UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

Facultad de Ingeniería

Sistema para el control de la información del personal académico de la División de Posgrado de la Facultad de Medicina

T E S I S

Que para obtener el título de

INGENIERO EN COMPUTACION

p r e s e n t a

DANIEL SOLIS GUILLEN

DIRECTOR DE TESIS: ING. MA. DEL CARMEN MALDONADO SUSANO

Ciudad Universitaria **1998**

ÍNDICE

AGRADECIMIENTOS.....viii

OBJETIVO.....ix

CAPÍTULO 1.DEFINICIÓN DEL SISTEMA. 1

1.1 INTRODUCCIÓN. 1

1.2 JUSTIFICACIÓN DEL SISTEMA. 3

- Definición del sistema. 3

1.3 ALCANCES Y RESTRICCIONES DEL SISTEMA. 4

- Alcances. 4
- Restricciones. 7

1.4 PROPUESTA Y SELECCIÓN DE UNA SOLUCIÓN. 8

- Acercamiento. 8
- Conceptos de Bases de Datos. 10
- Modelo. 11
- Modelo de Datos. 11
- Modelos lógicos basados en objetos. 11
- Modelos lógicos basados en registros. 11
  - ➢ Modelo Relacional. 12
  - ➢ Modelo de Red. 12
  - ➢ Modelo Jerárquico. 12
- Modelos físicos de datos. 12
- Enfoque decisivo. 12
- Modelo Entidad Relación o Entidad Asociación. 13
  - ➢ Componentes. 13
  - ➢ Diseño con el enfoque Entidad - Relación. 14
- Diseño de bases de datos relacionales. 15
- Modelo Relacional. 15
  - ➢ Objetivos del modelo relacional. 16

- Componentes del Modelo Relacional. 16
- Formas Normales. 17
  - Primera Forma Normal. 17
  - Segunda Forma Normal. 17
  - Tercera Forma Normal. 18
- Aplicación de los modelos Entidad – Relación y Relacional. 18
- Software de almacenamiento y control de información. 19
- Lenguaje de Definición de Datos. 19
- Lenguaje de Manipulación de Datos. 20
  - Lenguaje Estructurado de Consulta (SQL.) 20
- Lenguaje de Control de Datos 21
- DBMS. 21
  - Seguridad. 21
  - Control de concurrencia. 22
  - Respaldo y recuperación. 22
  - Utilerías de mantenimiento. 22
  - Catálogo. 23
  - Índices. 23
  - Vistas 23
  - Optimización 24
  - Herramientas para monitoreo 24
  - Administración de bases de datos. 24
  - Esquemas. 24
  - Funcionalidad de un DBMS. 25
    - Desventajas 26
  - Oracle. 26
- Software de desarrollo. 27
  - dBASE IV. 27
  - FoxPro. 27

- ➢ Paradox. 28
- ➢ Access. 28
- ➢ Clipper. 28
- ➢ Lenguaje C. 29
- Confrontación. 29
- Teoría de Herramientas CASE. 30

CAPÍTULO 2. ANÁLISIS Y DISEÑO DEL SISTEMA. 32

2.1 ANÁLISIS DEL SISTEMA. 32

- Diseño. 32
- Reunión de información. 33
- Flujo de Datos. 34

2.2 MODELO ENTIDAD – RELACIÓN. 36

- Identificación de entidades. 36
- Relaciones y Cardinalidades. 38
- Diagrama Entidad – Relación. 40
- Mapeo. 40

2.3 MODELO RELACIONAL. 41

- Diagrama Relacional. 41
- Descripción del Diagrama Relacional. 43
- Lectura de Conectividades. 44
- Descripción Física. 45
- Diccionario de Datos. 47
- Confrontación con el mundo real. 48

2.4 DISEÑO MODULAR DEL SISTEMA. 48

- Interacción y descripción de módulos. 49
  - ➢ Módulo Maestros. 49
  - ➢ Módulo Catálogos. 50
  - ➢ Módulo Reportes 50
  - ➢ Módulo Administración. 50
- Diagrama de árbol. 51

CAPÍTULO 3. DESARROLLO DEL SISTEMA.....53
3.1 ESTRUCTURA DEL SISTEMA.....53
- Tipos de Archivos.....53
  - ➢ Archivos de Texto.....54
  - ➢ Archivos de Dbase.....54
  - ➢ Archivos Indexados.....54
  - ➢ Archivos de Código Fuente.....54
  - ➢ Archivo Ejecutable.....54

3.2 INTEGRACIÓN DEL SISTEMA.....55
- Medicos.prg.....55
- Help.prg.....55
- Med.prg.....55
- Repor.prg.....56
- Admini.prg.....56
- Catal.prg.....56

3.3 DESARROLLO DEL SISTEMA.....56
- Módulo de Maestros.....57
  - ➢ Alta de datos personales.....57
  - ➢ Alta de datos académicos.....58
  - ➢ Edición de datos personales.....60
  - ➢ Edición de datos académicos.....61
  - ➢ Consulta de Rechazados.....61
  - ➢ Consulta de Aprobados.....61
  - ➢ Consulta de Concluidos.....61
  - ➢ Salir.....62
- Módulo de Catálogos.....62
  - ➢ Catálogo de Solicitudes.....63
  - ➢ Catálogo de Causas.....63
  - ➢ Catálogo de Unidades Administrativas.....63
  - ➢ Catálogo de Subunidades Administrativas.....63

- ➢ Catálogo de Materias 63
- ➢ Catálogo de Categorías 63
- ➢ Catálogo de Sedes Hospitalarias 63
- ➢ Catálogo de Niveles 63
- ➢ Catálogo de Funciones 63

• Módulo de Reportes. 64

- ➢ Forma Única. 64
- ➢ Por movimientos. 65
- ➢ Resumen de horas. 65

• Módulo de Administración. 66

- ➢ Modificación arbitraria. 66
- ➢ Cambio de Estatus. 66
- ➢ Respaldo. 67

• Teclas de función y combinaciones reconocidas durante la ejecución del sistema. 69

• Crecimiento e importancia de la información. 70

- ➢ Tablespace 71
  - ❖ *Dinámico* 71
  - ❖ Estático 71

• Reestructuración. 76

- ➢ Reingeniería 76

3.4 CODIFICACIÓN DEL SISTEMA. 77

CAPÍTULO 4.PRUEBAS DE VERIFICACIÓN Y VALIDACIÓN DEL SISTEMA. 78

4.1 PRUEBAS DE VERIFICACIÓN. 79

4.2 PRUEBAS DE MÓDULOS. 80

4.3 PRUEBAS DE INTEGRACIÓN. 81

• Pruebas de volumen. 83

• Pruebas del sistema. 84

• *Pruebas de aceptación.* 84

4.4 VALIDACIÓN DEL SISTEMA. 85

- Fragilidad 86
- Mantenimiento 86

CAPÍTULO 5. MANUAL DE USUARIO 87

- Descripción del sistema. 87
- La información desde un punto de vista conceptual, funcional y real. 87
- Vida del sistema. 88
  - Inicialización de la información. 88
  - Operación del sistema. 89
  - Cierre del ciclo académico. 89
- Instalación. 90
- Teclas de control. 90
- Teclas de movimiento en diversas pantallas. 91
- Ejecución. 92
- Módulo Maestros. 93
  - Inserción. 93
  - Consulta. 94
  - Edición o Modificación. 94
  - Operación del módulo Maestros. 94
    - Alta Datos Personales. 95
    - Alta Datos Académicos. 96
    - Edición Datos Personales. 99
    - Edición Datos Académicos. 101
    - Submódulos de consulta. 102
    - Salir. 102
- Módulo Catálogos. 103
  - Operación del módulo Catálogos. 103
    - Salir. 105
- Módulo Reportes. 105
  - Operación del módulo Reportes. 105
    - Forma Única 106

- ❖ Por movimientos 106
- ❖ Resumen de Horas 106
- ❖ Salir. 107

• Módulo Administración. 107

➢ Operación del módulo Administración. 107

- ❖ Alteración arbitraria. 108
- ❖ Actualización Finalizados. 108
- ❖ Construir Respaldo 109
- ❖ Salir 110

• Módulo Salir. 110

CONCLUSIONES. 111

APÉNDICE.

Figuras 1 – 27.

BIBLIOGRAFÍA.

AGRADECIMIENTOS.

Ciertamente no existen palabras para describir la grandeza del amor, bondad y misericordia que **Dios** tiene para quienes confían en El, sin embargo, con todo mi ser agradezco, amo y alabo al Dios de mi salvación por su amor, por su misericordia y por permitirme llegar a este punto de mi vida en donde puedo comprobar la fidelidad de sus promesas, ya que nunca me ha dejado ni me ha desamparado y siempre ha tenido palabras de aliento e instrucción para impulsarme a seguir adelante. Por esto y por mucho más, ahora puedo expresar y testificar:

¡Jehová de los ejércitos, dichoso el hombre que en ti confía!

Quien me mostró el camino de Dios, quien me enseñó mis primeros pasos, quien siempre se ha esforzado y luchado por darnos a mi hermana y a mi lo mejor, es una persona a quien admiro, respeto y amo con todas mis fuerzas; detrás de esa mirada tierna y amorosa, de ese corazón incansable, sabio y valiente se encuentra una gran mujer: mi **madre**, Antonia Guillén de Solís.

¡ Gracias mamá, que Dios te bendiga !

Agradezco asimismo a mi **padre**, Agustín Solís Salinas, la instrucción y sabiduría con la que nos ha dirigido a mi hermana y a mi, la protección y comprensión que siempre nos ha brindado, su apoyo incondicional y sus correciones hacia nosotros, sin las cuales no hubiésemos comprendido la realidad de la vida.

¡ Gracias papá, que Dios te bendiga !

A mi hermana Betzabeth Solís Guillén, a mis tíos Angeles Flores, Jaime Guillén, Lidia Guillén, Mario Guillén que de alguna forma han contribuido en distintos aspectos de mi formación tanto académica como humana, gracias, que Dios les prospere y bendiga sus hogares.

A mi directora de tesis la Ing. Ma. del Carmen Maldonado, por su apoyo y ayuda incondicional.

Daniel Solís Guillén. Mayo 1998.

OBJETIVO

Diseñar e implementar un sistema de información que permita el fácil y eficiente manejo de la información del personal académico de la División de Estudios de Posgrado de la Facultad de Medicina, para beneficio de dicha entidad. Logrando con esto un mejor servicio, rendimiento y una disminución de los costos de operación.

CAPÍTULO 1

DEFINICIÓN DEL SISTEMA

1.1 INTRODUCCIÓN.

En la División de Estudios de Posgrado de la Facultad de Medicina se llevan a cabo diversos trámites administrativos, entre los cuales se encuentra el de asignar a un profesor un determinado número de horas para impartir asignaturas en distintas instalaciones de diversas instituciones. Esta asignación se refiere a un tipo de **solicitud** y debida por alguna **causa** determinada.
Se trata de registrar las fechas de seguimiento del trámite (o solicitud) para controlar de una forma más ordenada la documentación. La aceptación o rechazo de la solicitud, está en función de las horas disponibles (en el banco de horas), horas asignadas con las que cuente el solicitante, además de la aceptación por parte del consejo. Una vez que se ha aceptado la solicitud, se actualizan el número de horas disponibles del nivel, categoría, partida y unidad administrativa que se otorgaron, generándose la **forma única**, la cual respaldará la validez del trámite.

Lo anterior es la fuente de información para que una vez almacenada se proceda a "explotar" la misma, es decir, generación de informes, consultas, actualizaciones.

La generación de reportes vincula ó relaciona información que se almacenó por distintos procesos, ya sea desde datos personales, académicos o del mismo trámite.
Las consultas proporcionan información de forma particular, así como las actualizaciones.

Con base en el análisis del sistema se obtendrá el diseño lógico y físico del sistema, identificando los requerimientos y limitaciones, de la misma forma, se especifica el problema por resolver y las posibles soluciones.

El diseño del sistema se define como el proceso para obtener modelos que representen a éste, de acuerdo al resultado obtenido en el análisis.

El proceso que se lleva a cabo para la formación de la "forma única" es la siguiente *(ver figuras 2 y 3)*:

La DEPI (Dirección de Estudios de Posgrado e Investigación) envía una forma a cada sede hospitalaria para que indiquen quién o quienes pueden impartir clases, informando cuáles y cuántas horas, si son titulares o ayudantes; una vez llenadas estas hojas se devuelven a la DEPI y es evaluada esta información, de acuerdo

al reglamento de la misma. Una vez que cubran los requisitos, se procede a enviar la información del profesor al Departamento de Personal Académico (DPA), donde se llenará una hoja llamada "forma única", la cual contiene datos personales del profesor y tipo de nombramiento; a continuación se le informará al interesado que firme la forma única, la cual se mandará al Consejo Técnico de la Facultad para que nuevamente sea evaluada, después será firmada por el titular de la dependencia y posteriormente regresará a la DEPI para ser enviada a la Dirección General de Personal (DGP) donde se le dará trámite para que se registre en la nómina, una vez que se verifique que la información es correcta se envía nuevamente a la DEPI para que informe al profesor a partir de qué quincena cobrará y recibirá su contrato.

El Departamento de Personal Académico recibe por parte de la Dirección General de Programación y Presupuestos (DGPP), una cierta asignación de recursos económicos mediante los cuales se deben cubrir las erogaciones correspondientes al pago de los profesores. Esto se realiza a través del llamado "Banco de Horas", que es la asignación global de horas/semana/mes, (cuota de sueldo mensual por una hora a la semana y en un mes de clase impartida), en los diferentes niveles tabulares existentes dentro de la dependencia para el cumplimiento de sus programas docentes.

El Consejo Técnico de la Facultad de Medicina, en forma conjunta con el Departamento de Dictaminación, genera una relación de médicos con capacidades docentes que cuentan con los requisitos necesarios para cubrir las plazas existentes. En esta relación se incluyen datos de tipo personal y curriculum, así como la documentación de tipo administrativo requerida para realizar el trámite correspondiente. Además de las altas de académico, otra de las funciones del departamento, es el manejo de los movimientos de bajas y licencias, por motivos tales como reclasificaciones, renuncias, defunciones, modificación en horas asignadas, promociones, controlar el número de horas que imparte el académico (profesor) las cuales son 40 horas como máximo, etc.

Una vez realizado el trámite interno, se debe mantener un seguimiento del movimiento efectuado al ser enviado a las diferentes dependencias involucradas en ese proceso tales como la Dirección General de Programación y Presupuestación, El Consejo Técnico de la Facultad de Medicina y La Dirección General de Personal, **conociendo en todo momento**, en qué lugar se encuentra la documentación y las fechas de entrada y salida de ahí. Posteriormente se recibe la documentación correspondiente y se solicita la firma del interesado para concluir con el proceso del movimiento realizado.

Del total de movimientos realizados, es necesario obtener diversos informes que ilustren la situación en que se encuentra la asignación de horas, en que asignaturas, en que sedes hospitalarias, la relación de personas que se tramitaron

en cierta quincena, el número de movimientos, en que proceso de avance se encuentra el trámite, etc.

Con base en lo anterior y, considerando la cantidad y frecuencia con la que deben realizarse estos trámites, se pretende desarrollar la implementación de un sistema integral que conjunte todas las funciones señaladas, además de las particulares que faltan por señalar. El sistema que se propone contaría con un desarrollo enfocado al nivel de recursos tanto de equipo como de capacitación del personal que lo utilizaría, y con la flexibilidad necesaria para adaptarse a las posibles modificaciones que pudieran requerirse en un futuro próximo.

1.2 JUSTIFICACIÓN DEL SISTEMA.

Un sistema se define como la interrelación de elementos o componentes que se relacionan entre sí, para realizar una función definida.

Definición del sistema.

Se requiere diseñar un sistema de información que permita el fácil y eficiente manejo de la información del personal académico de la DEPI, para lograr un mejor servicio al personal académico, llevar un control de horas asignadas, mantener un almacenamiento tanto de información personal como laboral (manteniendo de manera histórica la laboral o académica), elaboración de los informes que se manejan en el departamento, así como su fácil manejo en modificaciones de materias, sedes hospitalarias, número de horas, etc.

Por lo antes mencionado, el presente proyecto busca cumplir con los siguientes puntos:

- Tener un seguimiento administrativo actualizado, eficiente y sencillo; logrando con esto el acercamiento de cualquier usuario sin experiencia en sistemas de cómputo, a la automatización de procesos administrativos.
- Disponibilidad inmediata de la información a través de diversos criterios de búsqueda.
- Seguridad de la misma por medio de claves de acceso.
- Integridad de la información.
- Validación de cierta información bajo ciertas circunstancias y condiciones según se presente alguna situación.
- División de trámites según el estado actual en el que se encuentre.
- Generación de consultas e informes de acuerdo a la información actual con la que disponga el sistema.
- Capacidad para respaldar la información las veces que se requiera.

El Departamento de Personal Académico en la División de Estudios de Posgrado e Investigación (DEPI) de la Facultad de Medicina, en la actualidad no cuenta con un sistema de cómputo que satisfaga la necesidad de llevar un control bien definido de la información que se maneja. Tal es el caso de los movimientos del personal académico y control de horas que son asignados a la DEPI.

Actualmente esta institución cuenta con programas de cómputo, los cuales no satisfacen los requerimientos establecidos. No facilitan del todo las operaciones, no son totalmente óptimos; por lo tanto lo que se busca en este proyecto es conjuntar de alguna manera todas las funciones que realizan estos programas en uno sólo, prescindiendo de las que no son necesarias y aumentando las que aún se requieren y no existen.

Lo anterior es para automatizar de manera fácil, eficiente y ordenada no sólo la información almacenada, sino además los recursos tanto de software como de hardware.

Debido a que las variables que se deben considerar son las correspondientes a altas, bajas, una serie de diferentes causas, materias asignadas a cada profesor, así como obtener su sueldo dependiendo del nivel, categoría y partida a los cuales pertenezca, datos personales del mismo, control de horas del mismo, etc; manejar todo manualmente resultaría muy costoso en tiempo y dinero, por lo que se decidió contar con un sistema que no sólo almacene toda esta información, sino que además la ordene y presente de manera precisa y confiable.

1.3 ALCANCES Y RESTRICCIONES DEL SISTEMA.

Alcances.

El sistema deberá cumplir con las siguientes condiciones:

- **Eficiencia en los procesos del sistema.** Cualquier tamaño de información requiere no sólo de un seguro almacenamiento sino además de una fácil y rápida recuperación en las búsquedas ya sea para consultar o actualizar. Evidentemente esto es muy tardado, engorroso e inseguro si se lleva a cabo manualmente, además de requerir un considerable espacio físico para su almacenamiento. Por lo que con este sistema se busca reducir al máximo estos inconvenientes a través de la clasificación de información y modularidad del sistema, para que de esta forma el usuario interactúe fácilmente con la información.

- **Automatización y seguimiento.** Los trámites llevados a cabo son invariablemente los mismos hablando de manera general, ya que se habla de una recepción, un seguimiento, una dictaminación y una conclusión de los mismos. Todo esto tiene variantes pero son muy particulares, dependiendo del

tipo de trámite, por lo que este proceso se presta a la automatización, dentro de la cual se tienen consideraciones, restricciones, validaciones que en forma conjunta contribuyen al almacenamiento y seguimiento automatizados. El seguimiento es importante, porque así es como mantiene el control exacto de la situación del documento actual.

- **Flexibilidad a cambios.** La forma de ver todo el proceso administrativo en módulos es muy favorable no sólo para el usuario final, sino también para el programador, ya que de esta forma, se pueden identificar rutinas que satisfagan ciertos módulos. En este caso usando la filosofía de la programación estructurada, se puede lograr la modularidad. Bajo la implementación de un sistema modular, posteriormente se podrán considerar cambios que se realicen de manera fácil y eficiente y que además podrán interactuar de manera transparente con los demás módulos que no requieran de cambios, esto es con respecto a la programación. La flexibilidad podrá ser a cualquier nivel o capa, ya que se contará con: diagramas de procesos administrativos (del flujo de la información), diagramas que presenten un panorama general de la relación de la información entre sí, diagramas del almacenamiento físico y además diagramas modulares del sistema; lo que significa que se podrá contar con diversos puntos de partida para comenzar un rediseño (reingeniería), mantenimiento ó modificaciones del sistema.

- **Seguridad en el manejo de la información.** Cuando se habla de grandes volúmenes de información es importante contar con claves de acceso para mantener la información libre de alteraciones que puedan dañar a la institución. Estas claves deberán ser conocidas solamente por las personas que en realidad conozcan los procesos administrativos de la institución o que deban operar el sistema cotidianamente. Esto se refiere al establecimiento de una jerarquización de usuarios, en donde se otorgan y delimitan responsabilidades.

- **Evitar la redundancia de datos.** Dentro de los procesos administrativos que manejan mucha información, se puede dar el caso de que alguna información que ya existe en el almacén (entiéndase esto como un lugar en donde se guarda toda la información) se presente en un determinado momento como nueva y debido a que se desconoce si se encuentra almacenada por el gran cúmulo de información o quizás por error de captura o intento de abuso, se corre el riesgo de volver a almacenarla y tener de esta forma datos redundantes. La solución a este problema es tener perfectamente identificada la información con ciertos atributos o características y mantener de alguna forma una interacción con el usuario para informar la situación de la información almacenada. Se puede hacer uso de reglas programadas que prevengan y eviten la redundancia de información. Cabe señalar que las búsquedas previas al almacenamiento de la información pueden disminuir la magnitud de este problema.

- **Ejecución de consultas bajo diversos criterios.** Tal y como se mencionó anteriormente, dentro de los procesos de automatización, existen diversos atributos que identifican en cierta forma a cada ocurrencia o instancia (de una solicitud por ejemplo sería el número de expediente), además existirán ciertos atributos que podrán ser comunes entre distintas instancias del mismo tipo o misma clase. Lo anterior proporciona elementos suficientes para establecer criterios de búsqueda, y así es claro que, si se establecen búsquedas por alguna característica específica que no es suficientemente identificable para una ocurrencia que puede ser común a distintas del mismo tipo, lo más probable es que exista una o más de una ocurrencia, por otra parte si la búsqueda es sobre un atributo que es totalmente identificable para la ocurrencia, seguramente (y así deberá ser) existirá solamente una ocurrencia de la misma.

- **Generación de informes impresos.** La finalidad de estos reportes es mostrar información consistente, confiable y relacionada entre sí. De esta forma el usuario podrá tener un panorama general de la situación de la institución y de la información almacenada y procesada. Es por eso que se requiere la constante alimentación (captura) al sistema con información. En la medida de como se lleve a cabo la captura y almacenamiento de información en el sistema así será la confiabilidad y veracidad de los informes.

- **Información histórica para llevar un seguimiento de cada trámite realizado.** El sistema podrá almacenar información como tan grandes sean las capacidades del hardware, sin embargo, es necesario que una persona sea responsable de mantener la información al día, es decir, que la información que se considere importante se conserve almacenada o si se considera pertinente depurarla.

- **Módulo de Administración.** Para mantener el control de toda la información alguien deberá tener derechos (superusuario) de consultar, borrar, modificar, almacenar información en todo momento para que en el caso de un almacenamiento erróneo de información, se pueda corregir desde este módulo de manera sencilla y rápida. Debido a que la información desde este módulo estará disponible para cualquier acción, se contará con una segunda clave de acceso que permitirá el acceso a este módulo.

- **Generación de respaldos.** La información almacenada continuamente, deberá ser almacenada en dispositivos que funcionen como copias de respaldo o seguridad; esto es para poder recuperar la información en caso del extravío de la misma. Se puede llevar un control de versiones por días, generando respaldos diariamente por ejemplo.

- **Pantallas de ayuda interactivas.** Considerando que las personas que operarán el sistema no tienen mucha experiencia en el manejo de sistemas de cómputo, se contará con un módulo de ayuda en tiempo de ejecución del sistema, para ello sólo será necesario invocar la ayuda en donde se encuentre en ese momento posicionado el cursor para obtener información acerca de cómo operar el sistema en el punto situado.

- **Un manual de usuario.** Además de la ayuda en tiempo de ejecución del sistema, se contará con un manual de usuario en donde se habla en detalle de todas las acciones a realizar para operar de manera adecuada el sistema. Este manual tendrá un enfoque tanto general como particular acerca de las características e instalación del sistema.

Restricciones.

- **Ejecución del sistema en ambiente multiusuario.** Actualmente todo el almacenamiento se lleva a cabo de manera local. En un futuro este sistema podría tener una mayor funcionalidad si la información pudiera ser compartida por diferentes usuarios. De esa forma la información estaría actualizada en un solo lugar (servidor hablando de un esquema cliente - servidor) y podría ser más segura en cuanto a su almacenamiento y control de usuarios, ya que así se podrían establecer privilegios sobre cada uno.

- **Habilitación del dispositivo "Mouse" (Ratón).** Para una mayor funcionalidad en pantalla muchas veces es mejor usar un ratón que pueda interactuar de manera más amigable con el usuario, sin embargo, a veces se complica el funcionamiento debido a la falta de habilidad para usar este dispositivo.

- **No contar con un verdadero manejador de Bases de Datos**. El software que se utilizó para el desarrollo del sistema no cuenta con validaciones sobre la información almacenada, por lo que las restricciones (constraints) que son necesarias aplicar para mantener consistente la información (integridad de la información), se llevan a cabo mediante programación, lo cual es válido y funcional, pero lo mejor sería usar validaciones predeterminadas y predefinidas para una mayor optimización de los recursos. Por otra parte, el uso de índices como se implementan, significa la generación de un archivo adicional (llamado indexado), mientras que en una base de datos real, la generación de un índice significa la adición de un mecanismo de búsqueda sobre la misma tabla y no una tabla adicional.

- **Consultas Manuales.** La recuperación de información se hace vía programación, esto puede ser optimizado con un software que tenga integrado SQL (Standard Query Language), construyendo con esto consultas que pueden resultar un tanto complejas pero que mejoran el tiempo de respuesta y justifican la forma en como está diseñado el diagrama relacional.

- **Software de Tercera Generación.** El sistema se desarrolló con un software de tercera generación; actualmente existe software llamado de "cuarta generación", que es capaz de mantener una interfaz con el usuario totalmente gráfica, cuya filosofía es "una imagen dice más que mil palabras"; sin embargo, a pesar de que el sistema no es capaz de ser totalmente gráfico es interactivo y amigable con el usuario.

1.4 PROPUESTA Y SELECCIÓN DE UNA SOLUCIÓN.

Canalizando el problema que se tiene hacia una solución computacional, se han podido encontrar diferentes tipos de modelos que de alguna manera podrían representar la solución del problema.

El problema presentado, trabaja con información almacenada para modificarla, consultarla, borrarla (sólo en ciertos casos); en estas operaciones existirá información que requiera más actualización, o quizás sea de tan sólo consulta, información que se esté almacenando continuamente ó a largo plazo.

Para controlar toda esta información de manera óptima se debe tener un panorama claro, general y al mismo tiempo particular acerca de qué información estará disponible para el sistema, sus limitaciones y también la forma en como se relaciona algún dato con otro.

Evidentemente no se trata de almacenar la información arbitrariamente, sino que es necesario identificar generalidades así como particularidades. Por lo que es necesario buscar un algoritmo, teoría, modelo o herramienta conceptual que sea capaz de dar esta facilidad.

Pero lo anterior necesita ser complementado por un dispositivo que trabaje de forma automática, programada, mecánica, de tal manera que ejecute ciertas instrucciones dadas en algún momento, para que así, aprovechando estas características y además su velocidad pueda explotar al máximo lo que se conoce como el nivel lógico (no tan sólo el software sino además la solución lógica) y de esta forma interactuar para llegar a una solución sistemática. Esta herramienta es una CPU (Central Process Unit).

Acercamiento.

Se tiene el conocimiento de la existencia de diversos conceptos referentes a estructuras de datos, a continuación se describe brevemente cada uno de éstos en forma particular.

- **Pila:** Una pila es una lista lineal con una operación remover y otra de agregar. Ambas operaciones se realizan por un extremo. Si queda vacía al final de la operación, no hay error, si queda vacía antes, entonces hay error. Si no se vacía hay error. (El último en entrar es el primero en salir).

- **Cola:** Es una estructura lineal en donde las operaciones se realizan por ambos extremos. (El primero que entra es el primero en salir).

- **Cola doble:** Es una estructura de datos en la cual las operaciones de agregar y retirar se practican por ambos extremos, se puede comportar como una pila o como una cola.

- **Lista Circular:** En esta estructura se tiene un orden en el que a la última localidad de almacenamiento le sigue la primera. Si se maneja con apuntadores, solo se necesita uno.

- **Lista doblemente ligada:** En esta estructura se han incluido dos campos de liga, la liga derecha (antecesor) o liga izquierda (sucesor) o viceversa. Por la estructura de la lista es posible agregar o retirar un nodo conociendo cualquier nodo de la lista.

- **Grafo:** Estructura de datos no lineal en donde todos los puntos pueden estar unidos entre sí.

- **Arcos:** Elemento de un conjunto A que forma parte de una gráfica denotada como G=(A,R) donde los arcos o líneas se establecen por la relación R.

- **Nodo:** Elemento del conjunto A de una gráfica.

- **Árboles:** Un árbol es una gráfica G=(A , R) en la que:

  - El número de los nodos es igual al número de los arcos mas uno.
  - Todos los nodos son de grado interno 1, excepto un nodo llamado de grado cero.
  - No hay ciclos.
  - Cualquier trayectoria es simple.
  - etc.

Para operar datos usando los conceptos descritos anteriormente existen algoritmos de inserción y borrado, los cuales están perfectamente definidos y sólo basta interpretarlos de manera metodológica y plasmarlos en un lenguaje computacional.

El trabajar con este tipo de conceptos significa automatizar los algoritmos respectivos, lo cual implica incremento en el código (lenguaje máquina) y como consecuencia más requerimiento de espacio físico en memoria secundaria, además del decremento en la rapidez de las operaciones.

Estos conceptos son aplicables a datos o información que se requiere almacenar y operar de manera sistemática y controlada, sin embargo en este caso no es la mejor opción porque es mejor enfocarse a la resolución del problema (filosofía o contexto de la situación a automatizar) y no dar solución a problemas que posteriormente servirán para dar problemas a otros segundos (programar algoritmos para que éstos a su vez funcionen para programar las funciones administrativas).

Esto último es dar soluciones de segundo grado (como aquél que estudia las funciones del cerebro, su cerebro propio realiza un estudio de segundo grado) lo cual no es necesario, ya que existen herramientas que ya tienen integrados estos conceptos y sólo basta usarlos para dar solución al problema en sí.

Por lo descrito anteriormente, se encontró que la solución general a este problema es resolverlo por medio de conceptos de **Bases de Datos**. La razón principal es por el enfoque que tiene este concepto con respecto a información almacenada. En este proyecto se tomarán conceptos de bases de datos en algunos casos a nivel real o en otros a nivel simulación, pero finalmente el funcionamiento y contexto del sistema estarán apegados a los conceptos de Bases de Datos.

Conceptos de Bases de Datos.

Banco de datos. Es el conjunto de todos los datos de una empresa almacenados en diversos medios, es decir, todo el cúmulo de información es conocido como banco de datos.

Toda esta información organizada según sean los requerimientos del problema, es conocida como **Base de Datos** o también sistema de información.

Todos los datos (información) son organizados de acuerdo a un modelo para establecer diversas reglas ó acciones a realizar según sea el mismo.

Al modelo se le deberán ir generando las aplicaciones que se basan en las estructuras ya creadas, se podrá extender el modelo por nuevos requerimientos.

Modelo.

Un modelo es una herramienta formal para representar un fenómeno.

- Abstrae los hechos más importantes omitiendo detalles del fenómeno
- Reduce la complejidad para poder ser entendido.
- Es lo suficientemente poderoso para explicar el fenómeno, de manera general.

Los modelos son una representación de un *fenómeno. La representación* se obtiene por abstracción, se toma en cuenta lo general de un objeto y no de sus detalles. Los modelos de datos son una herramienta para visualizar las *interrelaciones de los datos* y *pretenden* capturar el significado parcial de los datos, ya que uno total no es posible. Lo anterior es aplicable dentro de cualquier disciplina.

El modelo de datos reflejará las políticas y requerimientos de la organización y no sólo de un área específica para poder visualizar las conexiones entre datos.

Modelo de Datos.

Para describir la estructura de una base de datos es necesario definir el concepto de *modelo de datos.* Éste es un *grupo de herramientas* conceptuales usadas para describir los datos, sus relaciones, su semántica y sus limitantes. Existe gran variedad de modelos de datos y pueden dividirse en tres grupos:

- Los modelos lógicos basados en objetos.
- Los basados en registros.
- Los modelos físicos de datos.

Modelos lógicos basados en objetos.

Estos modelos se utilizan para describir los *datos en el nivel conceptual*, se caracterizan por permitir bastante flexibilidad en la estructuración y hacen posible especificar claramente las limitantes de los datos. Existe una gran variedad de estos *modelos, algunos de los más* conocidos son:

- El modelo entidad – relación.
- El modelo binario.
- El modelo semántico de datos.

Modelos lógicos basados en registros.

Los modelos lógicos basados en registros se utilizan para describir los datos en el nivel conceptual. A diferencia de los basados en objetos, estos modelos sirven para especificar tanto *la estructura lógica general de la base de datos* como una

descripción en un nivel más alto de la implantación. Los modelos que han tenido mayor aceptación son los siguientes:

- Modelo Relacional.
- Modelo de Red.
- Modelo Jerárquico.

Modelo Relacional.

Los datos y las relaciones entre ellos se representan por medio de tablas, cada una de las cuales tiene varias columnas con nombres únicos. Este será uno de los temas fundamentales de este capítulo debido a que alrededor de este modelo están basados la mayoría de los DBMS.

Modelo de Red.

Los datos en este modelo se representan por medio de *registros* (en el sentido que la palabra tiene en Pascal o PL/1) y las relaciones entre los datos se establecen por medio de *ligas*, que pueden considerarse como apuntadores.
Los registros de la base de datos se organizan en forma de conjuntos de gráficas arbitrarias.

Modelo Jerárquico.

El modelo jerárquico es similar al de red en cuanto a que los datos y las asociaciones se representan por medio de registros y ligas, respectivamente.
Sin embargo el modelo jerárquico difiere del de red en que los registros están organizados como árboles y no como gráficas arbitrarias.

Modelos físicos de datos.

Estos modelos se refieren propiamente al diseño físico de una base de datos, es decir, todo aquello concerniente a la organización de archivos, métodos de acceso a la información almacenada en un archivo y el contenido de los registros recuperados de un archivo.

Enfoque decisivo.

Al comparar las características de cada uno de los modelos anteriormente mencionados, se puede vislumbrar que los modelos entidad - relación y relacional son los más viables a seguir por lo siguiente:

- El modelo relacional es un estándar en la informática, sin distinción de plataformas o proveedores.

- El modelo relacional permite realmente una verdadera manipulación de los datos de las organizaciones.
- El modelo relacional ha propiciado y acelerado el desarrollo de otras tecnologías que en conjunto resultan en el enriquecimiento de la información que posee una institución.
- La mayoría del software en el mercado está basado en ellos.
- Los otros modelos tienen algoritmos de operación en general un tanto más rebuscados, esto significa mayor retardo en el funcionamiento del sistema y en la programación.
- Con el modelo entidad - relación las modificaciones posteriores al sistema pueden ser más claras.
- Con el modelo relacional se visualiza un panorama claro de la distribución física de la información.
- Asimismo la información se conserva mejor distribuida tanto física como lógicamente con ambos modelos.
- El modelo entidad – relación describe de manera clara al mundo real.

Por lo que a partir de aquí se enfoca el desarrollo del sistema sobre un modelo entidad relación y relacional.

Modelo Entidad Relación o Entidad Asociación.

El modelo entidad - relación se basa en una percepción de un mundo real, que consiste en un conjunto de objetos básicos llamados *entidades* y de las *relaciones* entre esos objetos. Este modelo se desarrolló para facilitar el diseño de bases de datos permitiendo especificar el esquema de cualquier sistema de información.

Componentes.

- *Entidad* es un objeto que existe y puede distinguirse de otros. La distinción se logra asociando a cada entidad un conjunto de *atributos* que describen al objeto.
- *Atributo* es una función que mapea un conjunto de entidades a un dominio.
- *Asociación* ó *relación* es el vínculo que existe entre varias entidades.
- *Diagrama de entidad - relación* queda constituido por entidades, atributos que las distinguen y asociaciones que las relacionan.

La llave es un conjunto de uno o más atributos que, juntos, permiten identificar en forma única a una entidad dentro de un conjunto de entidades.

Por otra parte, se puede definir el grado de una relación como el número máximo de ocurrencias de una entidad E, que puede participar en una asociación con una sola ocurrencia de otra entidad.

Así, para un conjunto binario de asociaciones entre los conjuntos de entidades A y B, los tipos de ocurrencias que pueden tomar son las siguientes:

- **Una a una (1:1).** Una entidad en A está asociada sólo con una entidad en B, y una entidad en B está asociada sólo con una entidad en A.

- **Una a muchas (1:n).** Una entidad en A está asociada con cualquier número de entidades de B, pero una entidad de B puede relacionarse sólo con una entidad en A.

- **Muchas a una (n:1).** Una entidad de A está asociada sólo con una entidad en B, pero una entidad de B está vinculada con cualquier número de entidades de A.

- **Muchas a muchas (n:n).** Una entidad de A está relacionada con cualquier número de entidades en B y una entidad de B está asociada con cualquier número de entidades de A.

Diseño con el enfoque Entidad - Relación.

La primera etapa en el diseño de una base de datos debe llevarse a cabo a través del enfoque que ofrece el modelo entidad - relación, debido a que éste proporciona una visión conceptual abstracta de toda la información, que en un futuro será fácil reducir a cualquier modelo de más bajo nivel.

Cada objeto físico o abstracto, que se distinga dentro de la base de datos debe representarse como un conjunto de entidades. Enseguida se debe elegir el conjunto de atributos que identifican completamente a cada entidad del conjunto. Posteriormente se asocian las entidades que correspondan, después se deberán definir las llaves primarias de cada una de las entidades y finalmente la cardinalidad o número de entidades que corresponden a otro conjunto de entidades (relaciones).

Este enfoque conceptual quedará listo para reducirse al modelo relacional, para lo cual se obtendrán un conjunto de tablas a partir de las entidades y asociaciones definidas. Lo anterior es conocido también como **Mapeo** que va del modelo entidad - relación al modelo relacional; para ello se considera: representación de entidades, representación de relaciones y consideración sobre las llaves primarias.

Diseño de bases de datos relacionales.

El objetivo es generar un conjunto de esquemas de relaciones que permitan almacenar información con un mínimo de redundancia, pero que a la vez faciliten la recuperación de la misma. Una de las técnicas para lograrlo consiste en diseñar esquemas *normalizados*.

Algunos defectos que tendría alguna base de datos mal diseñada serían:

- Redundancia en la información.
- Incapacidad para representar cierta información.
- Pérdida de la información.
- Tiempo de recuperación excesivo, debido al abuso de la normalización.

Modelo Relacional.

Una de las mayores ventajas del modelo relacional es su uniformidad. Todos los datos son almacenados en tablas, con cada renglón en la tabla teniendo el mismo formato. Cada renglón en la tabla contiene cada objeto o relación en el mundo real. Las entidades correspondientes en el mundo real a través del modelo relacional deben responder a las preguntas hechas.

Enfoque Relacional:

- Un sistema de información de bases de datos relacional se organiza en forma de tablas.
- Las tablas se organizan en renglones y columnas.
- Cada columna se denomina campo y es información de un solo tipo para todas las instancias.
- Las relaciones de las tablas se logran a través de algunos de sus campos (columnas), los cuales reciben nombre especiales:
    - Llave primaria. Columna o grupo de columnas que identifica de manera única a cada renglón de la tabla.
    - Llave foránea. Es una columna o grupo de columnas que es llave primaria en alguna otra tabla y se encuentra en una o más tablas para relacionarse.

- Dentro de este contexto se maneja el término nulo, que sirve para hacer referencia a una ausencia de dato.

Objetivos del modelo relacional.

Edgar Codd propuso tres objetivos para el modelo relacional:

1. Independencia de datos. Diferenciar entre lo físico y lo lógico de los datos.

2. Comunicación. Para que los programadores y los usuarios finales tengan una buena comprensión y comunicación acerca de los datos.

3. Procesamiento de Conjuntos. Poder manipular grandes volúmenes de datos en una sola operación.

Componentes del Modelo Relacional.

1. Estructuras. Las estructuras de datos se emplean para representar en el modelo los objetos y sus propiedades.

2. Restricciones de integridad. Las restricciones de integridad se usan para tratar de asegurar que cualquier transformación a las estructuras de datos produzca un resultado consistente (constraints).

3. Operaciones basadas en álgebra relacional. Las operaciones representan las transformaciones que sufren los objetos (estructuras de datos), sujetas a las restricciones de integridad definidas en el modelo.

La estructura básica del modelo relacional es la **Relación**.

Relación.- (término matemático) que denota una conexión entre un par de objetos.

A continuación se listan las correspondencias entre los elementos del modelo entidad – relación y el relacional.

- Entidad. Es el nombre con el que se conoce una relación. Es un conjunto finito de atributos. Propiamente en el modelo relacional es el nombre de una tabla.

- Atributos. Características que complementan la información de la entidad. A cada atributo corresponde un dominio. En el contexto del modelo relacional es el nombre de una columna

- Dominio. Conjunto de valores que puede tomar un atributo. En este caso es el universo de valores que puede tomar un determinado registro.

- Relación. Es la extensión de la entidad. Conjunto finito de mapeos del producto cartesiano de los dominios. Todos los registros existentes o contenidos en una tabla con respecto a otra tabla.

- Tuple. Es un elemento de una relación. Un renglón o registro en particular

Formas Normales.

Para determinar si una estructura de datos cumple con los objetivos de unicidad e integridad se utilizan las *Formas Normales*, que son una serie de restricciones que se aplican sobre las estructuras relacionales para evitar irregularidades al manipular los tuples (registros).

Después de aplicar las formas normales, se tendrá como consecuencia en el modelo una maximización en la independencia de datos y una reducción del riesgo de pérdida de su integridad.

La teoría de las formas normales se aplica a estructuras relacionales, asignando un grado de normalización a una relación, en donde este grado depende del número de restricciones que cumpla.

La teoría de las formas normales surgió con la teoría del modelo relacional. Las primeras investigaciones sobre el tema las inició el Dr. Edgar Codd quien definió tres grados de normalización o formas normales; posteriormente se ha complementado la teoría enumerando formas normales adicionales, las cuales son escasamente encontradas y aplicadas en la práctica.

Primera Forma Normal.

Una entidad R está en primera forma normal cuando cada atributo de la relación tiene un solo valor. Un atributo no deberá contener conjuntos de valores.

Segunda Forma Normal.

Una entidad R en primera forma normal, estará en segunda forma normal si no existe un atributo no primario en R que dependa parcialmente de la llave R (en este caso la llave R es compuesta).

Tercera Forma Normal.

Una entidad R está en tercera forma normal si está en segunda forma normal y además no existe algún atributo no primario en R que dependa *transitivamente* de la llave de R.
Lo anterior en la práctica se refiere a lo siguiente:

Situando la atención del lector en el concepto de una tabla en el modelo relacional, se entiende que existe una "cruce" entre cada columna y cada renglón, este cruce deberá siempre contener solamente un dato, no podrá contener una lista de datos ó enumeración de los mismos; al cumplir esta condición se dice que el modelo cumple con la primera forma normal.

Por otra parte, cada tabla deberá contener campos llamados llaves primarias, *éstas algunas veces estarán* compuestas por más de un campo, en tal caso, todos los demás campos que no son parte de la llave primaria deberán depender de todos campos que forman la llave primaria, un campo que no pertenezca a la composición de la llave primaria, no deberá depender solamente de un campo de los que forman la *llave primaria (dependencia parcial)*, sino de todos. Al depender todos los campos que no forman la llave primaria de todos los campos que componen la llave primaria y cumplir con la primera forma normal, se dice que se ha cumplido la segunda forma normal.

Además, con respecto a los campos que no forman parte de la llave primaria, ninguno de ellos deberá mantener alguna relación con alguno de éstos mismos, esto es conocido como dependencia transitiva. Al no existir esta dependencia y cumplirse la seguna forma normal, se dice que se ha cumplido con la tercera forma normal.

Las tres formas normales ofrecen las herramientas suficientes para que la base de datos funcione óptimamente y mantenga la integridad de la información. Con esto, se reducen considerablemente los posibles errores de redundancia de información y se asegura la representación formal del mundo real en un modelo relacional.

Aplicación de los modelos Entidad -- Relación y Relacional.

Considerando las características particulares de los modelos entidad – relación y relacional, *se puede vislumbrar que sus conceptos* son totalmente aplicables al problema que se trata de solucionar con este sistema. Las soluciones que ofrecen satisfacen de manera amplia los diferentes enfoques panorámicos, esto es, relaciones entre entidades, definición de sus atributos, vista general de los archivos físicos y relación entre ellos, *así como la definición de sus distintos* atributos y /o campos.

Software de almacenamiento y control de información.

Se canaliza el desarrollo en este momento hacia la elección de un software de desarrollo y de almacenamiento. Para ello es necesario comentar los conceptos reales e ideales con los cuales se trabajará en la resolución del problema presentado.

Ya que anteriormente se han tocado los conceptos referentes a bases de datos relacionales, a continuación se presentan conceptos que son utilizados en este mismo contexto, pero de manera práctica, es decir, la relación que se guarda entre *distintos niveles de usuarios*, con la base de datos, interacción con la base de datos (diferentes tipos de lenguaje), operaciones sobre la misma, etc.

Una base de datos requiere de elementos de hardware y software.

El hardware es necesario para hacer frente a los grandes sistemas de cómputo, se compone de dispositivos de almacenamiento secundario, donde reside la base de datos física, junto con dispositivos asociados como unidades de control.

Por lo que respecta al software, los sistemas de bases de datos están compuestos por un conjunto de programas. Los programas se construyen atendiendo a dos objetivos básicos: accesar a la base de datos y crear una interfaz con el usuario que se encargue de solicitar los datos que se desean consultar, insertar, modificar o eliminar.

Los **lenguajes** con los que se construye la interfaz se les conoce como lenguajes anfitriones, también conocidos como **front - end**.

Existen además de los lenguajes de desarrollo de aplicaciones otros tipos de *lenguajes dentro del contexto de bases de datos*. Se distinguen los siguientes:

Lenguaje de Definición de Datos.

Un esquema de la base de datos se *especifica por medio de instrucciones* de este lenguaje. El resultado de la compilación de las sentencias en el DDL (Data Definition Language) es un conjunto de tablas que se almacenan en un archivo especial llamado diccionario de datos (un diccionario de datos es un archivo que contiene *metadatos*, es decir, datos acerca de los datos) y una serie de instrucciones que especifican los detalles de la implantación de los esquemas de bases de datos. (lo anterior es por ejemplo: creación alteración o eliminación de tablas, sinónimos, índices, secuencias, vistas).

Con este lenguaje pueden definir:

- Tablas.
- Vistas.
- Indices.
- Archivos Físicos.
- Llaves primarias y foránea.
- También se pueden borrar ó alterar estas definiciones con el mismo lenguaje.

En SQL algunas palabras para ello serían Create Table, Create Index, Create Sequence, Alter table, Drop index, Drop table, etc.

Lenguaje de Manipulación de Datos.

Se refiere a:

- Recuperación de información almacenada en la base de datos.
- Inserción de información nueva en la base de datos.
- Modificación y eliminación de información de la base de datos.

Por lo tanto, un DML(Data Manipulation Language) permite a los usuarios manejar o tener acceso a los datos que estén expresados por medio del modelo apropiado. Con este lenguaje podemos modificar el estado de los datos que residen en las tablas, respetando las restricciones de integridad:

- Agregar.
- Borrar.
- Modificar.
- Leer.

Cualquiera de las operaciones anteriores se puede ejecutar para uno o varios renglones de la tabla. No hay que olvidar que el lenguaje de manipulación de datos estará basado en el álgebra relacional, que a su vez se fundamenta en la teoría de conjuntos. Por lo tanto, cualquier postulado de manipulación de datos deberá ser capaz de operar con conjuntos de renglones.
Por ejemplo en lenguaje **SQL** (Structured Query Language) serían las sentencias: select (en todas sus variantes), update (para actualizaciones), insert (almacenamiento de nuevos datos).

Nota: El lenguaje SQL es muy utilizado en bases de datos relacionales, es la forma en cómo el usuario interactúa con la información que existe en su base de datos. Su sintaxis es muy trivial, resulta muy intuitiva la formación de sus sentencias, y es muy parecido al lenguaje coloquial humano.

Lenguaje de Control de Datos.

Con él se contemplan instrucciones para establecer privilegios sobre la información contenida en la base de datos a ciertos usuarios (los usuarios se identifican por medio de "nombres de usuarios" y claves de acceso). De esta forma se establecen niveles de usuarios que van desde usuarios que solamente pueden consultar cierta información hasta "superusuarios" (capaces de hacer cualquier tipo de operación o transacción sobre toda la información contenida en la base de datos).
Un ejemplo de SQL es la palabra: Grant.

DBMS.

Un Sistema Manejador de Bases de Datos es una herramienta que nos permite construir, administrar y explotar una base de datos.
Todo DBMS está basado en un modelo de datos, ya sea jerárquico, de red o relacional. El DBMS actúa como un intermediario entre la base de datos y los usuarios. Estos usuarios pueden ser los usuarios finales, los programadores o el administrador de la base de datos. Los componentes de un DBMS son:

Seguridad.

Dentro del DBMS debe existir un mecanismo para proteger los datos, y para poder dar o quitar autorizaciones a los usuarios.

Los tipos de autorización son:

- Lectura.
- Actualización.
- Borrado.
- Inserción.

Y se pueden otorgar sobre:

- Un grupo de tablas.
- Una tabla.
- Una vista (subconjunto de renglones y columnas).

La seguridad que provee el DBMS debe ser complementada por el sistema operativo. No existen sistemas de seguridad perfectos. Los DBMS suelen brindar la función de crear tiras de auditoría. Se deben considerar otras opciones como el encriptado de datos.

Control de concurrencia.

Para que un DBMS permita el acceso concurrente a los datos debe tener técnicas para asegurar la integridad de los datos. Se emplean técnicas de "locking" (aquellas que se refieren al bloqueo de objetos cuando están siendo utilizados por otro proceso o usuario) para poder asegurar la integridad.

Todo DBMS deberá estar construido a partir de la premisa de que las fallas son inevitables. Las fallas pueden clasificarse en cuatro tipos:

- Fallas locales a un programa en ejecución detectadas por el programa.
- Fallas locales a un programa en ejecución no detectadas por el programa (no hay espacio en disco).
- Fallas globales que afectan a todas las transacciones en ejecución (suspensión de la energía eléctrica).
- Fallas en los dispositivos de almacenamiento.

El primer tipo de falla deberá ser manejado por el programa, apoyándose en las herramientas que el DBMS proporciona. Las fallas del segundo y tercer tipo deberán ser manejadas por el DBMS en forma automática. Las fallas del cuarto tipo se deberán resolver apoyándose en utilerías de recuperación proporcionadas por el DBMS.

Respaldo y recuperación.

El DBMS debe contar con mecanismos que permitan recuperar los datos en caso de alguna falla. Los mecanismos de recuperación se basan en la redundancia controlada. Por tanto deberá contar también con procesos para poder respaldar la información.

Por lo general, los DBMS proporcionan tres elementos básicos para resolver alguna falla:

- Utilerías para hacer respaldos periódicos de las tablas.
- Mecanismos para registrar en una bitácora (archivo en donde se registra un historial de acciones efectuadas) todos los cambios efectuados a los valores de los datos en las tablas.
- Utilerías para poder restaurar o recuperar tablas a partir de los respaldos.

Utilerías de mantenimiento.

Para tener un buen rendimiento, todo DBMS debe contar con diferentes procesos para mantener y ajustar las estructuras de datos.
En general, las utilerías de mantenimiento se pueden agrupar en tres clases:

1. Reorganización Física: Este tipo de utilerías se emplea para reestructurar físicamente alguna tabla o índice con objeto de preservar las condiciones físicas con las que se diseñó originalmente, o bien, corregir alguna de estas condiciones.

2. Validación de Integridad: Son utilerías que chequean algún aspecto de la integridad de los datos.

3. Reportes y generación de estadísticas: Son utilerías para producir reportes sobre algún aspecto del DBMS, o bien, para actualizar las estadísticas que emplea el optimizador para seleccionar trayectorias de acceso a los datos.

Catálogo.

Es un conjunto de tablas donde se almacena información de estructuras de datos definidas, programas que hacen referencia a las estructuras y a ciertas autorizaciones sobre objetos.

Índices.

Un índice se puede definir por dos razones:

- Asegurar la unicidad de una llave primaria (no necesariamente siempre)
- Mejorar la velocidad de acceso a los datos de una tabla.

Por lo general existen diferentes tipos de índices:

- que no permiten valores duplicados.
- que sí permiten valores duplicados.
- que imponen un ordenamiento físico a los renglones de la tabla.

Vistas.

En términos generales a una vista se le da el mismo tratamiento que a una tabla. La principal diferencia entre una vista y una tabla se presenta al momento de pretender hacer una actualización, puesto que no cualquier vista puede ser actualizada. La tabla es actualizable y la vista es sólo un subconjunto de columnas y renglones. Las vistas se pueden crear con diversos propósitos:

- Tener un nivel adicional de independencia lógica entre datos y programas.
- Reducir la complejidad en las consultas hechas por usuarios finales.
- Dar nombres alternos a las columnas de una tabla.
- Proporcionar seguridad a nivel de contenido.

- Como medio para crear tablas sumarizadas, agregadas o con valores calculados.

Optimización.

En un DBMS relacional es indispensable la existencia de un optimizador, dada la naturaleza declarativa del álgebra relacional. Los optimizadores se basan en algoritmos de costo en los que se pondera el uso de la CPU y el I/O.

Herramientas para monitoreo.

El DBMS debe contar con herramientas que permitan obtener información sobre:

- La ejecución de programas, transacciones.
- Las trayectorias de acceso a los datos.
- etc.

El monitoreo es una actividad que se debe realizar en forma continua para analizar tendencias y corregir desviaciones. Sólo así se puede garantizar un servicio estable y tiempos de respuesta adecuados.

Administración de bases de datos.

Las tareas del DBA (Data Base Administrator):

- Hacer el diseño lógico y físico de las estructuras de datos.
- Definir perfiles de seguridad.
- Definir la estrategia de respaldo, recuperación y mantenimiento.
- Monitorear el DBMS y las aplicaciones que lo usan, dando solución a problemas.
- Revisar el diseño de las aplicaciones que usarán el DBMS.
- Dar asesoría técnica a usuarios.

Esquemas.

Dentro de un DBMS podemos ver los datos desde tres niveles diferentes o ESQUEMAS que son:

1. Esquema Externo. Es el más cercano al usuario final y es cómo este visualiza las estructuras de datos que usa. Generalmente un usuario sólo usa un subconjunto de estas estructuras; estos subconjuntos de datos son definidos por medio de vistas lógicas.
2. Esquema Conceptual. En este nivel están definidas todas las estructuras de datos y sus relaciones. Los usuarios a este nivel son el administrador de datos y el administrador de la base de datos.
3. Esquema Interno. En este esquema se ven las estructuras a nivel de almacenamiento físico, por ejemplo:

    - El espacio físico que se requiere para cada estructura.
    - Su localización.
    - Índices que tiene.
    - Tipos de archivos físicos que utiliza.
    - Etc.

Los usuarios son soporte técnico y el administrador de la base de datos.

En la arquitectura a tres niveles existen dos tipos de mapeos, que son la forma en que se comunica cada esquema o nivel con el siguiente:

a) De cualquier Esquema Externo al Esquema Conceptual y viceversa. Este mapeo consiste en transformar una petición de acceso a los datos a un postulado equivalente pero a nivel conceptual (de EE a EC). Al obtener los datos, el DBMS los transforma de EC a EE y se realiza el mapeo inverso.

b) Del Esquema Conceptual al Esquema Interno y viceversa. Este mapeo transforma un postulado en una lista de operaciones para el acceso físico a los datos usando la trayectoria de acceso más óptima. Posteriormente los datos se mapean a una estructura relacional.

Funcionalidad de un DBMS.

Un DBMS es una colección de programas que son interfaz con los programas de aplicación y manejan los datos a favor de los programas de aplicación. Un programa de aplicación puede acceder a los datos en almacenamiento secundario a través de un programa llamado método de acceso.

En un DBMS, el programa de aplicación requiere datos desde el DBMS. El DBMS determina qué dato es necesitado, si el usuario es el autorizado para ver el dato y dónde el dato es localizado en almacenamiento secundario. El DBMS puede presentar datos en el mismo formato para programas existentes y también presentar datos con un nuevo formato a los nuevos programas que están siendo instalados. Esta facilidad es llamada independencia de datos. Los datos no redundantes es uno de los mayores objetivos de un DBMS. El DBMS examina

cada requerimiento desde un programa de aplicación para asegurarse que el programa tiene la autorización para manipular el dato que ha requerido.

El DBMS también provee integridad de los datos, esto es, el dato representa lo que significa. La información puede estar contenida en un gran contenedor y ser consultada por muchos usuarios, de esta manera ser compartida y descentralizada, lo cual resulta mejor que tener toda la información en cada terminal, para lo anterior se requiere un control de tráfico.

Desventajas.

Las desventajas de un DBMS son las siguientes:

- Requieren de mucho espacio para almacenamiento.
- Se requiere la capacitación de personal.
- Presentan bajo rendimiento cuando las peticiones son muchas y por ello hay mucha espera.
- Su tamaño contribuye a su relativo alto costo.
- Los requerimientos de hardware son altos.
- La ejecución de millones de instrucciones para soportar un DBMS consume un significativo número de ciclos por computadora, así como una gran cantidad significativa de almacenamiento primario y secundario.
- Cuando sucede algún error o falla en el sistema o periféricos, el tiempo de reparación puede ir desde minutos hasta días.

Un ejemplo claro de un DBMS es Oracle.

Oracle.

Es un DBMS en toda la extensión de la palabra, en el mercado es una de las herramientas en bases de datos más cara. Puede ser instalado en Windows NT o Unix y ser totalmente transparente para el usuario. Hace uso del lenguaje SQL, además de tener integradas instrucciones propias de él llamadas "SQL Plus", además tiene la capacidad de manejar la información almacenada en sus tablas por medio de PL\SQL (Procedural Language), esto es, manipular con los lenguajes SQL y SQL*Plus vía programación, en donde el código y la ejecución del mismo residen en el servidor respectivo, lo que incrementa la velocidad de ejecución y optimización de espacio en el cliente. Hace uso de la tecnología cliente - servidor. Además de tener otros productos como SQL*Forms, Administrator, Developer 2000, Oracle Web Server (desarrollo de páginas dinámicas en Internet), etc.

Software de desarrollo.

Existe una amplia variedad de software que está enfocado al modelo relacional, a continuación se describen algunos de ellos, para que de esta forma se confronten y se pueda decidir que software es el que se ajusta a las necesidades del problema:

dBASE IV.

De entre las facilidades que ofrece, aunque no son del todo versátiles ni rápidas, se encuentra la creación de consultas, formas reportes y etiquetas. dBASE IV incluye la opción de utilizar SQL para BD relacionales. Una de sus limitaciones es que su velocidad de procesamiento es baja, debido a que los programas se ejecutan por medio de un intérprete, es decir que no genera un código binario, sino que cada instrucción se ejecuta según se va encontrando, además de que para la ejecución de un programa se requiere de todo el software, el programa no es independiente. A pesar de que dBASE IV genera un código ejecutable, no hay mucho avance por el lado de velocidad, se requieren 2 MB en RAM, procesador 286 o superior.

FoxPro.

Entre sus características está la emigración que tuvo a plataformas UNIX y Macintosh, permite una velocidad de ejecución grande, ya que además de tener un modo de operación como intérprete, permite generar una versión ejecutable que se puede distribuir junto con un programa de runtime. FoxPro extendió el lenguaje original de DBASE para darle mayor funcionalidad y manejo de multiusuarios, además de incluir un generador automático de pantallas y reportes.
La versión que apareció para Windows, además de permitir el diseño con la interfaz gráfica y el uso de SQL para hacer consultas, integra la capacidad de usar BD residentes en servidores tales como Oracle y Sybase.
Esto último aumenta la posibilidad de crear aplicaciones cliente - servidor en las que se optimiza el acceso a los datos siempre que éstos no se tengan localmente, sin embargo una consecuencia del cambio de plataforma (de DOS a Windows) es que Fox Pro para Windows disminuye su rendimiento en cuanto a la velocidad de ejecución (para aplicaciones locales) e incrementó en forma considerable sus requerimientos de espacio en disco y memoria RAM (Memoria de Acceso Aleatorio). Requiere Microsoft Windows 3.0 o superior, PC con Intel 386 o mayor, 4 MB de memoria, 14.5 MB en espacio en disco, Monitor EGA o de mayor resolución. Trabaja en plataformas tales como Novell Netware, Windows for Workgoups, Windows NT, LAN Manager.

Paradox.

Desde su inicio ha mostrado un esquema de funcionamiento que ofrece una idea un poco más cercana a un DBMS: en lugar de archivos maneja tablas como ORACLE, contiene el concepto de vistas lógicas, realiza validación de datos, integridad referencial, maneja índices primarios y secundarios y seguridad de archivos. Además mantiene una tendencia a administrar en forma coherente todos los elementos de un proyecto. Sus versiones para Windows incluyen el modelado de datos en forma visual para formas y reportes, un ambiente de desarrollo orientado a objetos, capacidad de utilizar simultáneamente datos de Paradox y DBASE y la conectividad de servidores a bases de datos. Puede utilizarse como front - end, como base de datos en redes locales y en modo local independiente de servidores, tanto como para desarrollo de aplicaciones de alto volumen como de menor escala. Trabaja en Novell Netware, Windows for Workgroups 3.11, Windows NT 3.11, LANtastic. Requiere Windows 3.1, procesador 386, 8 MB de memoria RAM, 20 MB en disco duro, monitor EGA.

Access.

Contiene elementos de un DBMS formal tales como: manejo de tablas, índices, llaves primarias y foráneas, integridad referencial, soporte de transacciones, seguridad de datos, y consultas mediante SQL. Cabe mencionar que este lenguaje tiene mucha semejanza con Visual Basic para Windows, y que este último a su vez está basado en Access, por los que es posible realizar programas en Visual Basic que exploten los datos de Access. Es un producto dual, sirve como front - end en un esquema cliente-servidor y como base de datos local, está orientado a usuario final. Requerimientos mínimos: Windows 3.11, procesador 386, 6 MB en memoria, 17.8 MB en disco duro, monitor EGA. Trabaja en Novell Netware, Windows for Workgroups, Windows NT, LAN Manager y Windows 95.

Clipper.

Es un lenguaje de programación de bases de datos completo, con todas las herramientas necesarias para desarrollar aplicaciones de bases de datos totalmente independientes. Clipper nació como un compilador del lenguaje DBASE III para generar el código binario para PC. Clipper permite básicamente manipular el lenguaje de programación de dBASE III, con algunas pocas restricciones, sobre todo en cuanto a la consulta de datos. Clipper permite interactuar con otros lenguajes como "C", ensamblador, etc; y generar aplicaciones para red debido a su concurrencia.

Es muy rápido, debido a que Clipper compila su propio dialecto de dBASE produciendo así aplicaciones muy rápidas. Como Clipper es un lenguaje de programación de alto nivel, tiene un considerable número de funciones y comandos que están diseñados específicamente para la creación de aplicaciones

que se requieran implementar, evitando así la generación de cientos de líneas de código. Las aplicaciones desarrolladas producen un programa ejecutable totalmente independiente, de esta manera no se visualiza el código fuente.

Lenguaje C.

Es un lenguaje usado con mucha frecuencia en cálculos estadísticos, en problemas que requieren de cierta precisión numérica; maneja rutinas gráficas que ofrecen una cierta interfaz agradable y amigable, sin embargo, se requiere un considerable número de líneas para lograr lo anterior, debido a que es un lenguaje de medio nivel.

Confrontación.

Existe una considerable diferencia entre cada uno de los lenguajes mencionados anteriormente, algunos ofrecen una interfaz más amigable entre la aplicación y el usuario, es decir que su ambiente en tiempo de ejecución es más descriptivo y representativo para el usuario por sus componentes gráficos, sin embargo requieren de características de hardware elevadas, otros de manera sencilla cumplen parcialmente con la teoría de bases de datos, pero a través de programación un tanto más detallada, se puede simular lo que sea necesario, algunos otros no son capaces de ejecutarse de otra forma más que con su ambiente de desarrollo, lo que implica que cada instalación de aplicación, implica más espacio físico en memoria secundaria.

Atendiendo a lo anterior, se busca un software con las siguientes características:

- que fuese un sólo archivo ejecutable que no dependa de su ambiente de desarrollo, para economizar espacio en disco duro con el objeto de almacenar información útil para la institución y no para desarrollo de programación.
- que el programa ejecutable residente en memoria no implique la disminución o retardo en la operación del mismo.
- que tenga los requerimientos mínimos de hardware, de acuerdo a los equipos con los que cuenta actualmente la UNAM.
- que tenga la flexibilidad y accesibilidad para cualquier usuario (al menos nivel técnico) para el mantenimiento del programa y posibles modificaciones futuras.
- que sea capaz de interactuar con el sistema operativo MS-DOS, con instrucciones predefinidas, sin hacer uso de lenguaje ensamblador que es bajo nivel.

Por lo tanto el software que más se adapta a las necesidades es: Clipper. La versión que se utilizará será Clipper 5.2.

Teoría de Herramientas CASE.

CASE (Computer Aided Software Engineering).- Ingeniería de Software Asistida por Computadora.

Las herramientas CASE son términos genéricos que se refieren a la automatización del desarrollo del software. Existen varios niveles de Herramientas CASE, cada uno de ellos se refiere a cada etapa en el ciclo de vida del sistema en desarrollo o en crecimiento.
Con este tipo de herramientas, el tiempo dedicado al desarrollo del sistema es en su mayor parte ocupado en el análisis y diseño, más que en la programación con aciertos y errores. Mejora la calidad de análisis al realizar validaciones automáticas en los modelos a través de métodos perfectamente definidos.

La razón de tocar el tema de herramientas CASE es porque los diagramas de procesos, de módulos, entidad - relación y relacional, así como el diccionario de datos presentados en este trabajo han sido generados con herramientas CASE.

Indiscutiblemente son herramientas que no sólo ahorran trabajo, sino que además debido a las rutinas automatizadas de validación que tienen integradas, son confiables. Lo anterior no significa que el diseñador se separa del análisis y permite que la herramienta realice todo el trabajo, al contrario, por lo mismo que la herramienta está automatizada, si no se está plenamente consciente y se tienen perfectamente definidas todas las variables y situaciones que se van a modelar, puede ocurrir que la herramienta "arroje" resultados erróneos para la situación en particular, sin embargo será una respuesta buena para la herramienta como tal por los valores de entrada que generó el usuario.

Las Herramientas que se utilizaron fueron: SILVERRUN-ERX, SILVERRUN ERX-RDM y SILVERRUN-RDM (generación de diagrama entidad - relación y relacional respectivamente).

En este caso los conceptos que manejan estas herramientas son de bases de datos relacionales.
Para generar el diagrama entidad - relación, es necesario que se tenga totalmente identificada a cada entidad (esto no lo hace la herramienta), posteriormente se hace uso de los mecanismos automatizados del software para establecer las conexiones entre las mismas. A través de preguntas en un ambiente totalmente gráfico, la herramienta trabaja de manera interactiva con el usuario, por lo que el usuario (desarrollador) **deberá** tener absolutamente todo el conocimiento de la situación o problema que se está modelando y diseñando. La herramienta es capaz de verificar errores, hacer correcciones y ofrecer posibles soluciones.

Asimismo para la generación del modelo relacional se hace uso de una etapa intermedia que se encarga de convertir el diagrama entidad relación al diagrama relacional.

Una vez que se obtiene el resultado anterior, se podrá manipular el archivo para generar valores nulos, llaves foráneas, índices, constraints (reglas de integridad y validación sobre la base de datos), etc., para lo cual también se trabaja en forma interactiva con la herramienta para finalmente obtener el modelo relacional que es el modelo físico de la base de datos.

De acuerdo a lo anterior se puede observar la interacción entre el software y el usuario, no se trata de que la herramienta trabaje por sí sola, sino que trabaje en forma conjunta con el usuario.

En sistemas que manejan grandes volúmenes de información (esto es, grande número de entidades, relaciones, constraints ó reglas de integridad, hablando de bases de datos), estas herramientas ayudan satisfactoriamente no solo para diseñar, sino también para la puesta en marcha de una base de datos (lo que se conoce como levantar una base de datos).

Es importante considerar esto porque el objetivo de la ingeniería es lograr la optimización de métodos y recursos con el fin de encontrar y aplicar la solución a diversos problemas. Por lo tanto, las herramientas CASE pueden contribuir a la generación de resultados con un alto grado de confiabilidad en conjunción con técnicas y teorías aplicadas.

CAPÍTULO 2

ANÁLISIS Y DISEÑO DEL SISTEMA

2.1 ANÁLISIS DEL SISTEMA.

La metodología de desarrollo de sistemas puede variar de acuerdo a cada forma de trabajo de cada autor o a los requerimientos o necesidades que requiera un sistema en determinadas circunstancias, pero finalmente de manera general lo que se busca es recolectar toda la información posible que sea real, reducir los inicios falsos, reciclamiento indebido, callejones sin salida, situaciones indefinidas, la disminución en el costo, tiempo del desarrollo y vida del sistema y además que el sistema sea el que realmente los usuarios desean y necesitan.

Diseño.

Una idea general del diseño es la siguiente: un proceso sujeto a una metodología, un proceso en el que la aplicación de la metodología adecuada permite al responsable generar un producto efectivo y eficiente. La evolución del diseño de software es un proceso continuo que se ha ido produciendo durante las últimas décadas (desde los criterios para el desarrollo de programas modulares y del análisis y diseño estructurado), en el que cada metodología de diseño introduce heurísticas y notaciones propias.

1. **Diseño orientado a procesos** está enfocado a la funcionalidad del problema y a sus soluciones; la información puede representarse como un flujo continuo que sufre una serie de transformaciones (**procesos**), conforme va de la entrada a la salida.

2. **Diseño orientado a los datos** está enfocado a la información manipulada por los procesos; en vez de concentrarse sobre los procesos, utilizan la estructura de la información como el conductor de la derivación del diseño.

3. **Diseño orientado a objetos** está enfocado a una mezcla de los dos anteriores, vinculando a objetos de datos específicos los procesos que pueden ser ejecutados en ellos; se mencionan ventajas significativas por su sencillez y su habilidad para ser reutilizables.

4. **Diseño orientado a eventos** está enfocado en eventos que ocurren y que deben ser atendidos durante la ejecución de la solución; cuenta con mucho respaldo al ser generalmente reconocido por su utilidad en aplicaciones línea.

La mayoría de los anteriores conceptos de diseño, así como los diferentes tipos, se integran en el presente sistema (algunos con cierta simulación). Algunos se pueden entender como etapas, por ejemplo el primero se refiere al flujo de datos, es decir, las diferentes etapas que puede tener una situación dentro del problema planteado, las consideraciones de sus diferentes bifurcaciones y todos sus posibles estados; el segundo al esquema físico de la base de datos, esto es, la estructura física que contendrá toda la información, esta estructura debe ser diseñada de acuerdo a las condiciones que se establezcan en el problema o situación, de acuerdo a las reglas del negocio.

Los conceptos tercero y cuarto se refieren a la programación en sí. Manejando ambientes gráficos, visuales, se hace uso de objetos (botones, radio buttons, check box, listas, tablas), que están predefinidos por el mismo software y que sólo es necesario seleccionarlos y definir sus propiedades para que en tiempo de ejecución proporcionen la funcionalidad deseada.

En el presente sistema no existe el tercer concepto como tal, sin embargo se establece una simulación, tal es el caso de la función Browse(), la cual despliega una ventana que contiene información de una tabla en las coordenadas definidas previamente.

El cuarto concepto se integra en el sistema cuando el usuario decide realizar alguna acción (evento) dentro del sistema, ya sea la selección de un menú, la invocación de una ventana por medio de alguna función predefinida en el sistema, etc. Esto es porque cualquier acción, ocasionará la ejecución de algún procedimiento o función programados.

Reunión de información.

Para recolectar la información necesaria e identificar todo el contexto del sistema ha sido preciso establecer pláticas con personal que lleva a cabo estas labores, que identifica cuáles son los requerimientos necesarios para tener un mejor control y funcionamiento en cuanto a sus labores se refiere. Es por eso que el usuario final tiene un papel muy importante en el desarrollo del sistema por que es él quien establece las fronteras del mismo sistema para beneficio propio y de la institución.

En la interacción con el usuario se deben establecer preguntas estratégicas de tal forma que sea el mismo usuario quien proporcione las soluciones a determinadas situaciones y defina de manera clara y precisa las variantes.

De acuerdo con toda la información recabada se han identificado diferentes situaciones, tiempos de proceso y de operación, las cuales nos marcan en cierta forma el punto de partida para comenzar a integrar y plasmar lo que es el mundo real en un diagrama de bases de datos relacional.

Viendo de manera general cada situación o ente que interactúa con otras, se pueden establecer y definir de manera terminante cada una de ellas, para que de esta forma se comience a vislumbrar de forma general pero no por ello borroso sino totalmente claro, el camino que sigue la información tanto en el sistema como administrativamente (confrontación del mundo real con el diseño de bases de datos relacionales).

Flujo de Datos.

Con esto se trata de dar una idea general de los procesos administrativos, a los cuáles se pretende dar solución. Puede entenderse como un enfoque resumido, que busca relacionar los conceptos puramente administrativos, todavía sin dirigirlo a un entorno de bases de datos relacionales o de programación.

Nota: En los diagramas de flujo o procesos que se muestran se ha usado una simbología en particular, cada símbolo representa una cierta acción a realizar o denota algún proceso en particular. (Ver figura 1).

Todo gira en torno a una solicitud requerida por personal académico.

Un diagrama de procesos administrativo es el siguiente *(Ver Figura 2)*:

1. Un académico puede solicitar un movimiento administrativo, este movimiento se refiere a manipular horas de impartición de clases, horas asignadas previamente o cualquier otra causa.

2. Si el académico está dado de alta como empleado, se procede a la captura de la solicitud, en caso contrario se almacena por primera vez (se contrata) al individuo, esto último es otro tipo de trámite.

3. La solicitud que se captura hace referencia principalmente al tipo de solicitud, la causa de la misma, el nivel que depende de la categoría y a su vez la partida a la cual pertenezca, etc; para que dependiendo de lo anterior se consulte el banco de horas, entrando así a una serie de procesos de validaciones y correcciones (en caso de que sea necesario).

4. El banco de horas actuará de acuerdo al tipo de solicitud de que se trate en el periodo correspondiente, se verificará la disponibilidad de horas y se dará continuidad a la solicitud (evaluación) si se cumplen las condiciones, en caso contrario, se detiene la captura.

5. Una vez que la información capturada es válida, se procede a almacenarla como una solicitud. Esta solicitud se almacena en un archivo de TRAMITES,

para posteriormente entre a un proceso de EVALUACION, cuyo resultado será la actualización constante de fechas de seguimiento. Esta actualización se llevará a cabo en el almacén de TRAMITES.

6. Se mantiene un seguimiento para conocer cuándo se encuentra el trámite ó solicitud en qué escritorio. Todo esto es llamado, conocido o encapsulado como un periodo de evaluación, registrando las fechas de cada movimiento. El proceso de EVALUACION consiste en lo siguiente: del almacén de solicitudes, la DEPI consulta y envía un listado de las mismas hacia distintas sedes hospitalarias, estableciendo comunicación con la DEPI sobre las mismas, una vez que las sedes hospitalarias encuentran que las solicitudes se encuentran debidamente requisitadas, se manda(n) la(s) solicitud(es) a la Dirección de Personal Académico (DPA), quien finalmente decidirá si se genera o no la forma única (indispensable para el seguimiento), una vez generada, se convoca al solicitante para que la firme, hecho lo anterior entra a otra evaluación, esta vez por parte del Consejo Técnico, quien finalmente decidirá si se aprueba o no la solicitud, en caso de aprobación se firma el documento por el titular de la Dependencia, mandándose a la DEPI, para que ésta a su vez envíe a la DGP (Dirección General de Personal) las solicitudes aprobadas y se pueda registrar en nómina al solicitante (en caso de que así sea, ya que pueden ser contratados sólo temporalmente), hecho lo anterior se regresa a la DEPI para generar el contrato respectivo y fechas de cobro. *(ver figura 3.)*

7. Si no hubiese sido aceptado el trámite, se liberan las horas comprometidas, pasando a ser nuevamente asignadas (las que pertenecen a la combinación unidad administrativa, nivel, categoría, partida), y la solicitud quedaría almacenada como rechazada (llevando así un histórico). En caso contrario, si se ha aprobado la solicitud, se procede a alterar las horas en el banco de horas (esto se refiere a ocupar o liberar horas) según el tipo de la misma. Estas horas originalmente (cuando se inició el trámite) quedaron en un estatus de comprometidas, ahora son ejercidas (en el caso que sean horas para ocupar y serán disponibles en caso de que sean horas a liberar).

8. Una vez que ha sido concluido el periodo en el cual se aprobó el movimiento ó trámite (vigencia), se procede a almacenar la solicitud como finalizada o concluida (cambio de estatus). De esta forma se alterarán las horas que estaban siendo ejercidas por el trámite respectivo, pasando a ser disponibles.

9. En cualquier momento de operación administrativa están disponibles las acciones de la generación de consultas y reportes, tanto de la información que forma el flujo de procesos como de la información catalogada y que es constante (con menos frecuencia de alteración).

Así pues, se pueden identificar los diferentes momentos administrativos en la situación o problema presentado, que de alguna forma resultan más generales y

que con los cuales se puede partir para comenzar un análisis detallado, por lo que con esto se podrán representar de manera esquemática las variantes y **relaciones** de cada momento.

Nota: Cabe señalar que la figura 3 se encuentra encapsulada dentro de la figura 2 a través del proceso denotado como Evaluación, por lo que en tiempo de diseño, cada proceso puede contener a su vez procesos encapsulados.

2.2 Modelo Entidad – Relación.

De acuerdo a lo anterior se comenzarán a identificar los entes que resultan claves o descriptivos para el presente problema. Cada una de estas situaciones implica la relación de información que a su vez viene a ser más particular y detallada. Estos entes en el contexto de bases de datos y más aún en el modelo **entidad - relación,** se conocen como entidades y la información detallada que implica cada una de éstas son sus respectivos atributos.

Identificación de entidades.

Las entidades se pueden distinguir porque son aquellos entes, características ó situaciones que describen todo el problema de forma conjunta y coadyuvan a que la institución realice los movimientos administrativos correspondientes (en este caso).

Las entidades son todas las variables "globales" o factores que intervienen en el proceso de vida de la institución.

La identificación de las entidades se reduce a buscar aquellas variables, sustantivos ó situaciones que existan dentro de cada tipo de problema y que de alguna manera deberán relacionarse para interactuar en la ejecución del sistema.

Existirán algunas otras entidades que funcionarán como asociativas, esto es, aquellas que no son identificables como sustantivos propiamente dentro del problema, sino que ayudan a "conectar" dos entidades identificadas previamente (estas entidades se vislumbran mejor en el modelo relacional, cuando se traslada o mapea el modelo entidad – relación a su respectivo modelo relacional, y se dan cuando se trata de una relación de muchos a muchos).

Al identificar las entidades, se procede a identificar los atributos de cada una de ellas, esto es también para aquellas entidades que funcionan como asociativas.
Los atributos son todas las características que guarda cada entidad y pueden referirse al comportamiento de la misma como aislada y algunos otros al comportamiento de la misma con respecto a todo el sistema (campos ó llaves de tipo foráneo).

Los atributos de tipo foráneo no son muy visibles o explícitos en el modelo entidad – relación, sino hasta el modelo relacional.

Por lo que una vez identificadas las entidades y los atributos de cada una, el segundo paso es definir las relaciones entre cada una de ellas, así como sus conectividades (cardinalidad ó correspondencia).

Nota: Hay que recordar que la cardinalidad se refiere al mínimo y máximo número de correspondencias que guarda una entidad con respecto a la que se encuentra en el otro extremo de la relación.

A continuación se procederá a identificar las entidades, los sustantivos que son descriptivos para el problema propuesto.

Se tienen las siguientes:

1. Maestro
2. Trámite
3. Materia
4. Sede Hospitalaria
5. Institución
6. Jefes
7. Solicitud
8. Causas
9. Función
10. Nivel
11. Códigos Programáticos
12. Categorías
13. Unidad Administrativa

Los sustantivos anteriores se pueden relacionar a través de un lenguaje coloquial sin entrar en términos de bases de datos todavía de la siguiente manera:

Un profesor puede tener en un determinado período ninguno o varios trámites, los cuales se referirán a un determinado tipo de solicitud por alguna causa, además el profesor en algún trámite determinado puede tener un nivel determinado en alguna categoría y a su vez ésta pertenecer a una partida determinada.

El profesor, dependiendo de cada trámite podrá tener alguna función docente además de la que solicita (labor voluntaria de servicio a la comunidad académica o institucional).

El trámite que se lleva a cabo en turno se refiere a algún movimiento concerniente a alguna materia, dicha materia pertenecerá a una unidad administrativa directamente ó quizás a una subunidad administrativa directamente y como consecuencia ésta última a su respectiva unidad administrativa de la misma

forma. La materia sólo podrá impartirse en ciertas sedes hospitalarias, en donde cada sede hospitalaria pertenecerá a una institución gubernamental.

Existirá personal directivo actualizado constantemente (según sea el actual en ese momento de la institución).

Cada unidad administrativa, partida, categoría, nivel (estas tres últimas son dependientes) y periodo académico almacenarán un conjunto de horas para controlar además del juicio dictaminador la liberación de las horas solicitadas debido a un trámite.

Así pues, de esta forma, se ha enunciado la relación que guarda cada sustantivo mencionado previamente con respecto a otro, para dar con esto un panorama del contexto del problema hablando solamente con respecto a las entidades identificadas anteriormente.

Relaciones y Cardinalidades.

A continuación se comienza a particularizar más en cada situación, estableciendo con más precisión las relaciones y ocurrencias de cada evento y situación en el sistema, así como sus conectividades.

El modelo entidad - relación está entendido de la siguiente forma:

Todo gira en torno a un tramite o solicitud. Esta solicitud puede ser alta o baja, es decir asignación ó liberación de horas para que el solicitante (profesor) pueda impartir clases sobre una materia determinada.

Para el trámite además de especificar el tipo de solicitud, se determina la causa por la cual se lleva a cabo dicho trámite. Existen diferentes causas para cada tipo de solicitud las cuáles irán en función de la misma.

La administración de horas (el proceso de asignar o liberar horas) está determinada por el nivel que tenga el solicitante, la unidad administrativa a la cual pertenezca la materia sobre la cual se trate, la categoría que tiene el solicitante y por la partida a la cual pertenezca. Lo anterior significa que existe un conjunto de horas válidas en un periodo (año escolar) el cual pertenece a la combinación unidad administrativa, nivel, categoría, partida.

Además de lo anterior por cada trámite se debe señalar si el solicitante tiene alguna función, esto es, si además de impartir clases se dedica a alguna otra labor académica para el beneficio de la institución y/o de los alumnos.

Un **maestro** representa una persona que podrá participar en el proceso del sistema, dicho **maestro** podrá tener desde 0 (cero) hasta "n" número de **trámites**;

entendiendo un trámite como un proceso administrativo que va desde el momento que es solicitud hasta convertirse (en caso de ser aprobada) en un movimiento llevado al cabo. El trámite se relaciona a su vez con varias entidades, deberá tener relación solamente con una materia, una causa, un tipo de solicitud, un nivel, una categoría y una partida.

La **materia** se refiere a la asignatura de que se trate, ya sea para asignar o liberar horas de impartición. Cada materia pertenece a uno y sólo un registro de la entidad **unidad administrativa**. Mientras que una **unidad administrativa** (un registro) podrá tener desde uno hasta "n" registros (ocurrencias) en la entidad **materia**.

La relación entre una materia y una sede hospitalaria se define como una relación de muchos a muchos, en este caso nombrada en el diagrama entidad - relación como "imparte"; posteriormente en el modelo relacional se podrá visualizar que esta relación se convierte en una tabla. A su vez una sede hospitalaria se relacionará con una institución pero en una relación de uno a muchos.

Por otra parte, una ocurrencia de la entidad unidad administrativa se relaciona con la misma entidad para expresar un idea de jerarquización. Esto es, la lectura de las conectividades se entiende de la siguiente forma: una unidad administrativa (subunidad administrativa) pertenecerá a solo una unidad administrativa (o a ninguna si es unidad administrativa), mientras que en sentido contrario, una unidad administrativa se relacionará con cero o más unidades administrativas (subunidades administrativas). *Esta relación es conocida como recursiva en la teoría de bases de datos.*

El tipo de solicitud se refiere a:

- Alta.
- Baja.
- Licencia con sueldo.
- Licencia sin sueldo.

Mientras que la causa se refiere al motivo por el cual se lleva a cabo el trámite, dependiendo del tipo de solicitud.

Con respecto al nivel se tiene que por cada trámite que tenga un profesor, dicho trámite tendrá relación con uno y solo un nivel, dicho nivel tendrá a su vez relación con sólo una categoría y ésta a su vez se relacionará con sólo una partida de donde se obtienen los pagos de los profesores. Esta serie de relaciones en sentido inverso se entendería de la siguiente forma:

Una ocurrencia de la entidad **partida** podrá relacionarse con una o hasta "n" **categorías** (ocurrencias), a su vez un **categoría** se relacionará con una o hasta

"n" ocurrencias de la entidad **nivel**. *Se entenderá en el desarrollo de este documento con la palabra "ocurrencias", lo que se refiere en bases de datos a registros.*

Existe un control de horas de donde (además del dictamen del Consejo) se determina el conceder la petición de la solicitud en trámite. Dicho control es por unidad administrativa, partida, categoría y nivel teniendo como información las horas asignadas, comprometidas, ejercidas y disponibles en un periodo específico.

Lo que resulta de la relación de una ocurrencia de la entidad unidad administrativa con la ocurrencia de la entidad nivel es precisamente el control de horas mencionado en el párrafo anterior. Debido a lo anterior esta entidad podría verse como una entidad asociativa.

Diagrama Entidad – Relación.

Se construyó un Diagrama Entidad Relación, el cual es el modelo conceptual de nuestra base de datos.

Se presenta el diagrama entidad - relación, *según lo descrito anteriormente.*

Cada **rectángulo** representa una entidad, cada **círculo** u **óvalo** representa la relación o asociación entre cada entidad según corresponda y cada **línea** conecta a cada relación señalando en cada uno de sus extremos la conectividad o el número de ocurrencias que tendrá un registro en la entidad contraria. A cada asociación se le ha identificado con un nombre, esto es, según sea la acción a realizar entre las entidades que relaciona. *(Ver figura 4).*

Mapeo.

Una entidad representará una tabla, una asociación representará una tabla *sólo si* tiene conectividades en ambos extremos de muchos a muchos, en caso contrario solo conecta a las tablas respectivas, dependiendo de la llave primaria de cada entidad, se generarán las llaves foráneas en cada tabla (en este caso se conoce como padre hijo la que recibe llaves, y tabla padre aquella que otorga la llave primaria en la tabla contraria).

Una vez hecho esto, se dice que el resultado de lo anterior es un modelo relacional. Posteriormente se puede "afinar" el modelo con índices sobre algunos campos (lo que mejorará el acceso), conectividades (borrado en cascada), constraints, triggers (no usados en el presente sistema), etc.; esto último quizás ya sea parte del mantenimiento de la base de datos.

A continuación se describe el modelo relacional, resultado del análisis y diseño desarrollado previamente.

2.3 Modelo Relacional.

Existe una correspondencia entre los objetos de cada modelo. Esto es:

- Las entidades se convierten en tablas.
- Los atributos corresponden a campos.
- Las *entidades asociativas* son transformadas en tablas propias manejándose indistintamente como tales.
- Se identifican explícitamente los campos o llaves de tipo foráneas, así como las llaves primarias (tanto simples como compuestas).
- Las ocurrencias se refieren también como en el *modelo entidad – relación*, a los registros de cada tabla.

Diagrama Relacional.

Se tiene que una ocurrencia de la tabla **maestro** participará o estará relacionada desde cero hasta "n" veces en la tabla **tramite,** y un registro de la tabla **trámite,** se referirá a uno y sólo un registro en la tabla de **maestro.** Así pues un registro de la tabla **trámite** corresponderá con un mínimo de uno y un máximo de uno con las tablas: **solicitud, función, causa, nivel, materia,** y **sede.** *(Ver figura 5).*

Mientras que cada registro de cada una de las tablas mencionadas tendrá una ocurrencia en la tabla **trámite** desde cero hasta "N" veces. Esto significa que una ocurrencia de cada una de estas tablas podrá estar relacionada (tantas veces se requiera) o no (ninguna vez) con un algún registro de la tabla **tramite**.

Para esta serie de relaciones, en la tabla **tramite** existe un campo que funciona como llave foránea para cada una de las diferentes tablas con las cuales se relaciona.

Analizando la tabla **tramite**, se pueden distinguir algunas características que son la aplicación de la teoría de bases de datos, utilizándola como ejemplo.

- Contiene llaves de tipo foráneo (campos) de los cuales algunos no podrán aceptar el valor nulo, cuando sea estrictamente necesaria la conexión con otras tablas, por ejemplo con la tabla **maestro** y aquellos que puedan ser *nulos o vacíos serán aquellas relaciones* que no sean indispensables.

- Llave primaria. Se refiere al campo que de manera única es capaz de identificar cualquier ocurrencia dentro de la entidad y que describe de manera general a la tabla dentro del sistema. En este caso el *número de expediente es*

suficiente para identificar cualquier ocurrencia dentro de la tabla. Para identificar la llave primaria es necesario minimizar todas las posibles causas, por ejemplo, para el diseño de esta instancia, originalmente se pensó que la llave primaria estaría formada por el número de expediente y por el RFC del profesor, sin embargo en el momento de **aplicar las leyes de normalización ó formas normales** (reglas definidas por Edgar Codd y se utilizan para minimizar errores de integridad referencial en la base de datos), se encontró que la segunda ley falló, se modificó la propuesta de la formación de la llave primaria y se cumplió hasta la tercera ley exitosamente.

- Si se ha hablado de relaciones recursivas, de un establecimiento de jerarquización entre ocurrencias de una misma tabla y de un almacenamiento histórico, esta tabla en particular **tramite** podría diseñarse como recursiva (desde luego desde el modelo entidad – relación); sin embargo esto lo define el problema mismo. Se hubiera hecho así sólo si las reglas administrativas requirieran que el número de expediente de un trámite que evidentemente pertenece a un profesor tuviera relación con otro número de expediente sin que necesariamente éste último se relacionara con el mismo profesor.

La tabla **partidas** funciona como padre de las tablas **categorías** y **nivel**. La llave primaria de la tabla **partidas** es heredada a la tabla **categorías,** para que con la clave de categoría juntas formen una llave primaria compuesta, ésta última es heredada a la tabla **nivel,** y de la misma forma, con una clave de **nivel** se forma una llave primaria compuesta en ésta última tabla, es decir los campos partida, clave de categoría y clave de nivel.

Lo anterior se determinó porque existen niveles de diferentes categorías e incluso partidas que se identifican con el mismo nombre y misma clave, por lo tanto por cada nivel es necesario saber a qué categoría y partida pertenece para poder asignar un sueldo respectivo.

Se tiene un catálogo de horas o banco de horas que es la tabla **totalHrs**, la cual se forma por la llave compuesta de la tabla **nivel** y además la llave primaria de la tabla **u_admini** (catálogo de unidades administrativas). Estas columnas forman una llave primaria compuesta propia de la tabla **totalhrs.** De esa forma se puede tener perfectamente identificada la situación en horas que tiene cada unidad administrativa con respecto a un nivel en una categoría y partida específicos en un periodo determinado. Un periodo es un año escolar ó académico dentro del cual se llevan a cabo los trámites.
Con respecto a la tabla **u_admini** se tiene que es una tabla recursiva, esto se refiere a que un registro (hijo) apuntará a otro (padre) de tal forma que se forma una jerarquización de datos, en este caso de unidades administrativas y subunidades. Se diseñó de esta forma por las siguientes razones:

- En una sola tabla se tienen almacenadas *todas las unidades administrativas* con sus respectivas subunidades.
- En caso de crearse una nueva subunidad por medio de los campos apuntadores a registros se puede encontrar y almacenar su respectivo padre.
- Pueden existir subunidades de subunidades, es decir hijos de hasta "n" generaciones.
- Y porque cada registro comparte las mismas características propias de la entidad y sólo se diferencian por la razón de jerarquía.

Cada registro de la tabla **u_admini** podrá tener desde cero hasta "n" registros en la tabla **materia,** esto significa que una unidad administrativa (entendiéndola como "padre" o "hijo") puede tener un número determinado de **materias**. Y que cada materia (asignatura) corresponde única y exclusivamente a un registro de la tabla **u_admini.**

Las materias que están relacionadas con las unidades administrativas, lo están con la sede hospitalaria que es en donde se imparte tal asignatura. En el modelo físico se tiene la tabla **sede.** Debido a que en una sede se pueden impartir varias materias y una materia impartirse en diferentes sedes, se puede visualizar que esta relación *es de muchos a muchos*, por lo que se crea una tercer tabla llamada **sh_mat,** la cual es una tabla asociativa entre las tablas **sede** y **materia.**

Una sede hospitalaria corresponde a una Institución (institución gubernamental o particular) y una institución tendrá desde una hasta "n" sedes hospitalarias. La tabla **inst** funciona como un catálogo de instituciones.
La tabla **jefes** es empleada para almacenar en ella los nombres del director y secretario que en el momento se encuentren en turno dentro de la Facultad de Medicina. Es una tabla que no requiere relación alguna con una tabla específica y cuando dicha información es requerida por algún reporte se accesa al último registro conservando con esto un historial de los directivos de la facultad.

Descripción del Diagrama Relacional.

Se tienen diversos elementos en el diagrama relacional presentado.

Registros: Un registro es una ocurrencia en una determinada situación del problema real, en este caso, de cada tabla. En el diagrama no aparecen.

Columnas: Son los campos o características que guarda cada registro. Algunas columnas guardan ciertas características para identificar de manera única al registro del que se trate, algunas otras columnas sirven para relacionar una

situación con otra y otras columnas sirven para almacenar información propia de la ocurrencia. En el diagrama son los nombres que están dentro de cada tabla.

Tabla :Rectángulo que contiene información sobre algo específico, identificada por un nombre genérico. Contiene las columnas de las ocurrencias que participan en ella. Cada columna tiene asociado un tipo (llave primaria, llave foránea, no nulo) el cual se presenta en cada tabla.

Dentro del sistema existe una tabla que aparece aislada, dicha tabla no requiere de una relación con alguna otra porque simplemente es de consulta para alimentar algunos reportes. La forma en cómo está modelado el problema no se presta a relacionar a los directivos de la institución, sin embargo puede participar dentro del sistema.

Líneas: Representan la relación existente entre una y otra tabla.

Cardinalidad, Conectividad: Es el par de caracteres que tiene cada tabla junto a ella misma.

Esta conectividad representa el número de veces que participará un registro u ocurrencia de la tabla origen en la tabla del otro extremo de la relación. La conectividad expresa un valor mínimo y un valor máximo (valor mínimo, valor máximo). Cuando una conectividad está subrayada significa que la llave primaria de la tabla contraria participará de forma compuesta o a veces sola como una llave primaria en la tabla respectiva. Tal es el caso de la llave primaria compuesta por los campos clave de materia y clave de sede en la tabla SH_MAT, viendo sus conectividades.

Lectura de Conectividades.

La forma más fácil de leer las conectividades es de la siguiente forma (es una forma de representar las relaciones, existen diversas, y en el presente sistema se optó por seguir ésta):

Se supone que se tiene una "tabla padre" llamada A, y una "tabla hijo" llamada B. Junto a la tabla A se tiene un par (conectividad), que son "w", "x". Junto a la tabla B se tiene un par "y", "z". Esto se leería de la siguiente forma:

Un registro de la tabla A (identificado por su llave primaria) existirá en la tabla B como mínimo "w" veces y como máximo "x" veces. Y un registro de la tabla B pertenecerá a la tabla B desde "y" veces hasta "z" veces.

Lo mismo se aplica para cuando una tabla sea recursiva, en tal caso se hace que la tabla A sea igual a B (en la descripción anterior) y se sigue el mismo procedimiento.

Ejemplo 1:

En la Tabla de ***unidad administrativa*** *se tiene una relación recursiva, lo que expresa se puede leer de la siguiente forma:*

"Una ocurrencia de la tabla ***unidad administrativa*** *(porque puede ser unidad o subunidad), puede tener desde "Cero" hasta "N" ocurrencias asociadas o relacionadas a ella (se habla del padre) y (en sentido contrario), una ocurrencia tendrá asociada desde "una" y hasta "una" (debe existir) ocurrencia relacionada a ella (se habla del hijo)."*

Ejemplo 2:

En las tablas ***maestro*** *y* ***trámite*** *se tendría lo siguiente:*

"Una ocurrencia de la tabla ***maestro****, puede participar desde "Cero" hasta "N" veces en la tabla* ***tramite*** *y una ocurrencia de la tabla* ***tramite*** *pertenece o se relaciona a solamente "una" ocurrencia en la tabla Maestro."*

Lo que significa que un profesor puede tener varios trámites, si se encuentra por primera vez ninguno, pero un trámite pertenecerá solamente a un profesor específicamente.

Los ejemplos anteriores denotan la claridad y facilidad que nos proporciona la teoría de bases de datos relacionales para representar un problema. La facilidad y claridad no sólo es para el usuario final sino también para el desarrollador. La representación gráfica es no solamente general sino que expresa la idea que guarda cada situación por medio de las conectividades.

Descripción Física.

De acuerdo a lo que se trató en el capítulo 1 con respecto a la relación existente entre el modelo entidad - relación y el modelo relacional, se presenta a continuación la definición formal del Modelo Relacional, el cual es el modelo físico del sistema.

Se describirán las características y funcionamiento que tiene cada tabla y su relación entre sí.

Tablas.

1. INST: Contiene una clave para identificar a la institución, el nombre de la institución y las siglas de la misma. Funciona como tabla padre de la tabla SEDE.

2. SEDE: Almacena información del nombre de la sede, una clave identificable y un campo que funciona como llave foránea con respecto a la institución a la que pertenece.

3. SH_MAT: Guarda la relación entre materias y sedes para impartirse. Es una tabla conocida como asociativa (en el modelo entidad – relación se refería a una relación de muchos a muchos).

4. MATERIA: Contiene la clave y nombre respectivo de cada materia, así como la clave de la unidad administrativa a la cual pertenece (subunidad si es el caso).

5. U_ADMINI: Dentro de ella se almacena la clave, nombre de la unidad administrativa y la clave de la unidad a la que pertenezca en caso de que sea subunidad el registro de que se trate, si no es así, el valor de este último campo tendrá un valor vacío (nulo). Este tipo de tablas son llamadas "Recursivas", porque cada registro puede tener relación con ninguno o hasta N registros (hablando del registro padre), y solamente uno si se trata del registro hijo.

6. PARTIDAS: Tabla que guarda la clave de cada partida, formando en conjunto con los demás campos lo que se conoce como **código programático**.

7. CATEGORIA: Guarda la clave y descripción de una determinada categoría dependiente de una partida. Para identificar la descripción de una categoría es necesario definir la partida y la clave de categoría.

8. NIVEL: Almacena clave y descripción de niveles de acuerdo a una determinada categoría y a su vez a una determinada partida. Lo anterior da la idea de que un nivel no es identificable por sí solo para establecer un sueldo, ya que esto depende directamente de la categoría y partida a las cuales pertenezca.

9. TOTALHRS: Tiene la funcionalidad de un banco de horas, en donde se guarda el número de horas asignadas (las que se otorgan para disponer de ellas), comprometidas (que están en evaluación por un determinado trámite), ejercidas (que han sido otorgadas por un trámite), disponibles (las que se pueden comprometer, que no están en uso), de acuerdo a la combinación de unidad administrativa, nivel, categoría, partida y un periodo académico (en este caso anual, que va desde agosto hasta julio del próximo año).

10. JEFES: Simplemente almacena los nombres del director y secretario de la institución para alimentar algunos reportes, así como una clave consecutiva para guardar un histórico. No se encontró sentido el relacionarla con alguna otra tabla, debido a que inclusive, administrativamente, un personal directivo, no está vinculado específicamente con una situación solamente, sino que en cualquier momento puede estar habilitado para relacionarse con cualquier situación en general o en particular.

11. SOLICITUD: Catálogo que contiene la clave y descripción de los diferentes tipos de solicitudes que puede tener un trámite.

12. FUNCION: Catálogo que contiene la clave y descripción de las diferentes actividades extracurriculares puede tomar un solicitante.

13. CAUSA: Son las diferentes causas que puede tomar un trámite, las causas son de acuerdo al tipo de solicitud que se seleccione.

14. MAESTRO: Se almacenan los datos personales de cada profesor, cada uno es identificado de manera única por su RFC.

15. TRAMITE: Cada registro en esta tabla se identifica por un número de folio que funciona como llave primaria y que es formado (vía programación) por el año natural en curso con un consecutivo; la secuencia de consecutivos termina cuando termina el año mencionado. Almacena valores de campos pertenecientes a otras tablas que sirven para establecer la relación con cada una de ellas (llaves foráneas), datos particulares del trámite y fechas a través de las cuales se puede tener un seguimiento del trámite, es decir cuándo y dónde estuvo el documento en evaluación.

Diccionario de Datos.

Para toda base de datos relacional se debe tener un panorama totalmente claro del almacenamiento de la información, tanto general como particular. Hasta ahora se han descrito la forma de relacionar la información y su almacenamiento de manera general. A continuación se describe cada campo que constituye a cada tabla, proporcionando el significado de lo que se presentó en el diagrama relacional con el verdadero nombre del campo que tiene en la base de datos (en este caso archivos .dbf), el tipo de datos que almacenará y la longitud máxima del mismo. *(Ver figuras 6 a 10).*

- **Name** es el nombre de descriptivo del mismo campo o de la tabla.
- **Coded Name** es el nombre físico del campo o de la tabla.
- **Row Lenght** es la longitud total de posiciones enteras que ocupa un registro.

- **Cat.** puede ser Base...(campo propio de la tabla), Foreig...(campo que funciona como llave foránea).
- **Domain** es el tipo de dato que puede almacenar el campo correspondiente.
- **Len** es la longitud del campo
- **Dec** es el número de posiciones decimales que puede tener un campo de tipo numérico.

Confrontación con el mundo real.

El esquema físico anteriormente descrito cumple con las siguientes afirmaciones:

- Un Maestro podrá tener desde cero hasta "n" trámites.
- Un trámite es exclusivamente referente a un profesor.
- Un trámite podrá tener *únicamente un tipo de solicitud, una función* (de profesor), una causa del mismo, una materia asociada, una sede hospitalaria, un nivel (y al hablar de nivel, una categoría y una partida respectiva).
- El trámite en turno podrá no tener necesariamente una función relacionada.
- Una partida podrá tener desde una hasta "n" categorías, una categoría podrá tener desde uno hasta "n" niveles.
- Un conjunto de horas (*disponibles, comprometidas, ejercidas, asignadas*) pertenecerá a una unidad administrativa y a un nivel (el nivel será identificado totalmente por la categoría y partida a los cuales pertenezca).
- Una unidad administrativa podrá tener desde cero hasta "n" subunidades administrativas, y una subunidad podrá tener solamente una unidad administrativa (padre).
- En una sede hospitalaria se *imparten desde cero hasta "n" materias*, y una materia se puede impartir en una o en "n" sedes hospitalarias.
- Una institución podrá tener desde una hasta "n" sedes hospitalarias.
- Una unidad administrativa (de cualquier nivel) podrá tener desde una hasta "n" materias y una materia pertenece a una y solo una unidad administrativa.
- El personal directivo podrá interactuar en el sistema de acuerdo al que se encuentre actualmente en ese momento, *manteniendo una historia de aquellos* directivos que han estado activos.

2.4 DISEÑO MODULAR DEL SISTEMA.

Se establecen cuatro módulos principales, que reflejan cuatro modos de operación del sistema ó acciones a llevar a cabo. Dentro de cada uno de ellos se establecen a su vez submódulos que realizan tareas más específicas. Estas tareas pueden ser de captura, de consulta, de actualización, de edición, de despliegue.

- **MAESTRO:** En el cual se lleva a cabo todo el almacenamiento de información referente a un profesor, desde datos personales hasta datos académicos o de

trámites. Dentro de este módulo también se incluyen consultas y modificaciones de la misma información, así como información histórica de apoyos otorgados y finalizados.

- **CATÁLOGOS:** Dentro de este módulo se encuentra información que sirve para alimentar al sistema a través de claves con descripciones que identifican un valor único dentro del catálogo. Esta información catalogada no puede ser modificada desde el sistema por un usuario a menos que tenga la clave de acceso para ello.

- **REPORTES:** Se tienen contemplados reportes predeterminados, tales como la **forma única**, listado de profesores activos con toda la información consecuente y un resumen de horas quincenal por unidad administrativa, partida, categoría y nivel.

- **ADMINISTRACIÓN:** Para habilitar este módulo es necesario teclear la clave de acceso ya que dentro de él se podrá modificar desde el sistema toda la información almacenada. A través de este módulo se podrán generar respaldos de información, alimentar al sistema con nueva información catalogada, actualización del Banco de Horas, cambio de estatus de solicitudes, modificar e inclusive borrar registros.

- **SALIR:** Terminación del programa.

Se ha descrito un diagrama de procesos, el cual pretende plasmar de una manera clara todas y cada una de las acciones que se pueden llevar a cabo dentro del sistema a nivel usuario final. *(Ver figuras 11 a 20).*

Interacción y descripción de módulos.

El módulo CATÁLOGOS proporciona información en forma de consulta a los módulos MAESTROS y CATÁLOGOS. MAESTROS genera información, la cual alimenta a REPORTES. El módulo ADMINISTRACIÓN, es quien podrá actualizar toda la información de los tres módulos de forma arbitraria. Además, en todo momento se podrá invocar el proceso de terminación del sistema. *(Ver figura 11)*

El módulo **Maestros** implica lo siguiente:

Existe el proceso de almacenar nuevos profesores en el sistema, así como actualizar y consultar su información existente personal. Debe existir un profesor para iniciar un trámite asociado al mismo, en tal caso, se almacena un nuevo trámite el cual estará en un periodo de evaluación, manteniendo capacidades para ser consultado y modificado, dependiendo del dictamen; finalmente se almacenará como **rechazado** o **aprobado**. Existe un tercer tipo conocido como

concluido, el cual será afectado directamente por un proceso de administración solamente. *(Ver figura 12).*

El proceso de almacenar nuevos profesores implica a su vez la captura del mismo, la comparación con los profesores existentes y sólo en caso de que no exista el que se capturó, entonces se almacenará. *(Ver figura 13).*

El proceso de almacenar un nuevo trámite implica consultar cierta información catalogada para la captura del mismo, validar la información capturada, validar la relación materia - sede, revisión y validez de las horas correspondientes en el banco de horas; una vez hecho lo anterior, se procede al almacenamiento del nuevo trámite. *(Ver figura 14).*

La edición de los datos académicos consiste en presentar solamente aquellos que se encuentran en un periodo de evaluación, en el momento que adquieren una determinada dictaminación, pasarán a ser identificados por su respectivo estatus *(Ver figura 15).*

El módulo de **Catálogos** contiene a su vez nueve submódulos, en donde cada uno representa un diferente catálogo sobre cierta información en particular. No existe comunicación entre ellos, y es por eso que se presentan aislados. *(Ver figura 16).*

El módulo de **Reportes** consiste en tres procesos generales. *(Ver figura 17)*

- El llamado Resumen de Horas, el cual reúne información de los almacenes Unidades Administrativas, Categorías, Banco de Horas, Niveles y Partidas para generar el correspondiente informe o reporte.

- Forma Unica, que hace uso de la información almacenada en Trámite, diversos catálogos y datos personales para presentar el reporte que se genera cuando es aprobada la solicitud.

- Por Movimientos, en donde se extrae información de Trámite, diversos catálogos y datos personales para mostrar en forma de reporte una especie de historial de todos los trámites expedidos hasta el momento durante todo un año.

El módulo de **Administración** comprende la manipulación de toda la información contenida en toda la base de datos de una forma arbitraria, la transferencia de estado de los tramites concluidos a finalizados y la construcción de respaldos. *(Ver figura18).*

El cambio o transferencia de los trámites aprobados a concluidos consiste en seleccionar todos aquellos trámites cuya vigencia halla caducado y que se

encuentren todavía como aprobados para asignarles el nuevo estado. *(Ver figura 19).*

Por otro lado, la generación del respaldo de la información consiste en considerar el tamaño de la información a respaldar, definir la ruta destino (con ciertas validaciones) y proceder al almacenamiento. *(Ver figura 20).*

Diagrama de árbol.

Todas las acciones posibles a ejecutar pueden ser jerarquizadas, de acuerdo al contexto sobre lo que traten.
Para el usuario final, contar con este panorama puede resultar más claro y entendible, ya que de esta forma conocerá todas las generalidades y particularidades con las que cuenta el presente sistema. *(Ver figuras 21 a 25).*

Por tanto, el sistema para el control de Información del Personal Académico de la DEPI maneja una serie de procesos que pueden ser enunciados jerárquicamente de la siguiente forma:

MAESTROS:

- Alta datos Personales.
  - Captura.
  - Comparación con el almacén (validación).
  - Almacenar el registro.
- Alta Datos Académicos.
  - Validación de Información capturada.
- Edición Datos Personales.
- Edición Datos Académicos.
- Consulta Rechazados.
- Consulta Aprobados.
- Consulta Concluidos.

CATALOGOS:

- Catálogo Solicitudes.
- Catálogo Causas.
- Catálogo Unidades Administrativas.
- Catálogo Subunidades Administrativas.
- Catálogo Materias.
- Catálogo Categorías.
- Catálogo Sedes Hospitalarias.
- Catálogo Niveles.
- Catálogo Funciones.

REPORTES:

- Forma Unica.
- Por Movimientos.
- Resumen de Horas.

ADMINISTRACION:

- Manipulación de Tablas.
- Cambio de trámites Aprobados a Concluidos.
  - Selección de aquellos menores a la fecha actual.
- Construcción de Respaldo.
  - Información sobre el tamaño actual.
  - Captura del subdirectorio (validaciones).

CAPÍTULO 3

DESARROLLO DEL SISTEMA

3.1 ESTRUCTURA DEL SISTEMA.

Dentro de cualquier sistema se pueden identificar diversas variables, algunas que son indispensables, otras que son complementarias, ó quizás algunos procedimientos que sólo sirven para dar un toque estético al sistema, algunas otras que marcan la diferencia entre dos o más sistemas refiriéndose a la competitividad en el mercado existente.

Cada variable está asentada sobre una plataforma la cual es la mejor para llevar a cabo su labor o es su solución o entorno de trabajo, por lo que por ejemplo, veamos la siguiente situación: una variable que tiene información en cierto momento sobre frecuencias auditivas, para que esta variable tomara la información que le concierne hubo un factor que le proporcionó esa información porque la función de ese factor es exclusivamente esa, sería erróneo pensar que el mismo factor en las mismas circunstancias de operación generará información para alimentar variables de tipo térmicas (aquellas que se refieren a la temperatura) por ejemplo.

Lo que se pretende ilustrar con lo anterior es que un sistema para llevar a cabo su trabajo en forma integral requiere de ciertas variables que son alimentadas por diversos factores, trasladando esta idea a un entorno computacional y de forma más particular a un sistema de almacenamiento de información sostenido por conceptos de bases de datos relacionales, se tiene que es necesario emplear diferentes tipos de archivos, así como variables y métodos de acceso, para que de esta forma, se logre una interacción armónica para el buen funcionamiento del sistema.

Tipos de Archivos.

Por lo tanto, los tipos de archivos empleados que interactúan en este sistema son los siguientes:

- Archivos de texto (*.txt)
- Archivos de Dbase (*.dbf)
- Archivos indexados (*.ndx)
- Archivos de código fuente (*.prg)
- Archivo ejecutable (*.exe)

Archivos de Texto.

En este sistema son usados para almacenar dentro de ellos todo el texto correspondiente a la información desplegada como ayuda en tiempo de ejecución del sistema, por lo que pueden ser alterables de manera arbitraria sin modificar o afectar el funcionamiento del sistema. Sólo son archivos con caracteres que no pueden ser alterados o modificados desde el sistema, por lo que son desplegados con la restricción de sólo lectura.

Archivos de Dbase.

Haciendo uso de estos archivos se puede hacer la simulación de tablas y vistas de una base de datos real, por lo que en ellos se almacena toda la información en forma tabular (columnas y renglones), para que posteriormente mediante programación se opere con ellos en consultas, modificaciones, inserciones y borrado de registros.

Se puede decir que estos archivos se usan como tablas y como vistas, siendo las vistas generadas a partir de las tablas, pero finalmente ambos tipos son archivos con extensión .dbf, siendo solamente una convención conceptual su diferenciación.

Archivos Indexados.

Partiendo de un archivo con extensión .dbf, se puede crear un archivo indexado con respecto a uno o varios campos especificados. Esta creación es para optimizar el acceso a determinados registros (consulta), de esta forma se obtiene mejor funcionamiento en tiempo de respuesta.
El proceso de indexación y la búsqueda misma de la información por este medio son algoritmos que ya están determinados y sólo es necesario establecer parámetros tales como definición de los campos para ejecutar su funcionamiento. Sin embargo el uso de un archivo más por cada índice es una desventaja de la cual se habló en capítulos anteriores *(Ver capítulo1)*.

Archivos de Código Fuente.

Son los archivos que contienen instrucciones en lenguaje de Clipper, en ellos es donde se aplican los conceptos de programación estructurada y donde se establecen las ligas con cada tipo de archivo. Después de una compilación se genera un archivo con extensión .obj (archivo listo para ser ligado), después de esto se liga, generándose así un archivo ejecutable (.exe).

Archivo Ejecutable.

Es la conversión de las instrucciones anteriores a lenguaje máquina, es un archivo que no depende de un ambiente específico para ejecutarse (como los ejecutables

de Dbase), sino que lo hace por sí solo, interactuando con archivos que contienen información para complementar la funcionalidad del sistema a través de las ligas establecidas (archivos con extensión .exe).

3.2 INTEGRACIÓN DEL SISTEMA.

Se cuenta con seis programas de código fuente llamados, cada uno de ellos contiene rutinas o sentencias de control para el funcionamiento procedural del sistema.

- **MEDICOS.PRG**
- **HELP.PRG**
- **MED.PRG**
- **REPOR.PRG**
- **ADMINI.PRG**
- **CATAL.PRG.**

Dentro de **MEDICOS.PRG** se encuentra lo que es la función principal, que contiene el menú desplegado en el programa (menú principal) y de acuerdo a la opción seleccionada se llaman a las funciones y procedimientos correspondientes.

El programa tiene habilitada la tecla de función <F1>, la cual al oprimirse manda a llamar según los parámetros establecidos (variable, nombre de función o procedimiento, línea en los cuales se encuentra posicionado el cursor o la ejecución de una acción, lo cual es transparente para el usuario) el archivo de texto correspondiente. Los archivos de texto se encuentran clasificados en estructuras de control dentro del programa fuente llamado **HELP.PRG**.

En **MED.PRG** se tienen todos los procedimientos y funciones utilizados en el primer menú llamado "MAESTROS", tales como actualizaciones inserciones y consultas a la base de datos referentes a trámites y a la información personal de profesores.

Además contiene el código fuente que forma la implementación de todos los procedimientos y funciones que interactúan en el sistema y que de alguna manera coadyuvan al funcionamiento integral del sistema (funciones de claves de acceso, diferentes tipos de funciones de mensajes, definición de cuadros o ventanas).

En **REPOR.PRG** se encuentran las rutinas que sirven para generar los tres posibles reportes considerados en el sistema. Primeramente todo se concentra en una función general que toma la decisión de cualquier ejecución dependiendo de la acción del usuario.

Dentro de **ADMINI.PRG** están consideradas todas las posibles opciones que se presentan en el menú de administración, esto es, edición de tablas, construcción de respaldos y actualización de movimientos.

En el programa **CATAL.PRG** se encuentra el código referente a los nueve catálogos que sirven como consulta y asignación de valores en la captura de trámites administrativos.

La razón de tener seis archivos es para llevar un control más preciso de la ubicación de las funciones y procedimientos que puede ejecutar el programa, además de que se facilita el desarrollo e implementación del mismo en tiempo de compilación, ya que no es necesario compilar todo el programa cuando solamente se han alterado las líneas de alguna función específica. Además para futuras modificaciones del mismo, puede resultar mucho más fácil la alteración de cada proceso ó función que así lo requiera.

Se compila cada archivo con extensión .prg y posteriormente se ligan los seis archivos para formar un solo archivo ejecutable llamado **MEDICOS.EXE,** el cual al ejecutarse es el que va a contener el ambiente del sistema y control del mismo.

El archivo ejecutable tendrá acceso a cada archivo con extensión .dbf para cualquier operación de consulta o actualización en caso de que así sea necesario.

Los archivos indexados son generados en el momento del requerimiento de la consulta, por lo que generalmente al instalarse el sistema por primera vez, no aparecerán en el directorio actual; al generarse estos archivos, las consultas y búsquedas se hacen sobre ellos siendo de esta forma dinámicos.

Se ha integrado el concepto **vistas** en el sistema. Una vista es generada a partir de una tabla cualquiera (en este caso un archivo con extensión .dbf) y está formada por una selección de ciertos registros y campos. Cuando se va a efectuar la consulta de una tabla que tiene muchos registros (diferenciados por un campo estatus), se genera una vista para restringir la búsqueda y para que la recuperación sea más rápida, de esta forma la vista contendrá registros con un sólo estatus, la búsqueda será más directa y por lo tanto más eficiente.

3.3 DESARROLLO DEL SISTEMA.

A continuación se describe el como y cuando se ejecutan los diferentes procedimientos y funciones considerados en el sistema, la secuencia de acciones

a realizar, y la bifurcación de las acciones o decisiones que el sistema es capaz de realizar.

El programa principal (medicos.prg) consiste en una sola función, previo a la ejecución de la misma, se ejecuta una serie de instrucciones que no pertenecen propiamente a alguna función o procedimiento.

Al iniciarse la ejecución del programa, se detecta si el monitor es a color o blanco y negro, dependiendo de lo anterior se inicializan las variables correspondientes para la visualización de los colores, se establecen relaciones entre algunas teclas de función (teclas que estarán ligadas durante toda la ejecución del programa a *una específica acción*) con sus respectivos procedimientos asociados, se inicializan algunas variables globales, después de presentar una "Carátula Informativa" del programa y en ese momento dentro de la función principal, se ejecuta la función "Passw" la cual evaluará si el texto tecleado es válido, de no serlo el programa abortará y regresará al sistema operativo. En caso contrario se despliega un menú de cinco opciones, cada opción representa un módulo. Estos módulos son los siguientes:

Cada módulo presenta un submenú, dentro del cual se presentan las opciones posibles a ejecutar.

MÓDULO DE MAESTROS.

Al seleccionar esta opción se despliega un menú con ocho opciones:

1. Alta de datos personales.

En donde se almacenan datos propios de un profesor *para que posteriormente* pueda participar en el almacenamiento de solicitudes de trámites administrativos. Al ser seleccionada esta opción manda a llamar a la función Altas(), la cual despliega una pantalla de captura con cada uno de los campos correspondientes y al final se pregunta si se procede o no al almacenamiento del registro.

En caso de ser afirmativa la respuesta se lleva a cabo una búsqueda sobre la tabla Maestro del RFC tecleado, si no se encuentra, se almacena el registro, si se encuentra se despliega un mensaje de error. Esto simula la violación de un constraint (regla de integridad referencial) de llave primaria en un esquema de bases de datos real, es decir, que no pueden existir dos ó más registros con el mismo valor de su llave primaria.
En este caso cada registro es identificado por el RFC, y por lo tanto este campo es la llave primaria.

Se han considerado algunas validaciones de captura sobre algunos campos, en el RFC (cuatro caracteres alfabéticos y los demás numéricos, sin ser obligatorios los

de la homoclave), estado civil (soltero, casado, viudo, divorciado, otro), código postal (todos los caracteres deberán ser numéricos) y la nacionalidad (mexicana o extranjera).

2. Alta de datos académicos.

Se almacena el trámite que solicite un profesor y para relacionarlo se usa el RFC del profesor, para esto, considerando que el programa lo manejan personas inexpertas en un contexto de cómputo y para darle facilidad y optimización de captura, en el momento que se solicita dentro de esta pantalla la captura del RFC, se puede invocar una ventana al oprimir <Alt-R>, la cual contiene un listado de todos los RFC´s (con sus respectivos nombres), de los cuales podrá seleccionar el deseado y con esto evitar errores en la captura del mismo. Dentro del trámite se registran número de horas, materia, causa, tipo de solicitud, partida, categoría, nivel, etc; algunos valores son alimentados por medio de catálogos, que pueden ser invocados con la tecla de función correspondiente (la definición de las teclas de función con cada procedimiento se ejecutó en el programa principal, previamente explicado).

Al ser seleccionada esta opción, también despliega una pantalla de captura. De la misma forma se pregunta si se quiere almacenar el registro en caso afirmativo se lleva a cabo una búsqueda del RFC capturado (en la tabla Maestro), si se encuentra se almacena el registro, si no, se despliega un mensaje de error, puede ser debido a una captura errónea del RFC o que no exista en la tabla. Es por eso que previo a la asignación de un trámite a un profesor, es necesario teclear la información personal del mismo. Lo anterior funciona como violación a una llave foránea (Parent not found), es decir, que debe existir un registro padre por cada registro hijo.

La llave primaria de la tabla trámite está formada por un número consecutivo (Secuencia), el cual es construido por la concatenación del los dos últimos dígitos del año, una diagonal y el próximo consecutivo de acuerdo al anterior registrado en la tabla. De esta forma no puede haber error en la identificación de un trámite. Cabe señalar que la construcción de esta clave o folio no es construido sino hasta que se ha validado toda la información capturada, por lo que una vez creado el folio o número de trámite es debido a que la captura y la serie de validaciones fueron exitosas.
Esto último es para evitar la creación de un número cada vez que se validen los datos capturados y desperdiciar números de folio.

Otra característica de la creación de la llave primaria de la tabla **trámite** es que ya se tiene contemplado el cambio de centuria, es decir, actualmente se toman los dos últimos dígitos del año en curso, sin embargo, para el año 2000 los dos últimos dígitos son ceros y para la creación del folio se podría generar un resultado erróneo. Por tal motivo se ha considerado un mecanismo especial de

programación para este caso, formando de la misma forma el folio, con los dos últimos dígitos.

En diversos sistemas, dependiendo de la Institución, se toman diferentes acuerdos ó decisiones acerca de cómo manejar el cambio de siglo, algunos optan por almacenar el año con los cuatro dígitos, algunos siguen considerando los dos últimos dígitos sobreentendiendo con esto que un año del siglo XX no es reconocido; en este caso se optó por esta opción por reglas de la institución simplemente.

Asimismo, antes de ser almacenado el registro, se ejecutan algunas funciones de validación que son la verificación de valores no nulos, es decir valores de campos que no deben estar vacíos, esto último simula la presencia de constraints de campos no nulos, si la función respectiva detecta que un campo que no debe estar vacío lo está, manda un aviso sin almacenar el registro; otra validación es la de consistencias, es decir que la materia esté relacionada con la sede hospitalaria capturada y además que la materia pertenezca a la unidad o subunidad administrativa capturada o seleccionada de catálogos.

Otra validación es que la combinación unidad administrativa, partida, categoría, nivel, periodo cuente con suficientes horas en la tabla **TOTALHRS** (banco de horas), para esto es necesario verificar que las fechas conocidas como "A Partir de" y "Con limite" estén dentro de un periodo escolar y así determinar a qué periodo pertenecerá la solicitud (el periodo escolar se está tomando en cuenta que es desde agosto de un año hasta julio del próximo año), cabe señalar que no siempre un trámite se referirá a peticiones de horas, sino que también se pueden liberar por alguna razón; una validación más que se lleva a cabo es que un profesor no puede tener más de 40 horas en trámites activos en el año académico, para ejecutar esta validación, se buscan todos los números de folio del RFC seleccionado cuyas fechas almacenadas como **a partir de** y **con límite** se encuentren en el periodo académico actual; para ello se ejecuta una función llamada "Periodo_Tramite", la cual se encarga de devolver el valor del periodo al cual pertenecen las dos fechas (fecha inicio, fecha fin) que recibe como parámetros.

Una observación sobre el banco de horas es que solamente se relacionarán el nivel, categoría y la partida con unidades administrativas (padres), debido a que, al seleccionar una subunidad administrativa (en caso de que así sea) en la captura de la solicitud, se identifica inmediatamente la unidad a la cual pertenece y de esta forma se busca y se opera sobre la unidad (padre), afectando con esto a la misma.

3. Edición de datos personales.

Funciona para consultar y/o modificar los datos personales de un profesor. Se ejecuta una función llamada Consulta1(), en donde dentro de un arreglo (vector) se definen las variables (campos) que se desplegarán en forma tabular en la pantalla, a su vez esta función invoca a otra (propia de Clipper) llamada Dbedit(), a la cual se le pasan como parámetros el arreglo mencionado, otro con los encabezados de cada columna, y una función definida por el usuario en donde se especifica cada acción que se puede hacer dentro de la forma tabular (es en donde se interceptan los eventos generados por el usuario desde el dispositivo de entrada): posicionarse hacia arriba, hacia abajo, con las flechas, campos hacia las cuatro direcciones, página hacia arriba, página hacia abajo, detección del primer registro ó del último.
Por ejemplo al llegar al último registro, despliegue de un mensaje de "Ultimo Registro", lo mismo arriba. Este mensaje se presenta por medio de una función llamada "Mensaje" a la cual se le pasa como parámetro el texto y se encarga de desplegarlo con color, dentro de un cuadro. Esta función se usa en todo el programa.

En el caso de que se oprima <Enter> sobre algún registro, se ejecuta una función llamada Modifi() la cual despliega una pantalla de captura con todos los campos con información (según se hallan grabado) para poder ser modificados. Presionando <Esc> se podrá cerrar esta ventana.
En caso de presionar <F2> sobre el listado tabular, se ejecuta una función llamada "Busca1" la cual despliega una ventana de captura pidiendo el RFC a buscar, posteriormente se encarga de buscar sobre el listado la cadena de caracteres tecleada en la ventana anterior, en caso de encontrarlo se posiciona en el registro correspondiente.

Al presionar <F3> se ejecuta una función llamada "Busca2", requiriendo que se tecleé el apellido paterno, se busca y en caso de encontrarlo se posiciona en el correspondiente registro.

Las búsquedas anteriores se llevan a cabo mediante índices según el campo de búsqueda del cual se trate. Así que si por ejemplo, un apellido paterno aparece varias veces en el listado, y se busca por él, el programa se posiciona en el primero que encuentre y debajo de él los demás registros con el mismo apellido.

En el caso de que se presione la tecla <Esc>, desaparece la forma tabular de la pantalla y se transfiere el control al menú anterior. Para esto antes de mostrar la forma tabular se usa una función llamada savescreen(), mandando como parámetros el tamaño de la pantalla a grabar y una variable en donde se guarda. Para recuperarla se utiliza la función restorescreen().

4. Edición de datos académicos.

Sirve para consultar los datos del trámite de un profesor, si no ha sido rechazado ó aprobado el trámite (es decir está en evaluación todavía) se podrán modificar los datos del mismo (generalmente deberán ser las modificaciones sobre las fechas de seguimiento del trámite).

5. Consulta de Rechazados.

6. Consulta de Aprobados.

7. Consulta de Concluidos.

Estos tres últimos submódulos funcionan para consultar las solicitudes que actualmente tienen el estatus correspondiente.

En los últimos cuatro módulos se usa el concepto de "Vistas" (físicamente es un archivo con extensión .dbf que contiene sólo un subconjunto de la tabla tramite.dbf). Se usan cuatro vistas, las cuatro contienen los mismos campos que tiene la tabla trámite. La vista que se usa en **edición datos académicos** es formada por todos aquellos registros que tienen un estatus "En evaluación", la vista que se usa en **consulta de rechazados** es formada por todos aquellos registros que tienen un estatus de "Rechazado" y lo mismo para las otras dos según el estatus.

Estas cuatro opciones al seleccionarse provocan la ejecución de una misma función llamada "Consulta2", cada opción le asigna un valor propio a una variable definida como global, la cual contiene el nombre de la vista a usar. La vista (en este caso cada vista es una tabla temporal, que sólo sirve para visualizar los registros según sea el estatus correspondiente) es la que se usa para visualizar en forma tabular registros y es llenada de acuerdo al valor que tenga la variable global.

Esta forma tabular es construida por la función Dbedit() con los campos y nombres de registros (2 vectores), el nombre de la vista y el nombre de la función definida por el usuario, los cuales son mandados como parámetros.

Al igual que en la consulta de datos personales, la función Dbedit() usa una función definida por el usuario, en este caso se llama "mueve2", se clasifican las acciones a tomar y en caso de oprimir <Enter> y solamente si el listado presentado es de todas las solicitudes en trámite, se podrá acceder a una pantalla de modificaciones, en caso contrario, simplemente se despliega la consulta (forma tabular).

La función "mueve2" responde a la tecla de función <F2> presentando una ventana para capturar el RFC a buscar en el listado presentado. Dependiendo de la opción seleccionada, se busca sobre la vista según el estatus correspondiente.

Cabe señalar que las modificaciones hechas (refiriéndose a la opción de Edición de datos académicos solamente) no se hacen sobre las tablas manejadas como vistas, estas solamente sirven para tener un control más clasificado sobre los tipos de solicitudes que pueden existir.

Las modificaciones se hacen sobre la tabla **trámite**, y cada vez que se ejecuta alguna de estas opciones, se actualiza la vista correspondiente.

Por otra parte, la forma de modificar los estatus de una solicitud o trámite, varía dependiendo de lo que se trate, es decir, la modificación de los estatus de rechazados y aprobados se da en una solicitud desde el módulo o ventana de modificaciones (anteriormente mencionado), mientras que el de concluidos desde el módulo de Administración.

8. Salir.

La opción de **Salir** simplemente borra de la pantalla el menú presentado con las opciones anteriores, recuperando desde una variable la información (modo caracter) de la pantalla grabada (savescreen() y restorescreen()).

MÓDULO DE CATÁLOGOS.

Un catálogo es una información constante durante la operación del sistema y que sirve para alimentar los campos para su captura e identificar las claves capturadas previamente (si se está consultando); cada descripción corresponde a una clave que puede ser numérica o alfanumérica, esta clave solamente sirve para identificar cada descripción de cada catálogo.

Al iniciarse la ejecución del programa, dentro del módulo MEDICOS.PRG se asocian procedimientos con teclas de función, éstas van desde <F4> hasta <F12> consecutivamente. Los procedimientos que se asocian con estas teclas son precisamente las que ejecutan y despliegan la información de catálogos.

Es por eso que al seleccionar la opción de catálogos desde el menú principal, se despliega un submenú listando todos los catálogos con su respectiva "tecla rápida" (que en tiempo de captura se oprime y se despliega en pantalla el catálogo correspondiente, para facilidad del capturista) o desde el menú seleccionándolo directamente.

Al seleccionar la opción de catálogos se ejecuta la función "eligecat()". Después al seleccionar uno de los catálogos (ya sea desde menú o por la respectiva "tecla

rápida"), se ejecuta su respectivo procedimiento. Los catálogos se visualizan a través de una forma tabular en la cual se puede desplazar con las teclas de página o de movimiento. Esta forma es creada por medio de la función Dbedit(), y los nueve catálogos hacen uso de la misma función definida por el usuario llamada "cata". Era ineficiente, crear una función de usuario por cada catálogo, ya que lo único que los diferencia son los campos mismos y la funcionalidad de la forma tabular es la misma para todos. Desde el procedimiento ejecutado se llenan los vectores de nombres y campos según el catálogo del cual se trate y se manda a llamar a la función "cata" con sus parámetros respectivos. De esta forma se economizan muchas líneas de código.

Se manejan nueve catálogos, los cuales son:

- **Catálogo de Solicitudes.** Se muestran todos los tipos de solicitudes que puede tomar un trámite. Corresponde al archivo "solicitu.dbf"
- **Catálogo de Causas.** Se listan todas las posibles causas del trámite, dependiendo del tipo de solicitud el capturista deberá seleccionar una causa determinada. Se leen los datos del archivo "causa.dbf".
- **Catálogo de Unidades Administrativas.** Son todas las posibles unidades administrativas que no tienen "padres". Se crea una vista del archivo "u_admini" en donde los registros son nulos ó vacios, y por esta razón, no se muestra este campo.
- **Catálogo de Subunidades Administrativas.** Se muestran las unidades que tienen "padres", en este caso el nivel máximo de generación es 1 (uno). Debido a que la tabla es recursiva se creó una vista (tabla temporal físicamente) que muestra las subunidades con su respectiva descripción y "padre". Esta vista es creada con todos los registros de la tabla "u_admini" que no contengan en su campo "clv_padre" nulo o vacío.
- ***Catálogo de Materias.*** *Son todas las asignaturas almacenadas en el sistema* identificadas por claves con sus respectivas descripciones. El archivo respectivo es "materia.dbf".
- **Catálogo de Categorías.** Se muestran la descripción y la partida a la que pertenece cada categoría del archivo "categori.dbf", debido a que la llave primaria está compuesta por clave de categoría y partida es necesario desplegar ambos campos para su identificación.
- **Catálogo de Sedes Hospitalarias.** Todas aquellas sedes que se encuentran disponibles para poder impartir en ellas clases. Corresponde al archivo "sede.dbf".
- **Catálogo de Niveles.** Se muestra la clave de nivel, clave de categoría y partida a las cuales pertenece el nivel, asimismo de muestra el sueldo por cada nivel y su descripción. Cada sueldo dependerá de la combinación categoría, partida, nivel ya que estos tres campos forman la llave primaria de esta tabla. Se lee la información desde el archivo "nivel.dbf".
- **Catálogo de Funciones.** Cada profesor puede o no tener funciones adicionales a las que desarrolla en su horario normal, puede impartir

asesorías, participar como organizador en eventos, pertenecer a algún comité, etc. Cada una de éstas es catalogada y puede ser asignada o almacenada en un determinado trámite, debido a que durante la vigencia del trámite desempeñará tal función. Esta información se encuentra almacenada en el archivo "funcion.dbf".

Para poder interactuar, es decir, mostrar y borrar las pantallas de catálogos, se hace uso de las instrucciones savescreen y restorescreen (propias de Clipper).

Los catálogos al igual que toda la información, se podrán corregir y/o aumentar desde el módulo de administración.

MÓDULO DE REPORTES.

Un reporte es un desplegado de información relacionado entre sí, cuyo objetivo es la visualización rápida, eficiente y veraz de la situación de la Institución u organismo en alguna particularidad.

Actualmente se cuenta con la generación de tres reportes ó informes:

1. Forma Única.

Es un informe que se genera una vez que la solicitud ha sido aprobada. Al seleccionarse se ejecuta el procedimiento **edit_tram**, se abre una **vista** con respecto a la tabla **trámite**, esta vista contiene solamente los registros cuyo estatus sea igual a "APROBADO". Al abrir esta vista, se invoca a la función definida por Clipper, llamada Dbedit(), la cual a su vez invoca a "tramitando" (función creada por el programador). Se selecciona uno de los registros mostrados en forma tabular y se genera la **forma única**.

La generación de esta forma implica la consulta (interna del programa) a diferentes tablas, recuperación de información a través de las llaves foráneas almacenadas en la tabla trámite, conversión de la cantidad numérica del sueldo a letras y finalmente la generación por impresora. El sueldo es identificado de acuerdo al nivel categoría y partida al que pertenezca el profesor en ese trámite.

Con respecto a la conversión numérica de números a letras, se tiene lo siguiente:

Se calcula la longitud de la cantidad numérica (leída desde la tabla) y se identifica cuántas posiciones de unidades, decenas y centenas se están ocupando (es posible convertir como máximo hasta una cantidad de 10 dígitos, contando el punto y dos decimales), en caso de que no se complete un conjunto de unidades, decenas y centenas, se "llena" la cadena con ceros, después pasa a un ciclo de lectura para convertir dígito por dígito, a través de funciones llamadas "centenas()", "decenas()" y "unidades()". Cada una de ellas hacen referencia a

arreglos que tienen predefinidos los valores según sea la posición y el dígito que está siendo revisado. Lo anterior se puede hacer hasta 2 veces (esto es con el fin de limitar al ciclo y suponiendo que los trabajadores no ganan más de 9,999,999.99 pesos a la quincena). Los centavos no se convierten a letra, solamente se despliega su cantidad numérica concatenándola con "/00".

2. Por movimientos.

En él se muestran todos los registros que tienen un estatus de "APROBADO", es decir son aquellas solicitudes que fueron otorgadas. Se presentan solamente aquellos registros cuyo número de expediente se refiere al año en curso (en este caso se verifican los dos primeros dígitos del expediente para verificarlo con el año). Para esto primeramente se consulta la tabla "Tramite" para revisar quienes cumplen con las condiciones mencionadas, almacenando los registros en un arreglo; posteriormente se recuperan las descripciones de las llaves foráneas (claves) almacenadas en cada registro (para lo cual se consulta cada tabla del campo al que se refiera).

La información desplegada es la más genérica y descriptiva que identifica a cada trámite.

3. Resumen de horas.

Es un control sobre las horas que tiene disponibles un nivel perteneciente a una categoría y partida con respecto a una unidad administrativa. Este reporte se lleva a cabo quincenalmente, la información es consultada de la tabla "Totalhrs" y debido a que se tiene un periodo (año académico) por cada conjunto de horas, se identifica la fecha en que se genera el reporte y según en el periodo que se encuentre se despliega la información concerniente a tal periodo. Es importante notar que se genera de acuerdo al año académico y no al lectivo.

La generación se hace a través de ciclos, combinando todos los elementos de tres vectores:

- Unidades administrativas (sólo los padres, cuatro registros).
- Partidas (tres registros).
- Categorías (dos registros).

Se hace una especie de producto cartesiano (en sql se le conoce como un query de tipo join; si se permiten valores nulos se le llama outer join), se suma entre elementos de los vectores para dar un subtotal y a través de transferencia de valores entre elementos se forma una matriz que es la que finalmente se presenta.

MÓDULO DE ADMINISTRACIÓN.

Dentro de este módulo se podrá modificar toda la información de todas las tablas, es por eso que para poder entrar a él, es necesario capturar una clave de acceso ya que se despliega en pantalla una ventana que requiere la contraseña inmediatamente después que se selecciona este menú. Si se reconoce la clave, se despliega un menú, en donde cada opción es una tabla de todo el sistema además de otras opciones, en caso contrario no se podrá visualizar el menú.

Modificación arbitraria.

Se invoca una función propia de Clipper llamada Browse(), la cual edita la tabla de que se trate, enviando como parámetros las coordenadas de la ventana de edición. A través de la tecla "Del" se podrán borrar registros *(Ver Fragilidad, Capítulo 4)* con <Enter> sobre un registro se cambian valores y con flecha hacia abajo sobre el último registro se adiciona un nuevo registro.

Para grabar los datos modificados, borrados o anexados se utiliza la instrucción "pack", de esta forma se registrarán las acciones hechas bajo la responsabilidad del usuario, siendo cambios irreversibles (si se borró la información no se podrá recuperar, desde luego, a menos que se lea un respaldo que se halla generado previamente con el sistema, lo cual será explicado más adelante).

Cambio de Estatus.

Otra función con la que cuenta este módulo es con la opción de saber qué solicitudes que se encuentran actualmente como aprobadas, han llegado a la conclusión de su vigencia; en este caso, deberá cambiar el estatus del registro, pasando de un estatus **aprobado** a un estatus de **finalizado**.

Para listar todas las solicitudes con las características mencionadas se utiliza la *función Dbedit(), se crea una vista con todos los registros* cuyo estatus sea "aprobado", dicha vista es la que se edita, una vez que se selecciona el registro que se quiere cambiar de estatus, se manda a ejecutar un procedimiento que hace esta operación.

Es responsabilidad del usuario (que tenga acceso a este módulo), el tener al día este tipo de información; la libre selección de los registros cuya vigencia ha culminado es de mucha ayuda para el usuario, ya que así podrá conocer cuántos y cuáles registros son los que cumplen esta condición.

Evidentemente se liberan las horas respectivas del banco de horas para coincidir con el manejo de horas, esto es, dependiendo del tipo de solicitud al que se refería el trámite (alta o baja según sea el caso).

Respaldo.

Además dentro de este módulo se tiene la facilidad de respaldar la información en cualquier unidad destino (no necesariamente deberá existir el subdirectorio).

El procedimiento de respaldo consiste en lo siguiente:

Una vez que se ha seleccionado esta opción, el programa despliega en pantalla el número de bytes que se requieren para guardar los quince archivos con extensión .dbf (conceptualmente tablas), el usuario podrá elegir entre continuar o abortar la operación en caso de que no se tenga disponible el espacio mencionado. Para calcular el total de espacio ocupado por los quince archivos, se hizo uso de la siguiente expresión:

Valor_Acumulado=Valor_Acumulado+Header()+1+(Lastrec()*Recsize())

donde:

Lastrec() es el número del último registro en la tabla que está en uso.
Recsize() es el número del tamaño en bytes que ocupa cada registro de la tabla en uso.
Header() es el número en bytes que ocupa el nombre del archivo como tal.

Valor Acumulado : es una variable propia del procedimiento, almacena el valor total de cada archivo que está en uso de manera acumulativa.

Siempre se le agrega una unidad más de lo que resulta la expresión.

Posteriormente se presenta una ventana en donde se capturará la ruta destino (especificando unidad con toda la ruta completa), al oprimir <Enter>, se verifica la existencia de espacio suficiente en la unidad especificada y además que realmente exista la ruta capturada, en caso que no exista espacio suficiente, se visualizará en pantalla un mensaje con el aviso correspondiente.

Si no existe la ruta capturada (cualquier nivel ó subdirectorio), el sistema procede a crear la ruta en la unidad especificada. Posteriormente, se almacenarán los quince archivos en la ruta especificada o creada recientemente.

Para determinar si existe espacio suficiente en el destino señalado, se hace uso de la función "Diskspace()" con la instrucción:

"Run &[comando]"

Se ejecuta un comando en el sistema operativo directamente(como si se estuviese posicionado en el prompt del mismo), en este caso cambio de unidad

y/o directorio, con la función Curdir() se verifica la existencia del subdirectorio, esto es, primeramente se asigna con esta función a una variable la ruta actual de trabajo (la original), una vez que se han ejecutado las instrucciones de cambio de unidad y/o de ruta, se compara el valor de la función Curdir() con el de la variable anterior, si son iguales, significa que no existió el subdirectorio capturado, en caso contrario se continúa con el proceso de respaldo normalmente.

La validación de la existencia de la ruta capturada consiste en truncar esta cadena en subcadenas según se encuentre el caracter "\", se ejecuta el comando CD (Change Directory de MS-DOS) para verificar su existencia, en caso de que no exista, el subdirectorio actual será el mismo al que se tenía antes de ejecutar el comando CD, en ese caso se ejecuta el comando MD (Make Directory), una vez creado se vuelve a ejecutar el comando CD para entrar recién subdirectorio creado. Este control se encuentra dentro de un ciclo, que termina hasta construir (si no existe) o posicionarse en el último nivel de la ruta especificada.

Una vez que se efectuó la verificación y validación de todo lo concerniente a los parámetros del módulo de **respaldo**, se cambian la unidad de trabajo (en caso de que así sea) y el subdirectorio capturado hacia la unidad y subdirectorio normal de trabajo para realizar la copia con una instrucción de Clipper (Copy to [ruta]) por cada archivo.

Este módulo ofrece facilidad para el usuario en cuanto a copias de la información, debido a que en el subdirectorio de trabajo, originalmente (cuando se instala por primera vez el sistema) se tendrán solamente quince archivos con extensión .dbf, pero a medida que se hacen consultas, se generarán algunas vistas (que simplemente son nuevos archivos .dbf) y algunos archivos indexados, por lo que alguien que quisiera copiar la información real y única, se tardaría en identificar los archivos que contienen la información realmente importante o base, mientras que en el programa basta con especificar la ruta destino y se copiarán los archivos que realmente son los esenciales para construir el respaldo.

Para lograr la verificación de la existencia de un subdirectorio en una determinada unidad de trabajo (A, B o C), se hizo uso de algunas instrucciones invocadas desde Clipper hacia el sistema operativo. Lo anterior en un lenguaje de bajo nivel (aquellos lenguajes que en donde las instrucciones son a nivel de maquina, es decir, directamente con el procesador) se hubiese podido completar con un número mayor de líneas de código (por ejemplo con interrupciones de ensamblador, hablando de Turbo C), lo que hubiese ocasionado una ejecución más lenta, es por ello que en este sentido, se considera una ventaja y justificación del uso de Clipper.

Teclas de función y combinaciones reconocidas durante la ejecución del sistema.

Por medio de la siguiente instrucción:

set key <tecla o combinación> to <nombre función o procedimiento>

se pueden predefinir teclas que ejecuten un cierto procedimiento o función al ser pulsadas.

Lo anterior no aplica para <F1>, ya que el propio lenguaje Clipper al detectar el presionado de esta tecla, busca el archivo llamado **help.prg** para proceder según las condiciones establecidas. Por lo que esta tecla está reservada para la ejecución de módulos de **ayuda.**

- <F1>: Presentación de una ventana de ayuda en cualquier posicionamiento del cursor o focus.
- <F2>: Búsqueda por RFC en las pantallas de edición de registros.
- <F3>: Búsqueda por apellido paterno en las pantallas de edición de registros.
- <F4>: Presentación del catálogo de solicitudes.
- <F5>: Presentación del catálogo de causas.
- <F6<: Presentación del catálogo de unidades administrativas.
- <F7>: Presentación del catálogo de subunidades administrativas.
- <F8>: Presentación del catálogo de materias.
- <F9>: Presentación del catálogo de categorías.
- <F10>: Presentación del catálogo de sedes hospitalarias.
- <F11>: Presentación del catálogo de niveles.
- <F12>: Presentación del catálogo de funciones.
- <Esc>: Abortar cualquier operación, tales como cerrar alguna ventana, algún menú, finalizar el sistema, truncar captura, cancelar cambios capturados en la pantalla de modificaciones, etc.

- <Alt-R>: Para presentar un listado de los RFC con su respectivo nombre existente en el sistema.

- <Enter>: Para confirmar cualquier operación, para seleccionar una opción o submenú, etc.

- <Page Up>: Presentar página anterior al listado actual en pantalla.

- <Page Down>: *Presentación de la página siguiente* cuando se trate de ventanas que contienen un listado de registros.

- Flechas : Para navegar o posicionarse en cualquier renglón de un menú; en listados para posicionarse sobre el campo ó registro deseado.

Crecimiento e importancia de la información.

Es obvio que el almacenamiento continuo de información en un sistema de datos, ocasionará un crecimiento en el volumen de los datos, de tal forma que es necesario saber administrar de manera óptima la información almacenada, formar información histórica según requiera la institución o las reglas del negocio.

En la actualidad todavía existen empresas o instituciones que conservan aquellos grandes archiveros que además de no tener un almacenamiento y seguridad adecuada de los datos, ocupan demasiado espacio físico, ocasionan el excesivo desperdicio de recursos de papelería, exceso de peso en los inmuebles (a largo plazo ocasionan daños), las búsquedas de la información son además de lentas, inseguras y poco confiables, es muy probable tener demasiada información redundante y un problema (probablemente el mayor) que puede ser irreversiblemente perjudicial para la institución, por ejemplo:

El hecho de tener una copia de toda esa información en caso de algún incendio, temblor, terremoto o simplemente robo significa redoblar todos los recursos (lo cual no es nada fácil con las condiciones expuestas) y por lo tanto benéfico y muy sano para la misma, sin embargo no es así, si ocurriera cualquiera de los hechos anteriormente mencionados, seguramente la Institución comenzaría a declinar o a luchar por restablecer su punto en el cual se encontraba originalmente y comenzar de nuevo, en vez de prevenir estos puntos y proseguir en el desarrollo propio.

Con lo anterior se trata de reflejar un panorama claro y real de la enorme ayuda que significa contar con el almacenamiento eficiente y optimizado que ofrecen las herramientas tanto de software como de hardware, además de las teorías y *métodos que pueden ser usados para administrar los recursos.*

Debido a que el problema que se trata se refiere al almacenamiento de información y por lo tanto se usan los conceptos de bases de datos, es necesario

especificar una relación del crecimiento de la información, para que así se realicen las consideraciones pertinentes.

Como se mencionó en capítulos anteriores, el actual sistema cuenta con quince archivos con extensión .dbf (en teoría de bases de datos, tablas).
Como es de suponerse, cada una de las tablas puede crecer, algunas crecerán más que otras, debido a que algunas sirven sólo para intercambiar insertar y actualizar datos, mientras que otras solo de sirven de consulta. Transportando esta idea general a un esquema real de bases de datos (en este caso de Oracle) se tiene lo siguiente:

Una base de datos tiene un crecimiento de información de dos tipos:

- Dinámico
- Estático

Un **tablespace** es un concepto de bases de datos que se refiere al almacenamiento lógico de los datos. Siendo este espacio una zona física del dispositivo en donde se almacenan los datos.

Se dice que el tipo ***dinámico*** *se refiere a* aquellos datos que se "mueven" con mucha frecuencia en la base de datos, es decir aquellas que con mucha frecuencia ocasionan inserciones y actualizaciones en la base de datos. Mientras que el tipo **estático** se refiere a la información que muy pocas veces se actualizará en la base de datos, es decir, la mayoría de las veces es de consulta y regularmente son catálogos.

Un índice es un mecanismo para el eficiente acceso a los datos almacenados en una determinada tabla. Por lo que estos mecanismos y tablas deben asignarse y definirse dentro de un determinado tipo de tablespace, esto es:

- Tablespace Dinámico para tablas.
- Tablespace Estático para tablas.
- Tablespace Dinámico para índices.
- *Tablespace Estático para* tablas.
- Tablespace para uso de usuarios (como reserva).

Regularmente los índices de cada tabla corresponden al mismo tablespace de la tabla respectiva.

Un ejemplo práctico de lo mencionado anteriormente es el siguiente:

```
RDM Interface   : RDM for ORACLE, version 2.4.3, 1996
Action          : Forward Engineering
Server Name     : Oracle7
RDM File        : rdmcyr10.rdm
RDM Schema      : Model 1.0
Physical File   :
Physical Schema :
Date and Time   : 97/10/09, 06:31:48 pm   USUARIO:SUPCR.
*******************************************************************************
*/

/*******************************************************************************

Options Selected for the Forward Engineering:

        Output Options:
                Interleave Statements: NO
                Lowercase Keywords: NO
                Add Comments in Headers: NO
                All Constraints Through ALTER: NO
        Generation Options:
                Default Values for Columns: NO
                Table and Column Comments: NO
                Physical Storage: YES
                Parallel Options (v7.1): NO
        Referential Integrity: Foreign Keys

*******************************************************************************/


/*******************************************************************************
SECTION: DATABASE
*******************************************************************************
*/
--<none>


/*******************************************************************************
```

```
SECTION: TABLESPACES
********************************************************************************
*/

/*-------------------------------------------------------
CREATE TABLESPACE TS_CATREP_DINAMICO
-------------------------------------------------------
*/
CREATE TABLESPACE TS_CATREP_DINAMICO
  DEFAULT STORAGE (INITIAL 10240 K
           NEXT 100 K
           MINEXTENTS 2
           MAXEXTENTS 121
           PCTINCREASE 0);


/*-------------------------------------------------------
CREATE TABLESPACE TS_CATREP_ESTATICO
-------------------------------------------------------
*/
CREATE TABLESPACE TS_CATREP_ESTATICO
  DEFAULT STORAGE (INITIAL 10240 K
           NEXT 100 K
           MINEXTENTS 2
           MAXEXTENTS 121
           PCTINCREASE 0);


/*-------------------------------------------------------
CREATE TABLESPACE TS_CATREP_IDX_DINAMICO
-------------------------------------------------------
*/
CREATE TABLESPACE TS_CATREP_IDX_DINAMICO
  DEFAULT STORAGE (INITIAL 10240 K
           NEXT 100 K
           MINEXTENTS 2
           MAXEXTENTS 121
           PCTINCREASE 0);


/*-------------------------------------------------------
CREATE TABLESPACE TS_CATREP_IDX_ESTATICO
-------------------------------------------------------
*/
```

```
CREATE TABLESPACE TS_CATREP_IDX_ESTATICO
  DEFAULT STORAGE (INITIAL 1024 K
                   NEXT 100 K
                   MINEXTENTS 2
                   MAXEXTENTS 121
                   PCTINCREASE 0);


/*----------------------------------------------------
CREATE TABLESPACE TS_USERS
----------------------------------------------------
*/
CREATE TABLESPACE TS_USERS
  DEFAULT STORAGE (INITIAL 10 K
                   NEXT 10 K
                   MINEXTENTS 1
                   MAXEXTENTS 121
                   PCTINCREASE 50);

/*****************************************************************************
SECTION: ROLLBACK SEGMENTS
*****************************************************************************
*/
--<none>


/*****************************************************************************
END.
*****************************************************************************
*/
```

El ejemplo anteriormente ilustrado trata de ejemplificar las condiciones, especificaciones y los parámetros que requiere una base de datos para prever, controlar y documentar su crecimiento.

A continuación, en la tabla 1, se presenta una relación del crecimiento que tendrá cada tabla (archivo con extensión .dbf) según el tamaño del registro.

La tercer columna se refiere al lugar en donde se ubicarían las respectivas tablas tomando el concepto de tablespace dinámico y estático. Con lo anterior, esta misma columna proporciona una idea del crecimiento de la base de datos, es decir entre más tablas estáticas se consideren, menos crecimiento se tendrá y de esta forma se optimizará espacio. *(Ver gráfica en la figura 26).*

Así pues, lo que es necesario realmente, es implementar un modelo de tal forma que cuente con una normalización adecuada para lograr accesos más rápidos y un óptimo almacenamiento.

NOMBRE DE TABLA	TAMAÑO EN BYTES POR REGISTRO	DINÁMICO (D) ESTÁTICO (E)
CATEGORI	64	E
CAUSA	28	E
CO_PRO_P	13	E
FUNCION	29	E
INSTITU	49	E
JEFES	123	E
MAESTRO	206	D
MATERIA	53	E
NIVEL	51	E
SEDE	55	E
SH_MATER	15	E
SOLICITU	14	E
TOTALHRS	52	D
TRAMITE	292	D
U_ADMIN!	39	E

Tabla 1

Con lo anterior se ilustra la importancia y ventaja total que ofrecen las herramientas que proporcionan tanto el software como el hardware, en donde, sabiendo administrar los recursos se puede lograr un rendimiento ó desempeño (performance) incluso competitivo.

Cuando la información ha crecido demasiado (porque así debe de ser por la naturaleza misma del sistema y condiciones), es necesario hacer uso de dispositivos que almacenan cantidades enormes de información, algunos de estos dispositivos pueden ser cintas ó cartuchos que pueden almacenar hasta "n" número de gigabytes (10^{12}). Por supuesto es mucho más fácil almacenar y conservar información en disquetes, cartucho o cintas, así como crear archivos de respaldo de la información como prevención de posibles pérdidas y seguridad de la misma.

Con esto se hace notoria la importancia que significa el adecuado manejo de la información para la institución y la importancia por sí misma.

Reestructuración.

En todo sistema siempre es necesaria la retroalimentación en cuanto a requerimientos y funcionamiento de diversos módulos entre aquellos que desarrollan el mismo y los usuarios. Es muy común encontrarse con la situación de que una vez terminado el producto, los requerimientos han aumentado o a veces han cambiado. Por lo que siempre es necesario tener un panorama del sistema lo más general posible en el cual exista la capacidad para modificar en forma radical inclusive la estructura del mismo.

Lo anterior se refiere a cambios por parte del usuario, además de éstos, existen aquellos que se refieren a la competitividad en el mercado.

¿Qué hacer cuando la competencia tiene mejor tiempo de respuesta?, ¿Cómo responder a la mejor calidad que ofrece la competencia?, ¿Copiar las técnicas de desarrollo y producción de la competencia ó implementar nuevas?. Esta serie de preguntas nos conllevan a seguir y fijar la vista en una meta: aumentar la capacidad para competir en el mercado y utilizar al máximo los recursos disponibles mediante la reducción de costos (**Reingeniería**).

En el caso del desarrollo de sistemas se busca mejorar los procesos, muchas veces es mejor comenzar un diseño por n - ésima vez desde cero que realizar modificaciones sobre módulos que por el momento satisfacen la necesidad pero que en un futuro resulta dudosa su funcionalidad.

Algunos expertos consideran que existen siete condiciones que deben formar parte del proceso de reingeniería para que llegue a un término exitoso:

1. Habilidad para orientar el proceso de reingeniería de acuerdo con una metodología sistemática y amplia.
2. Administración coordinada del cambio para todas las funciones del negocio que se vean afectadas.
3. Habilidad para evaluar, planear e implementar el cambio sobre una base continua.
4. Habilidad para analizar el impacto total de los cambios propuestos.
5. Habilidad para visualizar y simular los cambios propuestos.
6. Habilidad para utilizar estos modelos sobre una base continua.
7. Habilidad para asociar entre sí todos los parámetros administrativos de la compañía.

Sin embargo pueden variar las formas de desarrollo según las necesidades y objetivos de cada empresa ó esencia del sistema. Por tanto, es preciso saber perfectamente en qué situación se encuentra la empresa y hasta qué punto se debe llegar, asentando todo sobre bases firmes y realistas.

Los costos de los nuevos sistemas se han reducido a una fracción de lo que pudieron ser debido al mejoramiento introducido con la reingeniería y con los CASE.

Se considera parte del desarrollo de este sistema el tema porque puede darse el caso que en un futuro los movimientos administrativos de la DEPI puedan cambiar y entonces sea necesario aplicar un proceso de reingeniería al presente sistema, o más aún, que precisamente esos cambios administrativos se realicen por medio de procesos amplios de reingeniería.

3.4 CODIFICACIÓN DEL SISTEMA.

El programa fuente está codificado en lenguaje Clipper versión 5.2. En general se constituye por un programa principal, un programa de funciones y otro de control de ayuda.

La forma en cómo se unen estos programas es a través del comando (que se ejecuta desde el prompt del sistema operativo):

```
rtlink fi programa1,programa2, programa3, programa4, programa5, programa6
```

Cada programa que se envía como parámetro al comando rtlink es un archivo con extensión .obj, el cual es el resultado de la compilación de cada uno de su respectivo programa fuente (en lenguaje Clipper). La compilación es a través del comando "Clipper (parámetro)", en donde el parámetro es el nombre del archivo fuente con extensión .prg, dicho comando es ejecutado desde el prompt del sistema operativo.

Para editar y codificar los programas se hizo uso del editor de MS-DOS llamado "EDIT". No se requiere de un editor especial para compilar los programas fuente en Clipper, solamente aquel que guarde la información como texto sin ningún caracter adicional (formatos de centrado, de espaciado, de párrafos, caracteres especiales, etc.).

A continuación se presenta una porción del código fuente del módulo principal del sistema.

```
#Include"c:\clipper5\include\Box.ch"
#Include"c:\clipper5\include\Inkey.ch"
#Include"c:\clipper5\include\Dbedit.ch"
// Checa el tipo de pantalla para el uso de los colores


  If iscolor()
     public Borrado:="w+/b,n/w"
     public Color:="w/bg"
     public Mens :="w/r"
     public Colonn3:="w/b"
     public Colonn :="b/bg,n/w"
     public Colorear:="w/bg"
     public Cuadro:="b/bg,n/w,,,w+/b"
     public Letras:="n/bg,n/w,,,n/b"
     public Sinprompt:="w/b,n/w,,,n/b"
     public Menuhoriz :="b/bg,n/w"  // ,,,w/b
     public Decara :="w/b"
     public Colorin2:="w+/r"
     public Sale :="w/n,n/w,,,n/w"
  else
     public Borrado :="w/n,n/w,n,n,n/w"
     public Color :="w+/n,n/w,n,n,n/w"
     public Mens :="w+/n,n/w,n,n,n/w"
     public Colonn :="w/n,n/w,n,n,n/w"
     public Colorear :="w/n,n/w,n,n,n/w"
     public Cuadro :="w+/n,n/w,n,n,n/w"
     public Letras :="w+/n,n/w,n,n,n/w"
     public Sinprompt :="w/n,n/w,n,n,n/w"
     public Menuhoriz:="w+/n,n/w,n,n,n/w"
     public Decara :="w+/n"
     public Colorin2 :="w+/n+"
     public Colorin3.:="w/n"
     public Sale :="w+/n,n/w,n,n,n/w"
  endif


  setcolor(Borrado)              // * Color estandar de fondo
  clear

  setcolor(Cuadro)               // * Color de para encabezado de
◊ Sistema
  Ventana(5,5,15,75)
  @ 8,16 say "-- Division de Posgrado de la Facultad de Medicina --"
  setcolor(Letras)               // * Color para las letras de cuadro
  @ 10,25 say "     ( Proyecto de Tesis )"
  @ 6,10 say "Sistema para el Control de la Informacion del Personal
◊ Academico"
  @ 14,21 say "Presione cualquier tecla para continuar ."
```



```
set cursor off
Berror()
wait ""
setcolor(Sinprompt)            // * Elimina el prompt
set cursor on
//setcolor(Borrado)            // *  Color estandar de fondo
clear
set date french

// ** ** Inicio de Procedimientos ** **
set key K_F12 to funcion
set key K_F11 to nivel
set key K_F10 to sedes
set key K_F9 to categ
set key K_F8 to materias
set key K_F7 to su_acad
set key K_F6 to u_acad
set key K_F5 to causas
set key K_F4 to solicitudes
SET KEY K_ALT_R to edita_rfc
set wrap on
public labase =0,entra
public cat:=0,actual =0
public c[10],d[10],d1[6],u[10]
public cantidad,numeros
store space(30) to cantidad
store space(9) to numeros
store space(3) to bloque
STORE SPACE(08) TO VISTA

c[1].=" "
c[2] ="CIENTO"
c[3] ="DOSCIENTOS"
c[4] ="TRESCIENTOS"
c[5] ="CUATROCIENTOS"
c[6].="QUINIENTOS"
c[7]:="SEISCIENTOS"
c[8]:="SETECIENTOS"
c[9]:="OCHOCIENTOS"
c[10] ="NOVECIENTOS"

d[1].=" "
d[2] ="DIECI"
d[3] ="VEINTI"
d[4]:="TREINTA"
d[5] ="CUARENTA"
d[6]:="CINCUENTA"
d[7] ="SESENTA"
d[8].="SETENTA"
d[9]:="OCHENTA"
```

```
d[10]="NOVENTA"

d1[1]="DIEZ"
d1[2]="ONCE"
d1[3]="DOCE"
d1[4]="TRECE"
d1[5]="CATORCE"
d1[6]="QUINCE"

u[1]=" "
u[2]="UN"
u[3]="DOS"
u[4]="TRES"
u[5]="CUATRO"
u[6]="CINCO"
u[7]="SEIS"
u[8]="SIETE"
u[9]="OCHO"
u[10]="NUEVE"
Principal()

procedure Principal()
op=1
 clear
 entra=passw()
if entra==1
 do while .t.
  Cara()
  setcolor(Menuhoriz)
  ventana(1,5,3,74)
  @ 1,7 SAY "Fecha "
  ??date()
  @ 1,63 SAY "<F1> Ayuda"

  set message to 24 center
  @ 2,7 prompt "Maestro" message "Manejo de Informacion sobre solicitudes
  y profesores"
  @ 2,20 prompt "Catalogos" message "Informacion identificada por
  claves"
  @ 2,34 prompt "Reportes" message "Visualiza reportes predeterminados
  por impresora "
  @ 2,47 prompt "Administracion" message "Alteracion arbitraria de
  informacion "
  @ 2,65 prompt "Salir" message "Finaliza la sesión."
  menu to op
  do case
     case op=1
          Eleccion()
          Setcolor(Borrado)
      case op=2
```



```
          Elegecat()
          Setcolor(Borrado)
     case op=3
          Reportes()
          Setcolor(Borrado)
     case op=4
          entra =password()
          if entra==1
            Administracion()
            Setcolor(Borrado)
          else
            Mensaje(    'No puedo entrar')
          endif
     otherwise
            close all
            setcolor(Sale)
            clear
            quit

     endcase

  enddo
else
  Mensaje( 'Clave Invalida')
  Mensaje( 'No puedo Continuar . ')
  Setcolor(Sale)
  clear
  quit
endif
Return
```

CAPÍTULO 4

PRUEBAS DE VERIFICACIÓN Y VALIDACIÓN DEL SISTEMA

Idealmente, los procesos para el desarrollo de software deberían llevarnos a generar sistemas que cumplieran con las necesidades del negocio, sin embargo, dado que no hay proceso perfecto, sería irreal desarrollar software y no probarlo.

En la gran mayoría de las organizaciones el proceso de pruebas es necesario para compensar las deficiencias ya sea de los procesos o de la forma en la que éstos son realizados.

Podríamos entender con pruebas, como el proceso de mostrar que un programa o sistema desarrolla todas las funciones en forma correcta ó que demuestra la ejecución de las funciones para las que fue construido.

El enfoque de ejecución de pruebas para comprobar que algo trabaja, es no hacer nada para disminuir la probabilidad de encontrar defectos existentes. Siempre que en el proceso de desarrollo y mantenimiento se sigan introduciendo defectos en el software, el costo total de desarrollo implicará un fuerte costo de retrabajar o reconstruir los componentes.

El concepto de pruebas ha pasado de un proceso de evaluación posterior a la construcción de programas, a un concepto en que el proceso de pruebas es parte integral de cada fase del ciclo de vida del desarrollo de sistemas. No es un eslabón más, sino una parte vital del ciclo de vida.

Cualquier proyecto de desarrollo necesita de una **definición de requerimientos.** En este caso en particular, el manejo tanto de entrada como de salida debe ser confiable, seguro y claro, así como la interacción entre el usuario y el programa (especialmente por los usuarios que tienen poca o ninguna experiencia en el manejo de sistemas de cómputo).

La definición de requerimientos derivará en lo siguiente:

- **Requerimientos Funcionales.-** Atributo medible y factible de ser probado que debe ser poseído por una función que será realizada por el producto.

- **Requerimientos Estructurales.-** Atributo técnico y de infraestructura del sistema a desarrollar, que debe ser medible y factible de ser probado.

Con relación al **contenido** de los requerimientos, existe controversia en cuanto a la necesidad de que sean "completos y sin ambigüedades", mantenerlos sin cambios, sin embargo, este punto de vista es contrario a la naturaleza misma del

software, ya que precisamente nace para atender necesidades del usuario ó para solucionar problemas y por lo tanto es susceptible de cambios.

Es indudable que el producto evoluciona en el tiempo con las necesidades de los usuarios, por lo que un sistema puesto en marcha se encuentra en constante mantenimiento.

De acuerdo a una determinada convención, se distinguen dos grandes métodos de pruebas que se realizan en las fases de los proyectos de sistemas:

Las **pruebas estáticas** son el proceso de revisión y validación de la *documentación desarrollada* en las diferentes fases del ciclo de vida del proyecto. Una de las más importantes actividades que se deben realizar en este tipo de pruebas es la **revisión y validación de la documentación del proyecto,** entre otras:

- Objetivos del proyecto.
- Requerimientos.
- Diseño.
- Especificaciones de programas.
- Código.
- Objetivos de pruebas.

Las **pruebas dinámicas** son la ejecución real de los programas y puede ser vista como un pequeño ciclo de vida en sí mismo; son una "**serie de pasos**" que inician con pequeños volúmenes de datos y un alto número de ejecuciones (pruebas unitarias ó de módulos), que continúan hacia grandes volúmenes de datos con *menor número de ejecuciones.*

4.1 PRUEBAS DE VERIFICACIÓN.

Cualquier problema en general, para su solución, debe primeramente de representarse de alguna forma para poder visualizar su tamaño, sus variantes, límites, de tal forma que visto en un panorama esquemático o representativo se pueda llegar no sólo a una solución, sino poder tener varias opciones a tomar, de tal forma que se solucione así el problema.

Dependiendo del tipo de problema, se podrán tener **diferentes formas de representación**, en nuestro caso, se pueden mencionar las siguientes:

- Lenguaje natural. - Fuente misma por parte del usuario, donde se establecen las fronteras del sistema.
- *Lenguaje estructurado.* - *Vocabulario específico* relativo al problema o requerimiento.

- Modelo de estructura de datos. - Modelo que exprese la forma en como se relacionan y almacenan los datos.

- Modelo de flujo de datos. - Todos los procesos reales, vistos de manera general y procesos que puede tomar la información.

- Diccionario de datos. - Descripción detallada del archivo, tablas o lugar conceptual físico en donde reside la información.

Las anteriores formas de representación pueden combinarse y es común ver en algunos proyectos el lenguaje natural, con modelos de flujo de datos y diccionario de datos. Es importante tomar en cuenta que existen herramientas CASE que nos auxilian en la creación y mantenimiento de la mayoría de las formas de representación mencionadas.

La razón de hacer mención de formas de representación de sistemas en general, es porque las pruebas de verificación tomarán como punto de partida estas características, en donde se confrontarán las situaciones, procesos y problemas reales para arrojar datos que confirmen la generación de las salidas y los resultados del sistema.

Estas pruebas se cumplen en el momento que se confrontan los movimientos administrativos con toda la información plasmada en distintos tipos de diagramas y estos describen fielmente a la situación real. Es por eso que aquí se vislumbra y enfatiza la importancia de contar con un diagrama de procesos *(figuras 2, 3, y 11 a 20)* el cual identifique de manera general y particular (procesos hijo y padre), un diagrama que represente todas y cada una de las relaciones con cada entidad en un contexto de conceptualización *(figura 4)*, un diagrama en el cual se tenga un panorama claro y específico sobre lo que se tiene físicamente *(figura 5)*. y la globalización de los diferentes procesos que pueden interactuar en la ejecución del sistema según se quiera proceder. *(Figura 21 a 25).*

4.2 PRUEBAS DE MÓDULOS.

Es la categoría de pruebas que con más frecuencia se ejecutan en los proyectos de sistemas (en muchos casos nunca planeadas ó documentadas) y son las primeras pruebas de ejecución de código de un programa o componente.

Lo primero que se debe hacer es desarrollar un plan de trabajo, que debe evolucionar durante el diseño, la codificación y las revisiones detalladas.

Al emplear **casos de prueba**, sabemos lo que deben hacer y por consecuencia, ya tenemos desarrollados los resultados esperados. Es fundamental evitar el

desperdicio o desarrollo de casos de prueba tipo "úsese y tírese". Si se está diseñando un programa y sus pruebas, debemos proteger el plan de pruebas; ejecutar la prueba conforme al plan y si es necesario actualizarlo (en el ambiente línea, las pruebas unitarias equivalen a la prueba de una tarea o de una unidad de *trabajo y su documentación* no es fácil, pero es la única manera de mostrar como "corrió" (se ejecutó) la prueba y cuales fueron sus resultados.

La documentación debe consistir en el plan de pruebas, con casos de prueba, resultados esperados y los resultados obtenidos.

Una vez finalizada esta fase, se procede a llevar a cabo las pruebas de integración, mencionadas anteriormente para avanzar en la implantación del sistema.

Es importante considerar distintos casos de prueba, variarlos, para no caer en un módulo desarrollado específicamente para un caso; se deberán considerar todas las posibilidades (aún la que parezca más insignificante) para terminar un producto más robusto.

En el caso de la función que crea subdirectorios, fue necesario que el parámetro que recibe, que es la ruta completa, se fragmentara, para poder verificar la existencia de cada nivel de subdirectorio, dependiendo de ello tomaba una acción alternativa, crear el subdirectorio ó entrar a él.

Esto da una idea clara de la importancia de considerar todas las posibilidades para que un módulo trabaje realmente en toda su extensión, ya que son variantes que sirven para mejorar su funcionamiento. Además es necesario delegar responsabilidades a cada función, en cuanto a toma de decisiones, para poder en un futuro (en la etapa de pruebas de integración) contar con un módulo confiable, que garantice siempre la generación de sus resultados y mantener un desarrollo modular para futuras posibles modificaciones.

4.3 PRUEBAS DE INTEGRACIÓN.

Son las realizadas a todo el sistema o a un grupo lógico de programas para probarlos en forma simultánea con la intención de asegurar que los datos y controles sean pasados adecuadamente entre programas e identificar problemas que surgen por la interacción o comunicación entre ellos.

Uno de los problemas con este tipo de pruebas es pensar que tienen el mismo objetivo que las pruebas de sistema. Realmente las pruebas de integración deben ejecutarse en fases, por ejemplo: en la fabricación de automóviles, el motor se prueba una vez que se tiene armado, antes de instalarlo en el automóvil.

Las pruebas aplicativas se deben ver de la misma forma y ejecutar varias pruebas de integración al nivel de subsistemas, para identificar defectos y corregirlos al menor costo (en este caso el costo se refiere a pérdida de tiempo por alguna variable no considerada que ocasione una salida errónea y trabajar con información falsa).
Las pruebas deben ejecutarse en forma estructurada.

Los pasos que se siguen en el flujo lógico de una prueba de integración son los siguientes:

1. Diseño de casos de prueba de programa.
2. Ejecución de la prueba de integración.
3. Verificar resultados.
4. ¿Errores detectados?
5. Si el paso anterior es verdadero, se procede a la depuración de el (los) programa(s), en caso contrario se libera la etapa de pruebas de integración.
6. Ejecución de prueba(s) unitaria(s).
7. Se regresa al punto (2).

Durante los pasos 1, 2 y 3, se está constantemente llevando a cabo la documentación de las mismas. *(Ver figura 27)*

En resumen, se toman los casos de prueba originales y se actualizan para la prueba. Se utilizan los planes para ejecutar las pruebas y nuevamente se registran los resultados en la documentación de pruebas. Al identificar los defectos, se corrigen y se vuelven a ejecutar las pruebas para verificar las correcciones.

Debido a que el desarrollo del sistema se basa en procedimientos y funciones, este tipo de pruebas contribuye de manera creciente a la implantación total del sistema.

Un ejemplo de ello fue la conversión de cantidades numéricas a letras para reportar los sueldos con letra. Se hizo una simulación más con un parseo (revisión de caracter por caracter) de la cantidad numérica, dependiendo de la posición en donde se encontraba el apuntador (elemento del arreglo actual), se mandaba a llamar ya fuera a la función de centenas, decenas o unidades según el caso, una vez hecho lo anterior, se iba concatenando cada subcadena generada por el

"parser" (revisión sintáctica) con la cadena resultante. Fue necesario establecer la comunicación entre cada módulo y su correcta sincronización, para que finalmente se reportara la cantidad final.

Otro ejemplo de este tipo es el que se presentó en la creación de respaldos; se tiene que una función general se comunica con otra para obtener información acerca del tamaño en bytes que ocupa actualmente la información, se informa al *usuario, se pide la ruta destino y se verifica que la ruta tenga* los caracteres necesarios para reconocerse cada ruta (en caso de no contar con lo necesario, se corrige internamente), después se verifica que realmente exista espacio suficiente en el destino comunicándose con otra función, la cual determinará pasar el control a otra función con ciertos mensajes para que esta última busque la ruta especificada, si no existe, será ésta la que invoque a otra función a su vez, la cual tendrá como objetivo la creación en la unidad especificada del subdirectorio que no se encontró.

Por lo que esto es un claro ejemplo de la necesidad de llevar a cabo pruebas de integración, para establecer buena comunicación entre elementos ó módulos del sistema.

Pruebas de volumen.

Generalmente este tipo de pruebas es inducido por cambios en el consumo de recursos y se ejecutan para conocer los efectos de un cambio, por ejemplo: en el software, en la administración de los archivos o en las bases de datos del sistema.

Este tipo de pruebas se ejecuta para identificar errores relacionados con el tiempo de respuesta, almacenamiento, tiempos de acceso, concurrencia, etc.

Existen herramientas que pueden ayudar a los propósitos de las pruebas de volumen y de otra forma, este tipo de pruebas se hace con los volúmenes reales de información requeridos para un crecimiento esperado, por lo cual el plan de pruebas debe incluir la recuperación o respaldo de la información necesaria.

Ha sido necesario almacenar volúmenes un tanto grandes de información para simular la realidad del funcionamiento del sistema, evidentemente, en función del tamaño de información, se decrementará la velocidad y tiempo de respuesta del sistema. Con estas pruebas, se ha logrado la afinación de algunos detalles, especialmente creación de índices y vistas *(Ver capítulo 1)* para el más rápido acceso.

Sin embargo, es claro que a medida que avance la cantidad de información almacenada, se hará más lenta la búsqueda y recuperación de información, por lo que se recomienda en este caso, establecer límites de tiempo para la renovación

que se encuentre almacenada en los archivos, creando respaldos, información histórica, dependiendo de la necesidad de la institución.

Pruebas del sistema.

Son una combinación de pruebas en un ambiente muy similar a producción que incluyen pruebas funcionales y que al ser realizadas exitosamente, garantizan que el sistema cumple con los requerimientos.

Viendo el sistema como un todo, ha sido necesario efectuar pruebas de este tipo, de tal forma que se busque la falla de cualquier caso que no se halla considerado, con lo cual se podrán tener aún mejoras en cada uno de los procesos, sin perder de vista que son afinaciones y que contribuirán a la robustez, madurez y fortalecimiento del mismo sistema.

Una ejemplo de estas pruebas es el almacenamiento masivo de profesores al sistema, posteriormente la asignación de trámites de todos ellos (capturando exhaustivamente, haciendo uso de las teclas predefinidas), en donde cada uno podrá tener más de uno considerando las validaciones pertinentes, generar formas únicas de acuerdo a su aprobación, generar informes del banco de horas para consultar los movimientos ó situación que se tiene de acuerdo a cada solicitud, revisión de vigencia de cada trámite (en caso de terminación, cambio de estatus), creación de respaldos en distintas rutas, haciendo búsquedas, revisando que las validaciones consideradas realmente ayuden al eficiente manejo y almacenamiento de la información, etc.

Pruebas de aceptación.

Son las pruebas finales con la participación del usuario (él las ejecuta), antes de la instalación en producción del sistema, con el objeto de asegurar que las necesidades o requerimientos originales del negocio se satisfacen y generalmente son realizadas en una fase del proyecto en la que no se tiene el tiempo de reaccionar.

De no satisfacerse las necesidades o requerimientos originales, el problema surge cuando se debe determinar lo que se debe hacer, por lo que pueden ocasionar instalaciones de sistemas con una lista de mejoras desde antes de que inicie la producción del sistema.

Idealmente, las pruebas de aceptación deben ser similares a la prueba de manejo de un nuevo automóvil. No deben presentarse interrupciones, probablemente se tengan algunos pocos cambios menores o ajustes, pero no deben presentar problemas críticos, ya que si aparecen problemas o se reportan defectos, significa que las pruebas anteriores no fueron adecuadas, que no se ejecutaron

debidamente, que no se especificaron totalmente los requerimientos ó que no se efectuó una debida comunicación.

4.4 VALIDACIÓN DEL SISTEMA.

La puesta en marcha del sistema, de acuerdo a lo expuesto y desarrollado, se debe a la relación de distintos temas, conceptos, teorías, metodologías. El sistema no funciona solamente debido al manejo de variables, o interacción de instrucciones en diferentes módulos de programación, sino que además participan elementos teóricos desde el análisis y diseño hasta la implementación e interacción con el usuario.

Por lo que el adecuado funcionamiento del sistema no se refiere solamente al hecho de que el programa presente buena estética o amigabilidad con el usuario sino que además halla sido representado de manera amplia (y esto no implica redundancia), modelado de manera óptima y clara para posibles cambios futuros.

La forma en como interactúa Clipper con el programador es sencilla, existen muchas funciones que ya están predefinidas y que sólo esperan valores paramétricos para arrojar algún dato o resultado requerido, por lo que se ahorran muchas líneas de código fuente.

El sistema en sí, sin ser gráfico (GUI- Graphical User Interface), ofrece al usuario al trabajar en modo caracter un ambiente amigable y modular con ayuda interactiva. Por otra parte, existen validaciones que contribuyen a la disminución de errores en la manipulación de la información.

Con esto también se ilustra que no es necesario hasta cierto punto hacer uso de software que es más complejo y completo, sino que es suficiente hacer uso de una herramienta sencilla (y no por ello sin importancia) para satisfacer la necesidad o dar solución al problema, y en ello va implícita la selección de software, en donde, en función de la complejidad del problema se debe seleccionar la adecuada herramienta.

La esencia e importancia del presente sistema es que es un problema diseñado con conceptos de bases de datos relacionales, actualmente la mayoría de los proyectos en grandes, medianas y pequeñas empresas, hacen uso de estos conceptos, debido a su claridad de interpretación, confiabilidad, facilidad de modelaje y accesibilidad. Son conceptos que son muy aplicables en la práctica y que por lo tanto satisfacen al ser humano las necesidades de organización y almacenamiento de información.

Fragilidad.

Por otra parte, desde un principio, se fijaron alcances y fronteras; una de las limitaciones se refirió al no contar con un verdadero manejador de bases de datos, sobre esta característica se comentará lo siguiente:

Supóngase que se desea eliminar un profesor de la tabla "MAESTRO", y este profesor tiene relacionados "n" trámites en la tabla "TRAMITE", al eliminar el registro de la "tabla padre", los registros pertenecientes al registro eliminado y que están almacenados en la "tabla hijo" seguirán allí y no tendrán ningún significado en la representación del mundo real con el modelo relacional diseñado.

Por tanto, en el momento de eliminar un registro de una "tabla padre" debe ejecutarse una regla de integridad conocida como "constraint" en una verdadera base de datos para evitar la inconsistencia mencionada anteriormente.

Actualmente el sistema, en particular, en el módulo de administración es muy abierto, en el sentido de permitir alterar la información arbitrariamente, no considera reglas de validación, ni simulación de "constraints" (reglas sobre los datos desde la base de datos vía programación), por lo que por ese lado presenta una cierta fragilidad en el manejo de la información, depositando toda la responsabilidad en el usuario administrador.

Mantenimiento.

Dentro de este capítulo se comentó acerca del mantenimiento constante que sufre todo sistema. Durante el mantenimiento que se puede considerar en el presente sistema, se vislumbra un cambio considerable que en nada afecta la implementación del mismo y que puede ser una herramienta útil para el usuario administrador.

Esta adición en la funcionalidad del sistema se refiere a la verificación de la integridad de la información, es decir, una nueva opción dentro del módulo de administración que se encargue de buscar todos aquellos registros que se encuentran en todas las "tablas hijo" y que no exista su respectivo registro en su "tabla padre". Finalmente generando un informe del resultado, teniendo como opción la corrección de los errores en la base de datos.

CAPÍTULO 5

MANUAL DE USUARIO

Descripción del sistema.

A continuación se presenta un panorama general del sistema desde una perspectiva del sistema operativo y modular.

El sistema para el control administrativo del personal académico de la División de Estudios de Posgrado de la Facultad de Medicina consta de los siguientes módulos para su funcionamiento:

- Programa ejecutable: **MEDICOS.EXE**.
- Archivos de Ayuda: **ARCHIVOS.TXT** (33 archivos).
- Tablas de almacenamiento de datos: **ARCHIVOS.DBF** (15 tablas).
- Archivos indexados (generados por el propio sistema según se realicen búsquedas).

Estos tres módulos deberán de instalarse en un mismo subdirectorio, ya sea en raíz o en alguno propio. Se puede ejecutar desde una unidad que no sea desde el disco duro, pero la ejecución evidentemente será más lenta, por lo que se recomienda instalarse en disco duro.

La información desde un punto de vista conceptual, funcional y real.

La información que gira en torno al sistema y que coadyuva para su funcionamiento integral puede ser de diferentes tipos:

- La que es próxima a ser almacenada como nueva, la cual generalmente serán datos de profesores.
- La que es propia del sistema y permite alimentar los movimientos administrativos de la institución. Es aquella que se encuentra en catálogos.
- La que es generada a través de consultas por pantalla. Generalmente se refiere al desplegado de datos personales y académicos de un profesor.
- La que es generada por reportes o compuesta. Es información relacionada por mecanismos de programación y engloba información de más de una tabla (archivo con extensión .dbf en este caso).
- La que es desplegada a través de la ayuda interactiva y es de sólo lectura.

- La que es finalmente almacenada en todas y cada una de las tablas, la cual es el conjunto o interacción de las mencionadas anteriormente.

Se distinguen estos tipos de información para que se sitúe el usuario o lector en el tipo de información que maneja el sistema y a la forma de cómo se puede obtener, estableciendo así las fronteras y alcances del manejo de información que el sistema presenta.

Vida del sistema.

Debido a que el presente sistema se diseñó de acuerdo a un periodo académico, se pueden distinguir tres etapas en este periodo:

1. Inicialización de la información (comienzo del ciclo académico).

2. Operación del sistema.

3. Cierre del ciclo académico.

Inicialización de la información.

Al inicio del ciclo académico (o si se prefiere a finales del anterior), se deberá inicializar la información, de tal forma que la misma se refiera al ciclo académico por iniciar.
Se pueden enunciar los nombres de las tablas que con más frecuencia a inicio de un periodo académico deberán ser reinicializadas:

- TRAMITE.
- TOTALHRS.
- MAESTRO.

Ya que la persona que se encargue de manejar el módulo de administración es la responsable de mantener la información congruente, será ella quien determine qué información es válida para el próximo periodo académico, esto es, de acuerdo al valor que contenga el campo "status" en la tabla "Tramite", se podrán identificar aquellos trámites que hallan culminado o que no se encuentren vigentes para entonces, por lo que serán éstos los que se deberán eliminar.

En teoría **todos** los registros de la tabla "Tramite" deberán ser eliminados (ya que todos pertenecen a un periodo académico específico), sin embargo será el administrador quien determine esta acción arbitrariamente.

Con respecto a la tabla TOTALHRS, se actualizarán los números de horas correspondientes a la combinación unidad administrativa – partida – categoría – nivel, en cada campo según se especifique (horas disponibles, horas asignadas).

Por otra parte, el administrador deberá conocer e identificar a los profesores que para el periodo próximo a iniciar, ya no participarán por cualquier razón; en tal caso, deberá eliminar físicamente los registros correspondientes de la tabla MAESTRO y además asegurarse que no exista ningún registro (hijo) relacionado al mismo en la tabla TRAMITE (para cuidar la integridad de los datos).

Precaución.- *Es de vital importancia el asegurar que ningún registro de la tabla MAESTRO que halla sido eliminado físicamente se relacione con uno o varios registros en la tabla TRAMITE para* ***evitar información inconsistente****; por lo que se recomienda borrar primeramente los registros correspondientes de la tabla TRAMITE y posteriormente el registro "padre" de la tabla MAESTRO.*

Las restantes tablas serán actualizables no necesariamente al inicio del periodo académico, debido a que se considera que su contenido no varía mucho y es más estática. *(Ver Capítulo 3. Crecimiento e importancia de la Información).*

Operación del sistema.

Se refiere a la ejecución del sistema con toda la información suficiente para su adecuado funcionamiento, en donde el sistema responderá a la ejecución de consultas, inserciones, actualizaciones, búsquedas, generación de informes, generación de respaldos, administración, etc.

Cierre del ciclo académico.

Al finalizar el ciclo académico se **deberán** generar respaldos de la información (para seguridad de la institución ó información histórica), *posteriormente*, se llevará a cabo la depuración de la información, llegando con esto a un periodo de transición del periodo académico actual con el próximo (según el punto 1).

Los tres puntos anteriores presentan un panorama general del ciclo de vida del sistema en donde estrictamente se consideran dos momentos (visto desde un punto de vista funcional):

- Actualización de la información.
- Renovación ó inicialización de la información.

Además se resalta la importancia de conservar actualizada la información para el sano funcionamiento del sistema, así como el conservar la integridad de la misma para que la información almacenada en la base de datos represente fielmente al problema o situación en el mundo real.

Nota: Cualquier palabra que se encuentre entre los símbolos < > (picoparéntesis), significa que es el nombre de la tecla que puede ser pulsada u oprimida para ejecutar la acción correspondiente. También pueden ser combinaciones de teclas.

Instalación.

Existirá un disquete que contiene almacenados los siguientes tipos de archivos:

- Archivos con extensión .dbf (los que contendrán toda la información).
- Archivos con extensión .txt (archivos de tipo texto, utilizados para la ayuda interactiva).
- Archivos con extensión .prg (contienen el código fuente de todo el programa).
- Archivo con extensión .exe (el sistema ejecutable).

La instalación simplemente se refiere a la copia de todos los archivos anteriormente mencionados en el disco duro, dentro de un mismo subdirectorio. Para hacer la copia de todos se puede capturar la siguiente instrucción:

```
c:\copy a:\*.* c:\medicos
```

la instrucción anterior es un comando del sistema operativo DOS, lo que ocasionará que todos los archivos se copien en el subdirectorio medicos.

Los requerimientos mínimos para instalarse y ejecutarse son los siguientes:

- Computadora 80286.
- 2 MB en RAM.
- 1 MB en disco duro.
- Para almacenamiento posterior se recomienda mínimo lo doble, según la administración e información que se requiera almacenar.

Teclas de control.

Dentro del sistema se tienen consideradas ciertas acciones que están ligadas exclusivamente con ciertas teclas. Estas están habilitadas en cualquier posicionamiento del cursor (cuadro intermitente ó acción a realizar) dentro del sistema y son importantes para casos en los cuales no se tenga una idea clara sobre la acción a tomar (ayuda interactiva).

Cabe señalar que simplemente con pulsar la tecla <Esc> (Escape), si el cursor está situado en cualquiera de estos menús, el sistema terminará su ejecución, devolviendo así el control al sistema operativo (desde donde originalmente se ejecutó la inicialización del sistema). En general esta tecla funcionará para cancelar la operación que se esté realizando en cualquier momento y/o lugar del programa o para cerrar cualquier ventana.

Otra tecla que siempre estará habilitada es la tecla de función <F1>, la cual desplegará ventanas de ayuda en todo el sistema. Esta ventana de ayuda desplegará información acerca del objeto, pantalla, campo o menú en donde se encuentre posicionado el cursor (pequeño cuadro que parpadea en la pantalla ó zona iluminada dentro del sistema que se mueve de acuerdo al movimiento de las teclas marcadas con flechas). De esta forma se establece un funcionamiento de ayuda interactiva entre el usuario y el sistema.

Para almacenar un registro en el sistema se recomienda que siempre se ejecute una previa búsqueda de la persona o registro que se requiere almacenar para evitar duplicación de información (aunque el sistema ejecuta ciertas validaciones), maximizar el ahorro de espacio físico en disco y evitar inconsistencia de la *información*.

Teclas de movimiento en diversas pantallas.

Dentro de cualquier ventana que se presente en el sistema, ya sea debido a la pulsación de cualquier tecla de función, combinación de teclas, ó selección desde un menú, se podrán hacer uso de las siguientes teclas para controlar cualquier acción que se desee realizar:

<Esc>: Cierra la ventana abierta, (sólo en la captura de profesores y trámites ocasionará la aparición de una pregunta decisiva en la parte inferior de la misma, dependiendo de la respuesta se procederá), cancela cambios.
<Enter>: Confirmación de una operación ó una pregunta.
<Tab>: Tabulador, en pantallas de captura funciona para posicionar el cursor en el siguiente campo.
<Shift-Tab>: (Combinación) Para posicionar el cursor en el campo inmediato anterior.
<Flecha abajo>: En pantallas de captura sirve para posicionar el cursor en el siguiente campo.
<Flecha arriba>: Para posicionar el cursor en el campo inmediato anterior.
<Flecha izquierda>: En ventanas de captura es para el desplazamiento del cursor dentro del campo hacia la izquierda.
<Flecha derecha>: En ventanas de captura es para el desplazamiento del cursor dentro del campo hacia la derecha.

<Page Up>: En ventanas que muestran listados ó catálogos se utiliza para mostrar los siguientes registros.
<Page Down>: En ventanas que muestran listados ó catálogos se utiliza para mostrar los registros anteriores.

Al estar el cursor en el menú principal, se despliega en el inferior de la pantalla una ayuda rápida, tratando de explicar en pocas palabras la función que lleva a cabo cada una de las opciones de este menú. Los módulos son los siguientes:

Ejecución.

Una vez que el sistema ha sido instalado adecuadamente, se procede a ejecutar el programa de la siguiente manera:

Ya que el sistema se instaló en el subdirectorio llamado medicos, se capturará lo siguiente desde el sistema operativo:

C:\>cd medicos

C:\>medicos>medicos

La primera instrucción sirve para entrar al subdirectorio nombrado medicos, en donde se encuentra el archivo ejecutable que en este caso se llama medicos.

Después de capturar cada instrucción se presiona <Enter> para ejecutar el comando.

Inmediatamente aparecerá una pantalla de presentación del sistema, para continuar presione cualquier tecla, luego, aparecerá una ventana en donde será necesario teclear la contraseña para poder continuar con la ejecución del programa. Al teclear en esa ventana cada caracter será visualizado con asteriscos con el objeto de que la clave no se visualice en pantalla; queda inhabilitada la tecla <Backspace>, así que en caso de equivocación, no se podrá corregir el texto capturado y esto provocará el desplegado con el mensaje: "No puedo continuar..." por lo que el programa regresará el control al sistema operativo.

En caso contrario aparecerá una pantalla con cinco menús, los cuales clasifican *de manera general los módulos* ó diferentes acciones del programa a ejecutar.

La imagen 1 es una copia de la pantalla que se debería visualizar al ejecutar lo anteriormente descrito, es la pantalla principal del sistema.

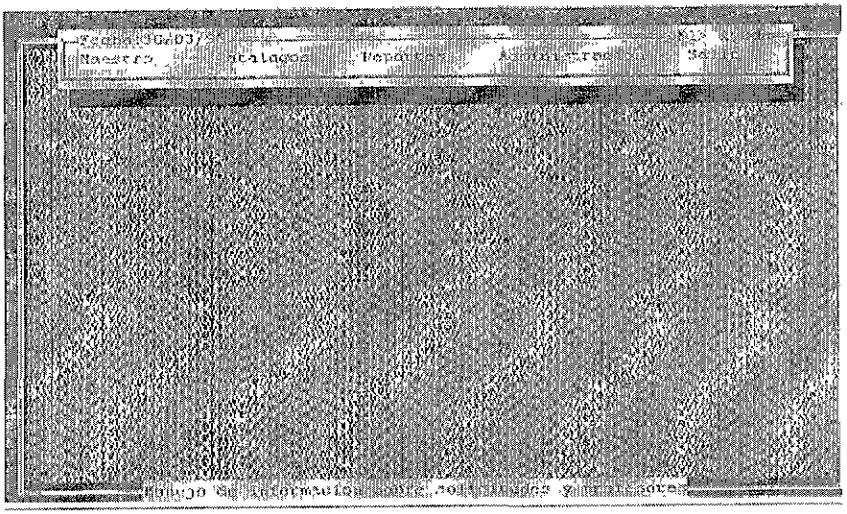

Imagen 1

Módulo Maestros.

En donde se podrá consultar, modificar o insertar información sobre un maestro, ya sea de tipo académico o personal. Este módulo se puede dividir a su vez en los siguientes submódulos:

- Inserción.- Almacenar por primera vez información al sistema.
- Consulta.- Desplegar en pantalla información de acuerdo a algún tipo discriminatorio.
- Edición o Modificación.- Modificar información previamente capturada, contando con la opción de ejecutar búsquedas bajo varios criterios.

A su vez, cada submódulo tiene la misma funcionalidad para diferente tipo de información, así se tiene:

Inserción.

- Se puede insertar por primera vez a un profesor. Por medio de programación se verifica que realmente no exista el registro a almacenar.
- Se puede insertar el trámite o solicitud de un profesor previamente almacenado para lo cual también se tienen ciertas validaciones.

Consulta.

- Se consultan todos los registros que han sido aprobados de acuerdo a la suficiencia de horas, validaciones y dictamen del comité.

- Se consultan todos los registros que han sido rechazados de acuerdo al dictamen del comité. En caso de no haber suficiencia en el banco de horas para dar trámite a la solicitud, ni siquiera se almacena tal solicitud, por lo que solamente habrá rechazos almacenados por causa del comité dictaminador.

- Se consultan todos los registros cuyo periodo de solicitud vigente ha concluido, lo que significa que estas solicitudes (que realmente ya no son porque fueron aprobadas y en ese momento dejan de ser solicitudes pasando a ser movimientos reales) fueron aprobadas. Es importante que la persona que sea responsable del módulo de administración tenga al día esta información, revisando los movimientos que se vencen de acuerdo a la fecha de vigencia con la fecha actual (esta pantalla se describe en el módulo de administración).

Edición o Modificación.

- Se edita o modifica información personal de un profesor, restringiendo la actualización a algunos campos desde este módulo. Esta modificación puede llevarse a efecto en cualquier momento.

- Se puede editar o modificar información de un trámite almacenado previamente, en donde generalmente se modificarán fechas que proporcionan información de cuando y donde se encuentra el documento. En el momento que el trámite ha sido aprobado o rechazado y se registra en el sistema, no podrá ser modificado, simplemente podrá ser consultado.

Operación del módulo Maestros.

Posicionándose con las teclas marcadas con "flechas" (son cuatro, apuntan hacia la izquierda, derecha, arriba, abajo) sobre el menú "Maestros" y después presionando <Enter>, se presenta un menú con ocho submódulos, dependiendo de la acción que desee tomar, también con las "flechas" y después presionando la tecla <Enter> se podrá ejecutar.

La pantalla real del sistema se ilustra en la imagen 2.

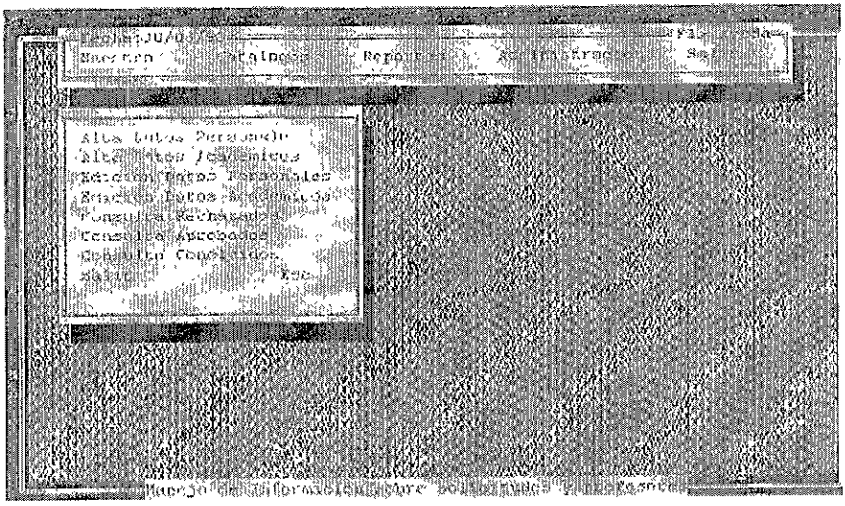

Imagen 2

Alta Datos Personales.

Se refiere al almacenamiento de profesores que por primera vez entrarán al programa. Al seleccionarlo, se presenta una pantalla en la cual se presenta un espacio por cada campo, con lo que se pide cada uno de los datos generales o personales del profesor. Para el RFC se tiene una especie de mascarilla, es decir que los primeros cuatro dígitos **deberán** ser letras y los demás números. Con la tecla de tabulador o "flecha abajo" podrá desplazar el cursor (zona iluminada para escribir) a través de los diferentes campos hacia abajo, para regresarse a los campos anteriores lo podrá hacer con la "flecha" hacia arriba o presionando al mismo tiempo <Shift> y <Tab>. Para desplazarse dentro de cada campo, será con las flechas izquierda o derecha según se requiera.

En los campos sexo, estado civil y nacionalidad será necesario teclear un caracter válido (con <F1>, que es la función asignada para invocar la ayuda interactiva, podrá consultar la lista de los valores válidos), de lo contrario no podrá proseguir con la captura. En caso de que ya no se requiera llevar a cabo el almacenamiento de la captura actual, con <Esc> se podrá anular esta operación, se preguntará en la parte inferior de la pantalla si se desea almacenar el registro, allí se podrá decir que no ("n" o "N") y el programa regresará el control al menú anterior. Al terminar de teclear toda la información se hace la misma pregunta (arriba mencionada) y se procede al almacenamiento. En la imagen 3 se muestra la pantalla a visualizar en pantalla.

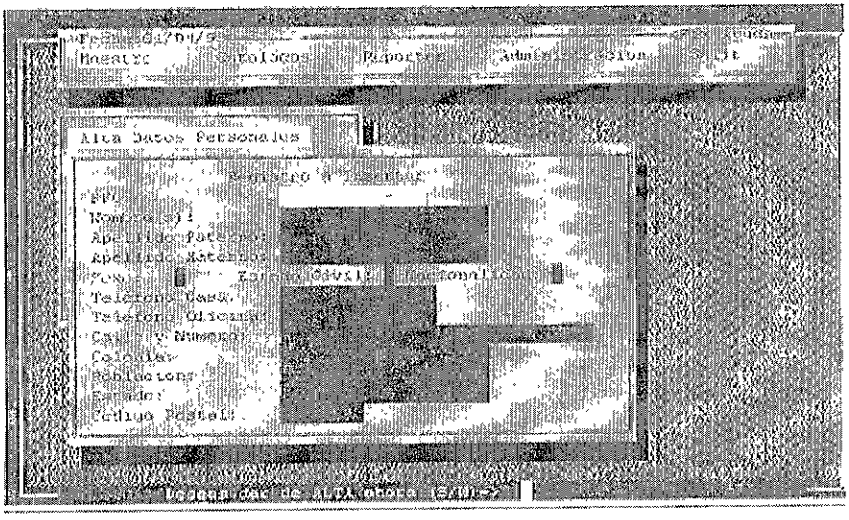

Imagen 3

El programa internamente hace una búsqueda del RFC capturado, si lo encuentra manda un mensaje diciendo que no se podrá almacenar el registro (debido a que el programa entiende esto como una duplicación de información) y da oportunidad para modificarlo (en caso de error de captura). Por tanto, no se podrá almacenar un registro cuyo RFC exista previamente. Se recomienda que las letras del RFC se capturen con mayúsculas simplemente para una mejor presentación, ya que el programa de cualquier forma las convertirá internamente en el momento del almacenamiento a mayúsculas. Además, no se podrá almacenar el registro si el RFC no está debidamente capturado, con sus diez ó trece (en caso de que no tenga homoclave el RFC) caracteres requeridos, de los cuales, los primeros cuatro deberán ser alfabéticos, y todos los demás numéricos.

Alta Datos Académicos.

Tiene como objetivo almacenar una solicitud de trámite.
Al presionar <Enter> sobre esta opción, aparecerá una pantalla en donde se capturará toda la información correspondiente, según el campo.

El primer campo que se presenta es el RFC; para su segura, confiable y rápida captura se recomienda presionar **<Alt-R>**, lo que ocasionará la presentación de una ventana en pantalla que contendrá un listado de todos los RFC´s que se pueden seleccionar para almacenar en un trámite. Presionando <Enter> sobre el registro deseado (para fijar el valor sombreado) y posteriormente <Esc> (para cerrar la ventana de RFC´s), se asignará el valor del RFC seleccionado sobre el campo de la pantalla de captura. Sin embargo si se desea, se puede teclear

directamente el RFC teniendo cuidado de las validaciones correspondientes mencionadas anteriormente.

Existe información requerida en esta pantalla que puede ser alimentada (recomendada para facilidad del capturista) por medio de catálogos, los cuales son invocados desde esta pantalla dependiendo de la tecla de función que se presione. Cada campo que tiene asociado un catálogo, tiene a su lado entre paréntesis la tecla de función correspondiente, de tal forma que al estar posicionado el cursor sobre el campo a capturar, si se presiona la tecla de función correspondiente, ocasionará que aparezca una ventana con el listado de todos los posibles valores que puede tomar el campo en turno.

Oprimiendo <Enter> (para seleccionar el registro) sobre el registro que se encuentre iluminado por el cursor, y posteriormente tecleando <Esc> (para cerrar la ventana del catálogo), en el campo de captura aparecerá la clave del valor seleccionado.
Por lo que toda la información que se requiera almacenar con catálogos, se almacenará con claves numéricas. En los reportes es donde se podrá visualizar la información con la traducción de las claves correspondientes, es decir con su descripción.

En el caso de nivel, categoría y partida al presionar la tecla de función correspondiente sobre el campo de la clave del nivel, automáticamente se asignará el valor respectivo de la categoría y la partida de que se trate el nivel seleccionado, siendo necesario solamente teclear <Tab> sobre los campos de categoria y partida para refrescar los valores asignados a los campos en pantalla y proseguir con la captura. Se menciona lo anterior porque el catálogo de nivel comprende la información de estos tres campos; el usuario podrá seleccionar estas combinaciones inclusive viendo el sueldo relacionado al puesto. Lo mismo ocurre al seleccionar el valor de la materia, se asigna automáticamente el valor de la unidad y subunidad (si es el caso) a la cual pertenece. La pantalla a visualizar en el sistema es la que se muestra en la imagen 4.

Esta pantalla de captura también contiene algunas validaciones:

- Mascarilla sobre el campo de RFC: (primeros cuatro caracteres alfabéticos, y los demás numéricos).
- En el campo de Partida aceptación de solamente tres partidas: 111,117 ó 142.
- En el campo de Afecta Nómina solamente "S" ó "N".
- Fechas en el formato "D/M/A". (Validando días y meses).

Con respecto a la segunda y tercera validación mencionadas, no se podrá continuar con la captura hasta introducir algún valor de los mencionados.

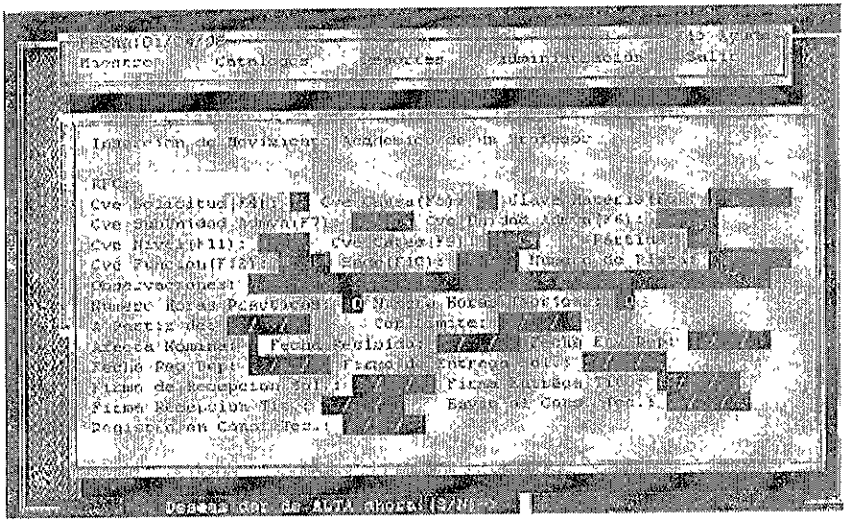

Imagen 4

Al presionar <Esc> sobre cualquiera de los campos, se transfiere el cursor al final de la pantalla para preguntar si se desea almacenar otro registro, ya que el programa entiende con esto que no se desea almacenar el registro que se estaba capturando, si se contesta que no, desaparece la pantalla de captura, regresando el control al menú anterior; si la respuesta es afirmativa se posiciona el cursor en el primer campo de la pantalla de captura para dar oportunidad de almacenar un registro nuevo.

Si la captura de todos los campos se lleva a cabo (que así debe de ser, excepto para las últimas fechas por ser de seguimiento), se pregunta en el inferior de la pantalla si se desea almacenar el registro, si la respuesta es afirmativa, el programa comienza a ejecutar una serie de validaciones sobre la información capturada (simulación de constraints, integridad referencial), si toda la información es congruente se almacena el registro, en caso contrario se despliegan mensajes correspondientes. Estos mensajes pueden referirse a cualquiera de las siguientes razones:

- Que no exista el RFC capturado.
- Que no se capturó información en campos que no deben ir vacíos tales como: materia, unidad administrativa, nivel, categoría, partida, sede hospitalaria.
- Que la materia no exista.
- Que la materia no se imparta en la sede hospitalaria capturada o que no pertenezca a la unidad administrativa especificada.

- Que la captura esté bien, sin embargo que no existan horas suficientes en la combinación unidad administrativa, nivel, categoría, partida, periodo especificados para almacenar la solicitud como "En evaluación". En este caso se presenta una ventana en donde se visualiza hasta cuántas horas se tienen disponibles para comprometer.
- Que la combinación anterior no se satisfaga y no se encuentre su respectivo banco de horas.
- Que la persona ya cumplido con más de cuarenta (40) horas en trámite durante el año académico. Estas horas se refieren al total que se encuentran almacenadas más las que se pretenden tramitar.

En caso de que exista algún error de los anteriores, se devuelve el cursor al primer campo de la pantalla de captura para permitir la modificación de la información.
Al terminar el almacenamiento exitoso de un registro, se pregunta si se quiere almacenar un nuevo registro, si la respuesta es afirmativa, se presentará la misma pantalla de captura para dar oportunidad a la nueva captura, en caso contrario, desaparece la pantalla de captura y se regresa el control al menú anterior.

Edición Datos Personales.

Se refiere a la consulta o modificación de datos personales.

Al presionar <Enter> sobre esta opción, se despliega en la pantalla un listado con todos los profesores almacenados en el sistema.

Este listado comprende todos los campos que se capturan en la opción de Alta de Datos Personales. Con las teclas de flechas (con las cuatro direcciones) se podrá desplazar a lo largo y ancho de la ventana que contiene este listado.
Con <Page Up> y <Page Down> se podrán desplazar los registros hacia arriba o abajo de la ventana.

Sobre este listado se podrán hacer búsquedas, éstas son para agilizar la consulta de algún registro en especial. Sobre esta pantalla se podrá presionar <F2>, lo que ocasionará la aparición de una ventana pidiendo que se teclee el RFC de la persona a buscar, al oprimir <Enter>, el programa comienza la búsqueda sobre el listado presentado, si se encuentra, se posiciona en el registro especificado, de lo contrario mandará un mensaje con ese aviso. Según se muestra en la imagen 5.

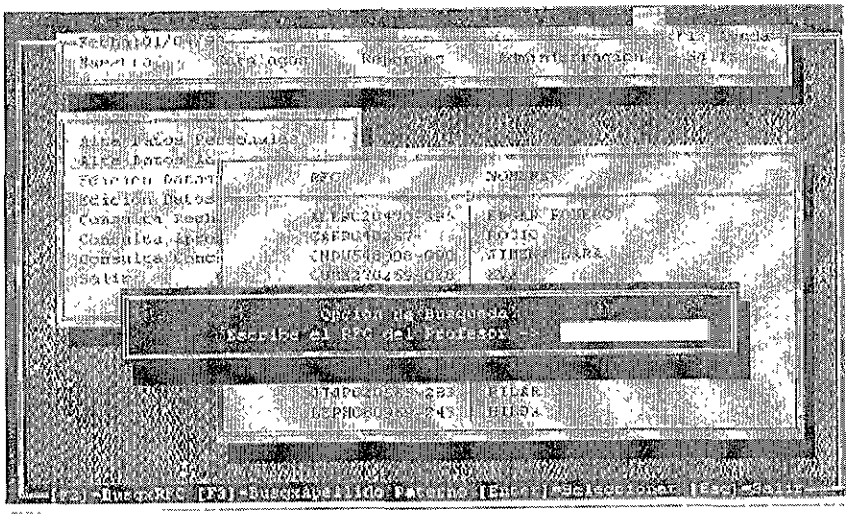

Imagen 5

Asimismo se podrá presionar <F3> para buscar por apellido paterno, y el mecanismo es el mismo. Cabe señalar que si se encuentran dos apellidos iguales, el programa se posicionará en el primero que encuentre, presentando a continuación (debajo de él) el o los siguientes registros con el mismo apellido.

Una vez que se tiene identificado el registro a consultar (la consulta se hace desde que se presenta el listado, ya que desde allí se pueden visualizar todos los campos del registro), se puede presionar <Enter> *si se quiere modificar la* información; de ser así aparecerá una pantalla mostrando todos los campos que pueden ser modificados (los que así no sean, solamente desde el módulo de administración, podrán ser cambiados). Y de igual forma que en las pantallas de captura el mecanismo de las teclas es el mismo para desplazarse en ella. Esto se muestra en la imagen 6.

El cursor iluminará el campo activo, se escribe la nueva información sobre él, y para registrar ese cambio se deberá presionar <Enter> sobre el campo modificado, de otra forma al oprimir <Esc> (la cual es para cerrar esta ventana), se perderán los cambios.
Los campos de fecha se deberán capturar con el formato: día/mes/año (ya que el programa valida esta información).

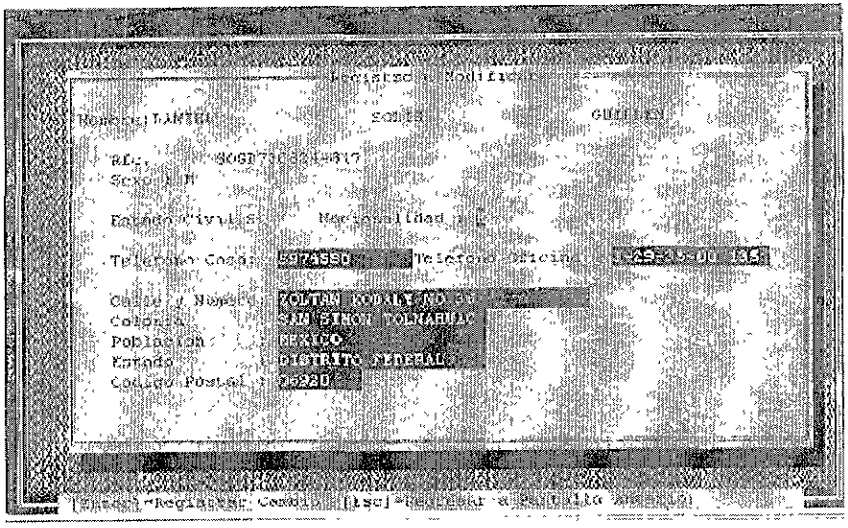

Imagen 6

Edición Datos Académicos.

Funciona de igual manera que el anterior, con algunas variantes.
Al ser seleccionado despliega un listado similar al anterior ordenado por número de expediente y funciona igual, solo que en este caso no se habilita más que una tecla de función para efectuarse una búsqueda, esta tecla busca por RFC y es <F2>. Al encontrar más de un registro con el mismo RFC se posicionará en el primero, presentando los demás a continuación.

Este listado presenta solamente aquellas solicitudes que se encuentren en evaluación ó en trámite, por tal motivo todos podrán ser modificados (regularmente y así deberá ser en fechas para llevar un seguimiento).
De la misma forma que en el submódulo anterior, se presenta una pantalla de modificación al presionar <Enter> sobre el registro deseado, habilitando las mismas teclas de captura y con la misma filosofía de modificación y navegación a través de los campos. En el momento que se detecta que la fecha de rechazo o la fecha de aprobación (minuta 8) han sido ocupadas, se cambia de estatus el registro, dejando de ser con esto una simple solicitud y pasando a ser un movimiento aceptado o rechazado. La pantalla de modificación se ilustra en la imagen 7.

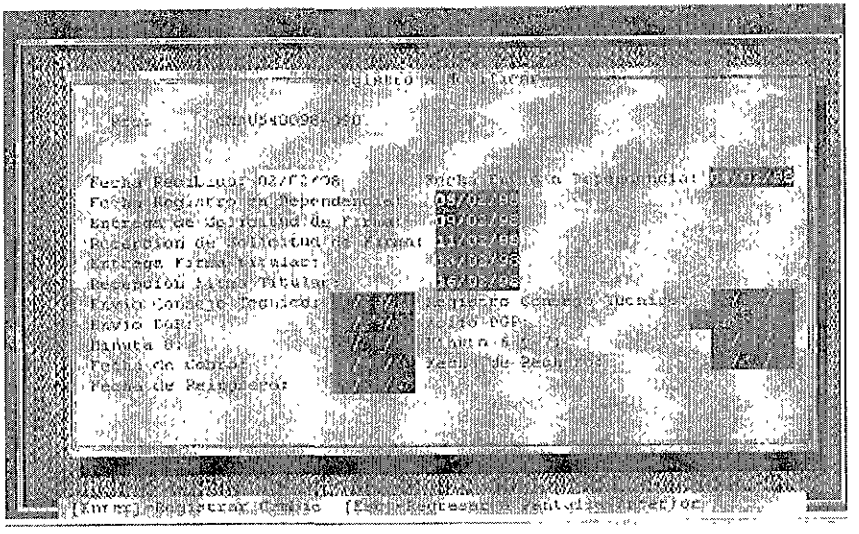

Imagen 7

Submódulos de consulta.

- Consulta de Rechazados.
- Consulta de Aprobados.
- Consulta de Concluidos.

Cada uno de estos presenta una ventana con un listado de registros dependiendo de la característica del trámite. En los tres, se presentan todos los campos capturados en la pantalla correspondiente. No pueden ser modificados debido a que su estatus es final e irreversible (conceptualmente) con excepción de los aprobados, los cuales podrán ser modificados a "concluidos" desde el módulo de administración.

De la misma forma, se puede desplazar el cursor a cualquier dirección con las teclas de movimiento mencionadas anteriormente.
Al igual que en el módulo anterior, se habilita la tecla de función <F2> para buscar por RFC en el listado que se encuentre.
Con la tecla de <Esc> se cierra esta ventana para volver al menú anterior.

Salir.

Al posicionarse sobre la opción "salir" del menú **maestro**, desaparecerá el *submenú, regresando el control al menú principal.*

Módulo Catálogos.

Dentro de este módulo existe información que será necesaria para alimentar los datos requeridos en el llenado de una solicitud y para **consultar** información almacenada que podría verse como propia del sistema. A cada catálogo se la ha asociado una tecla de función *(ver Módulo Catálogos, capítulo 5.2)*, por lo que al ser presionada la tecla respectiva aparecerá en pantalla el catálogo correspondiente.

Esta información no podrá ser modificada por el usuario, solamente por la persona que tenga acceso al módulo de administración. Esta misma persona será responsable de mantener dichos catálogos actualizados y consistentes para el eficiente, claro y buen almacenamiento de la información.

Operación del módulo Catálogos.

Al oprimir <Enter> desde el menú principal aparece un menú con las siguientes líneas:

- Catálogo de Solicitudes.
- Catálogo de Causas.
- Catálogo de Unidades Administrativas.
- Catálogo de Subunidades Administrativas.
- Catálogo de Materias
- Catálogo de Categorías.
- Catálogo de Sedes Hospitalarias.
- Catálogo de Niveles.
- Catálogo de Funciones.
- Salir.

Cada línea mencionada, tiene su propia tecla rápida asociada, para que en cualquier momento donde se presione esta tecla, se presente el catálogo correspondiente. Estas teclas se asignan desde <F4> hasta <F12> respectivamente a la lista anterior y la línea de Salir que es <Esc>.

Por lo tanto, para desplegar un catálogo en pantalla se puede hacer de dos formas:

- Presionando la tecla rápida asociada al catálogo correspondiente en cualquier momento.
- Seleccionando desde el menú principal la opción "Catalogos" y presionando <Enter> sobre la línea que contiene el nombre del catálogo deseado.

Podría ser que para el usuario fuese más fácil memorizar el significado de cada tecla de función para que en el momento que se requiera saber la descripción de una clave, se pueda presionar la tecla correspondiente y buscar la clave en el catálogo presentado.

Para desplazarse a través de la ventana de cualquier catálogo se podrá hacer con las teclas de "flechas", <Page Up> y <Page Down>.

Cuando se está capturando algún trámite administrativo, se puede presentar en pantalla cualquier catálogo por medio de su tecla rápida, en tal caso, al presionar <Enter> sobre un registro del catálogo presentado, se asignará el valor de la clave del registro en el que se oprimió <Enter> al campo correspondiente. Seguramente no se visualizará el valor asignado si el cursor no está posicionado en el campo apropiado, en ese caso bastará posicionar el cursor sobre tal campo para refrescar el dato en pantalla y poder visualizarlo.

En la imagen 8 se ilustra un ejemplo de este módulo con un catálogo abierto (de forma similar se visualizan todos los catálogos).

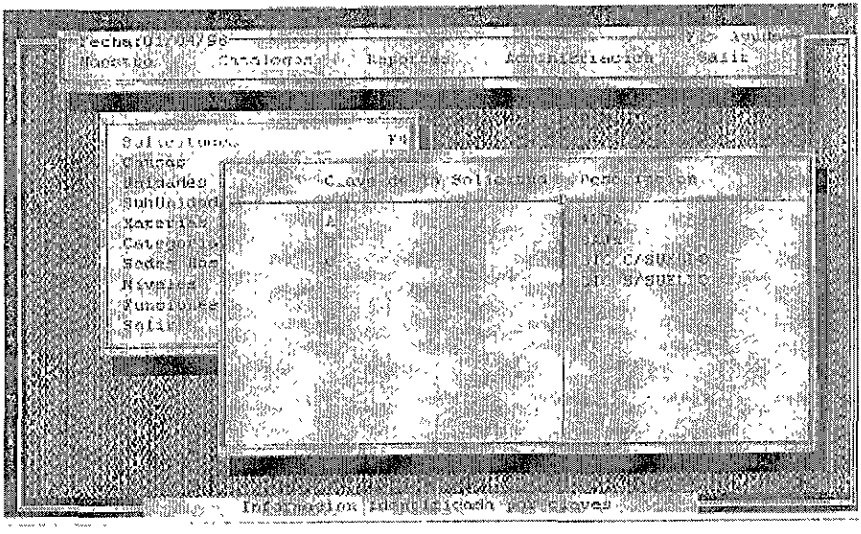

Imagen 8

Si se desea visualizar varios catálogos en pantalla se recomienda invocar a uno, cerrarlo y después invocar a otro. No hay necesidad de tener abiertos más de uno ya que están programados para desplegarse en las mismas coordenadas de la pantalla, por lo que de cualquier forma se sobrepondrían.

Precaución: *En caso de que se presionen consecutivamente las teclas de función que invocan a los catálogos sin cerrarse previamente cada uno de ellos, el sistema producirá un error y ocasionará el fin de la ejecución del mismo, devolviendo el control al sistema operativo. Si se estaba capturando algo para*

almacenar en el sistema y se hace esta serie de movimientos innecesarios, se perderá la información en pantalla, siendo irrecuperable, por favor no lo haga.

Salir.

Al posicionarse sobre la opción "salir" del menú **catalogos**, desaparecerá el submenú, regresando el control al menú principal.

Módulo Reportes.

Es toda aquella información desplegada en forma tabular por impresora que relaciona información almacenada previamente de una manera congruente y que por tanto, expresa la situación de la institución en alguna determinada particularidad.

Operación del módulo Reportes.

Posicionando el cursor desde el menú principal sobre esta opción y luego presionando <Enter>, se visualiza en pantalla un submenú con cuatro opciones, en donde cualquiera de éstas se podrá ejecutar presionando <Enter> sobre la línea deseada. En la imagen 9 se ilustra la pantalla a visualizar en el sistema.

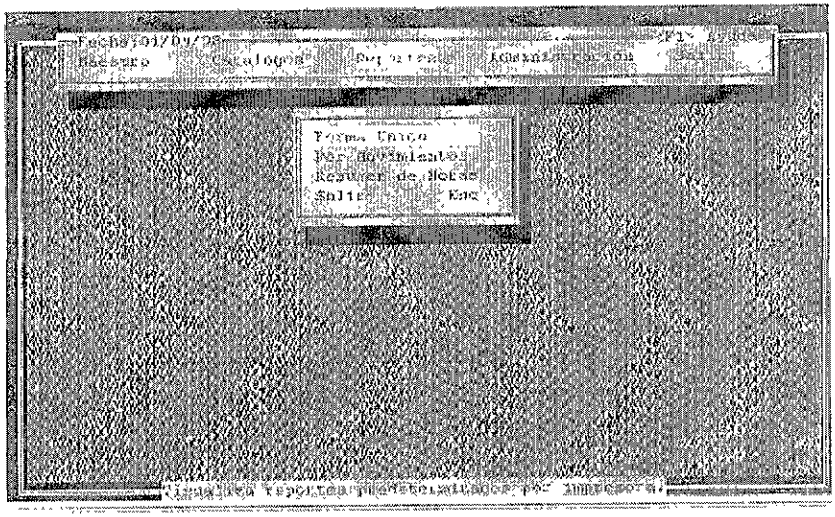

Imagen 9

Se consideran tres reportes:

1. Forma Única: Esta línea al ser seleccionada con <Enter> ocasiona que aparezca en pantalla una ventana que contiene el listado de todos aquellos trámites que han sido aprobados, presentando solamente el RFC y el número de expediente por registro. De esta forma el usuario podrá seleccionar el expediente del cual requiera generar su forma única simplemente con posicionar el cursor en el registro deseado y oprimir <Enter>; en ese momento el programa manda la información a impresora con medidas y distribución de los campos predeterminados.
La forma única es el documento que respalda al profesor para estar seguro de que le fue concedido lo que solicitó.
Al terminar la impresión se regresa el control al menú anterior.

2. Por movimientos: Se genera un listado en impresora de todos los trámites que han sido aprobados es decir válidos durante el año en curso (no académico), por lo que se listarán aquellos tramites con Año/XXXXX (Año se refiere a los dos últimos dígitos del año en curso). Basta con seleccionar esta línea con <Enter> para generar este reporte ya que no requiere de alguna intervención más del usuario.
Se recomienda que para este reporte se configure la impresora como desplegado horizontal y con la letra más compacta, ya que por el número y tamaño de campos desplegados, ocupa mucho espacio cada registro.

3. Resumen de Horas: Se resume de manera general el total de horas que contiene cada unidad administrativa (ya se incluyen las subunidades) por categoría, por nivel y partida.

Este reporte se lleva a cabo quincenalmente para que de esta forma se visualice el avance o contenido total de horas asignadas de cada detalle de manera más controlada.
El día en que se genere (según la fecha de la computadora), deberá pertenecer al periodo académico del que se trate, ya que el programa utiliza esta fecha para consultar el total de horas de los registros que tengan el periodo al cual pertenezca el día en que se genera el reporte. Se considera que un periodo académico es anual y va desde agosto hasta julio del año próximo.

Para visualizar este reporte se recomienda que se configure la impresora con tamaño carta y orientación vertical con tamaño de letra normal, sin embargo se pueden hacer pruebas y generarlo según se requiera.

Salir.

Al presionar <Enter> sobre "salir", desaparecerá el submenú del módulo de reportes, regresando al menú principal.

Módulo Administración.

De manera general se distinguen tres acciones a realizar dentro de este módulo:

1. Actualizar información de forma arbitraria en cualquier tabla.
2. Mantener al día el estatus de cada movimiento aprobado revisando la terminación de su vigencia.
3. Creación de respaldo de toda la información almacenada.

Es importante resaltar que el administrador debe tener todo el conocimiento acerca de cómo se relaciona la información, seguramente no será tan trivial el almacenar un trámite, como lo es un profesor, ya que en el primer tipo de almacenamiento se ejecutan mayores validaciones no sólo en cuanto a captura sino además en cuanto a relaciones.

Debido a que es importante la consideración de las relaciones en la base de datos para el adecuado almacenamiento de la información, a continuación se establece que por cada trámite se debe cumplir lo siguiente:

1. El nivel pertenece a una categoría, esta categoría pertenece a una partida y esta partida debe corresponder a una unidad administrativa (esto es para cumplir con el banco de horas). El sueldo del nivel varía de acuerdo a la categoría y partida a las cuales pertenece.
2. La materia debe pertenecer a una subunidad (si es el caso) ó en su defecto directamente a una unidad administrativa. Esta unidad administrativa es la misma de la que se habla en el punto (1).
3. Se deberá seleccionar la sede hospitalaria en la cual se impartirá la materia. Esta materia es la misma de la que se habló en el punto (2).

Conservando la adecuada relación de los tres puntos anteriormente mencionados, se podrá almacenar exitosamente un trámite, desde luego si hay suficiencia de horas, si el solicitante no tiene más de cuarenta horas (incluyendo las solicitadas) asignadas, y otras validaciones, así como de captura.

Operación del Módulo Administración.

Al seleccionar este módulo desde el menú principal con <Enter>, se presenta una ventana requiriendo una clave de acceso a este módulo para entrar. El tiempo de captura entre cada caracter (visualizados en pantalla como asteriscos) será de 10

segundos, si se excede, el sistema escribe cada 10 segundos un asterisco y será inútil intentar capturar nuevamente, sólo hasta volver a ejecutar la misma opción.
Si la clave es válida, se presenta un menú con todas las tablas que pueden ser alteradas, una opción de respaldo y otra para modificar el estatus de los trámites aprobados y que en ese momento ya no tienen vigencia. Este menú se ilustra en la imagen 10.

Alteración arbitraria.

Seleccionando cualquier opción con <Enter> que se refiera a una tabla del sistema, se presenta otra ventana con todos los registros y campos en forma tabular.

Si se desea modificar información de un campo, se presiona <Enter> sobre él y se teclea lo nuevo, para registrar el cambio se presiona <Enter> y automáticamente se posiciona el cursor en el siguiente campo.
Si se requiere borrar un registro, se posiciona el cursor sobre cualquier campo del registro y se presiona <Del>, inmediatamente aparecerá en la esquina superior derecha la palabra "Deleted", al cerrar esta tabla y abrirla nuevamente, el registro marcado ya no aparecerá, se habrá borrado físicamente.

Para "desmarcar" un registro como borrado se vuelve a presionar la tecla "Del" o "Supr" (según el teclado) antes de cerrar la ventana.

Para insertar un nuevo registro, sobre el último registro se presiona la tecla de "Flecha Abajo" y se comienza a capturar la información del campo respectivo.

Una vez que se ha presionado <Esc> sobre cualquier ventana, se cerrará la misma y cualquier cambio a la tabla quedará confirmado.

Es responsabilidad de quien tenga acceso a este módulo el mantener esta información coherente, íntegra y actualizada para el buen funcionamiento del sistema.

Dentro de este módulo es muy fácil alterar **toda** la información y por lo tanto peligroso en cuanto a la seguridad de la información. *(Ver fragilidad, capítulo 4).*

Actualización Finalizados.

Se refiere a la actualización del estatus de solicitudes que actualmente se encuentran aprobadas y su vigencia ha caducado.
Al seleccionarse, se presenta un listado de todas las solicitudes que hasta la fecha están aprobadas, mostrando el RFC, número de expediente, fecha inicio y fecha fin dentro de las cuales el trámite será válido. Una vez hecho lo anterior, se puede verificar la acción, consultando el módulo de "Consulta de Concluidos".

De igual forma será responsabilidad del encargado de este módulo el liberar horas (si así fuera el caso de alguna solicitud aprobada) y desde luego "mover" el estatus de "Aprobado" a "Concluido" del trámite. Por lo que el acceso a este módulo deberá ser diario (si es que diariamente sea posible que finalicen trámites válidos).

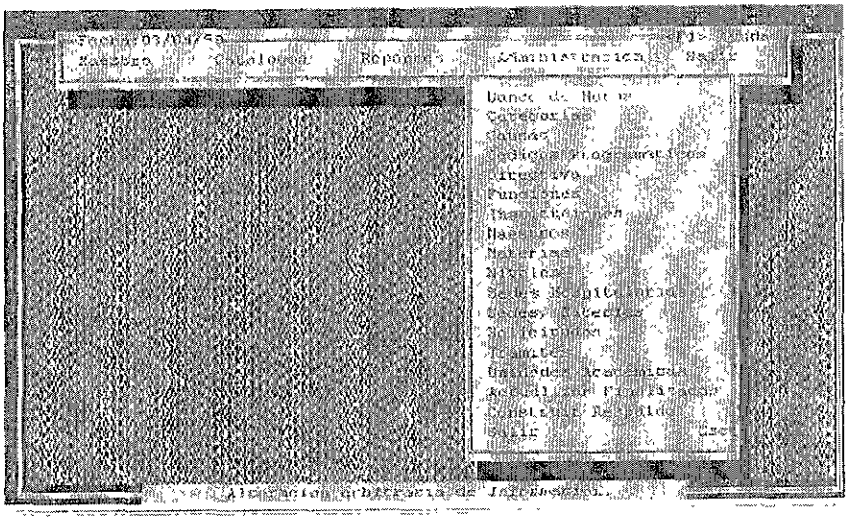

Imagen 10

Construir Respaldo.

Se refiere a la generación de copias de toda la información que maneja el sistema (es decir todas las tablas).

Presionando <Enter> sobre esta opción, ocasionará la aparición de una ventana que proporciona la cantidad de bytes que ocupa en ese momento toda la información almacenada. Para cerrar esta ventana sólo es necesario oprimir cualquier tecla, inmediatamente después, aparecerá un cuadro, dentro del cual se podrá capturar toda la ruta en donde finalmente se desea copiar toda la información.

Si el usuario está consciente que no cuenta con el espacio suficiente para copiar la información (según la ventana anteriormente mostrada en pantalla), en ese momento podrá cancelar la operación de respaldo a través de oprimir la tecla <Esc>, en caso contrario, oprimiendo <Enter> el sistema entenderá que se puede proseguir con este proceso.

Se llevan a cabo una serie de validaciones sobre lo que capturó el usuario, se verifica si efectivamente existe el espacio necesario para hacer la copia en el destino señalado, y si existe la ruta especificada; si no existe la ruta, se crean automáticamente los subdirectorios (el sistema sólo puede generar hasta quince niveles de subdirectorios) y posteriormente se efectúa la copia.

Si cualquiera de las validaciones no es satisfactoria (espacio insuficiente, unidad inválida, drive no preparado, etc.), se despliega un mensaje expresando la causa y se aborta la operación de respaldo, regresando, con esto, el control al menú de Administración.

Nota: *Se debe tomar en cuenta que los nombres de los subdirectorios no deben exceder los ocho caracteres (letras o dígitos), de lo contrario, se truncaran exactamente los nombres en el octavo caracter y con el nombre resultante se creará el subdirectorio.*

Salir.

Al presionar la tecla <Enter> sobre esta opción del menú presentado en Administración, el sistema desaparecerá este menú, regresando el control al sistema operativo; si se desea volver a entrar al menú de Administración, será necesario capturar nuevamente la clave de acceso.

Módulo Salir.

Al seleccionar esta opción del menú principal, el sistema terminará su ejecución y regresará el control al sistema operativo.
De la misma forma, posicionándose en cualquier parte del menú principal y presionando la tecla <Esc>, se transfiere el control al sistema operativo, terminando así el programa.

Cabe señalar que al presionar <Ctrl – Break> (Control – Pausa ó Control – Break, según el teclado) en cualquier punto, se terminará la ejecución del sistema. Esto es de gran ayuda para el caso cuando se requiera de una salida emergente.

CONCLUSIONES

El problema planteado ha sido solucionado por medio de una teoría computacional, que al aplicarse en forma conjunta con un software en particular, proporciona una solución que optimiza procesos administrativos con un considerable grado de confiabilidad.

En la medida de la magnitud o requerimientos que presente cada problema, se seleccionarán las herramientas adecuadas para resolverlo.

Con el software que se seleccionó se logró una comunicación entre el usuario y el sistema en sí, ya que ofrece una ayuda interactiva en cualquier momento, según la acción a realizar dentro del sistema. Con las facilidades otorgadas por el software, fue sencillo establecer este "canal" de comunicación (hombre - máquina).

Es muy común que el usuario cometa errores de captura, los cuales ocasionan un almacenamiento incierto, inseguro y a veces inconsistente, es por eso que por medio de diversas validaciones, se conduce al usuario a capturar datos que son de vital importancia para la identificación de la información.

De la misma forma se logra con este proyecto una distinción de usuarios, esto es, aquellos que tienen sólo la función de capturar y aquellos que son "dueños" de la información los cuales, son directamente responsables de la consistencia de la misma. Por medio de claves de acceso es como se restringen los permisos de alteración de información.

Por otra parte, se considera también la creación de respaldos de la información en cualquier momento, esto, con el fin de evitar extravío de la misma, conservación histórica y además de que es seguro, es esencial para la institución.

El sistema fue diseñado y planeado de manera procedural, para poder mantener un mejor control de la ejecución de cada función, optimización de código y facilitar su mantenimiento.

Se considera dentro de lo que se conoce como "mantenimiento" cualquier implementación o modificación sobre lo que existe actualmente desarrollado, nuevas funciones requeridas por el usuario, ejecución del sistema en un ambiente multiusuario (el software tiene funciones que permiten la ejecución del sistema para el usuario de manera transparente y que facilitan la programación), consideración del sistema a nivel facultad, nuevos reportes, etc.

Por lo tanto, con la solución y desarrollo implementados en este proyecto, se cumplieron todos los objetivos fijados, tales como:

- Aplicación práctica de una teoría computacional en un caso práctico y real que mantiene integridad en la información.
- Un sistema lo suficientemente *amigable para interactuar* con el usuario en tiempo de ejecución con mensajes o pantallas de ayuda según se requiera.
- Validación de la información y detección de errores en la captura.
- Seguimiento administrativo de la documentación.
- Disponibilidad de la información a través de distintos criterios de búsqueda.
- Seguridad en la información por medio de claves de acceso.
- Generación de consultas y/o reportes predefinidos (en este caso los más requeridos por los usuarios).
- Generación de copias de seguridad de tal forma que respalden la información actual, ó mantengan un control de versiones de la misma.
- Desarrollo procedural el cual facilitará en un futuro posibles *modificaciones, no* solo a nivel función o procedimiento, sino además a la intercomunicación entre los mismos y al sistema en sí.

No fue necesario hacer uso de herramientas que evidentemente ofrecen más facilidades tanto al usuario como al desarrollador, ya que con los recursos disponibles y los conceptos teóricos aplicados, se logró el control automatizado de una serie de procesos administrativos que mantiene integridad (sobre todo con el usuario final) y seguridad en la información interactuando con el usuario de una forma amigable con procesos de programación eficientes, confiables e intuitivos de tal forma que cualquier usuario podría hacer uso del mismo.

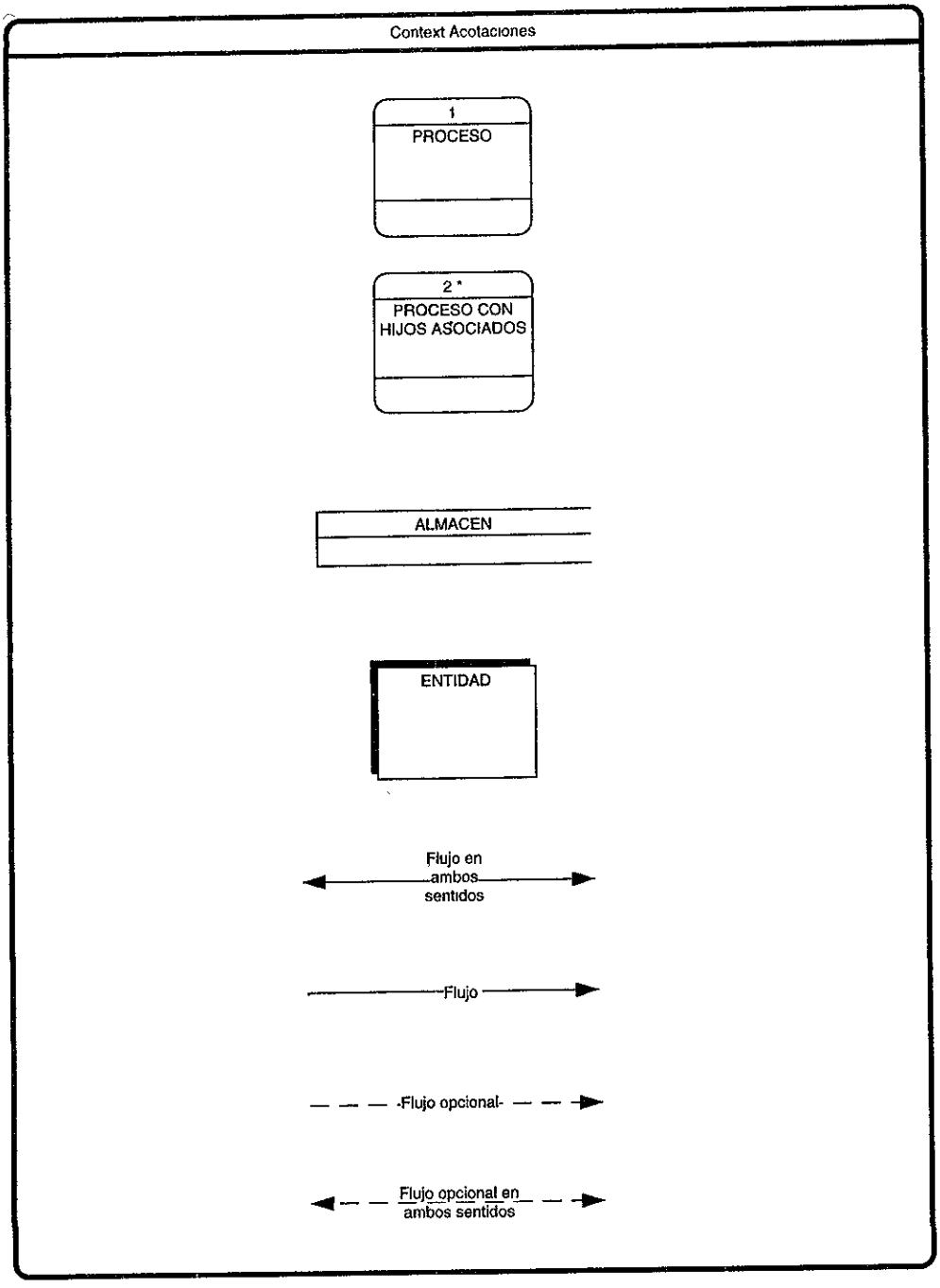
Context Acotaciones
1
PROCESO
2 *
PROCESO CON HIJOS ASOCIADOS
ALMACEN
ENTIDAD
Flujo en ambos sentidos
Flujo
Flujo opcional
Flujo opcional en ambos sentidos

Figura 1

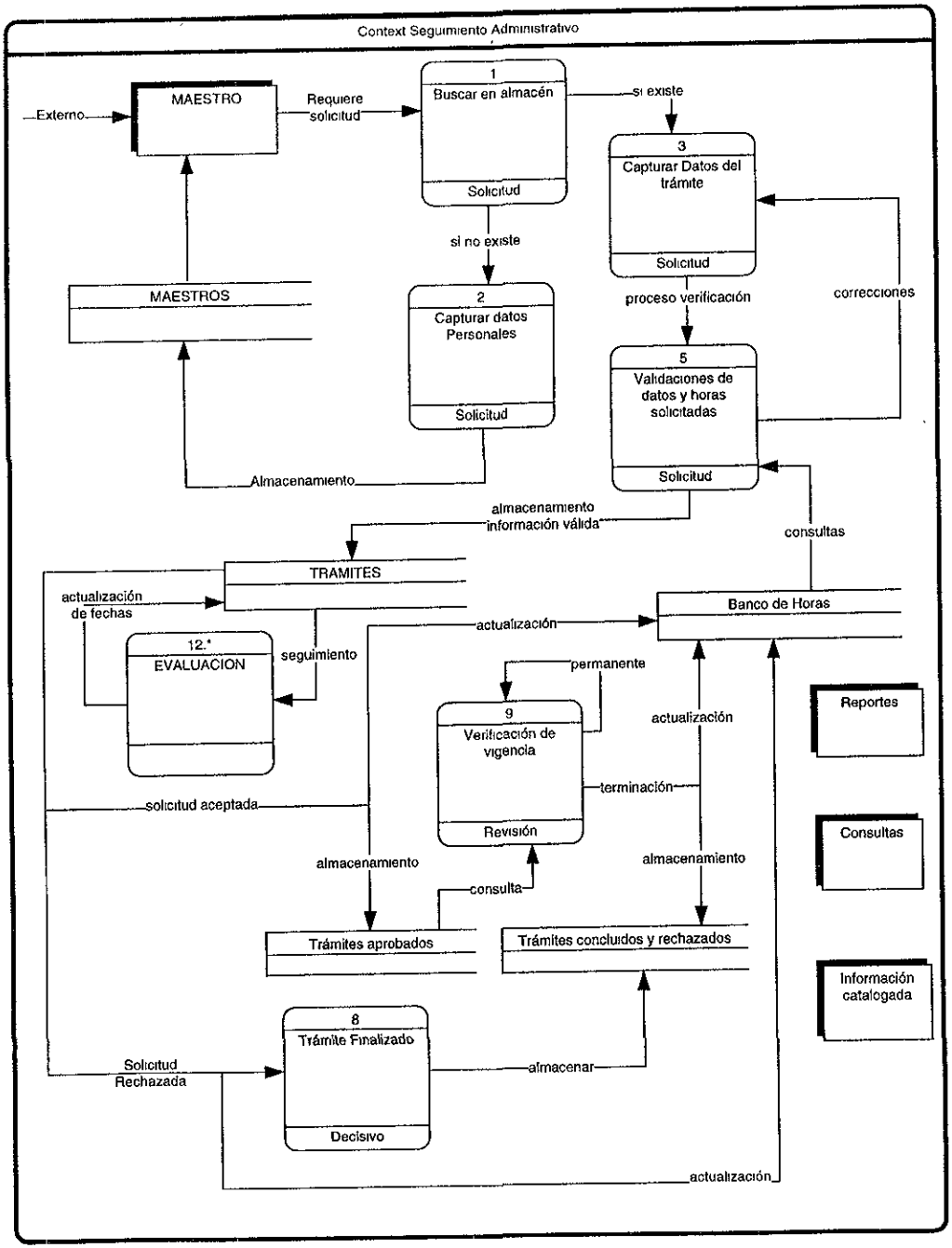
Context Seguimiento Administrativo
Externo
MAESTRO
Requiere solicitud
1
Buscar en almacén
Solicitud
si existe
si no existe
3
Capturar Datos del trámite
Solicitud
MAESTROS
2
Capturar datos Personales
Solicitud
proceso verificación
correcciones
5
Validaciones de datos y horas solicitadas
Solicitud
Almacenamiento
almacenamiento información válida
consultas
TRAMITES
actualización de fechas
Banco de Horas
actualización
12.*
EVALUACION
seguimiento
permanente
9
Verificación de vigencia
Revisión
actualización
terminación
Reportes
solicitud aceptada
Consultas
almacenamiento
consulta
almacenamiento
Trámites aprobados
Trámites concluidos y rechazados
Información catalogada
8
Trámite Finalizado
Decisivo
Solicitud Rechazada
almacenar
actualización

Figura 2

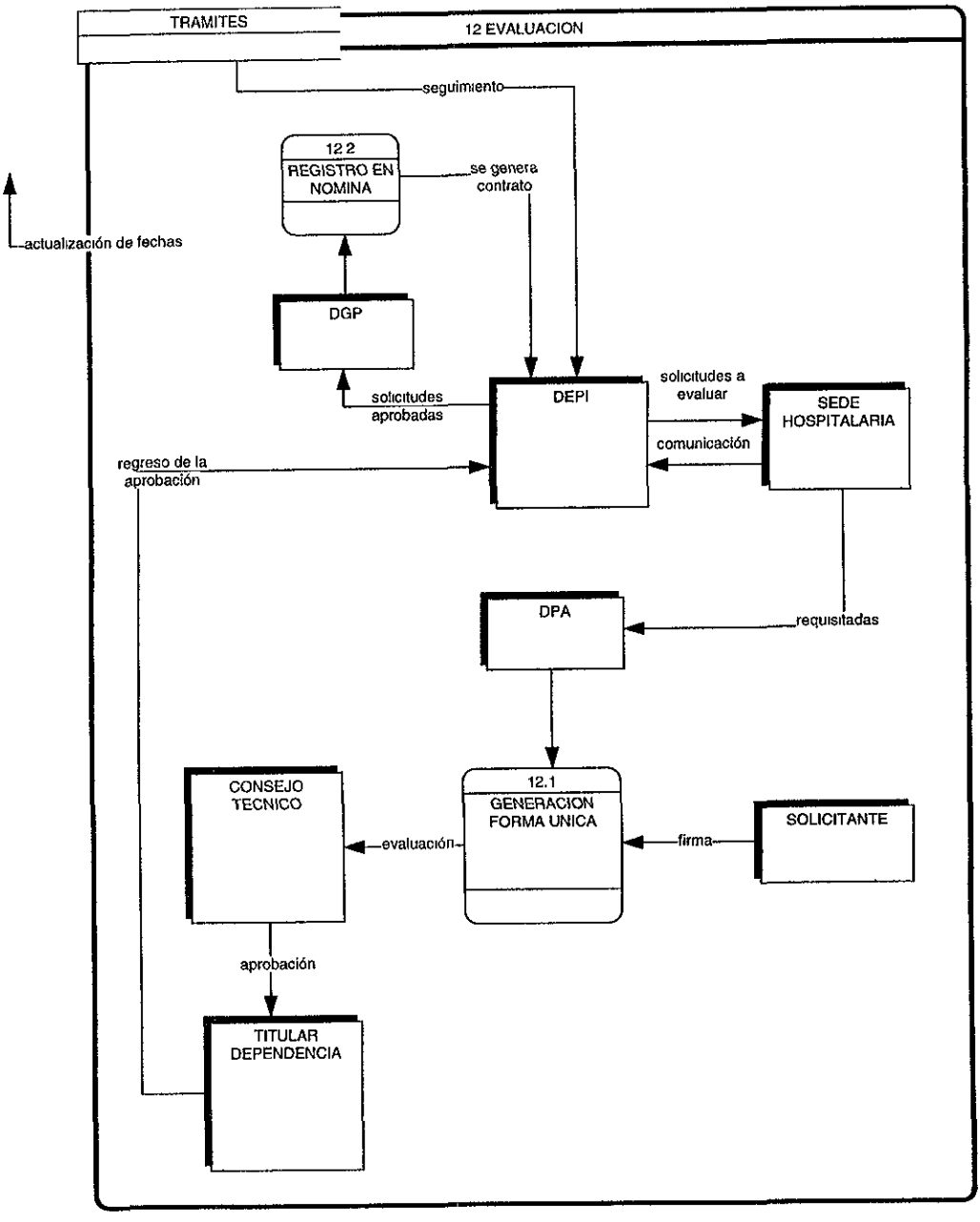
TRAMITES
12 EVALUACION
seguimiento
12 2
REGISTRO EN NOMINA
se genera contrato
actualización de fechas
DGP
solicitudes aprobadas
DEPI
solicitudes a evaluar
comunicación
SEDE HOSPITALARIA
regreso de la aprobación
DPA
requisitadas
CONSEJO TECNICO
12.1
GENERACION FORMA UNICA
evaluación
firma
SOLICITANTE
aprobación
TITULAR DEPENDENCIA

Figura 3

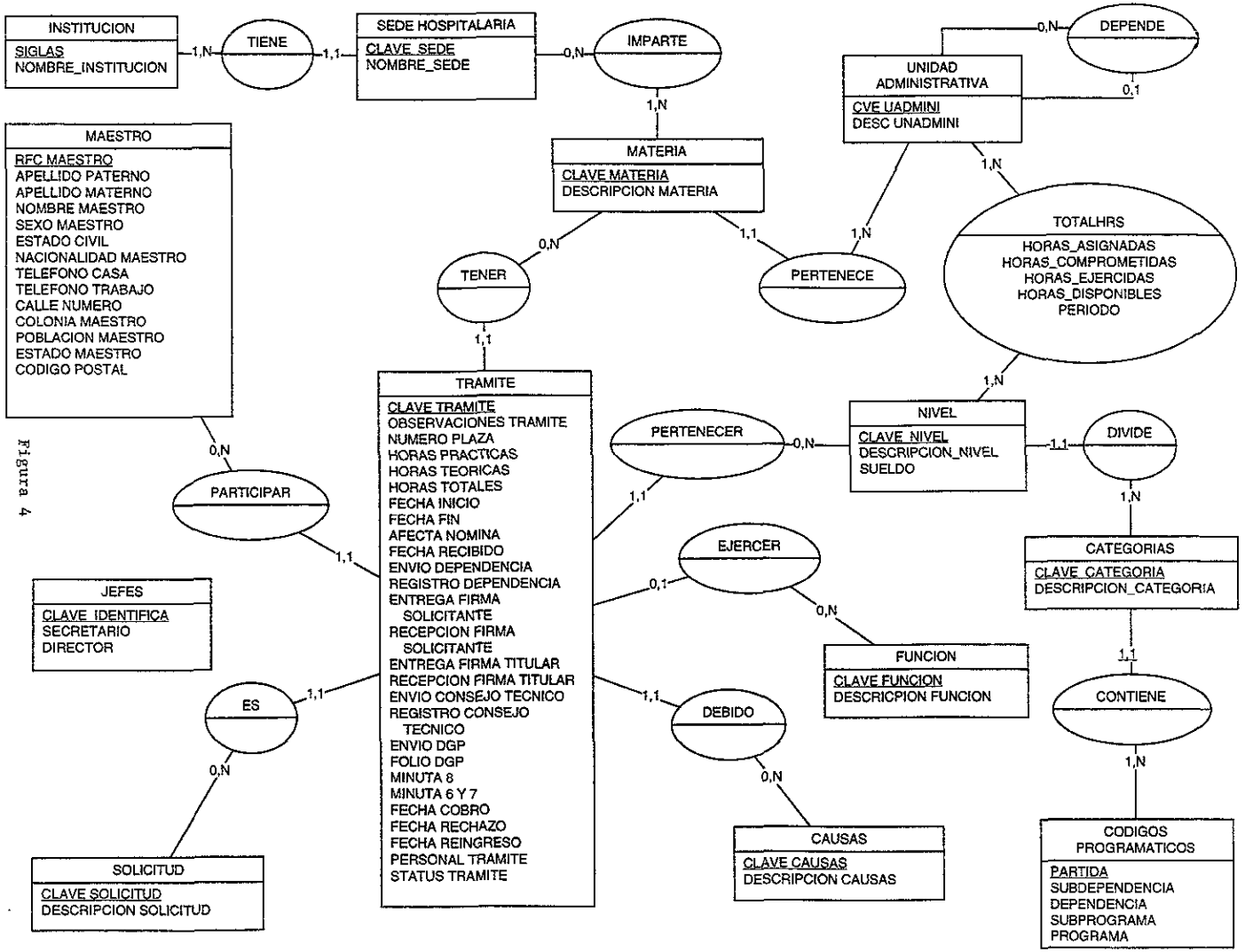
INSTITUCION
SIGLAS
NOMBRE_INSTITUCION
1,N
TIENE
1,1
SEDE HOSPITALARIA
CLAVE_SEDE
NOMBRE_SEDE
0,N
IMPARTE
1,N
MATERIA
CLAVE MATERIA
DESCRIPCION MATERIA
0,N
DEPENDE
0,1
UNIDAD ADMINISTRATIVA
CVE UADMINI
DESC UNADMINI
1,N
MAESTRO
RFC MAESTRO
APELLIDO PATERNO
APELLIDO MATERNO
NOMBRE MAESTRO
SEXO MAESTRO
ESTADO CIVIL
NACIONALIDAD MAESTRO
TELEFONO CASA
TELEFONO TRABAJO
CALLE NUMERO
COLONIA MAESTRO
POBLACION MAESTRO
ESTADO MAESTRO
CODIGO POSTAL
TENER
1,1
PERTENECE
1,N
TOTALHRS
HORAS_ASIGNADAS
HORAS_COMPROMETIDAS
HORAS_EJERCIDAS
HORAS_DISPONIBLES
PERIODO
TRAMITE
CLAVE TRAMITE
OBSERVACIONES TRAMITE
NUMERO PLAZA
HORAS PRACTICAS
HORAS TEORICAS
HORAS TOTALES
FECHA INICIO
FECHA FIN
AFECTA NOMINA
FECHA RECIBIDO
ENVIO DEPENDENCIA
REGISTRO DEPENDENCIA
ENTREGA FIRMA SOLICITANTE
RECEPCION FIRMA SOLICITANTE
ENTREGA FIRMA TITULAR
RECEPCION FIRMA TITULAR
ENVIO CONSEJO TECNICO
REGISTRO CONSEJO TECNICO
ENVIO DGP
FOLIO DGP
MINUTA 8
MINUTA 6 Y 7
FECHA COBRO
FECHA RECHAZO
FECHA REINGRESO
PERSONAL TRAMITE
STATUS TRAMITE
PERTENECER
0,N
1,1
NIVEL
CLAVE_NIVEL
DESCRIPCION_NIVEL
SUELDO
1,1
DIVIDE
1,N
CATEGORIAS
CLAVE_CATEGORIA
DESCRIPCION_CATEGORIA
1,1
CONTIENE
1,N
CODIGOS PROGRAMATICOS
PARTIDA
SUBDEPENDENCIA
DEPENDENCIA
SUBPROGRAMA
PROGRAMA
0,N
PARTICIPAR
1,1
EJERCER
0,1
0,N
FUNCION
CLAVE FUNCION
DESCRICPION FUNCION
JEFES
CLAVE_IDENTIFICA
SECRETARIO
DIRECTOR
ES
1,1
0,N
SOLICITUD
CLAVE SOLICITUD
DESCRIPCION SOLICITUD
DEBIDO
1,1
0,N
CAUSAS
CLAVE CAUSAS
DESCRIPCION CAUSAS

Figura 4

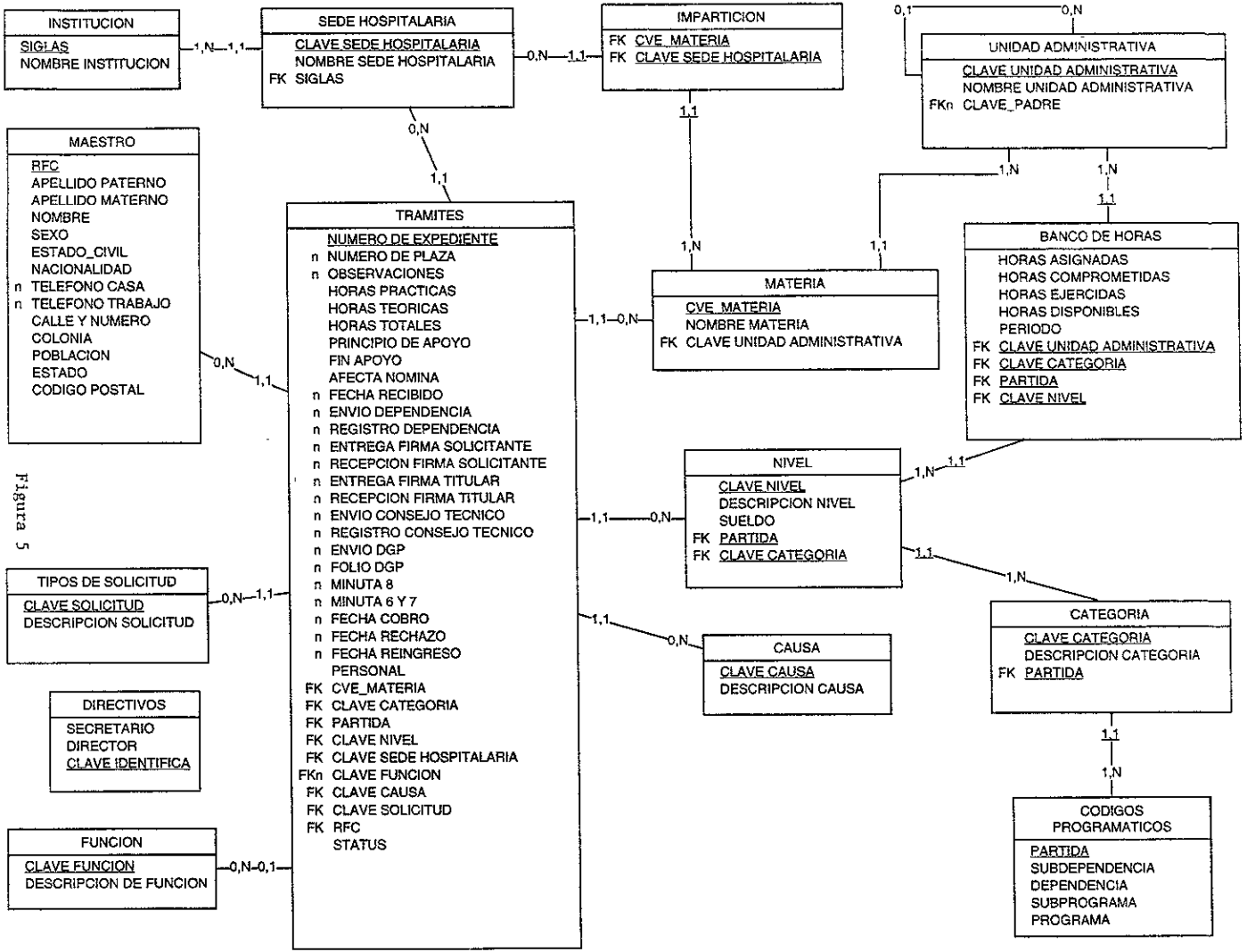
INSTITUCION
SIGLAS
NOMBRE INSTITUCION
1,N
1,1
SEDE HOSPITALARIA
CLAVE SEDE HOSPITALARIA
NOMBRE SEDE HOSPITALARIA
FK SIGLAS
0,N
1,1
IMPARTICION
FK CVE_MATERIA
FK CLAVE SEDE HOSPITALARIA
0,1
0,N
UNIDAD ADMINISTRATIVA
CLAVE UNIDAD ADMINISTRATIVA
NOMBRE UNIDAD ADMINISTRATIVA
FKn CLAVE_PADRE
MAESTRO
RFC
APELLIDO PATERNO
APELLIDO MATERNO
NOMBRE
SEXO
ESTADO_CIVIL
NACIONALIDAD
n TELEFONO CASA
n TELEFONO TRABAJO
CALLE Y NUMERO
COLONIA
POBLACION
ESTADO
CODIGO POSTAL
0,N
1,1
TRAMITES
NUMERO DE EXPEDIENTE
n NUMERO DE PLAZA
n OBSERVACIONES
HORAS PRACTICAS
HORAS TEORICAS
HORAS TOTALES
PRINCIPIO DE APOYO
FIN APOYO
AFECTA NOMINA
n FECHA RECIBIDO
n ENVIO DEPENDENCIA
n REGISTRO DEPENDENCIA
n ENTREGA FIRMA SOLICITANTE
n RECEPCION FIRMA SOLICITANTE
n ENTREGA FIRMA TITULAR
n RECEPCION FIRMA TITULAR
n ENVIO CONSEJO TECNICO
n REGISTRO CONSEJO TECNICO
n ENVIO DGP
n FOLIO DGP
n MINUTA 8
n MINUTA 6 Y 7
n FECHA COBRO
n FECHA RECHAZO
n FECHA REINGRESO
PERSONAL
FK CVE_MATERIA
FK CLAVE CATEGORIA
FK PARTIDA
FK CLAVE NIVEL
FK CLAVE SEDE HOSPITALARIA
FKn CLAVE FUNCION
FK CLAVE CAUSA
FK CLAVE SOLICITUD
FK RFC
STATUS
1,1
0,N
MATERIA
CVE_MATERIA
NOMBRE MATERIA
FK CLAVE UNIDAD ADMINISTRATIVA
1,N
1,1
1,N
1,1
1,N
1,1
BANCO DE HORAS
HORAS ASIGNADAS
HORAS COMPROMETIDAS
HORAS EJERCIDAS
HORAS DISPONIBLES
PERIODO
FK CLAVE UNIDAD ADMINISTRATIVA
FK CLAVE CATEGORIA
FK PARTIDA
FK CLAVE NIVEL
1,1
0,N
NIVEL
CLAVE NIVEL
DESCRIPCION NIVEL
SUELDO
FK PARTIDA
FK CLAVE CATEGORIA
1,N
1,1
1,1
1,N
CATEGORIA
CLAVE CATEGORIA
DESCRIPCION CATEGORIA
FK PARTIDA
1,1
0,N
CAUSA
CLAVE CAUSA
DESCRIPCION CAUSA
TIPOS DE SOLICITUD
CLAVE SOLICITUD
DESCRIPCION SOLICITUD
0,N
1,1
DIRECTIVOS
SECRETARIO
DIRECTOR
CLAVE IDENTIFICA
FUNCION
CLAVE FUNCION
DESCRIPCION DE FUNCION
0,N
0,1
1,1
1,N
CODIGOS PROGRAMATICOS
PARTIDA
SUBDEPENDENCIA
DEPENDENCIA
SUBPROGRAMA
PROGRAMA

Figura 5

Table

Table

Name : BANCO DE HORAS
Coded name : TOTALHRS
Row length : 51

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base .	HORAS ASIGNADAS	H_ASIG	<Logical>, 1 0, NUMBER	8	2
Base .	HORAS COMPROMETIDAS	H_COMPROME	<Logical>, 1.0, NUMBER	8	2
Base ...	HORAS EJERCIDAS	H_EJERCIDA	<Logical>, 1.0, NUMBER	8	2
Base ..	HORAS DISPONIBLES	H_DISPONI	<Logical>, 1.0, NUMBER	8	2
Base ...	PERIODO	PERIODO	<Logical>, 1.0, CHARACTER	2	
Foreig..	CLAVE UNIDAD ADMINIS. .	CLV_UADMI	<Logical>, 1.0, CHARACTER	4	
Foreig .	CLAVE CATEGORIA	CLV_CATEGO	<Logical>, 1.0, CHARACTER	5	
Foreig .	PARTIDA	PAR	<Logical>, 1.0, CHARACTER	3	
Foreig..	CLAVE NIVEL	CLV_NIVEL	<Logical>, 1.0, CHARACTER	5	

Table

Name : CATEGORIA
Coded name : CATEGORI
Row length : 63

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base ...	CLAVE CATEGORIA	CLV_CATEGO	<Logical>, 1 0, CHARACTER	5	
Base ..	DESCRIPCION CATEGORIA	NOM_CATEGO	<Logical>, 1.0, CHARACTER	55	
Foreig..	PARTIDA	PAR	<Logical>, 1.0, CHARACTER	3	

Table

Name : CAUSA
Coded name : CAUSA
Row length : 27

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base ..	CLAVE CAUSA	CLV_CAUSA	<Logical>, 1.0, CHARACTER	2	
Base ..	DESCRIPCION CAUSA	NOM_CAUSA	<Logical>, 1.0, CHARACTER	25	

Table

Name : CODIGOS PROGRAMATICOS
Coded name : CO_PRO_P
Row length : 12

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base . .	PARTIDA	PAR	<Logical>, 1.0, CHARACTER	3	
Base ...	SUBDEPENDENCIA	SD	<Logical>, 1.0, CHARACTER	2	
Base ...	DEPENDENCIA	DEP	<Logical>, 1 0, CHARACTER	3	
Base ...	SUBPROGRAMA	SP	<Logical>, 1.0, CHARACTER	2	
Base ...	PROGRAMA	PR	<Logical>, 1.0, CHARACTER	2	

Table

Name : DIRECTIVOS

Figura 6

Table

Coded name : JEFES
Row length : 122

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base .	SECRETARIO	NOM_SECRET	<Logical>, 1.0, CHARACTER	60	
Base ..	DIRECTOR	NOM_TITULA	<Logical>, 1 0, CHARACTER	60	
Base	CLAVE IDENTIFICA	CVE	<Logical>, 1 0, CHARACTER	2	

Table

Name : FUNCION
Coded name : FUNCION
Row length : 28

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base ..	CLAVE FUNCION	CLV_FUNC	<Logical>, 1.0, CHARACTER	3	
Base .	DESCRIPCION DE FUNCI. .	NOM_FUNC	<Logical>, 1.0, CHARACTER	25	

Table

Name : IMPARTICION
Coded name : SH_MATER
Row length : 14

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Foreig.	CVE_MATERIA	CLV_MATERI	<Logical>, 1.0, CHARACTER	8	
Foreig	CLAVE SEDE HOSPITALA .	CLV_SH	<Logical>, 1 0, CHARACTER	6	

Table

Name : INSTITUCION
Coded name : INSTITU
Row length : 48

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base ...	SIGLAS	SIGLAS	<Logical>, 1 0, CHARACTER	8	
Base .	NOMBRE INSTITUCION	NOM_INST	<Logical>, 1.0, CHARACTER	40	

Table

Name : MAESTRO
Coded name : MAESTRO
Row length : 205

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base ...	RFC	RFC	<Logical>, 1.0, CHARACTER	14	
Base ...	APELLIDO PATERNO	AP_PAT	<Logical>, 1 0, CHARACTER	20	
Base . .	APELLIDO MATERNO	AP_MAT	<Logical>, 1.0, CHARACTER	20	
Base ...	NOMBRE	NOMBRE	<Logical>, 1.0, CHARACTER	20	
Base .	SEXO	SEXO	<Logical>, 1.0, CHARACTER	1	
Base ..	ESTADO_CIVIL	EDO_CIVIL	<Logical>, 1.0, CHARACTER	1	
Base ...	NACIONALIDAD	NACIONALID	<Logical>, 1 0, CHARACTER	1	

Figura 7

Table

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base .	TELEFONO CASA	TEL_CASA	<Logical>, 1 0, CHARACTER	15	
Base ...	TELEFONO TRABAJO	TEL_TRABAJ	<Logical>, 1 0, CHARACTER	15	
Base ...	CALLE Y NUMERO	CALLE_NUM	<Logical>, 1.0, CHARACTER	30	
Base .	COLONIA	COLONIA	<Logical>, 1.0, CHARACTER	20	
Base	POBLACION	POBLACION	<Logical>, 1.0, CHARACTER	20	
Base ...	ESTADO	ESTADO	<Logical>, 1 0, CHARACTER	20	
Base ..	CODIGO POSTAL	COD_POS	<Logical>, 1.0, CHARACTER	8	

Table

Name : MATERIA
Coded name : MATERIA
Row length : 52

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base ...	CVE_MATERIA	CLV_MATERI	<Logical>, 1.0, CHARACTER	8	
Base .	NOMBRE MATERIA	NOM_MATERI	<Logical>, 1 0, CHARACTER	40	
Foreig...	CLAVE UNIDAD ADMINIS .	CLV_UADMI	<Logical>, 1.0, CHARACTER	4	

Table

Name : NIVEL
Coded name : NIVEL
Row length : 47

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base ...	CLAVE NIVEL	CLV_NIVEL	<Logical>, 1 0, CHARACTER	5	
Base ...	DESCRIPCION NIVEL	NOM_NIVEL	<Logical>, 1 0, CHARACTER	30	
Base ..	SUELDO	SUELDO	<Logical>, 1.0, NUMBER	4	2
Foreig.	PARTIDA	PAR	<Logical>, 1.0, CHARACTER	3	
Foreig...	CLAVE CATEGORIA	CLV_CATEGO	<Logical>, 1.0, CHARACTER	5	

Table

Name : SEDE HOSPITALARIA
Coded name : SEDE
Row length : 54

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base ...	CLAVE SEDE HOSPITALA...	CLV_SH	<Logical>, 1 0, CHARACTER	6	
Base ...	NOMBRE SEDE HOSPITA...	NOM_SH	<Logical>, 1.0, CHARACTER	40	
Foreig ..	SIGLAS	SIGLAS	<Logical>, 1.0, CHARACTER	8	

Table

Name : TIPOS DE SOLICITUD
Coded name : SOLICITU
Row length : 13

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base .	CLAVE SOLICITUD	CLV_SOLICI	<Logical>, 1.0, CHARACTER	1	

Figura 8

Table

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base	DESCRIPCION SOLICITUD	NOM_SOLICI	<Logical>, 1 0, CHARACTER	12	

Table

Name : TRAMITES
Coded name : TRAMITE
Row length : 155

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base	NUMERO DE EXPEDIENTE	NUM_EXPEDI	<Logical>, 1.0, CHARACTER	8	
Base ..	NUMERO DE PLAZA	NUM_PLAZA	<Logical>, 1 0, CHARACTER	8	
Base .	OBSERVACIONES	OBSERVA	<Logical>, 1.0, CHARACTER	50	
Base ...	HORAS PRACTICAS	NH_PRAC	<Logical>, 1.0, NUMBER	2	0
Base .	HORAS TEORICAS	NH_TEORICA	<Logical>, 1 0, NUMBER	2	0
Base ...	HORAS TOTALES	NH_TOTAL	<Logical>, 1 0, NUMBER	2	0
Base ..	PRINCIPIO DE APOYO	A_PARTIR	<Logical>, 1.0, DATE		
Base .	FIN APOYO	CON_LIMITE	<Logical>, 1 0, DATE		
Base ...	AFECTA NOMINA	AFECTA_NOM	<Logical>, 1.0, CHARACTER	1	
Base ..	FECHA RECIBIDO	F_REC	<Logical>, 1.0, DATE		
Base .	ENVIO DEPENDENCIA	F_ENV_DEP	<Logical>, 1 0, DATE		
Base ...	REGISTRO DEPENDENCIA	F_REG_DEP	<Logical>, 1.0, DATE		
Base .	ENTREGA FIRMA SOLICI..	E_FIR_SOL	<Logical>, 1 0, DATE		
Base ..	RECEPCION FIRMA SOLI...	R_FIR_SOL	<Logical>, 1 0, DATE		
Base .	ENTREGA FIRMA TITULAR	E_FIR_TIT	<Logical>, 1 0, DATE		
Base ...	RECEPCION FIRMA TITU ..	R_FIR_TIT	<Logical>, 1.0, DATE		
Base ..	ENVIO CONSEJO TECNICO	F_ENV_CT	<Logical>, 1 0, DATE		
Base ...	REGISTRO CONSEJO TE ..	F_REG_CT	<Logical>, 1.0, DATE		
Base	ENVIO DGP	F_ENV_DGP	<Logical>, 1.0, DATE		
Base ..	FOLIO DGP	FOLIO_DGP	<Logical>, 1.0, CHARACTER	10	
Base ...	MINUTA 8	F_MIN_8	<Logical>, 1 0, DATE		
Base .	MINUTA 6 Y 7	F_MIN6_7	<Logical>, 1.0, DATE		
Base . .	FECHA COBRO	F_COBRO	<Logical>, 1.0, DATE		
Base .	FECHA RECHAZO	F_RECHAZO	<Logical>, 1 0, DATE		
Base ...	FECHA REINGRESO	F_REINGRE	<Logical>, 1.0, DATE		
Base . .	PERSONAL	PERSONAL	<Logical>, 1 0, CHARACTER	15	
Foreig...	CVE_MATERIA	CLV_MATERI	<Logical>, 1.0, CHARACTER	8	
Foreig...	CLAVE CATEGORIA	CLV_CATEGO	<Logical>, 1.0, CHARACTER	5	
Foreig ..	PARTIDA	PAR	<Logical>, 1.0, CHARACTER	3	
Foreig...	CLAVE NIVEL	CLV_NIVEL	<Logical>, 1 0, CHARACTER	5	
Foreig...	CLAVE SEDE HOSPITALA ..	CLV_SH	<Logical>, 1.0, CHARACTER	6	
Foreig...	CLAVE FUNCION	CLV_FUNC	<Logical>, 1.0, CHARACTER	3	
Foreig...	CLAVE CAUSA	CLV_CAUSA	<Logical>, 1 0, CHARACTER	2	
Foreig. .	CLAVE SOLICITUD	CLV_SOLICI	<Logical>, 1.0, CHARACTER	1	
Foreig...	RFC	RFC	<Logical>, 1.0, CHARACTER	14	
Base ..	STATUS	STATUS	<Logical>, 1.0, CHARACTER	10	

Table

Name : UNIDAD ADMINISTRATIVA
Coded name : U_ADMINI
Row length : 38

Columns

Cat.	Name	Coded name	Domain	Len.	Dec.
Base ..	CLAVE UNIDAD ADMINIS ..	CLV_UADMI	<Logical>, 1 0, CHARACTER	4	
Base ...	NOMBRE UNIDAD ADMINI...	NOM_UADMI	<Logical>, 1.0, CHARACTER	30	

Figura 9

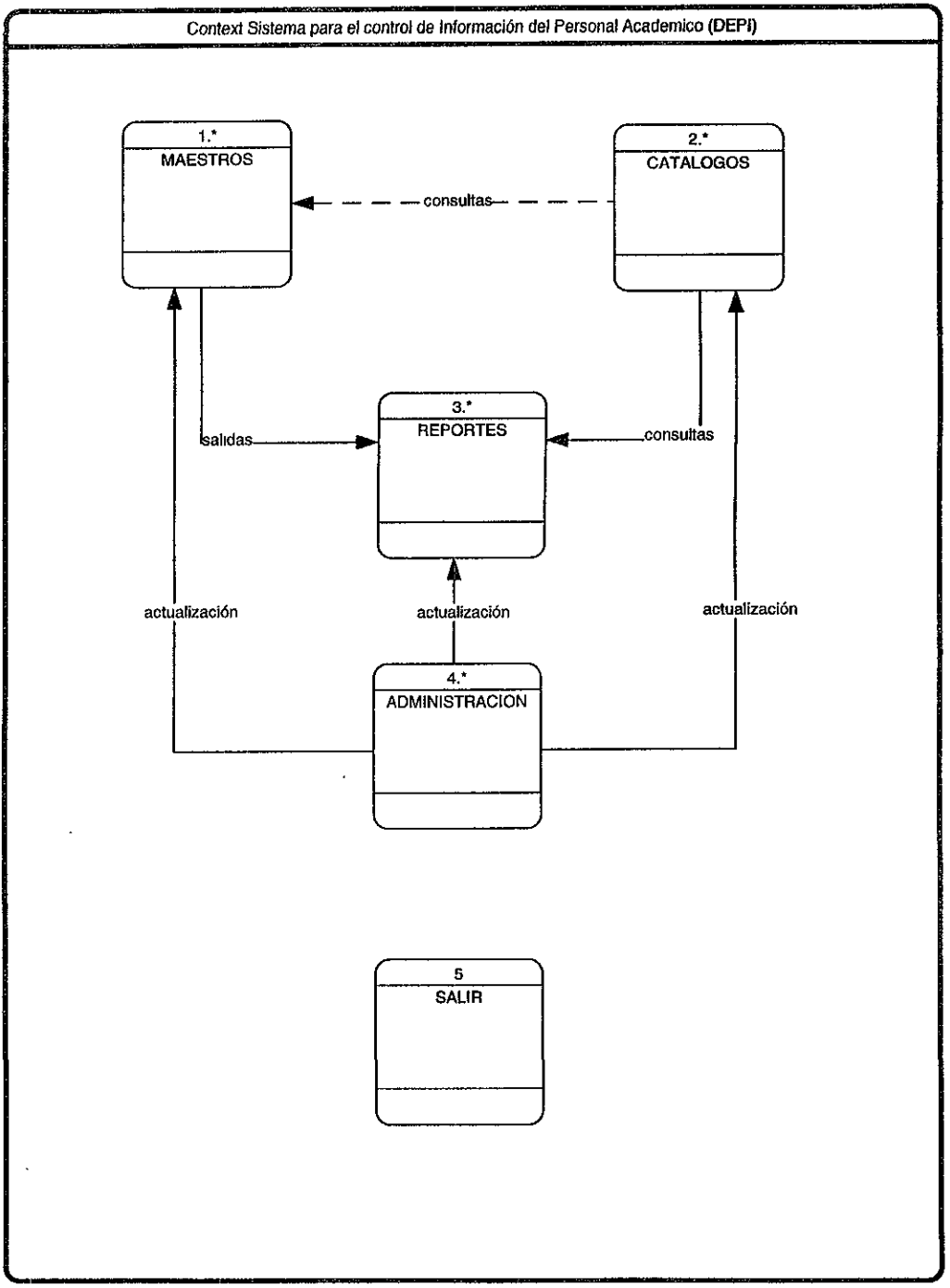
Context Sistema para el control de Información del Personal Academico (DEPI)
1.*
MAESTROS
2.*
CATALOGOS
consultas
3.*
REPORTES
salidas
consultas
actualización
actualización
actualización
4.*
ADMINISTRACION
5
SALIR

Figura 11

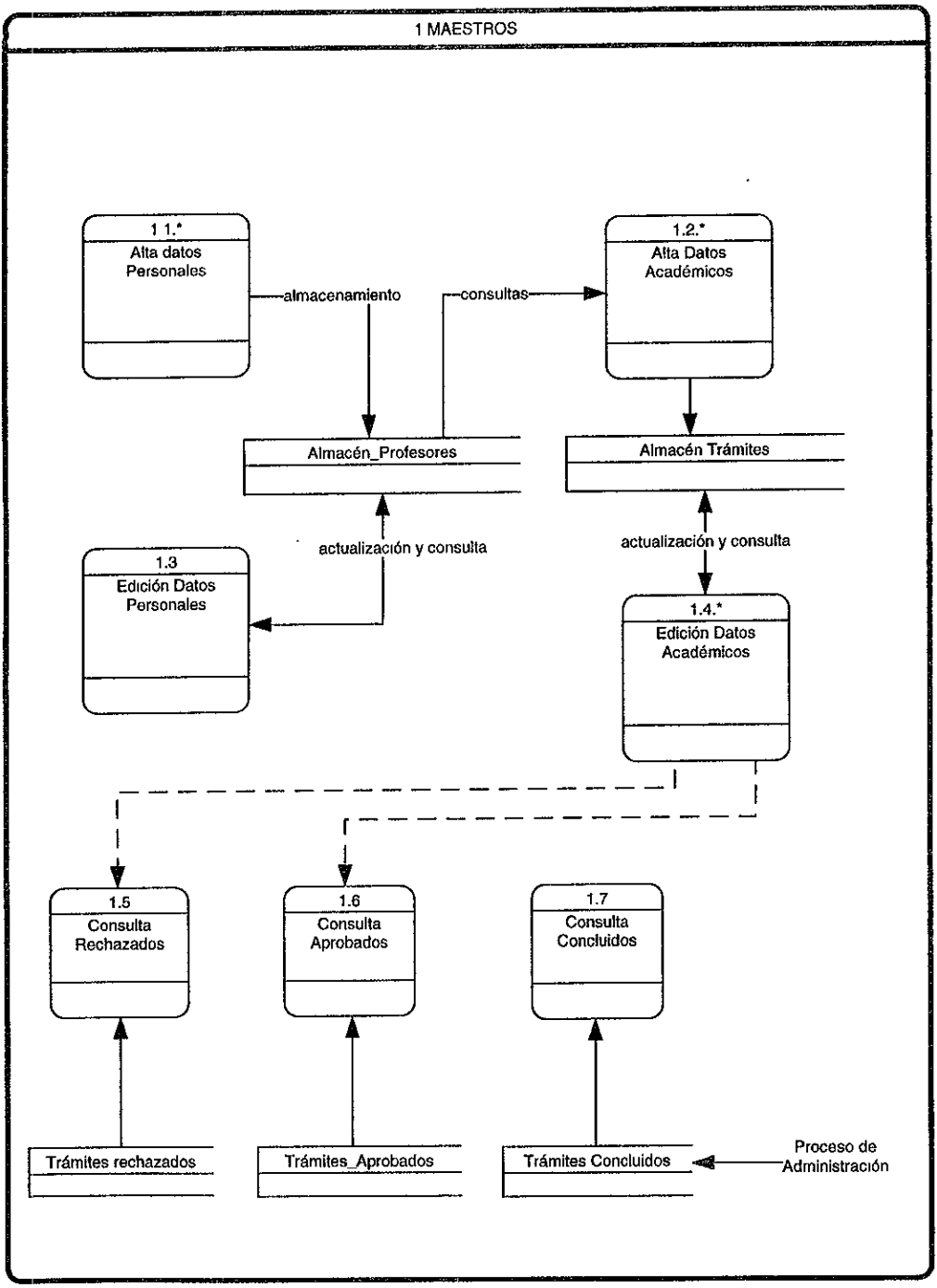
1 MAESTROS
1 1.*
Alta datos Personales
almacenamiento
consultas
1.2.*
Alta Datos Académicos
Almacén_Profesores
Almacén Trámites
actualización y consulta
actualización y consulta
1.3
Edición Datos Personales
1.4.*
Edición Datos Académicos
1.5
Consulta Rechazados
1.6
Consulta Aprobados
1.7
Consulta Concluidos
Trámites rechazados
Trámites_Aprobados
Trámites Concluidos
Proceso de Administración

Figura 12

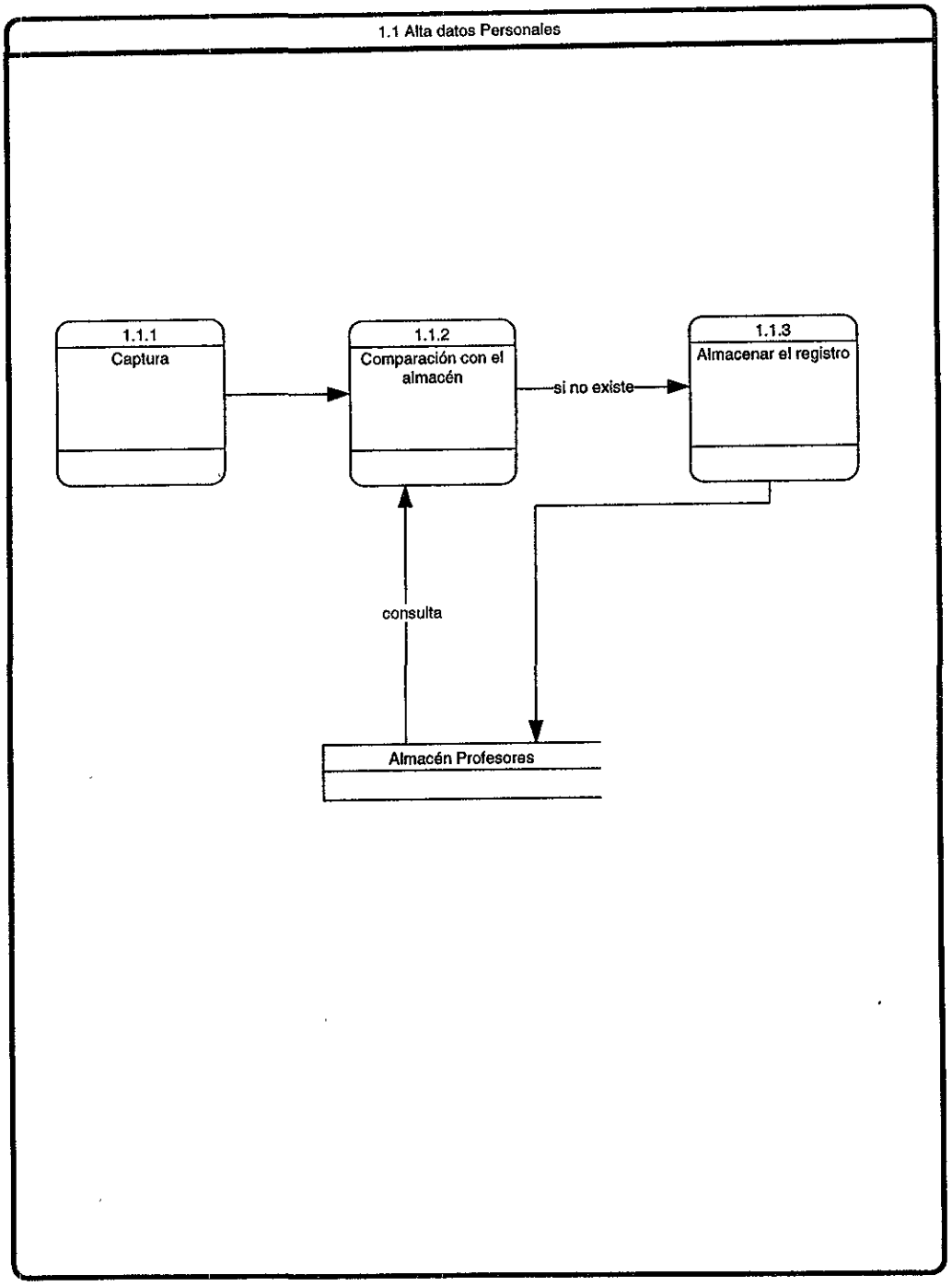
1.1 Alta datos Personales
1.1.1
Captura
1.1.2
Comparación con el almacén
si no existe
1.1.3
Almacenar el registro
consulta
Almacén Profesores

Figura 13

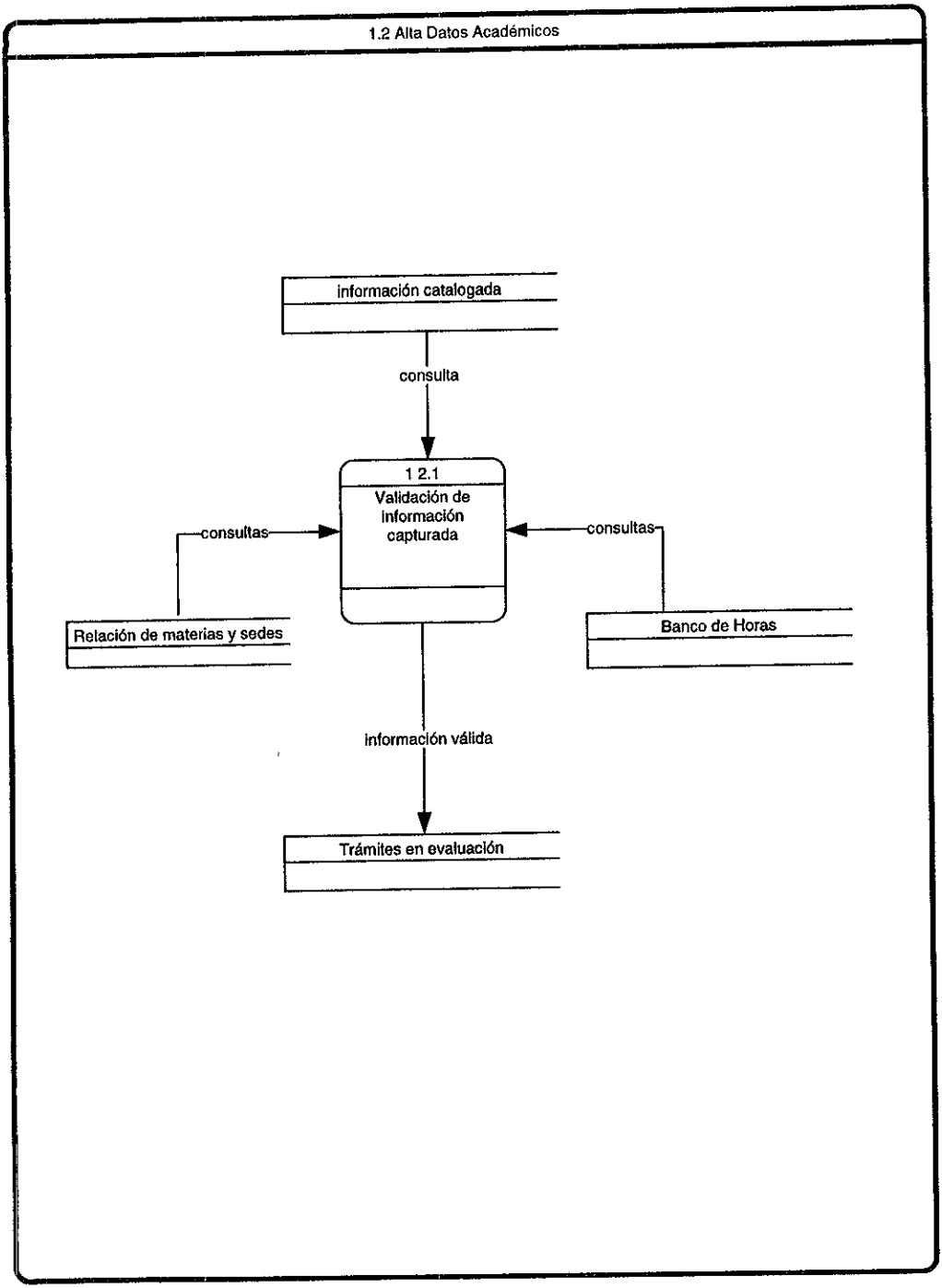
1.2 Alta Datos Académicos
información catalogada
consulta
1 2.1
Validación de información capturada
consultas
consultas
Relación de materias y sedes
Banco de Horas
información válida
Trámites en evaluación

Figura 14

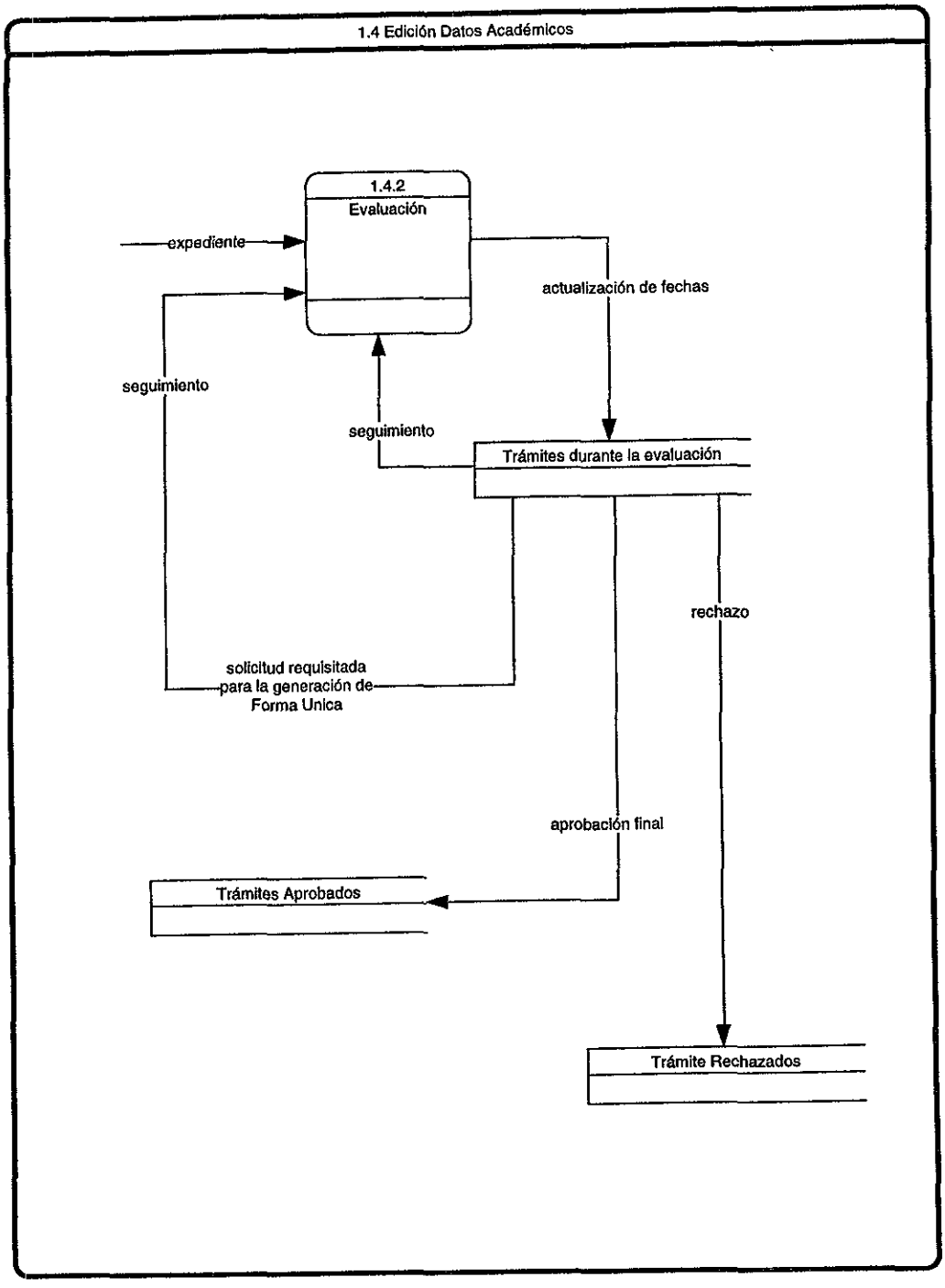
1.4 Edición Datos Académicos
1.4.2
Evaluación
expediente
actualización de fechas
seguimiento
seguimiento
Trámites durante la evaluación
rechazo
solicitud requisitada
para la generación de
Forma Unica
aprobación final
Trámites Aprobados
Trámite Rechazados

Figura 15

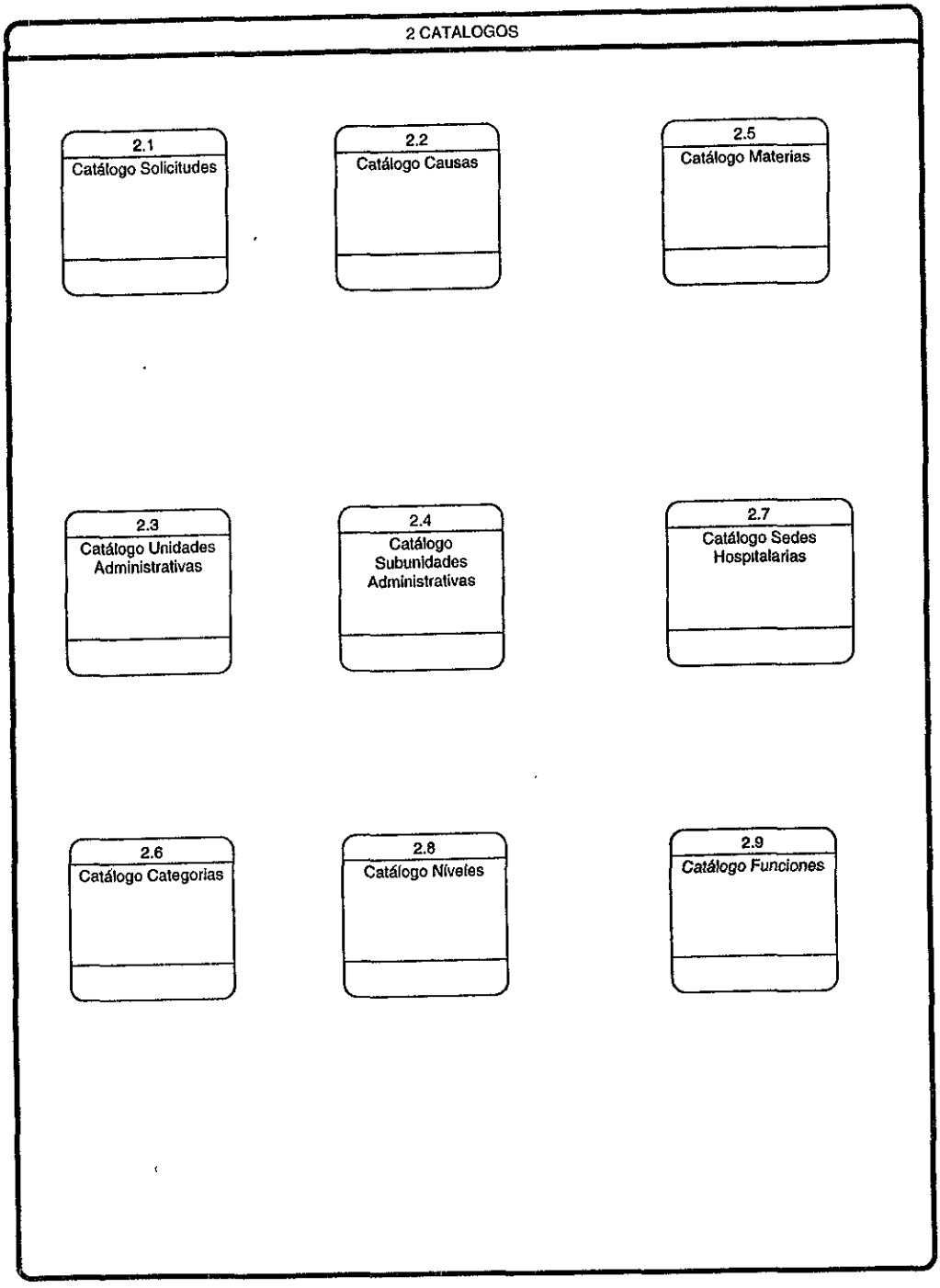
2 CATALOGOS
2.1
Catálogo Solicitudes
2.2
Catálogo Causas
2.5
Catálogo Materias
2.3
Catálogo Unidades Administrativas
2.4
Catálogo Subunidades Administrativas
2.7
Catálogo Sedes Hospitalarias
2.6
Catálogo Categorias
2.8
Catálogo Niveles
2.9
Catálogo Funciones

Figura 16

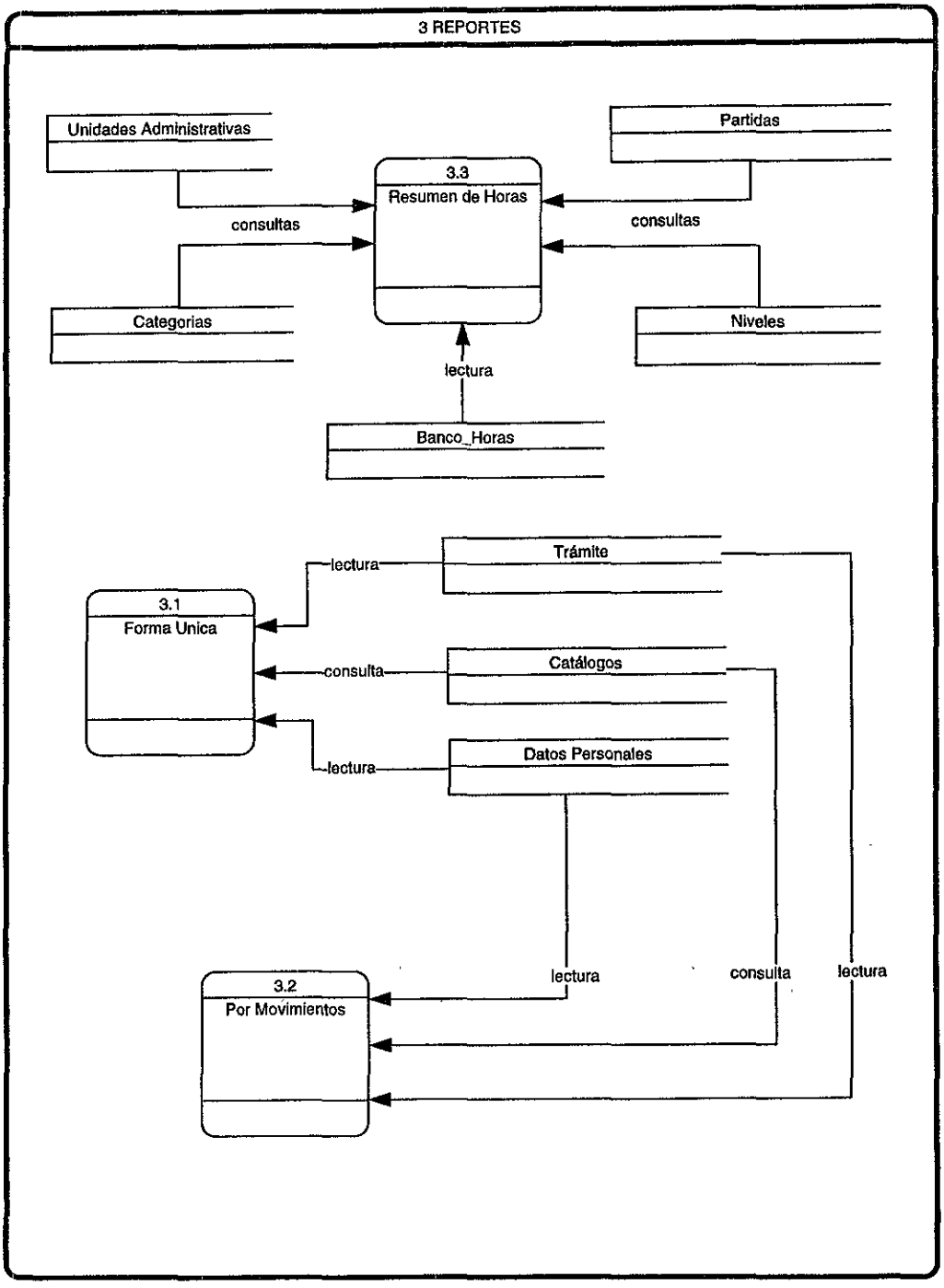
3 REPORTES
Unidades Administrativas
Partidas
3.3
Resumen de Horas
consultas
consultas
Categorias
Niveles
lectura
Banco_Horas
Trámite
lectura
3.1
Forma Unica
consulta
Catálogos
lectura
Datos Personales
lectura
consulta
lectura
3.2
Por Movimientos

Figura 17

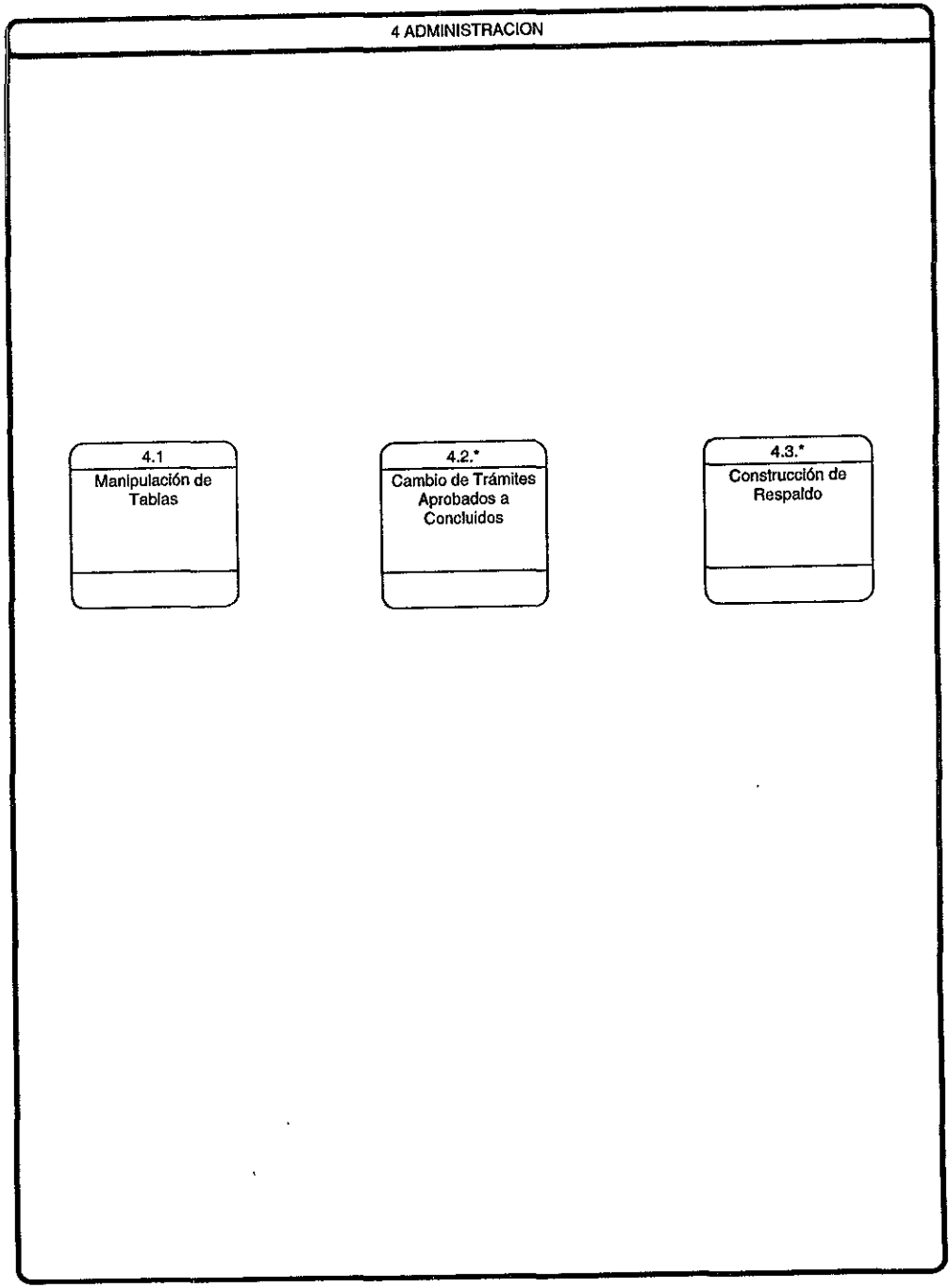
4 ADMINISTRACION
4.1
Manipulación de Tablas
4.2.*
Cambio de Trámites Aprobados a Concluidos
4.3.*
Construcción de Respaldo

Figura 18

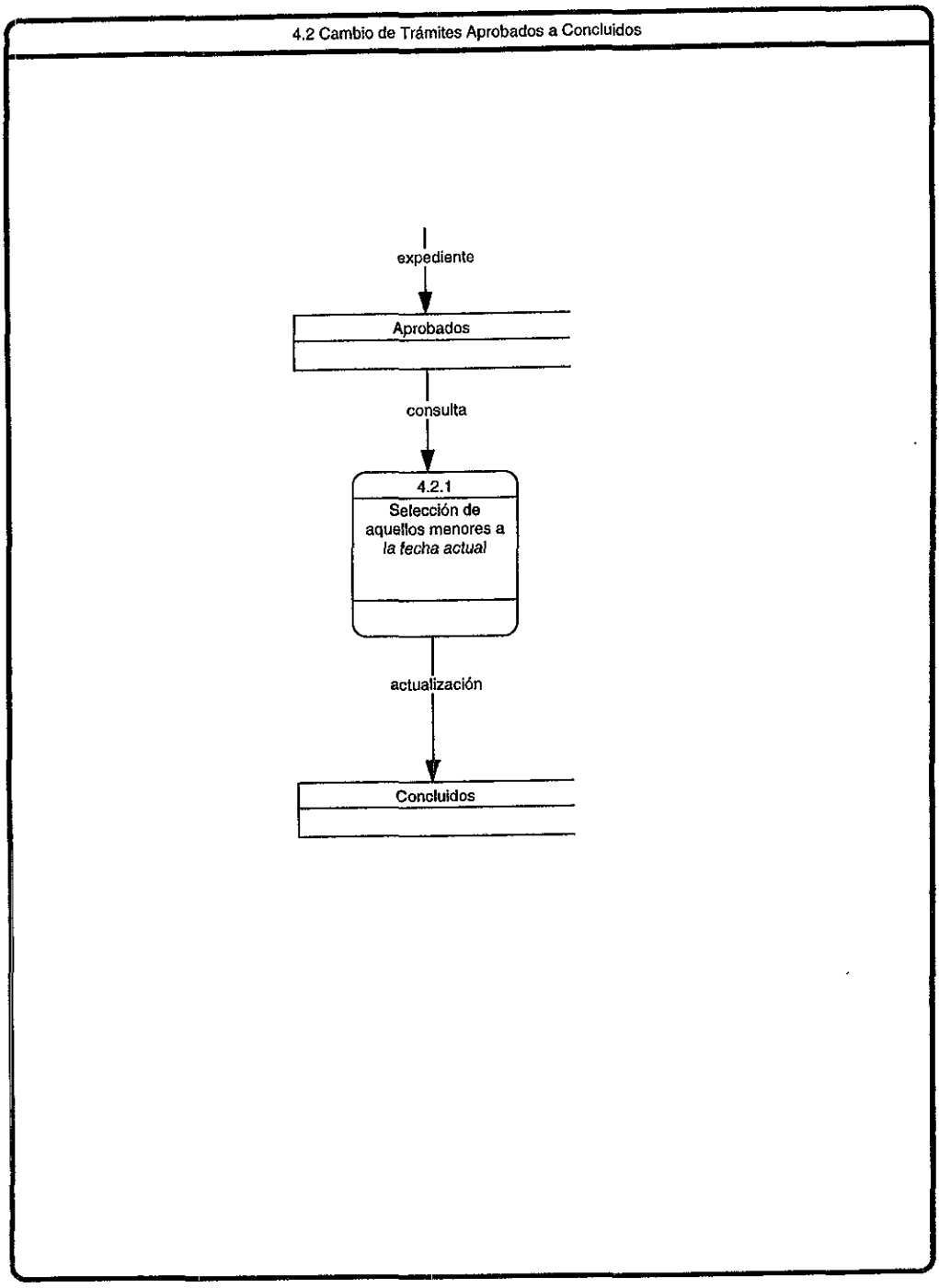
4.2 Cambio de Trámites Aprobados a Concluidos
expediente
Aprobados
consulta
4.2.1
Selección de aquellos menores a la fecha actual
actualización
Concluidos

Figura 19

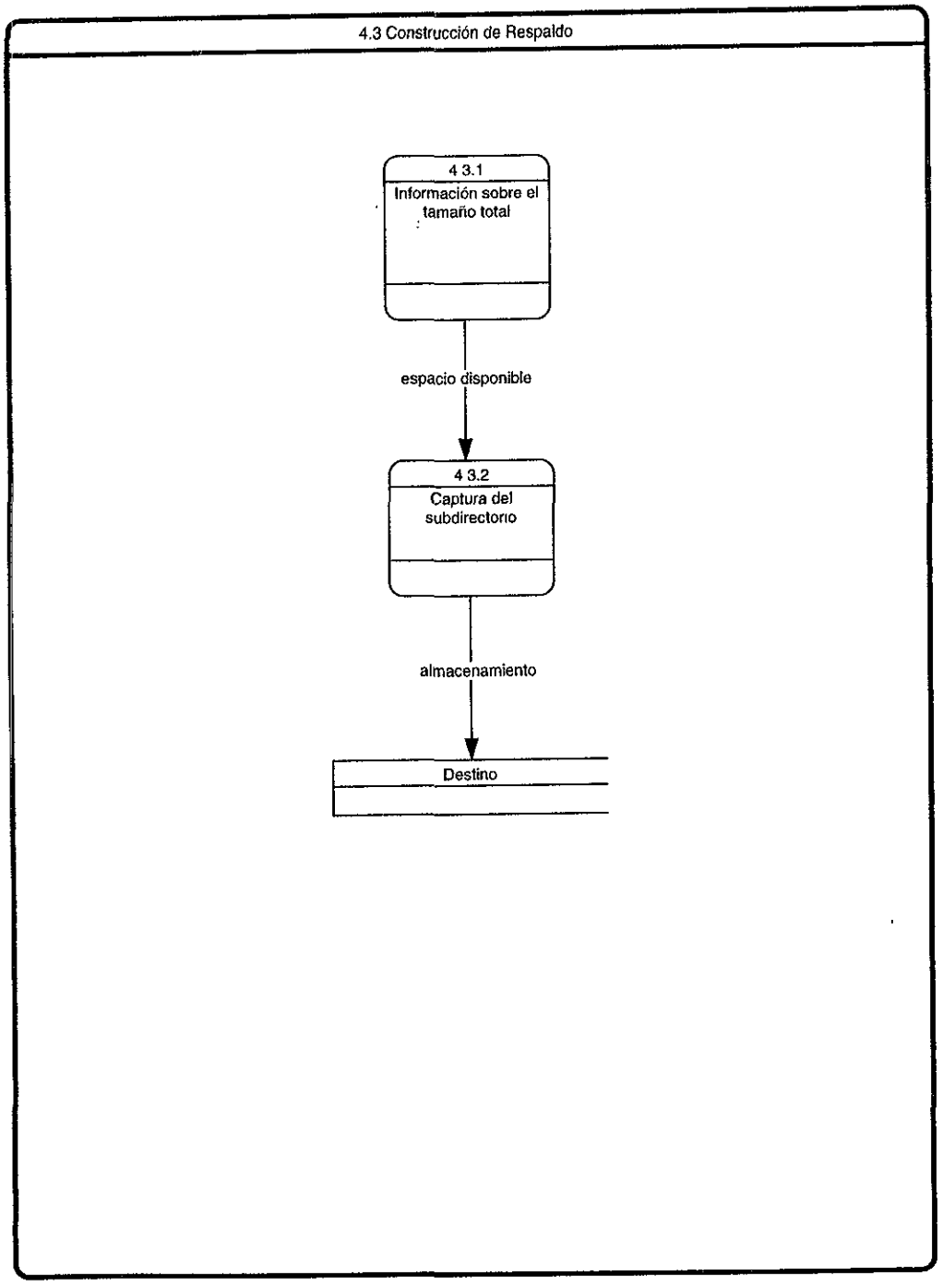
4.3 Construcción de Respaldo
4 3.1
Información sobre el tamaño total
espacio disponible
4 3.2
Captura del subdirectorio
almacenamiento
Destino

Figura 20

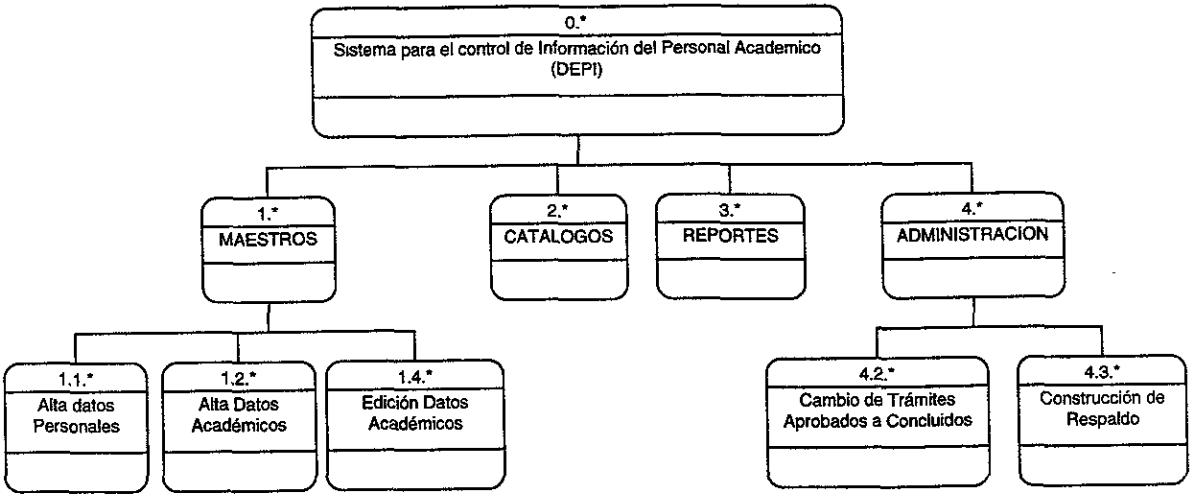
0.*
Sistema para el control de Información del Personal Academico (DEPI)
1.*
MAESTROS
2.*
CATALOGOS
3.*
REPORTES
4.*
ADMINISTRACION
1.1.*
Alta datos Personales
1.2.*
Alta Datos Académicos
1.4.*
Edición Datos Académicos
4.2.*
Cambio de Trámites Aprobados a Concluidos
4.3.*
Construcción de Respaldo

Figura 21

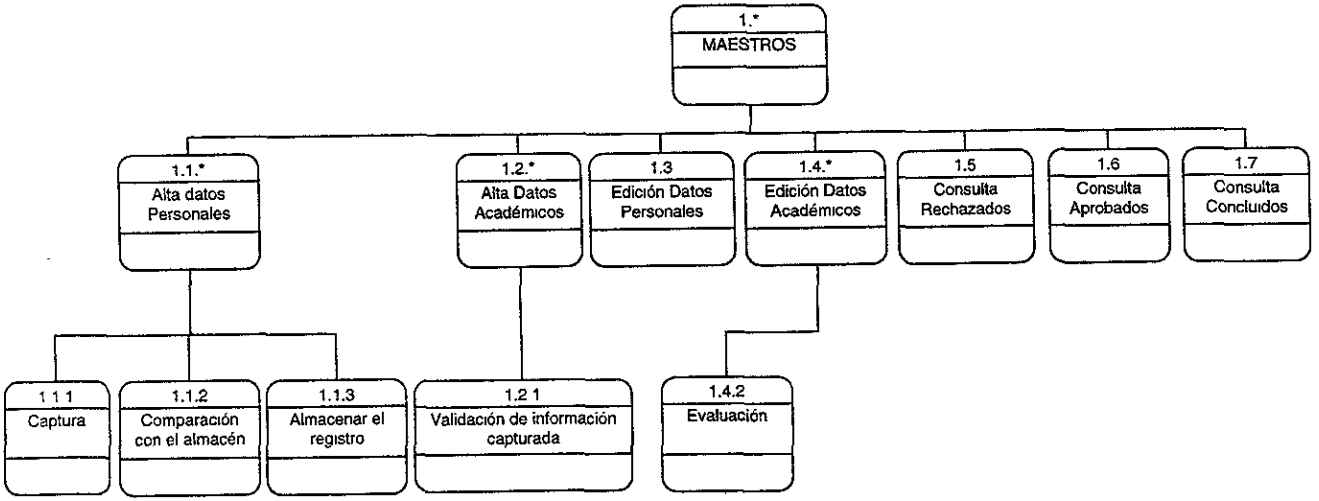
1.*
MAESTROS
1.1.*
Alta datos Personales
1.2.*
Alta Datos Académicos
1.3
Edición Datos Personales
1.4.*
Edición Datos Académicos
1.5
Consulta Rechazados
1.6
Consulta Aprobados
1.7
Consulta Concluidos
1.1.1
Captura
1.1.2
Comparación con el almacén
1.1.3
Almacenar el registro
1.2.1
Validación de información capturada
1.4.2
Evaluación

Figura 22

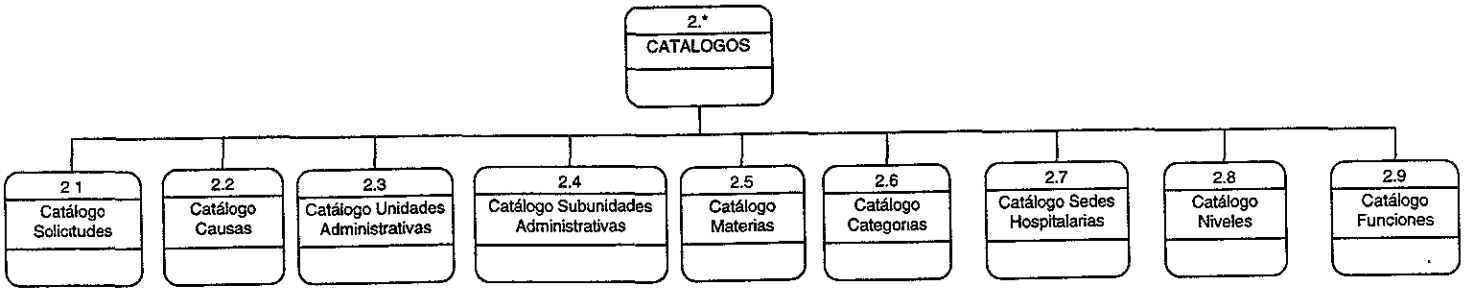
2.*
CATALOGOS
2.1
Catálogo Solicitudes
2.2
Catálogo Causas
2.3
Catálogo Unidades Administrativas
2.4
Catálogo Subunidades Administrativas
2.5
Catálogo Materias
2.6
Catálogo Categorias
2.7
Catálogo Sedes Hospitalarias
2.8
Catálogo Niveles
2.9
Catálogo Funciones

Figura 23

4.*
ADMINISTRACION
4.1
Manipulación de Tablas
4.2.*
Cambio de Trámites Aprobados a Concluidos
4.2.1
Selección de aquellos menores a la fecha actual
4.3.*
Construcción de Respaldo
4.3.1
Información sobre el tamaño total
4.3.2
Captura del subdirectorio

Figura 25

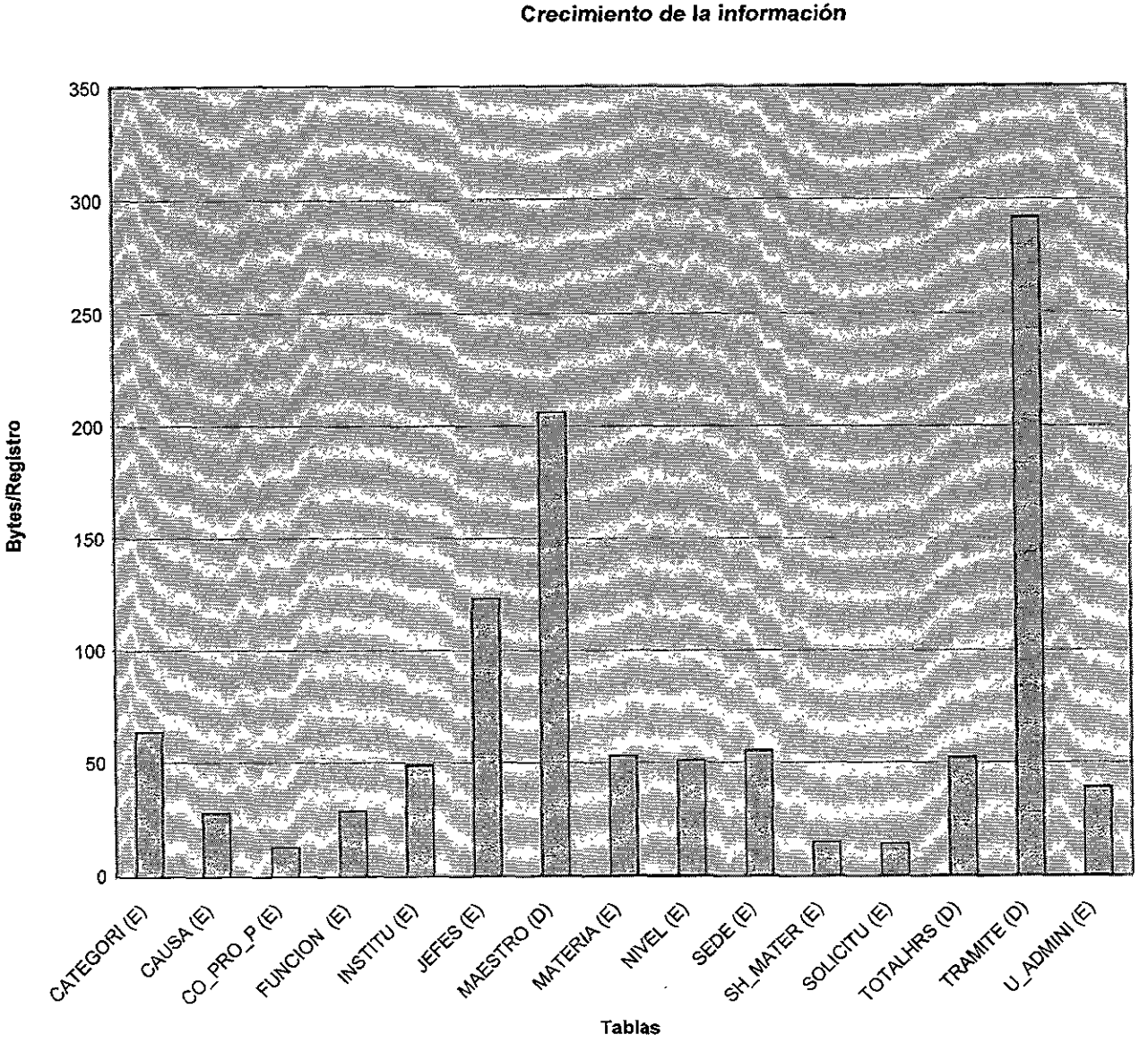
Crecimiento de la información
Bytes/Registro
350
300
250
200
150
100
50
0
CATEGORI (E)
CAUSA (E)
CO_PRO_P (E)
FUNCION (E)
INSTITU (E)
JEFES (E)
MAESTRO (D)
MATERIA (E)
NIVEL (E)
SEDE (E)
SH_MATER (E)
SOLICITU (E)
TOTALHRS (D)
TRAMITE (D)
U_ADMINI (E)
Tablas

Figura 26

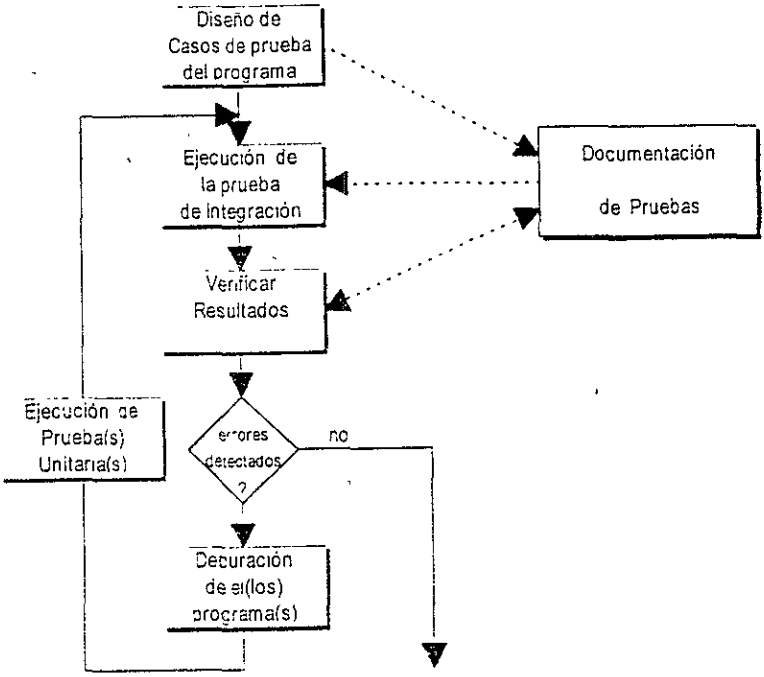
Diseño de
Casos de prueba
del programa
Ejecución de
la prueba
de integración
Verificar
Resultados
errores
detectados
?
no
Ejecución de
Prueba(s)
Unitaria(s)
Depuración
de el(los)
programa(s)
Documentación
de Pruebas

Figura 27

BIBLIOGRAFÍA

- Carlo Batini, Stefano Ceri, Shamkant B. Navathe. Diseño conceptual de bases de datos, un enfoque de entidades – interrelaciones. Addison – Wesley / Diaz de Santos. Estado Unidos. 1994.

- Guillermo Levine. Introducción a la computación y a la programación estructurada. Mc graw Hill. México D.F. 1990.

- Grupo Telsys – Ingenética. Análisis y Diseño de Sistemas de Computación. Ingenética capacitación. México D.F. 1994.

- Bancomer. Proceso de pruebas, Planeación y arquitectura Tecnológica. Calidad Aplicativa.

- Daniel Morris, Joel Brandon. Reingeniería, cómo aplicarla con éxito en los negocios. Mc Graw Hill.

- Infomedia. El modelo relacional y DB2.

- Infomedia. Diseño de bases de datos.

- M. en C. Ralf Eder Lange, M. en C. Rafael Martínez Casanova. Diagrama Entidad Relación, Diagrama de Transición de Estados.

- Apuntes del curso de Estructuras de Datos.

- Stephen J. Straley. Clipper 5.0, cubre versión 5.01. Megabyte Grupo Noriega Editores.

- Stephen J. Straley. Programming in Clipper, The definitive guide to the Clippper dBase compiler. Addison – Wesley Publishing Company Inc.

- José Javier García Badell. Clipper versión 5.0, guía del compilador para dbase III y dBase IV. Mc Graw Hill. España.

- Franciso Marín Quirós, Antonio Quirós Casado, Antonio Torres Lozano. Clipper summer '87. Grupo Eidos, coedición Macrobit.

- Brian Besser, John Benford, Michel Aknine. Oracle 7 para desarrolladores. Volumen 1, 2. Copyright Oracle Corporation. Enero 1994.

- Debby Kramer. Introduction to Oracle: SQL and PL/SQL using procedure builder. Volumer one, two. Copyright Oracle Corporation. Enero 1992.