionales, que produce normas internacionales industriales y comerciales.

Interfaz: Parte del programa informático que permite el flujo de información entre varias aplicaciones o entre el propio programa y el usuario. En sentido amplio, puede definirse interfaz como el conjunto de comandos y métodos que permiten la intercomunicación del programa con cualquier otro programa o elemento interno o externo.

Maxicomputadores: Servidores especiales que se utilizan para procesar grandes cantidades de informaciòn en tiempos relativamente pequeños.

Minicomputadores: Servidores de bajo rendimiento.

ORACLE: Sistema de gestión de base de datos relacional (o RDBMS por el acrónimo en inglés de Relational Data Base Management System), fabricado por Oracle Corporation.

SQL (Structured Query Language o Lenguaje de Consulta Estructurado): Lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas.

Traslape: Se define como las operaciones diversas de una base de datos distribuida en la que se repiten registros.

Tuplas: Se denomina a una fila de una tabla de una base de datos, es un elemento del conjunto que forman las relaciones.

Viewer: Pequeño visor para supervisar un proceso.