27
2EJ

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE CONTADURIA Y ADMINISTRACION

SISTEMA AUTOMATIZADO DE MOVIMIENTOS AL ORGANIGRAMA DE LA U.N.A.M.

SEMINARIO DE INVESTIGACION EN INFORMATICA

QUE PARA OBTENER EL TITULO DE:

LICENCIADO EN INFORMATICA

PRESENTA:

BLANCA LYDA SAUL PACHECO

1994

ASESOR DE SEMINARIO

L.A.E. MARIO NOVOA GAMAS

U/2065/96

FACULTAD DE CONTADURIA Y ADMINISTRACION

MEXICO D.F.

ACTUALIZADO A 1996

OCT. 9 1996

COORDINACION DE EXAMENES PROFESIONALES

TESIS CON FALLA DE ORIGEN

A mis padres, por el amor y motivación que siempre me han procurado, por sus regaños que han sido motor de cambio, por su paciencia y sabios consejos, gracias.

A mis hermanos, Isaac y Lupita, por su cariño, consejos y tremendas parrandas, por músicos, teatreros y extravagantes, gracias.

A José Luis, por su amor y comprensión, por su estímulo y futuro promisorio, gracias.

A las personas especiales que me han apoyado y ayudado a caminar, a saber y a crecer, que me han dado cariño y amistad incondicional, gracias.

A mis niños, por su amor y travesuras.

A Mario Novoa, mi asesor, por su tiempo y dedicación, por sus consejos y orientación en la realización del presente trabajo, gracias.

A Angel, Fernando, Gerardo y Ricardo que me permitieron abordar este tema, por sus consejos y apoyo, en especial al compadre, por su valiosa ayuda y paciencia, gracias.

A la Universidad Nacional Autónoma de México, por albergar a tantos estudiantes que como yo tratamos de aprovechar el conocimiento y experiencias que nos brinda.

ÍNDICE

PRÓLOGO

En ocasiones se piensa que los Licenciados en Informática son programadores que además tienen conocimiento de paquetes, lo cual no es del todo falso, pero realmente el objetivo de esta carrera, el que tratamos de alcanzar paso a paso, conforme vamos cubriendo el plan de estudios, es la de formarnos como profesionistas capaces de comprender las necesidades de información de los usuarios, documentarlas, analizarlas, optimizar los recursos, simplificar los sistemas de información y generar instrumentos útiles para la toma de decisiones y solución de problemas de planeación u organización.

Esa necesidad social es la que el Licenciado en Informática debe cubrir y por tanto tener siempre presente.

Lo anterior no significa limitar los alcances del profesionista, sino ubicarnos en nuestra realidad, ¿qué podemos esperar de un sistema cuyo análisis y diseño sea deficiente? pues nada, ya que una programación aunque esté bien hecha, no podrá resarcir las debilidades del diseño y mucho menos del análisis, que son las partes fundamentales en el desarrollo de sistemas, es por eso necesario conservar la esencia de esta licenciatura.

Por último, no quiero dejar a un lado el hecho de que elaborar la tesis es una buena oportunidad de aprender, corregir o fortalecer nuestros puntos débiles, capitalizar las experiencias y cúmulos de conocimiento adquiridos durante la carrera universitaria y desempeño profesional e incrementar la capacidad crítica para focalizar problemas y afrontarlos con métodos y exponerlos siguiendo técnicas de comunicación.

INTRODUCCION

Actualmente los movimientos a trabajadores universitarios que se traducen en modificaciones al organigrama de la UNAM, se realizan a través de oficios que viajan de una dependencia a otra, y cada una de estas dependencias lleva un control diferente, de acuerdo a sus políticas internas y nivel de desarrollo informático alcanzado.

Lo anterior ocasiona tiempos excesivos en la tramitación de movimientos al organigrama de la dependencia afectando tanto al trabajador demorando el tiempo para su procesamiento en nómina y a la dependencia en su afectación presupuestal entre otras cosas.

Es por ello que esta tesis plantea el desarrollo de una herramienta que permita a cada dependencia mejorar el control de los movimientos de su personal estableciendo interfases gráficas con información acerca de sus plazas y estructura organizacional, a fin de que los movimientos a los atributos de los objetos plazas o creación de objetos trabajador y modificación de sus atributos sean efectuados con la misma confiabilidad y seguridad que actualmente se realizan pero con información más rápida y oportuna para la toma de

decisiones y agilizando el tiempo de los trámites en la dependencia; dichos trámites en términos administrativos son los comunmente denominados: nuevo ingreso, reingreso, retabulación, baja, etc.

Se propone como hipótesis para el presente trabajo, la factibilidad de elaborar un análisis y diseño orientado a objetos aunado a una programación en Borland C++, para elaborar un sistema de cómputo que satisfaga los requerimientos de las dependencias en cuanto a información y en la agilización del movimiento para el trabajador.

Para tal efecto se utiliza una metodología de planeación que permite medir el desempeño y dar seguimiento a las actividades planteadas.

La tesis se conforma de seis capítulos y dos anexos, en el primer capítulo se plantea al lector los elementos a considerar como son el trabajador universitario, los diferentes movimientos que puede tener dentro de la Institución y las normas que los rigen. En el segundo capítulo se plantea el objetivo que se pretende alcanzar con la conclusión de este trabajo.

El tercer y cuarto capítulo que corresponden a las fases de análisis y diseño del desarrollo de un sistema respectivamente, fueron

desarrollados basados en la metodología orientada a objetos de Coad y Yourdon[1], en éstos se plantean los requerimientos, se definen las clases y objetos con los que se trabajará, las estructuras necesarias para su funcionamiento, los atributos y servicios de cada clase, la representación del dominio del problema, las interfases de interacción con el usuario y los manejadores de tareas y de datos que se requiere.

El quinto capítulo corresponde al control de proyecto utilizado para el presente trabajo y finalmente a manera de conclusiones, se plantea en el sexto capítulo las consideraciones generales derivadas del análisis y diseño del sistema desarrollado, los alcances y limitaciones del presente trabajo y alguna conclusión que de manera personal me representó este trabajo.

La fase de implementación en el desarrollo de sistemas se presenta como el Anexo A, en el cual se incluye el código de los programas desarrollados en Borland C++ versión 4.0 para Windows. Y el anexo B, presenta las pantallas y reportes del sistema resultante

[1] COAD Peter, YOURDON Edward, "OBJECT - ORIENTED ANALISYS", New Jersey, Prentice Hall, Inc.1991
COAD Peter, YOURDON Edward. "OJECT-ORIENTED DESIGN". New Jersey, Prentice-Hall, Inc. 1991

I. SITUACIÓN ACTUAL

I.1. El trabajador universitario.

La Universidad Nacional Autónoma de México cuenta con 2 grandes grupos de trabajadores: los académicos, quienes prestan sus servicios de docencia o investigación, y los administrativos, quienes prestan sus servicios generales de apoyo para cubrir las funciones sustanciales de la misma.

Cada uno de estos grupos contiene diferentes categorías y niveles; así pues el personal académico está integrado por:

1.- Técnicos académicos

2.- Profesores

3.- Investigadores

Los técnicos académicos pueden laborar tiempo completo (40 horas) o medio tiempo (20 horas), pueden tener categorías de auxiliar, asociado o titular, y en cada una de estas categorías hay tres niveles "A", "B", o "C".

Dentro de los profesores o investigadores encontramos a los

ordinarios de carrera, quienes laboran medio tiempo o tiempo completo con categorías Asociado o Titular, pudiendo ocupar niveles "A", "B" o "C".

En cuanto al personal administrativo, encontramos que está integrado por:

1.- Base

2.- Confianza

3.- Funcionario

Los trabajadores de base, pueden laborar 32, 40 o 48 horas dependiendo de la categoría que ocupen en cada rama. Se tienen 21 categorías en la rama administrativa, 13 categorías en la rama obrera, 11 categorías en la rama auxiliar de administración, 13 categorías de la rama especializada obrera, 45 categorías en la rama especializada técnica y 5 categorías en la rama profesional[1].

El personal de confianza puede laborar 32, 40 o 48 horas y ocupar cualquiera de las 106 categorías que pertenecen a este grupo.

---

[1] Se considera innecesario hacer mención de las categorías de cada rama ya que reciben el mismo trato, las diferencias, como se verá en capítulos posteriores, están dados por el tipo al que pertenecen, base, confianza o funcionario. En todo caso para mayor información puede remitirse al *Instructivo de Ejercicio y Catálogo Presupuestal.* UNAM 1994, o al *Catálogo de Puestos del Personal Administrativo de Base.* UNAM.

El personal funcionario labora 48 horas y puede ocupar cualquiera de las 109 categorías que contiene este grupo.

I.2. Los tipos de movimientos e instancias que intervienen.

Cada trabajador universitario se puede identificar dentro de un organigrama por la o las plazas que ocupa. Los movimientos a plazas ya sea académicas o administrativas, los podemos clasificar en tres tipos básicos:

- Creaciones
- Modificaciones
- Bajas

Estos se refieren a la incorporación, modificación o eliminación de un nuevo registro en la plantilla[2]. Estos registros contienen información sobre cada nodo o plaza, tales como: categoría, nivel, número de plaza, sueldo, etc. Para que un trabajador ocupe una de estas plazas debe existir previamente, por lo que las dependencias solicitan el movimiento correspondiente directamente a la Dirección General de

[2] Instrumento que utiliza la Dirección General de Programación y Presupuestación para controlar las plazas.

Programación y Presupuestación (DGPP). Esta dependencia también mantiene la información del trabajador que ocupa dicha plaza, información que obtiene de la nómina que procesa la Dirección General de Personal (DGP) o la indicación de que se encuentra vacante.

Independientemente cada trabajador puede ser referenciado dentro de la Institución por un identificador único, que en este caso es el registro federal de Hacienda (RFC); y cada plaza es ocupada por sólo un trabajador, de tal manera que el trabajador puede tener movimientos en su historia laboral de dos tipos: uno porque la plaza que ocupa sea modificada y otro porque el trabajador ocupe diferentes plazas. De tal manera los movimientos al trabajador quedan clasificados de la siguiente manera:

1.- Alta por nuevo ingreso

2.- Alta por reingreso

3.- Otro nombramiento

4.- Retabulación

5.- Reclasificación

6.- Permuta

7.- Licencia

8.- Baja

Estos movimientos al afectar a la nómina del personal universitario los tramita la Dirección General de Personal a petición de las dependencias, sin embargo a excepción del personal de base los trámites de contratación, modificación y baja se realizan en las propias dependencias.

La Coordinación General de Asuntos Jurídicos (CGAJ) tiene dentro de sus funciones dictaminar a solicitud del trabajador, los casos de modificaciones a los registros de personal derivados de actas administrativas o rescisiones de contrato al personal administrativo que afecten su situación laboral.

La Subdirección de Estudios Administrativos de la Dirección General de Personal (SEA-DGP) se encarga de dictaminar y recomendar la incorporación de un nuevo registro de personal administrativo al organigrama, así como de las modificaciones a éste por solicitud de la dependencia.

En algunos movimientos del personal académico se requiere que previo a la solicitud de movimiento, exista un dictamen favorable de la

Comisión Dictaminadora y una ratificación del Consejo Técnico de la dependencia o dependencias interesadas en que surta efecto la solicitud.

I.3. Normas que rigen los movimientos.

Las modificaciones que se efectúan al organigrama, están sujetas a disposiciones legales y normativas que establecen las instancias competentes de la misma Institución, siendo éstas las siguientes:

1.- Ley Orgánica de la Universidad Nacional Autónoma de México.

2.- Estatuto General de la Universidad Nacional Autónoma de México.

3.- Estatuto del Personal Académico de la Universidad Nacional autónoma de México.

4.- Estatuto del Personal Administrativo al servicio de la Universidad Nacional Autónoma de México.

5.- Reglamento Interior de Trabajo del Personal Administrativo al servicio de la Universidad Nacional Autónoma de México.

6.- Reglamento del Reconocimiento al Mérito Universitario.

7.- Contrato Colectivo de Trabajo del Personal Académico.

8.- Contrato Colectivo de Trabajo del Personal Administrativo.

9.- Normatividad Administrativa.

10.- Catálogo de Puestos del Personal Administrativo de Base.

11.- Instructivo de ejercicio y catálogo presupuestal.

12.- Instructivo de Profesorado de Carrera de Enseñanza Media Superior.

13.- Instructivo para asuntos migratorios de académicos extranjeros al servicio de la UNAM.

I.4. Requerimientos.

La dinámica de cambio a las características de categoría y nivel de los trabajadores es muy alta puesto que las dependencias necesitan tener una estructura organizacional que les permitan cumplir con sus funciones docentes y administrativas para las cuales fueron creadas, además de que los trabajadores al ser recursos humanos conllevan sus propias necesidades como pueden ser las meramente económicas o en su caso, de superación y reconocimiento, lo que incrementa la

cantidad de movimientos que una dependencia efectúa sobre su personal.

Los movimientos se refieren a la modificación de los atributos de algún nodo del organigrama, pueden ser por alta de un nuevo trabajador, cambio en la categoría o nivel de un trabajador ya existente o la baja del trabajador.

A cada nodo del organigrama se le denomina plaza y por las características de ésta, es decir, la categoría y nivel, tiene un valor asociado, el cual se denomina sueldo.

Para los movimientos de alta así como algunos de modificación que representan mayor erogación de dinero por parte de la dependencia, es necesario proyectar el nuevo sueldo por lo que resta del año en curso para determinar si con el presupuesto otorgado anualmente puede solventar el sueldo que deba pagar al trabajador por el alta o movimiento efectuado.

II. OBJETIVO DEL SISTEMA

II.1. Planteamiento del sistema.

Controlar a través de medios automatizados las nuevas contrataciones y modificaciones que una Dependencia requiera efectuar al personal adscrito a ella, conforme a los movimientos y tipos de personal planteados en el capítulo I, con absoluta confiabilidad y obedeciendo a los lineamentos que dictan las normas, reglamentos y estatutos de la Institución.

II.2. Funcionalidad del sistema.

El sistema permitirá al usuario, entre otras cosas modelar un organigrama que satisfaga sus necesidades funcionales y de organización, así como obtener información rápida y veraz del personal ya sea relacionada con su historia laboral o referente a sus datos personales.

El sistema funciona bajo las mismas circunstancias en las cuales se

realizan actualmente los movimientos al organigrama por la D.G.P., es decir que, de los procesos realizados en otras dependencias como los de la Dirección General de Programación y Presupuesto, la Dirección General de Asuntos del Personal Académico etc. y que afectan al organigrama de la UNAM, sólo se toman en cuenta los datos que deban entrar al sistema y los datos que deba arrojar para ser procesados por otros sistemas e instancias o para la toma de decisiones de los usuarios.

III. ANÁLISIS

La fase de análisis en el desarrollo de sistemas es una de las más importantes ya que en ella se plasman las necesidades por las cuales se requiere el sistema y se representa de manera abstracta el problema bajo estudio. De acuerdo a la metodología de Análisis Orientado a Objetos de Peter Coad y Edward Yourdon[1] esta fase se divide en cinco niveles:[2]

1.- Sujetos.

2.- Clases y Objetos.

3.- Estructuras.

4.- Atributos.

5.- Servicios.

Dichos niveles no requieren realizarse de manera secuencial, sino que se van completando de manera gradual conforme se avanza en cada uno de los niveles.

El primer paso para identificar el dominio del sistema e ilustrar gráficamente lo que el sistema debe realizar, con base en los

---

1 COAD Peter, YOURDON Edward, "OBJECT - ORIENTED ANALISYS", New Jersey, Prentice-Hall, Inc. 1991.

2 Ob. cit., p.54.

requerimientos mencionados en el capítulo I, será a través de las clases y objetos que abarca.

III.1. Identificación de clases y objetos.

La definición que Coad y Yourdon hacen sobre clases y objetos es la siguiente:

> "Objeto: Una *abstracción* de algo del dominio del problema, reflejando la capacidad del sistema para mantener información acerca de ese algo, o para interactuar con él, o ambas; una *encapsulación*[3] de valores de *atributos* y sus *servicios* exclusivos. (sinónimo: instancia)
>
> Clase: Una descripción de uno o más objetos con un conjunto uniforme de atributos y servicios, incluyendo una descripción de como crear nuevos objetos en la clase"[4]

La simbología para las clases son cuadros divididos en tres secciones, una para el nombre, otra para los atributos y la tercera para los servicios; los objetos se ilustran como cuadros de mayor dimensión que rodean a las clases, como se ilustra en la figura 1:

---

[3] La abstracción y encapsulación son 2 de los principios para manejar la complejidad del dominio del problema: Ob.Cit., cap. 1.

[4] Ob. cit. p.53

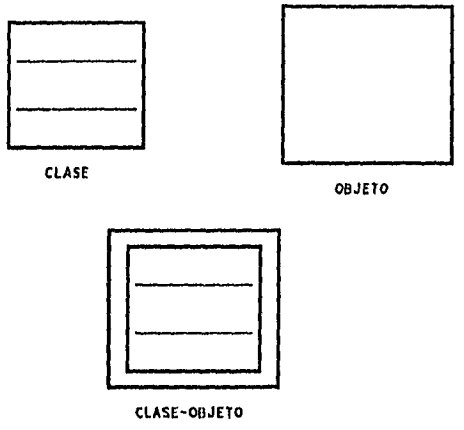


Figura 1

Las clases-objetos que se han identificado son los siguientes:

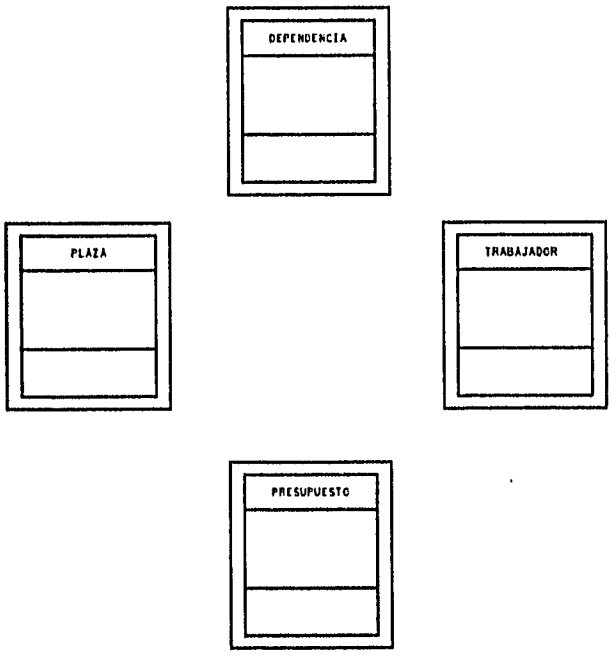


Figura 2

III.2 Identificación de sujetos y estructuras.

Los sujetos representan el tema del que estamos hablando y se definen como:

"Un sujeto es el mecanismo para guiar al lector (analista, experto en el dominio, administrador, cliente) a través de un modelo largo y complejo. Los sujetos son también útiles para organizar paquetes de trabajo en proyectos grandes,

basados en análisis orientado a objetos"[5]

Los sujetos pueden representarse de manera colapsada o extendida, la colapsada se representa por un rectángulo que tiene un nombre y un número exclusivo que lo identifica, la extendida se representa con un cuadro que abarque las clases-objetos que le correspondan y el número de identificación en cada una de las esquinas del cuadro[6]. En la figura 3 se ilustra el sujeto que se identificó para el análisis

Figura 3

Este sujeto agrupa a las clases-objetos que se trataron en el inciso anterior.

Algunas de estas clases-objetos se encuentran relacionados entre sí a través de estructuras que reflejan además del dominio del sistema, sus responsabilidades.

5 Ob.cit., cap 5, p. 106.

6 Ob. cit. p.110-112.

Coad y Yourdon definen estructura de la siguiente manera:

> "Estructura: La estructura es una expresión de la complejidad del dominio del problema que refleja las responsabilidades del sistema. El término 'estructura' es utilizado como término general que describe la forma generalización-especialización y la forma total-partes."[7]

La estructura de generalización-especialización puede interpretarse como 'es un tipo de' como "pino" es un tipo de árbol, y gráficamente se representa por un semicírculo entre las clases y las líneas de conexión van directamente de clase a clase. La estructura de total-partes ejemplifica una relación de un elemento y el total que integra, puede interpretarse como 'es parte de' como una manecilla es parte de un reloj; gráficamente se representa por un triángulo, las líneas de conexión van de objeto a objeto y en sus extremos se expresa la cantidad de elementos, mínimo y máximo que integran la relación.

La estructura que sustentan las clases-objetos aplicado a nuestro dominio es la que se presenta en la figura 4; la caja identificada con el número 1 representa la explosión del sujeto organigrama.

---

[7] Ob. cit., cap 4, p 79.

Figura 4

La estructura se interpreta como:

- Dependencia es un conjunto de por lo menos una plaza y cuan-do más de muchas plazas.
- Una plaza es parte de una dependencia.
- Dependencia es un conjunto de por lo menos un trabajador y

cuando más de muchos trabajadores.

- Trabajador es parte de una dependencia y cuando más de dos dependencias.

III.3. Identificación de atributos y servicios.

Cada uno de los objetos contiene datos, es decir información de la cual es responsable. Estos datos se denominan atributos y su definición es:

> "Un atributo es algún dato (estado de información) para el cual cada objeto en una clase tiene su propio valor."[8]

Los atributos sólo pueden ser manipulados por los servicios de su mismo objeto, si otra clase-objeto del sistema necesita accesarlo o manipular su valor deberá hacerlo a través de mensajes.

La definición de servicio y mensaje que hacen Coad y Yourdon en su libro de análisis es:

> "Un servicio es un comportamiento específico que un objeto es responsable de exhibir."[9]

> "Un mensaje de conexión modela la dependencia del proceso de un objeto, indicando una necesidad de servicio con el objeto de completar sus

---

8 Ob. cit., cap 6, p. 119.

9 Ob. cit., cap. 7, p. 143.

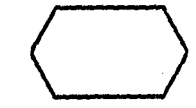
FALTA PAGINA

No. 24

1. Cuando se efectúa un movimiento de modificación al atributo RFC, plaza le asigna fecha de ingreso y envía la fecha al objeto trabajador para su registro.
2. Para crearse un nuevo objeto de plaza, ésta determina el sueldo que le corresponde y envía a dependencia su valor para que solicite presupuesto.
3. La dependencia eleva el valor del sueldo por el resto del año y solicita presupuesto a presupuesto.
4. Cuando efectúa movimiento de modificación a los atributos categoría o RFC, plaza solicita a dependencia que afecte su presupuesto.

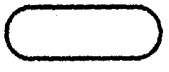
III.3.1. Definición de atributos y servicios:

Los atributos de cada una de las clases así como sus servicios se definen para enfatizar la necesidad de cada uno de ellos en el análisis. Para clarificar el estado de cada uno de los servicios se utiliza la simbología que Coad y Yourdon utilizan en su libro de AOO, siendo ésta la siguiente:



Condicion (if, precondicion, triggger, terminar)

Caja de texto



Loop (while, do, repeat, trigger/terminar)

Linea de conexion

Figura 6

Clase-objeto Dependencia:	
Atributo	Definición
Clave_dependencia	3 dígitos que identifican a cada dependencia de acuerdo al catálogo presupuestal de la UNAM[11]
Clave-subdepen	2 dígitos que identifican a cada una de las subdependencias de cada dependencia, conforme al catálogo presupuestal de la UNAM.

[11] UNAM. Instructivo de Ejercicio y Catálogo Presupuestal. UNAM. Edición anual.

Cod_Program	4 dígitos, 1 para la función, otro para el programa y los dos restantes para el subprograma que debe cubrir la dependencia.
Monto_presup_prop	Cantidad monetaria que se asigna a cada dependencia y que puede manejaria a su libre albedrío. (En este caso se refiere exclusivamente a la parte que corresponde a sueldos)

Servicio Afecta_pres_prop: incrementa o decrementa la cantidad monetaria asignada a la dependencia.

* Con base en el mensaje que recibe afecta el presupuesto de la dependencia de que se trate.

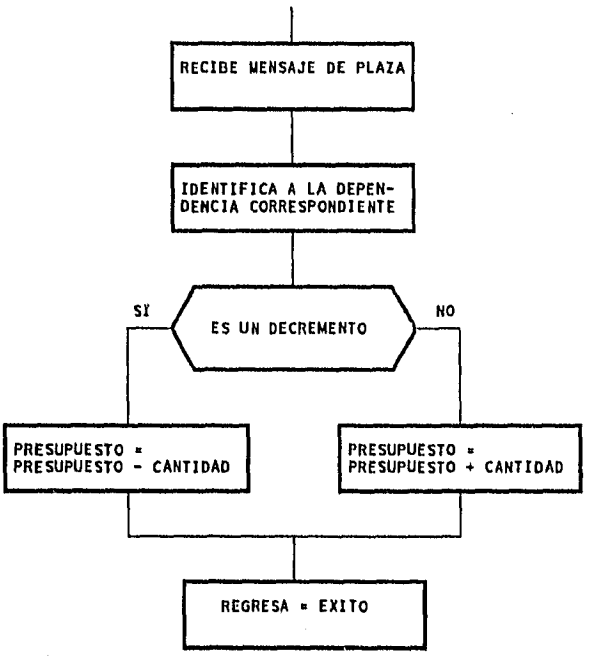


Figura 7

Servicio Solicita_presup: envía un mensaje a la clase-objeto Presupuesto.

*Recibe en un mensaje el nuevo valor requerido por una plaza.

*Eleva el valor por el número de meses restantes para finalizar el año.

*Envía un mensaje a la clase-objeto Presupuesto con el total del monto requerido.

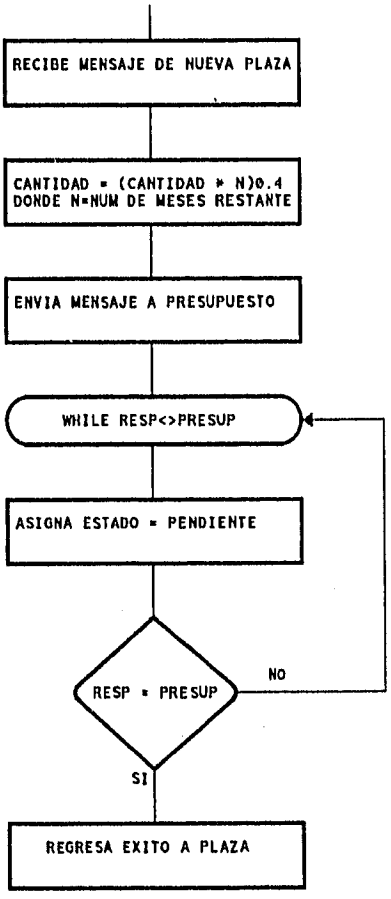

RECIBE MENSAJE DE NUEVA PLAZA
CANTIDAD = (CANTIDAD * N)0.4
DONDE N=NUM DE MESES RESTANTE
ENVIA MENSAJE A PRESUPUESTO
WHILE RESP<>PRESUP
ASIGNA ESTADO = PENDIENTE
RESP = PRESUP
NO
SI
REGRESA EXITO A PLAZA

Figura 8

Clase-objeto Plaza:	
Atributo	Definición
Numero	5 dígitos que identifican a cada plaza.
Categoría	4 dígitos; 2 caracteres alfabéticos que identifican el género de la categoría que sustenta cada plaza, conforme al catálogo presupuestal de la UNAM y 2 dígitos que identifican el nombre de la categoría dependiendo del género.
Sueldo	cantidad monetaria asignada a cada plaza mensualmente dependiendo de su categoría y nivel.
Estatus	indica si la plaza se encuentra ocupada o vacante.
Fecha_ocupacion	día, mes y año en el que una plaza es ocupada por un Rfc.

Numero_madre	5 dígitos para el número que identifica a la plaza de la cual depende.
Rfc	13 dígitos para el Registro Federal de Contribuyentes que asigna SHCP a los trabajadores.
Fecha_termino	día, mes y año en que debe desaparecer una plaza.

Servicio Determina_sueldo:

* Con base en los primeros 2 caracteres de la categoría que sustenta la plaza accesa al tabulador que le corresponda según su género.
* Con los siguientes 2 caracteres de la categoría ubica el renglón del tabulador que corresponda y obtiene el sueldo.

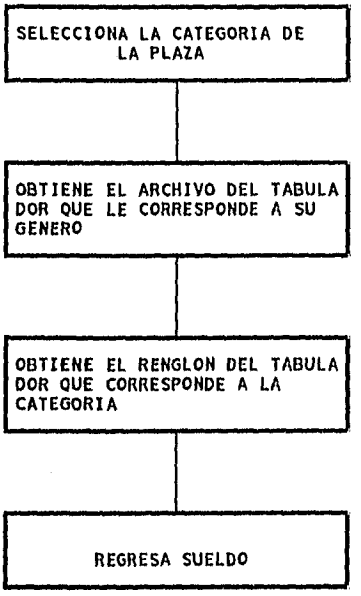


Figura 9

Servicio Determina_madre: establece el nodo (o la plaza) de la cual depende.

* Verifica el valor del atributo Numero_madre.
* Recorre las plazas hasta encontrar aquella que coincida con el número y genera la liga.

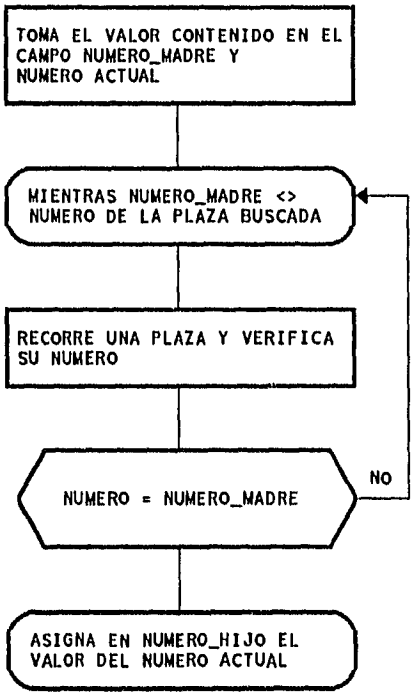


Figura 10

Servicio Crea, Modifica y Elimina, estos servicios se encuentran encapsulados en todos las clases de nuestro modelo y contienen especificaciones para crear un nuevo objeto dentro de la misma clase, modificarlo o eliminarlo.

Clase-objeto Trabajador:	
Atributo	Definición
Nombre	Apellido paterno, materno y nombre(s) del trabajador.
Rfc	Registro Federal de Contribuyentes que asigna la S.H.C.P.
Grado_estudios	Nombre del último grado de estudios comprobable con documentos del trabajador.
Antigüedad	Tiempo en años, meses y días que el trabajador ha laborado en una plaza desde su fecha de ingreso.

Servicio Registra_f_ingr:

* Con base en el mensaje del objeto Plaza, registra la fecha del sistema en la que el atributo Rfc obtuvo un valor diferente de 0.

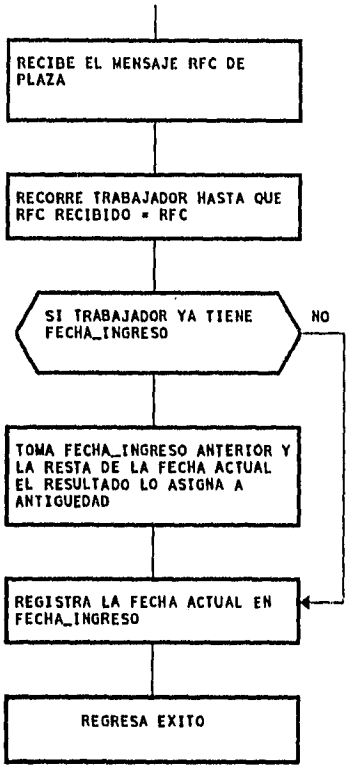

RECIBE EL MENSAJE RFC DE PLAZA
RECORRE TRABAJADOR HASTA QUE RFC RECIBIDO = RFC
SI TRABAJADOR YA TIENE FECHA_INGRESO
NO
TOMA FECHA_INGRESO ANTERIOR Y LA RESTA DE LA FECHA ACTUAL EL RESULTADO LO ASIGNA A ANTIGUEDAD
REGISTRA LA FECHA ACTUAL EN FECHA_INGRESO
REGRESA EXITO

Figura 11

Clase-objeto Presupuesto:	
Atributo	Definición
Monto_pres_afect	Cantidad monetaria que del presupuesto se ha erogado o comprometido.
Monto_pres_disp	Cantidad monetaria que queda disponible durante el resto del año y puede utilizar.
Clave_dependencia	Clave que identifica al objeto dependencia al que pertenece cada presupuesto.(en este caso son 3: División de estudios profesionales, división de estudios de posgrado y división de esducación contínua)

Serviclo Afecta_presup: afecta la cantidad del atributo Monto_pres_disp.

* Con base en el mensaje que recibe del objeto Dependencia identifica el presupuesto correspondiente y lo disminuye o incrementa en la cantidad especificada.

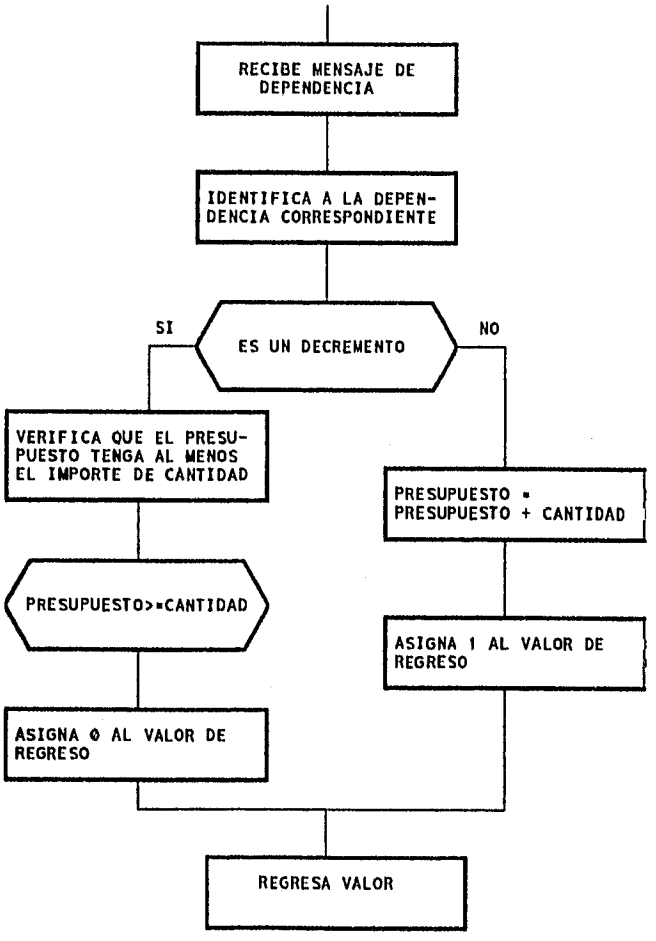


Figura 12

IV. DISEÑO

Con base en los resultados del capítulo anterior, en el cual se han desglosado los componentes del dominio del problema, lo que corresponde ahora es añadir los detalles para la implementación del sistema.

De acuerdo a la metodología de Diseño Orientado a Objetos (DOO) que se aplica en el presente trabajo, la fase de diseño abarca cuatro elementos:[1]

1.- Dominio del Problema.

2.- Interacción Humana.

3.- Administración de tareas.

4.- Administración de datos.

IV.1. Elementos del diseño.

El estudio sobre el problema y la separación de las partes para identificar su naturaleza, funciones y relaciones llevada a cabo en el

[1] COAD Peter, YOURDON Edward. "OBJECT-ORIENTED DESIGN". New Jersey. Prentice-Hall, Inc. 1991.

capítulo anterior, constituye los componentes del dominio del problema (CDP), que se convierte en el primer elemento del diseño, el cual podrá ser susceptible de modificaciones conforme se avance en el diseño y se detecten servicios o atributos necesarios para su resolución.

En el segundo elemento que corresponde a los componentes de interacción humana (CIH), se identifica a los usuarios del sistema, la manera en la que el sistema presentará su información y cómo el usuario comandará al mismo. Este elemento se convierte en el sujeto número 2.

El tercer elemento se refiere a los componentes del manejador de tareas(CMT), el cual al ser un sistema desarrollado para trabajar en ambiente windows se le dajará a éste el manejo de tareas.

El cuarto y último elemento se refiere al componente del manejo de datos(CMD), el cual se explicará en base al manejador Transac-SQL seleccionado para implementar el sistema.

IV.2. Componentes del Dominio del Problema (CDP).

El producto resultante de la fase de análisis constituye la base para

el diseño de este componente, en el cual se aplican correcciones o mejoras a los modelos creados. En este caso, dado el tiempo que se dedicó al análisis[2] y revisiones minuciosas, los errores detectados se aplicaron en los modelos durante el análisis, por tanto, para esta fase no se aplican adiciones o mejoras, así los componentes del dominio del problema se encuentran en el sujeto 1 y se muestran en la figura 13.

IV.2. Componentes de Interacción Humana (CIH).

Estos componentes se obtienen a través de identificar y clasificar a los usuarios, identificar las tareas que desarrollarán y las herramientas que utilizan, resultando que se cuenta con tres usuarios potenciales: Titular de la dependencia, Secretario Administrativo y Jefe de Personal.

[2] *infra.* Véase capítulo 6 Control del proyecto, en esta misma tesis..

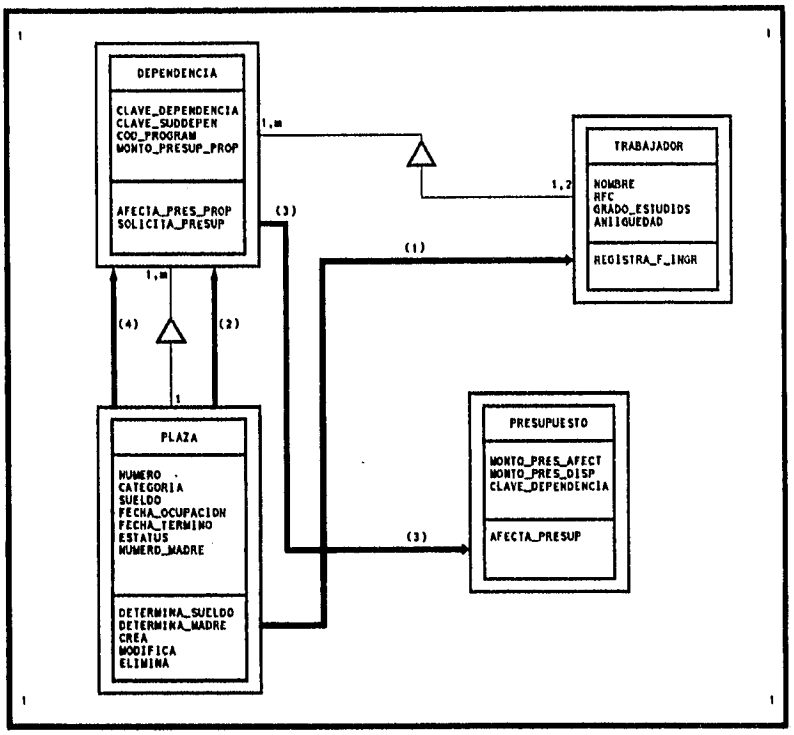


Figura 13

IV.3.1. Descripción de los componentes de interacción humana.

Persona: Titular de la Dependencia.

Nivel: Alto ejecutivo.

Propósito general de los titulares en cuanto a los organigramas: Desea ver la estructura que guardan las plazas de su dependencia para proponer cambios cuando se requieran.

Características en general de los titulares:

Edad: Entre 40 y 60 años.

Estudios: Maestría o Licenciatura.

Limitaciones: Tiempo.

Factores críticos:

Sólo desea ver ya sea en la pantalla o en un reporte rama por rama del árbol de su organigrama, el número y categoría de la plaza, así como el nombre del trabajador que la ocupa.

Las consultas las haría accesando el nombre de la persona que ocupa la plaza que sería cabezal de rama, es decir a partir del Subdirector, Coordinador o Jefe de departamento por ejemplo.

Nivel de uso computacional: Ocasional.

Tareas de utilidad:

Sus subordinados le presentan organigramas actuales y propuestos y el anállsis de las necesidades de movimientos a las plazas.

Evalúa los resultados del análisis a las necesidades y decide que plazas deben moverse, reclasificarse o desaparecer, decisión que transmite a los subordinados responsables.

Recibe los movimientos de los trabajadores a través de contratos (específicamente en la UNAM se denominan Formas Únicas)

Controla los recursos monetarios de la dependencia destinado al pago de plazas.

Persona: Secretario Administrativo.

Nivel: Ejecutivo medio.

Propósito general del Secretario respecto al organigrama:

Supervisar y en su caso realizar los movimientos requeridos a las plazas tales como cambios en el RFC que las ocupa, reclasificarlas o eliminarlas.

Características generales de los Secretarios:

Edad: Entre 25 a 50 años.

Estudios: Licenciatura.

Limitaciones:

Factores críticos:

Requiere ver mas de una ramificación del organigrama de la dependencia para mover una plaza de un lado a otro.

Desea ver todas las características de las plazas.

Nivel de uso computacional: Intermedio

Tareas de utilidad:

Identifica las plazas que sufrirán modificación dictada por el Titular de la dependencia.

Recibe bajas de trabajadores, a los cuales deberá eliminar de la plaza que ocupan.

Imprime y entrega el Organigrama requerido por el Titular (total o parcial).

Persona: Jefe de Personal.

Nivel: Administrativo.

Propósito: Obtener rápidamente la información de las plazas vacantes de una dependencia para contratar a personal.

Características:

Edad: Entre 30 y 50 años

Estudios: Licenciatura.

Limitaciones:

Factores críticos:

Sólo desea ver las plazas vacantes de la categoría y dependencia

que elija, y ver que el nombre y RFC de la persona que contrate aparezca en la plaza que haya seleccionado.

Nivel de uso computacional: Intermedio.

Tareas de utilidad:

Verifica qué plazas se encuentran vacantes para contratar personal.

Imprime listados de las contrataciones por periodos.

IV. 3.2. Comandos y sus jerarquías.

Los comandos para Interactuar con el sistema y sus jerarquías se presentan a los usuarios de diversas maneras, pudiendo ser a través de pantallas de menús, una barra con el menú o como una serie de íconos que se accionan al ser seleccionados, mientras más claros y amigables sean estos comandos permitirá una mayor integración del usuario al sistema, para el presente sistema se ha selecionado la barra de menú.

Sesion	Modificar	Consulta	Reportes	Ayuda

Figura 14

Para refinar la jerarquía de los comandos se considera la frecuencia de uso y la eficiencia de la amplitud contra la profundidad.

La barra de menú como se aprecia en la figura 14 constituye el menú principal y con los submenús correspondientes se aprecian en la siguiente figura:

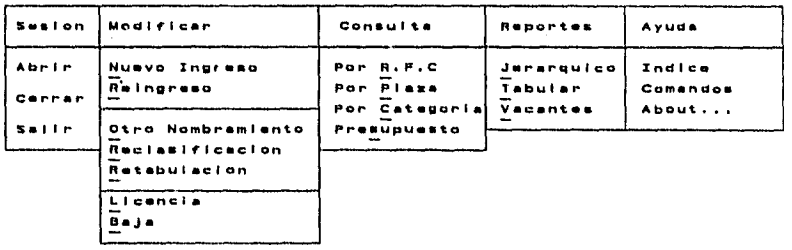


Figura 15

Las clases de los componentes de interacción humana quedan definidas en la figura 16, en la cual se muestra de manera colapsada al sujeto 1.

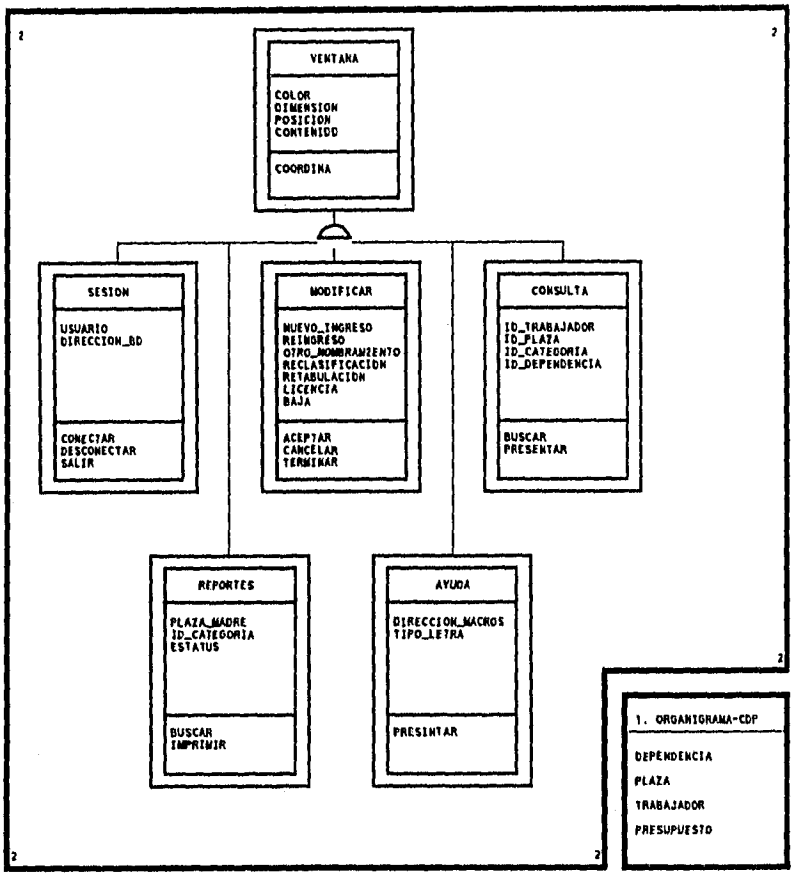


Figura 16

IV.3.3. Prototipos de pantallas.

Un factor importante señalado por Coad y Yourdon para lograr la

integración humana además de establecer el diseño de comandos y sus jerarquías es el de presentar a los usuarios un prototipo de interacción de manera que puedan expresar sus opiniones o refinar algunos detalles del modelo propuesto para su completa satisfacción.

A continuación se presenta un prototipo de las pantallas que contendrá el sistema que se desarrolle.

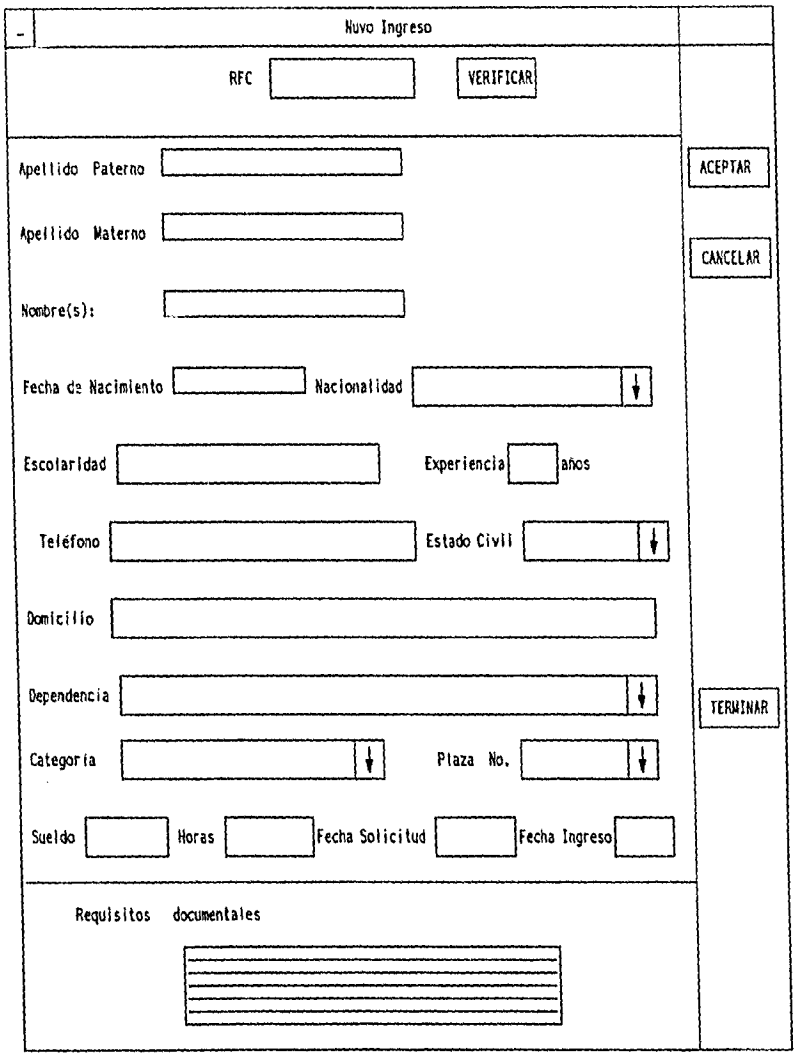

Nuvo Ingreso
RFC
VERIFICAR
Apellido Paterno
Apellido Materno
Nombre(s):
Fecha de Nacimiento
Nacionalidad
Escolaridad
Experiencia
años
Teléfono
Estado Civil
Domicilio
Dependencia
Categoría
Plaza No.
Sueldo
Horas
Fecha Solicitud
Fecha Ingreso
Requisitos documentales
ACEPTAR
CANCELAR
TERMINAR

Figura 17

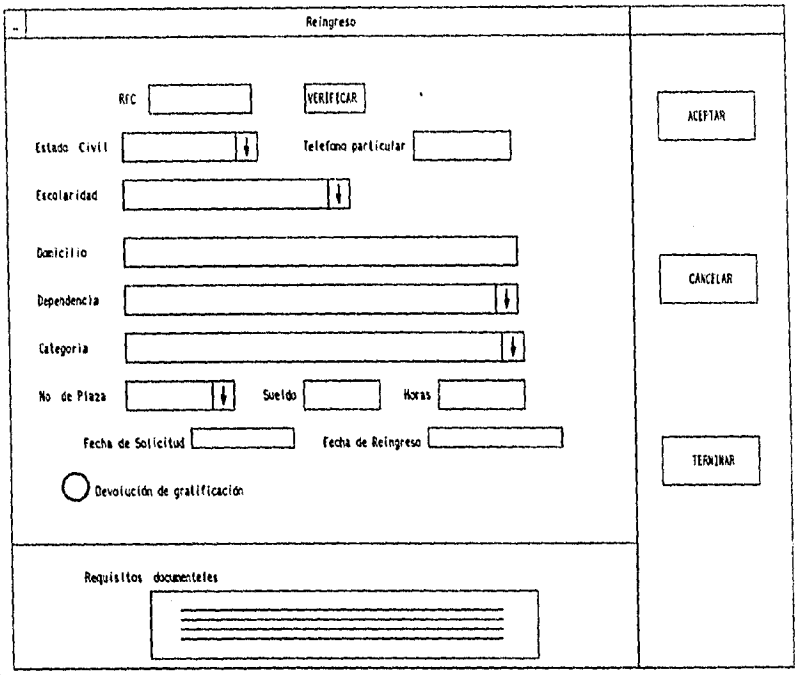

Reingreso
RFC
VERIFICAR
ACEPTAR
Estado Civil
Telefono particular
Escolaridad
Domicilio
CANCELAR
Dependencia
Categoria
No de Plaza
Sueldo
Horas
Fecha de Solicitud
Fecha de Reingreso
TERMINAR
Devolución de gratificación
Requisitos documentales

Figura 18

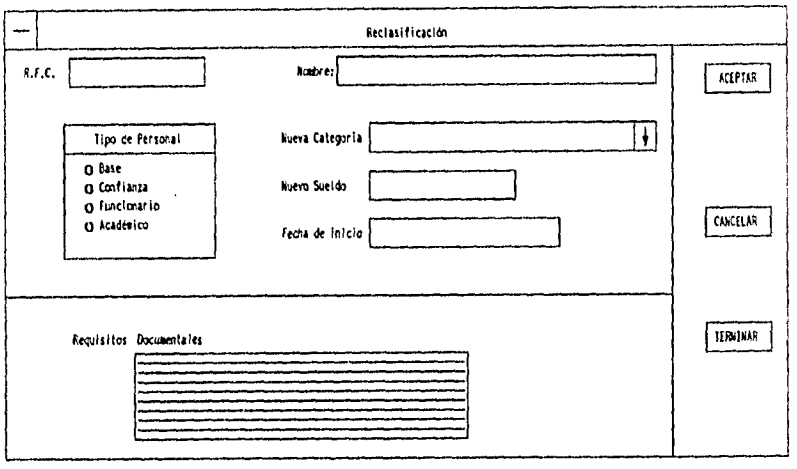


Figura 19

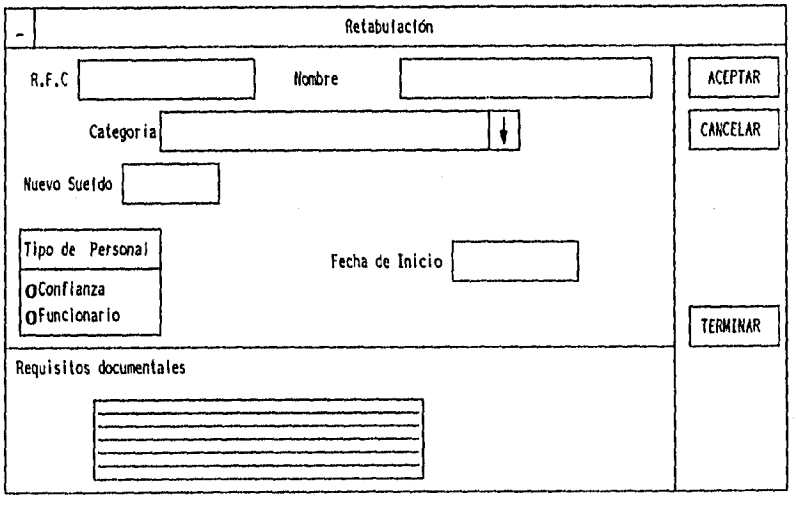


Figura 20

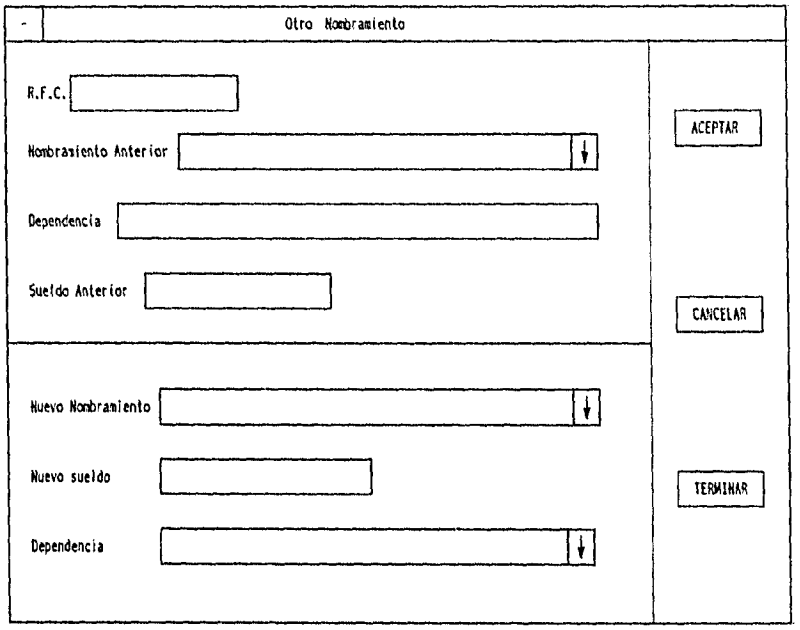


Figura 21

IV.4 Componente del manejador de tareas (CMT).

Para determinar estos componentes, es necesario primero especificar lo que se entiende por tarea, por lo que para tal efecto se ha tomado la misma definición que hace Coad y Yourdon en su libro de diseño al respecto, y posteriomente se ha seguido la estrategia que plantean para identificar cada una de las tareas requeridas por el sistema:

"Tarea: Otro nombre para un proceso (una cadena de actividades, definidas por su código). La ejecución concurrente de varias tareas es referenciado como multitarea."[3]

IV.4.1. Estrategia para identificar tareas.

La estrategia utilizada por Coad y Yourdon para identificar y diseñar las tareas aunadas a los servicios incuídos en cada una de ellas, se construyó a partir de las ideas de otro autor (Gomaa, Hassan), a quien hacen referencia en su libro de diseño[4], misma que se ha tomado para identificar las tareas de éste sistema y consta de lo siguiente:

- Identificar manejadores de eventos.
- Identificar manejador por tiempo de tareas
- Identificar tareas prioritarias y críticas.
- Identificar al coordinador de tareas.
- Medir cada tarea.
- Definir cada tarea.

---

[3] COAD Peter, YOURDON Edward. Ob. cit. cap 5. p. 72.

[4] [Gomaa, 1989] Gomaa, Hassan. "Structuring Criteria for Real-Time System Design", *ICSE'89 Proceedings*. IEEE/ACM, 1989.

El diagrama que se presenta en la figura 22 representa los componentes de manejador de tareas (CMT) para el sistema, los sujetos 1 y 2 aparecen de forma colapsada.

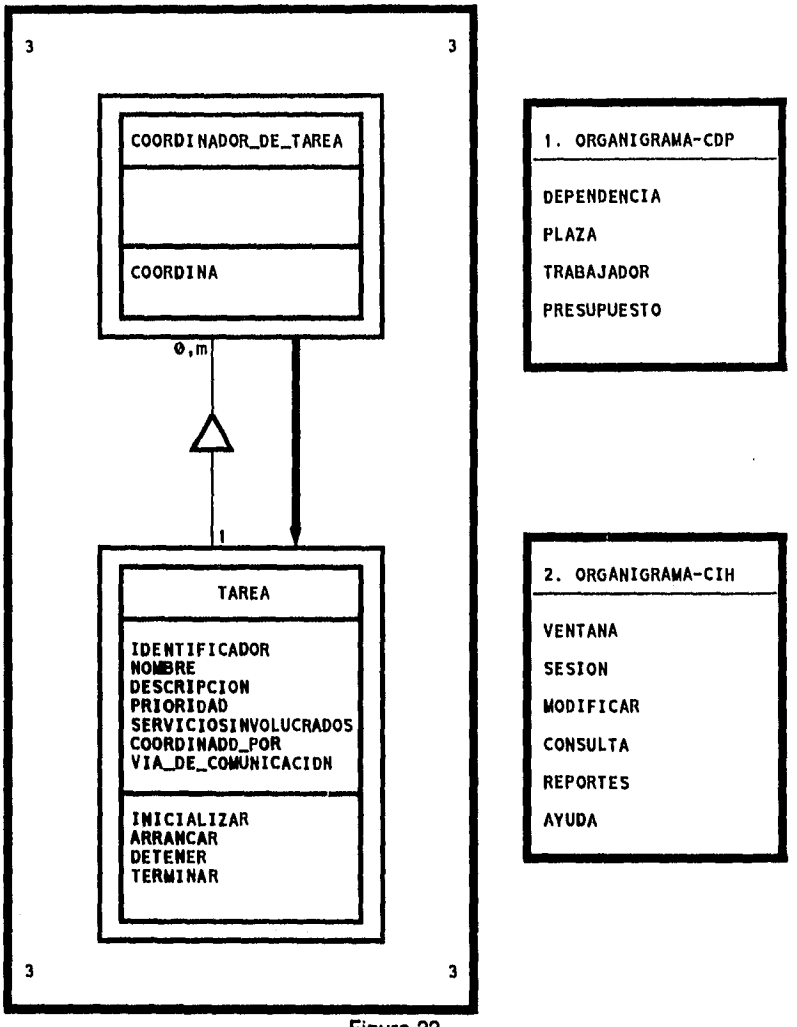

3
3
COORDINADOR_DE_TAREA
COORDINA
0,m
1
TAREA
IDENTIFICADOR
NOMBRE
DESCRIPCION
PRIORIDAD
SERVICIOSINVOLUCRADOS
COORDINADO_POR
VIA_DE_COMUNICACION
INICIALIZAR
ARRANCAR
DETENER
TERMINAR
3
3
1. ORGANIGRAMA-CDP
DEPENDENCIA
PLAZA
TRABAJADOR
PRESUPUESTO
2. ORGANIGRAMA-CIH
VENTANA
SESION
MODIFICAR
CONSULTA
REPORTES
AYUDA

Figura 22

Descripción de las tareas:

Tarea 1: AfectarPresupuesto

Descripción: Esta tarea es responsable de incrementar o decrementar el monto de los presupuestos propios o externos

Prioridad: media

Servicios involucrados:

Dependencia.Afecta_pres_prop

Presupuesto.Afecta_presup

Coordinado por: manejador de eventos

(1) Modificación de atributos de objetos plaza.

(2) Creación de un nuevo objeto plaza.

Vía de comunicación : obtiene los valores del mensaje que envían los objetos Plaza o Dependencia.

Tarea 2: DeterminarSueldo

Descripción: Esta tarea es responsable de determinar en base a la los atributos de la plaza su importe mensual.

Servicios involucrados: Plaza.Determina_sueldo

Prioridad: media

Coordinado por: manejador de eventos.

(1) Modificación de los atributos Categoría, Nivel o RFC del objeto plaza.

Vía de comunicación: obtiene sus valores de la base de datos.

Tarea 3: CoordinadorDeInteracciones

Descripcion: Esta tarea es responsable de la interacción humana, actualizar datos y determinar posiciones.

Servicios involucrados:

ServidorDeObjetos.Actualiza

Cursor.Determina_posicion

Trabajador.Registra_f_ingr

Prioridad: alta

Coordinado por: Manejador de eventos

(1) Interacción humana.

(2) Envío de mensajes de un objeto.

Vía de comunicación: Obtiene sus valores de la interacción con el usuario o del mensaje de las clases correspondientes.

Tarea 4: OrganizarNodos

Descripción: Esta tarea es responsable de determinar la jerarquía y la rama que le corresponde a cada elemento del organigrama.

Servicios involucrados:

Plaza.Determina_madre

Cursor.Posicion

Prioridad: Alta

Coordinado por: manejador de eventos

(1) Interacción humana.

(2) Modificación de atributos del objeto Plaza.

Vía de comunicación: Obtiene sus valores de datos accesados por el usuario a través de periféricos.

El sistema será desarrollado bajo ambiente Windows, de manera tal que la coordinación de tareas será realizada por Windows en su forma nativa a través de "Handles" o manejadores de eventos.

Windows es un sistema orientado a mensajes, por lo que los mensajes generados por los objetos que requieran ser despachados Windows los direcciona a la función de ventanas correspondiente la cual se determina mediante las estructuras generadas al registrar las clases y crear las ventanas asociadas a éstas; el direccionamiento de los mensajes se realiza utilizando las funciones de envío y disparo correspondientes.

IV.5. Componentes del administrador de datos (CAD).

Este componente para administrar los datos, tal como hacen mención en el libro de diseño de Coad y Yourdon, "...proporciona la infraestructura para el almacenamiento y recuperación de objetos desde el administrador de datos del sistema."[5]

IV.5.1. Especificación del administrador de datos.

Existen diversas formas para administrar los datos, sin embargo las que han tenido mayor auge han sido las que se refieren a archivos

planos, las relacionales y actualmente las orientadas a objetos. De ésta última Coad y Yourdon puntualizan[6] que los productos comerciales para estos manejadores, tienen dos vertientes, la que parece un manejador relacional extendido, y el que es una extensión del lenguaje de programación orientado a objetos. Por lo que respecta al presente sistema, se ha elegido como administrador uno meramente relacional.

La base de datos será implementada en SYBASE y se utilizará el lenguaje Transac-SQL para realizar las búsquedas (obtener datos de la base de datos), crear nuevos objetos dentro de la misma base, añadir nuevos datos o modificar los ya existentes.

IV.5.2. Definición de tablas en tercera forma normal.

El proceso de normalización consiste en agrupar a los campos de datos en tablas que representan a los objetos y sus relaciones. Este proceso permite asegurar que el modelo conceptual de la base de datos funcionará, lo cual no significa que un modelo no normalizado

---

5 COAD Peter, YOURDON Edward. Ob. cit. cap. 6, p. 80.

6 COAD Peter, YOURDON Edward. Ob.cit. cap. 6, p. 82.

no funciona pero si puede causar problemas cuando se intente aplicar modificaciones a la base de datos.

El proceso de normalización de una base de datos consiste de varios niveles, para efectos de aplicarlo a la presente tesis se tomó el procedimiento del autor Shakuntala Atre[7], siendo éste el siguiente:

"Un modelo de datos no normalizado consiste en el conjunto de registros utilizados por los programas de aplicación. El primer paso de la normalización consiste en eliminar las ocurrencias repetidas de campos de datos obteniendo una tabla de dos dimensiones. Un archivo de este tipo se encuentra en primera forma normalizada (1FN). En el segundo paso se identifican y establecen los campos llave y se relaciona esta llave con los demás campos de datos que le correspondan.

La división de la tabla an 1FN en una serie de tablas en las que cada campo sólo depende de la clave completa se llama segunda forma normalizada (2FN).

El tercer paso consiste en separar los campos de las segundas relaciones normales de manera que tengan una existencia independiente en la base de datos, es decir que no exista una dependencia transitiva entre los atributos que no son clave. Las tablas resultantes se encuentran en tercera forma normalizada (3FN)."

A continuación se presenta las tablas de la base de datos:

- Listando cada clase y sus atributos.
- Poniendo esa lista en una definición de tablas en tercera forma normal.

---

[7] ATRE, Shakuntala, "TECNICAS DE BASES DE DATOS. Estructuración en diseño y administración." Trillas 1988. cap. 5. p.138-157.

normal.

Después se define un esquema de la base de datos para las tablas en tercera forma normal.

CDependencias			
CAMPO	TIPO	LONGITUD	VALIDACION
cDependencia	Character	3	111-911
cSubDep	Character	2	01-99
vcNombreDep	VarChar	100	
fPresupuesto	Float	8	

CPlazas			
CAMPO	TIPO	LONGITUD	VALIDACION
cNumPlaza	Character	6	
cCategoria	Character	4	AA99
cPartidaPlaza	Character	3	111-799
cEstatusPlaza	Character	10	'Ocupada'/ 'Vacante'
cDependencia	Character	3	111-911
cSubDep	Character	2	01-99
cPlazaMadre	Character	6	
fSueldoPlaza	Float	8	
cHorasPlaza	Character	4	

CCategorias			
CAMPO	TIPO	LONGITUD	VALIDACION
cCategoria	Character	4	
vcDescCatego	VarChar	50	
fSueldoCatego	Float	8	
cHoras	Character	8	

CEscolaridad			
CAMPO	TIPO	LONGITUD	VALIDACION
cClaveEscol	Character	2	01-99
vcDescEscol	VarChar	100	

CNacionalidad			
CAMPO	TIPO	LONGITUD	VALIDACION
cNacionalidad	Character	1	
vcDescNacional	VarChar	50	

CTrabajadores			
CAMPO	TIPO	LONGITUD	VALIDACION
cRfcTrab	Character	13	
vcNombreTrab	VarChar	90	
dNacimientoTrab	Date	8	dd/mm/aa
cNacionalidad	Character	1	
cClaveEscol	Character	1	
cEstadoCivil	character	1	
vcDomicilio	VarChar	100	
cEstatus	Character	10	'Vigente' / 'No Vigente'

CEstadosCiviles			
CAMPO	TIPO	LONGITUD	VALIDACION
cEstadoCivil	Character	1	1-5
vcDescEdoCivil	VarChar	15	

THistoria			
CAMPO	TIPO	LONGITUD	VALIDACION
iConsecutivoHist	Int	4	
cRfcTrab	Character	13	
cDepHist	Character	3	413
cSubDepHist	character	2	1-3
cNumPlaza	character	6	
cCategoHist	Character	4	
dInicioHist	Date	8	dd/mm/aa
dBajaHist	Date	8	dd/mm/aa

El esquema de la base de datos queda de la siguiente manera:

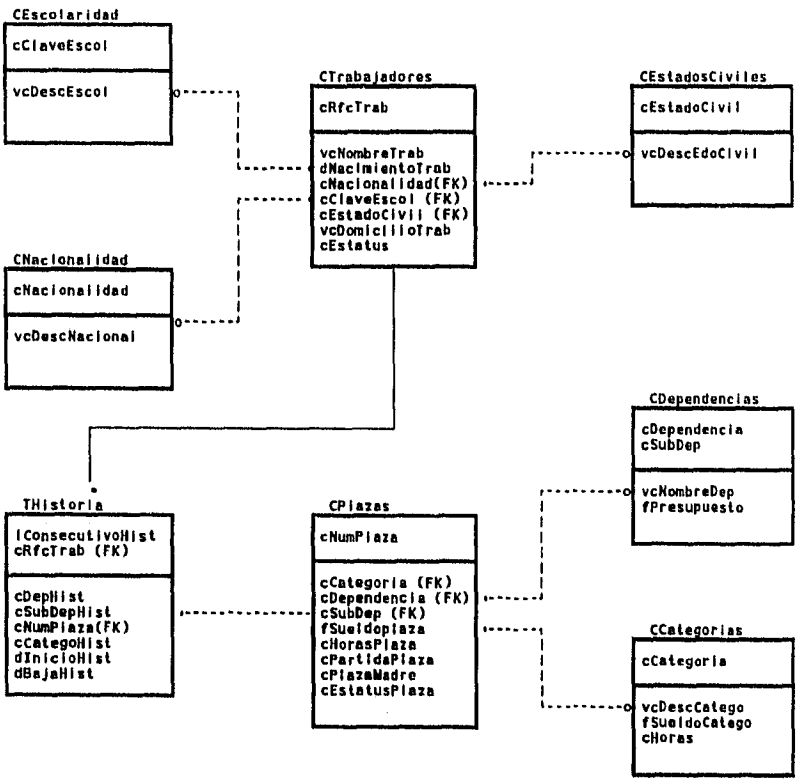


Figura 23

La clase objeto resultante, que corresponde al sujeto número 4 es el que se ilustra en la figura 24.

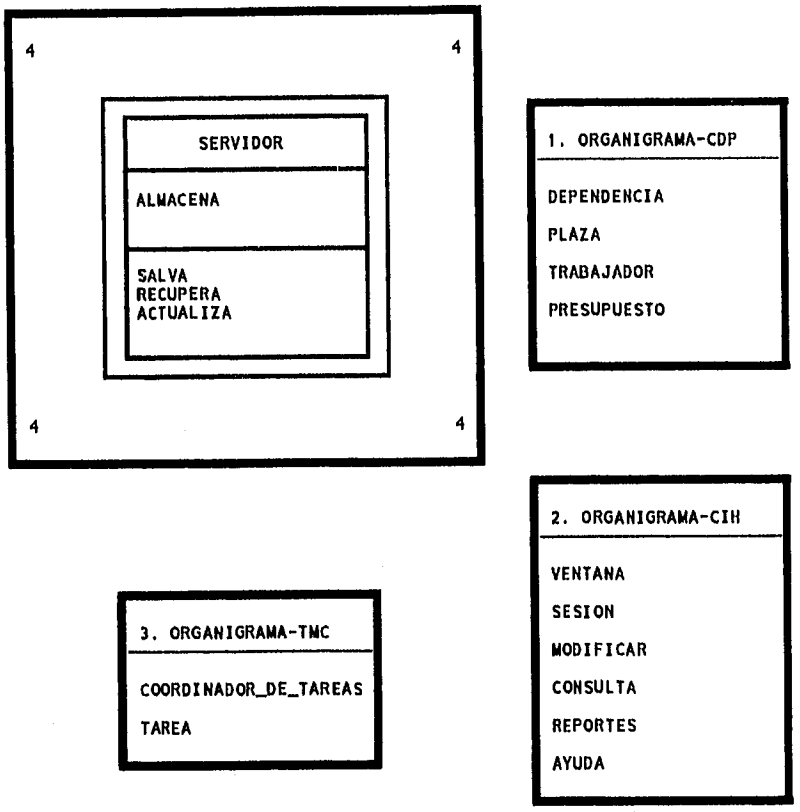

4
4
SERVIDOR
ALMACENA
SALVA
RECUPERA
ACTUALIZA
4
4
1. ORGANIGRAMA-CDP
DEPENDENCIA
PLAZA
TRABAJADOR
PRESUPUESTO
2. ORGANIGRAMA-CIH
VENTANA
SESION
MODIFICAR
CONSULTA
REPORTES
AYUDA
3. ORGANIGRAMA-TMC
COORDINADOR_DE_TAREAS
TAREA

Figura 24

V. CONTROL DEL PROYECTO

Para realizar cualquier tipo de trabajo es necesario controlar los aspectos de tiempo, actividades y recursos con el fin de llegar a su conclusión, para ello una vez definido el tema u objetivo a lograr, se puede comenzar por calendarizar las actividades o formular un plan de trabajo en el cual se establezca una relación de tiempo, eventos y realización que comprenda idealmente desde la concepción hasta las conclusiones del trabajo; es recomendable estipular límites de tiempo por cada etapa del mismo, lo cual permite concluir una secuencia ordenada de pasos establecidos y comparar día a día o semana a semana lo planteado con lo ejecutado.

> "El plan de trabajo es un compromiso escrito, susceptible de cumplirse en el tiempo determinado y así someterse a comparación con lo realizado. Operativamente representa una hoja de control y evaluación de acciones planeadas."[1]

El tiempo para el desarrollo de una tesis varia de acuerdo a la complejidad del tema y al dominio que sobre al metodología y

---

1 El Método Científico en la Investigación. Dr. Luis Vázquez Camacho. Editorial. Nueva Era 1990

herramientas requiera el caso. Al respecto encontramos autores que dicen:

"Digamos de entrada: no mas de tres años y no menos de seis meses. No mas de tres años porque si en tres años de trabajo no se ha logrado limitar el tema y encontrar la documentación necesaria, eso sólo puede significar tres cosas:

1) Ha elegido una tesis equivocada superioir a sus fuerzas.
2) Pertenece al tipo de eternos descontentos que querrían decirlo todo y sigue trabajando en la tesis durante veinte años, cuando en realidad un estudioso hábil tiene que ser capaz de fijarse unos límites, aunque modestos y producir algo definitivo dentro de estos límites.
3) Se le ha declarado la neurosos de la tesis; la deja de lado, la vuelve a coger, no se siente realizado, llega a un estado de gran dispersión, utiliza la tesis como excusa para muchas bajezas; éste no se doctora nunce.

No menos de seis meses; pues aunque queráis hacer el equivalente de un buen artículo de revista que no pase de los sesenta folios, entre estudiar el planteamiento del trabajo, buscar la bibliografía, ordenar los documentos y redacter el texto, seis meses pasan en un abrir y cerrar de ojos."[2]

A continuación se presenta el plan de trabajo expresado en gráficas

[2] Cómo se hace una tesis. Técnicas y procedimientos de investigación, estudio y escritura. Umberto Eco. Versión castellana de Lucia Baranda y Alberto Clavería Ibáñez. Gedisa editorial. Impreso en España. p37-38.

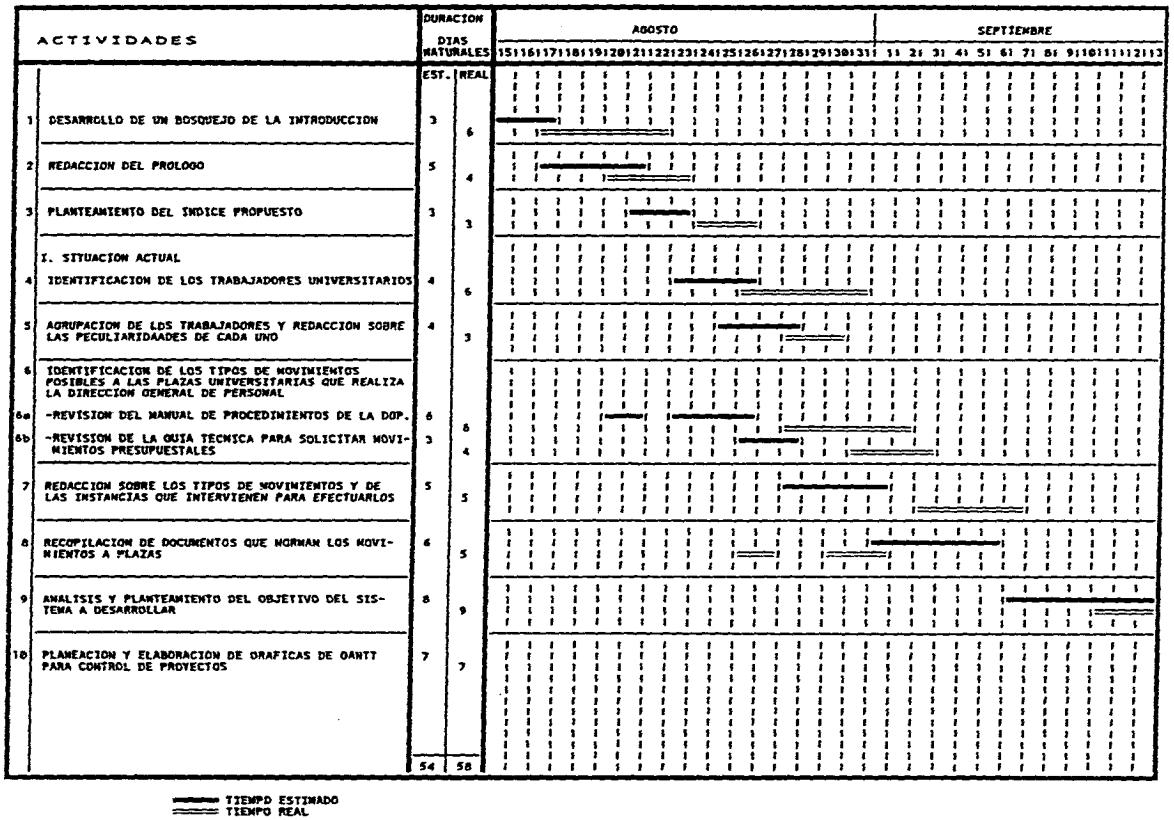
de Gantt que se siguió para el presente trabajo, en el cual se plantea el tiempo estimado con una línea gruesa y el tiempo real con línea doble.

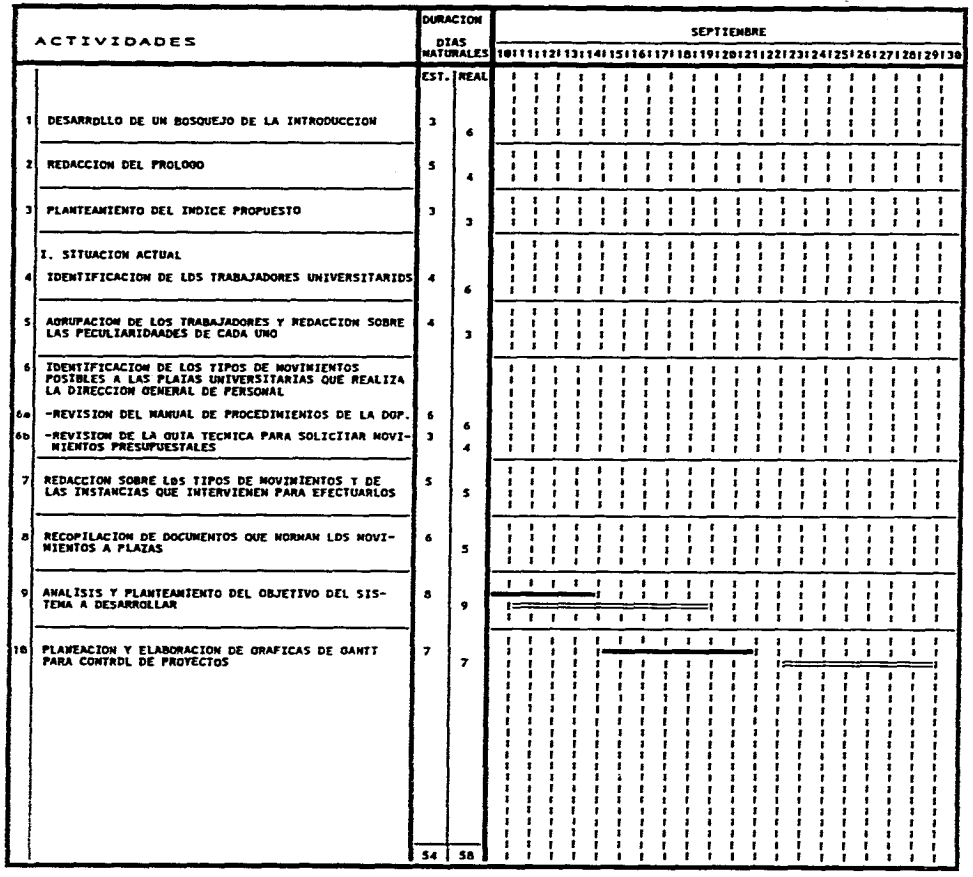
El plan de trabajo presentado se estructuró de acuerdo a las actividades que de manera secuencial deberían seguirse, comenzando desde plantear el índice, revisión de la documentación sobre el tema y recopilación de los requerimientos del usuario. Posteriormente se elaboró el plan de trabajo para la fase de análisis y diseño conforme a las actividades que la metodología seleccionada indica y se estimó el tiempo que llevaría realizarlas, así como el tiempo que tomaría elaborar las pruebas y modificaciones al código.

Conforme se avanzó en el programa de trabajo, se fué añadiendo las marcas del tiempo real que tomó concluir cada una de las actividades listadas de manera que reflejara el retraso u holgura en el tiempo estimado para realizar el proyecto.

SISTEMA AUTOMATIZADO DE MOVIMIENTOS AL ORGANIGRAMA

PLAN DE TRABAJO PARA LAS FASES PREVIAS AL ANALISIS (INTRODUCCION, SITUACION ACTUAL, OBJETIVO)

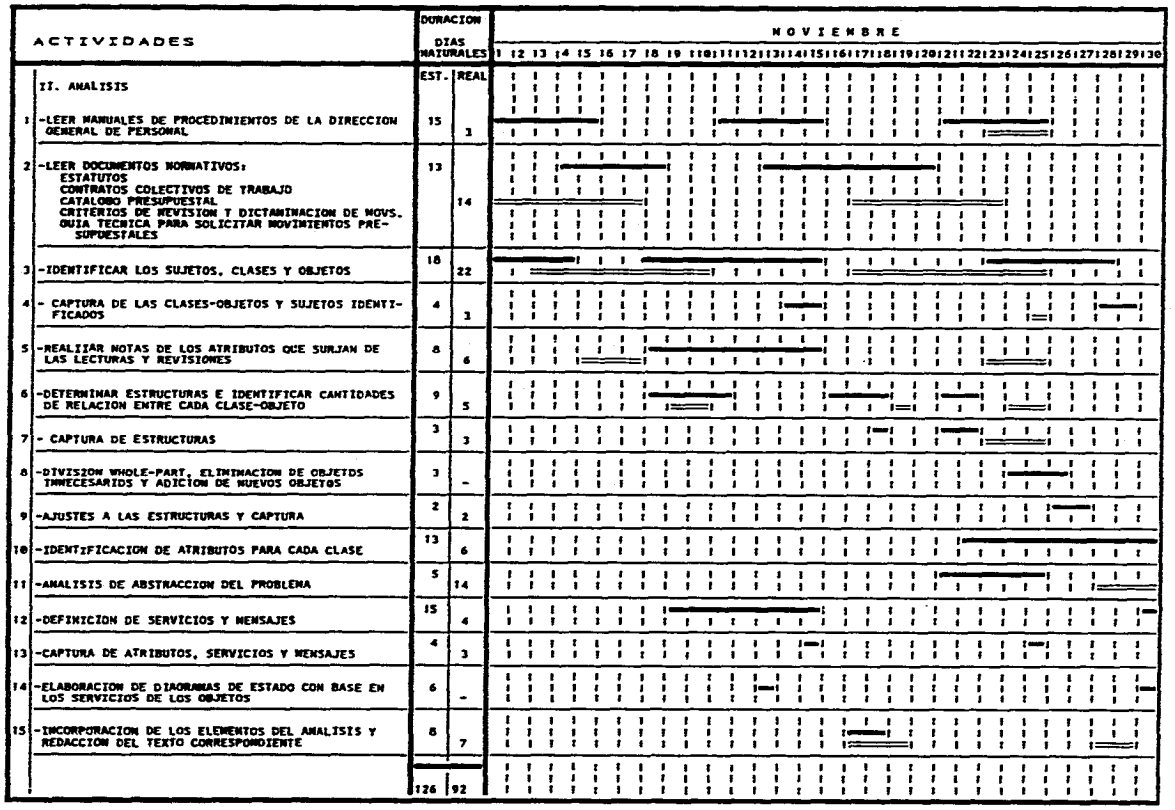



ACTIVIDADES
DURACION DIAS NATURALES
EST.
REAL
SEPTIEMBRE
10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
1 DESARROLLO DE UN BOSQUEJO DE LA INTRODUCCION 3 6
2 REDACCION DEL PROLOGO 5 4
3 PLANTEAMIENTO DEL INDICE PROPUESTO 3 3
I. SITUACION ACTUAL
4 IDENTIFICACION DE LOS TRABAJADORES UNIVERSITARIOS 4 6
5 AGRUPACION DE LOS TRABAJADORES Y REDACCION SOBRE LAS PECULIARIDADES DE CADA UNO 4 3
6 IDENTIFICACION DE LOS TIPOS DE MOVIMIENTOS POSIBLES A LAS PLAZAS UNIVERSITARIAS QUE REALIZA LA DIRECCION GENERAL DE PERSONAL
6a -REVISION DEL MANUAL DE PROCEDIMIENTOS DE LA DGP. 6 6
6b -REVISION DE LA GUIA TECNICA PARA SOLICITAR MOVIMIENTOS PRESUPUESTALES 3 4
7 REDACCION SOBRE LOS TIPOS DE MOVIMIENTOS Y DE LAS INSTANCIAS QUE INTERVIENEN PARA EFECTUARLOS 5 5
8 RECOPILACION DE DOCUMENTOS QUE NORMAN LOS MOVIMIENTOS A PLAZAS 6 5
9 ANALISIS Y PLANTEAMIENTO DEL OBJETIVO DEL SISTEMA A DESARROLLAR 8 9
10 PLANEACION Y ELABORACION DE GRAFICAS DE GANTT PARA CONTROL DE PROYECTOS 7 7
54 58
TIEMPO ESTIMADO
TIEMPO REAL

SISTEMA AUTOMATIZADO DE MOVIMIENTOS AL ORGANIGRAMA

PLAN DE TRABAJO PARA LAS FASES DE ANALISIS, DISEÑO E IMPLANTACION

	ACTIVIDADES	DURACION DIAS NATURALES EST.	REAL
	II. ANALISIS		
1	-LEER MANUALES DE PROCEDIMIENTOS DE LA DIRECCION GENERAL DE PERSONAL	15	3
2	-LEER DOCUMENTOS NORMATIVOS: ESTATUTOS CONTRATOS COLECTIVOS DE TRABAJO CATALOGO PRESUPUESTAL CRITERIOS DE REVISION Y DICTAMINACION DE MOVS. GUIA TECNICA PARA SOLICITAR MOVIMIENTOS PRESUPUESTALES	13	14
3	-IDENTIFICAR LOS SUJETOS, CLASES Y OBJETOS	18	22
4	- CAPTURA DE LAS CLASES-OBJETOS Y SUJETOS IDENTIFICADOS	4	3
5	-REALIIAR NOTAS DE LOS ATRIBUTOS QUE SURJAN DE LAS LECTURAS Y REVISIONES	8	6
6	-DETERMINAR ESTRUCTURAS E IDENTIFICAR CANTIDADES DE RELACION ENTRE CADA CLASE-OBJETO	9	5
7	- CAPTURA DE ESTRUCTURAS	3	3
8	-DIVISION WHOLE-PART, ELIMINACION DE OBJETOS INNECESARIOS Y ADICION DE NUEVOS OBJETOS	3	-
9	-AJUSTES A LAS ESTRUCTURAS Y CAPTURA	2	2
10	-IDENTIFICACION DE ATRIBUTOS PARA CADA CLASE	13	6
11	-ANALISIS DE ABSTRACCION DEL PROBLEMA	5	14
12	-DEFINICION DE SERVICIOS Y MENSAJES	15	4
13	-CAPTURA DE ATRIBUTOS, SERVICIOS Y MENSAJES	4	3
14	-ELABORACION DE DIAGRAMAS DE ESTADO CON BASE EN LOS SERVICIOS DE LOS OBJETOS	6	-
15	-INCORPORACION DE LOS ELEMENTOS DEL ANALISIS Y REDACCION DEL TEXTO CORRESPONDIENTE	8	7
		126	92



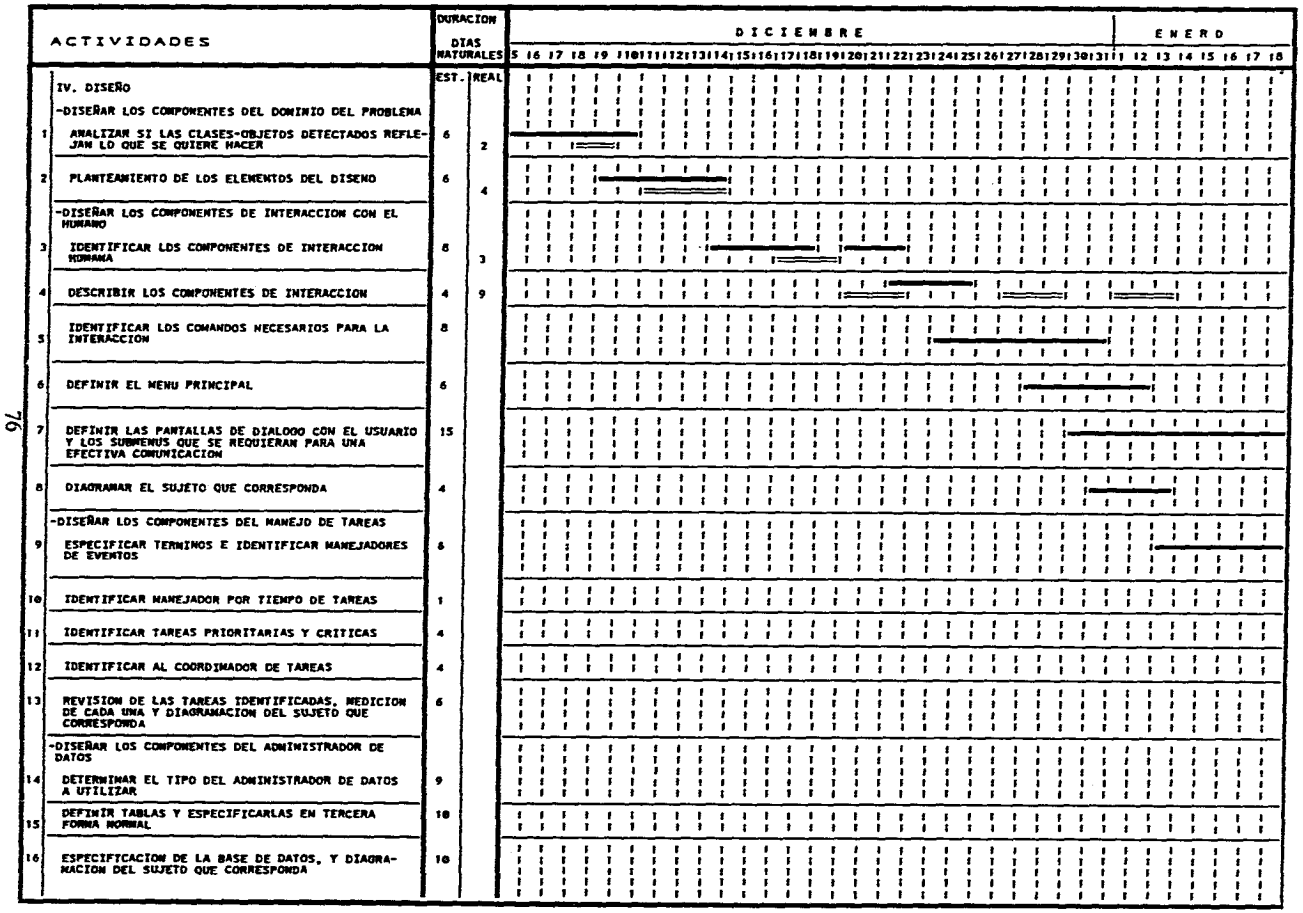
TIEMPO ESTIMADO
TIEMPO REAL

	ACTIVIDADES	DURACION DIAS NATURALES EST.	REAL	DICIEMBRE 30 1 2 3 4 5 6 7 8 9 10 11
	II. ANALISIS			
1	-LEER MANUALES DE PROCEDIMIENTOS DE LA DIRECCION GENERAL DE PERSONAL	15	3	
2	-LEER DOCUMENTOS NORMATIVOS: ESTATUTOS CONTRATOS COLECTIVOS DE TRABAJO CATALOGO PRESUPUESTAL CRITERIOS DE REVISION Y DICTAMINACION DE MOVS. GUIA TECNICA PARA SOLICITAR MOVIMIENTOS PRESUPUESTALES	13	14	
3	-IDENTIFICAR LOS SUJETOS, CLASES Y OBJETOS	18	22	
4	- CAPTURA DE LAS CLASES-OBJETOS Y SUJETOS IDENTIFICADOS	4	3	
5	-REALIZAR NOTAS DE LOS ATRIBUTOS QUE SURJAN DE LAS LECTURAS Y REVISIONES	8	6	
6	-DETERMINAR ESTRUCTURAS E IDENTIFICAR CANTIDADES DE RELACION ENTRE CADA CLASE-OBJETO	9	5	
7	- CAPTURA DE ESTRUCTURAS	3	3	
8	-DIVISION WHOLE-PART, ELIMINACION DE OBJETOS INNECESARIOS Y ADICION DE NUEVOS OBJETOS	3	-	
9	-AJUSTES A LAS ESTRUCTURAS Y CAPTURA	2	2	
10	-IDENTIFICACION DE ATRIBUTOS PARA CADA CLASE	13	6	
11	-ANALISIS DE ABSTRACCION DEL PROBLEMA	5	14	
12	-DEFINICION DE SERVICIOS Y MENSAJES	15	4	
13	-CAPTURA DE ATRIBUTOS, SERVICIOS Y MENSAJES	4	3	
14	-ELABORACION DE DIAGRAMAS DE ESTADO CON BASE EN LOS SERVICIOS DE LOS OBJETOS	6	-	
15	-INCORPORACION DE LOS ELEMENTOS DEL ANALISIS Y REDACCION DEL TEXTO CORRESPONDIENTE	8	7	
		126	92	

TIEMPO ESTIMADO
TIEMPO REAL

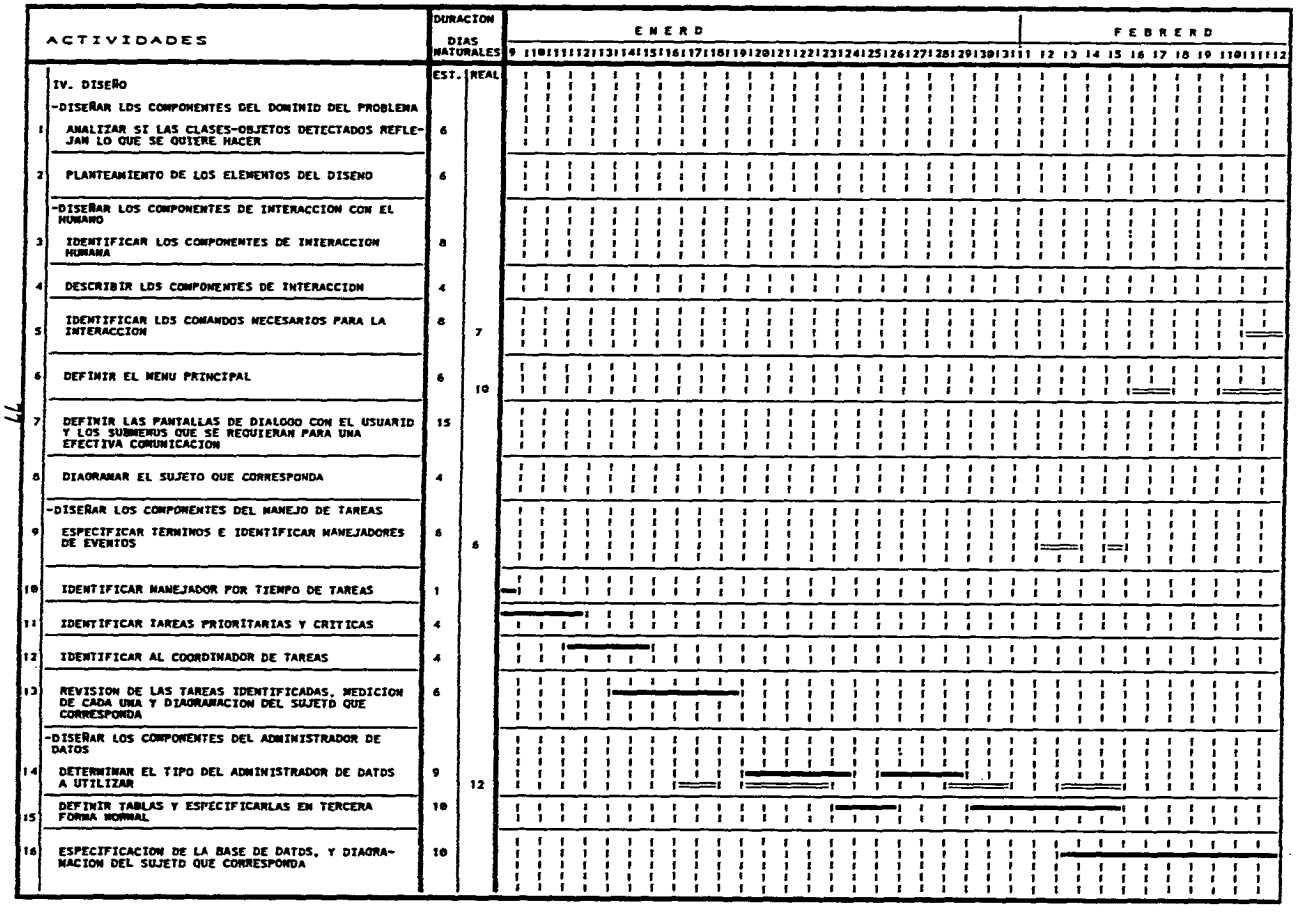
SISTEMA AUTOMATIZADO DE MOVIMIENTOS AL ORGANIGRAMA

PLAN DE TRABAJO PARA LAS FASES DE ANALISIS, DISEÑO E IMPLANTACION



SISTEMA AUTOMATIZADO DE MOVIMIENTOS AL ORGANIGRAMA

PLAN DE TRABAJO PARA LAS FASES DE ANALISIS, DISEÑO E IMPLANTACION

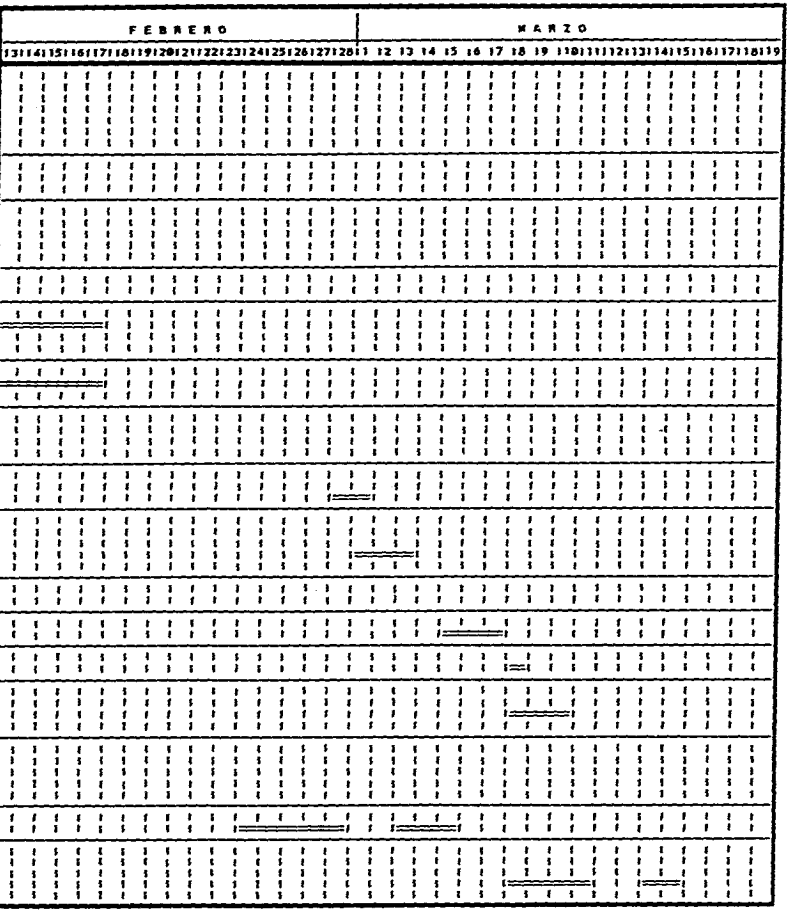


Control del proyecto

SISTEMA AUTOMATIZADO DE MOVIMIENTOS AL ORGANIGRAMA

PLAN DE TRABAJO PARA LAS FASES DE ANALISIS, DISEÑO E IMPLANTACION

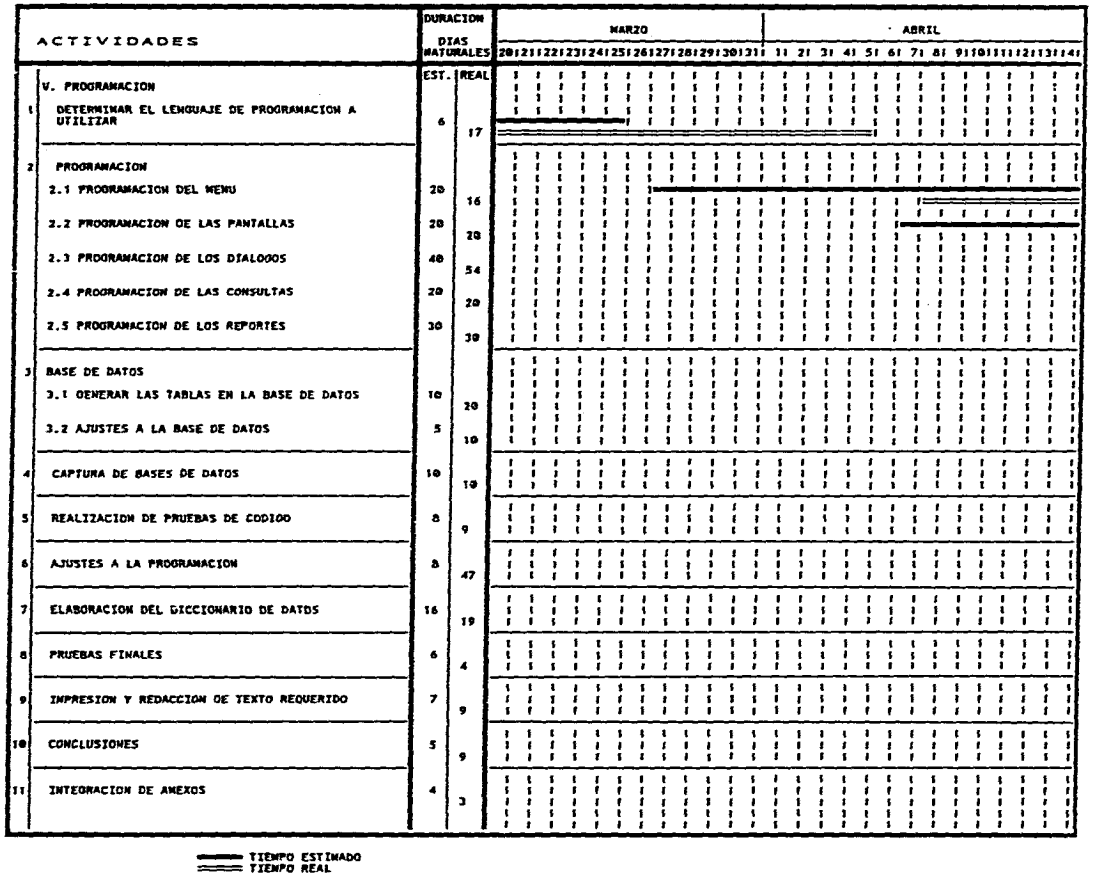
	ACTIVIDADES	DURACION DIAS NATURALES EST.	REAL
	IV. DISEÑO		
	-DISEÑAR LOS COMPONENTES DEL DOMINIO DEL PROBLEMA		
1	ANALIZAR SI LAS CLASES-OBJETOS DETECTADOS REFLEJAN LO QUE SE QUIERE HACER	6	
2	PLANTEAMIENTO DE LOS ELEMENTOS DEL DISEÑO	6	
	-DISEÑAR LOS COMPONENTES DE INTERACCION CON EL HUMANO		
3	IDENTIFICAR LOS COMPONENTES DE INTERACCION HUMANA	8	
4	DESCRIBIR LOS COMPONENTES DE INTERACCION	4	
5	IDENTIFICAR LOS COMANDOS NECESARIOS PARA LA INTERACCION	8	7
6	DEFINIR EL MENU PRINCIPAL	6	10
7	DEFINIR LAS PANTALLAS DE DIALOGO CON EL USUARIO Y LOS SUBMENUS QUE SE REQUIERAN PARA UNA EFECTIVA COMUNICACION	15	
8	DIAGRAMAR EL SUJETO QUE CORRESPONDA	4	2
	-DISEÑAR LOS COMPONENTES DEL MANEJO DE TAREAS		
9	ESPECIFICAR TERMINOS E IDENTIFICAR MANEJADORES DE EVENTOS	6	6
10	IDENTIFICAR MANEJADOR POR TIEMPO DE TAREAS	1	
11	IDENTIFICAR TAREAS PRIORITARIAS Y CRITICAS	4	3
12	IDENTIFICAR AL COORDINADOR DE TAREAS	4	1
13	REVISION DE LAS TAREAS IDENTIFICADAS, MEDICION DE CADA UNA Y DIAGRAMACION DEL SUJETO QUE CORRESPONDA	6	3
	-DISEÑAR LOS COMPONENTES DEL ADMINISTRADOR DE DATOS		
14	DETERMINAR EL TIPO DEL ADMINISTRADOR DE DATOS A UTILIZAR	9	12
15	DEFINIR TABLAS Y ESPECIFICARLAS EN TERCERA FORMA NORMAL	10	8
16	ESPECIFICACION DE LA BASE DE DATOS, Y DIAGRAMACION DEL SUJETO QUE CORRESPONDA	10	6



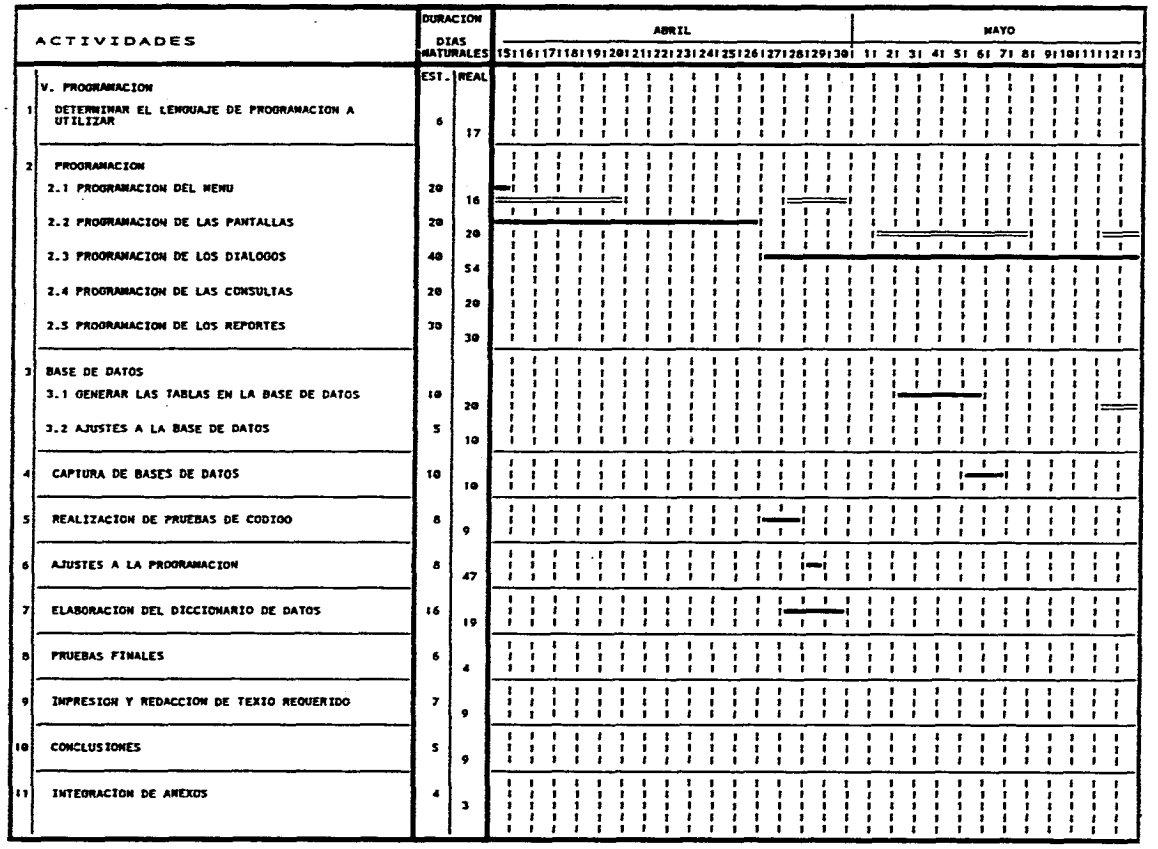
Control del proyecto

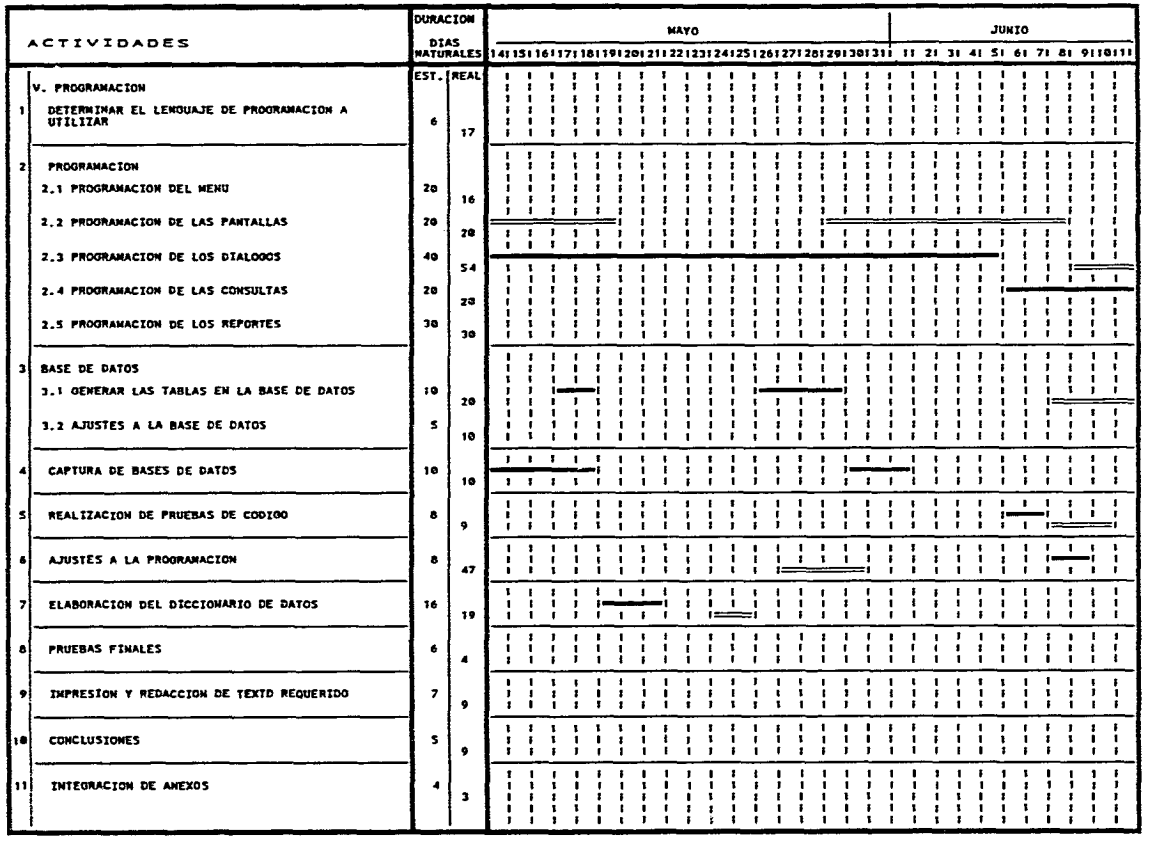
SISTEMA AUTOMATIZADO DE MOVIMIENTOS AL ORGANIGRAMA

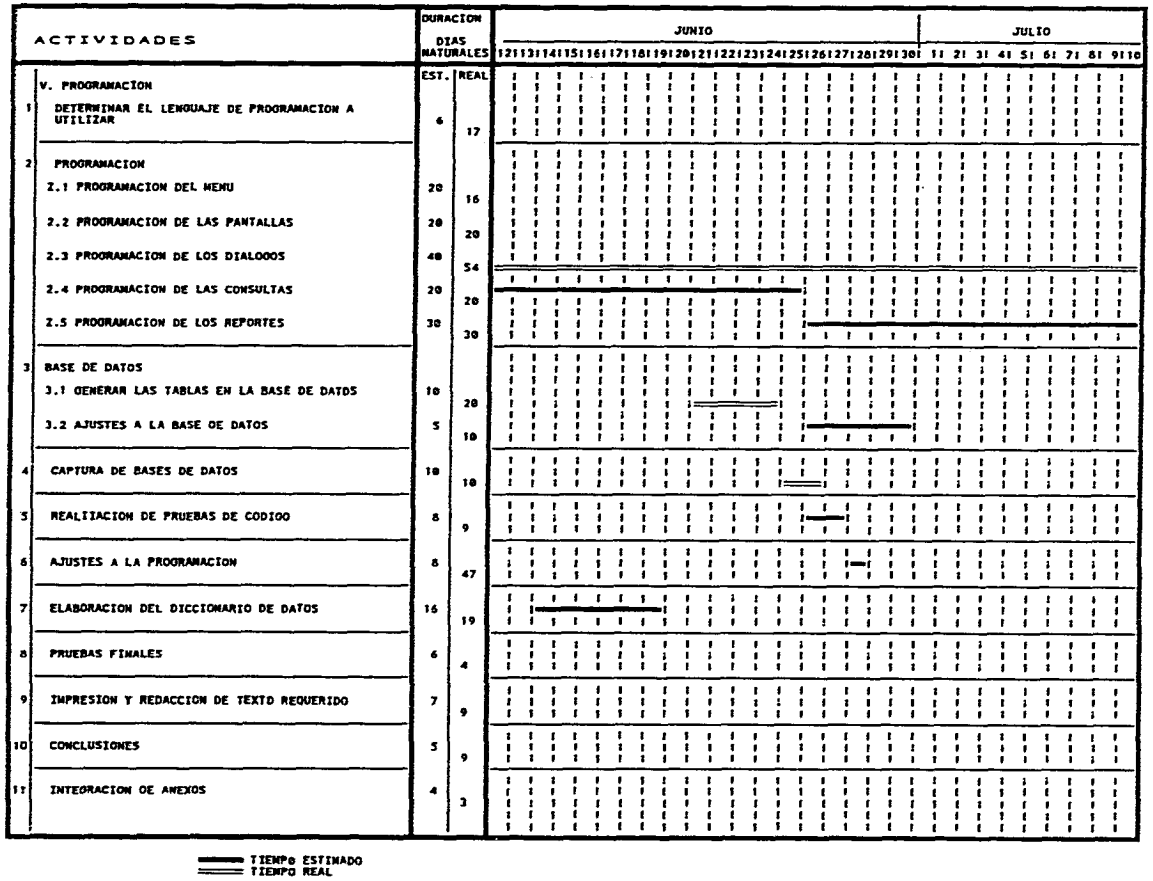
PLAN DE TRABAJO PARA LAS FASES DE ANALISIS, DISEÑO E IMPLANTACION



Control del proyecto


ACTIVIDADES
DURACION DIAS NATURALES
EST.
REAL
ABRIL
MAYO
15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 1 2 3 4 5 6 7 8 9 10 11 12 13
V. PROGRAMACION
1 DETERMINAR EL LENGUAJE DE PROGRAMACION A UTILIZAR 6 17
2 PROGRAMACION
2.1 PROGRAMACION DEL MENU 20 16
2.2 PROGRAMACION DE LAS PANTALLAS 20 20
2.3 PROGRAMACION DE LOS DIALOGOS 40 54
2.4 PROGRAMACION DE LAS CONSULTAS 20 20
2.5 PROGRAMACION DE LOS REPORTES 30 30
3 BASE DE DATOS
3.1 GENERAR LAS TABLAS EN LA BASE DE DATOS 10 20
3.2 AJUSTES A LA BASE DE DATOS 5 10
4 CAPTURA DE BASES DE DATOS 10 10
5 REALIZACION DE PRUEBAS DE CODIGO 8 9
6 AJUSTES A LA PROGRAMACION 8 47
7 ELABORACION DEL DICCIONARIO DE DATOS 16 19
8 PRUEBAS FINALES 6 4
9 IMPRESION Y REDACCION DE TEXTO REQUERIDO 7 9
10 CONCLUSIONES 5 9
11 INTEGRACION DE ANEXOS 4 3
TIEMPO ESTIMADO
TIEMPO REAL


ACTIVIDADES
DURACION
DIAS NATURALES
EST.
REAL
MAYO
JUNIO
14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 1 2 3 4 5 6 7 8 9 10 11
V. PROGRAMACION
1 DETERMINAR EL LENGUAJE DE PROGRAMACION A UTILIZAR 6 17
2 PROGRAMACION
2.1 PROGRAMACION DEL MENU 20 16
2.2 PROGRAMACION DE LAS PANTALLAS 20 28
2.3 PROGRAMACION DE LOS DIALOGOS 40 54
2.4 PROGRAMACION DE LAS CONSULTAS 20 23
2.5 PROGRAMACION DE LOS REPORTES 30 30
3 BASE DE DATOS
3.1 GENERAR LAS TABLAS EN LA BASE DE DATOS 10 20
3.2 AJUSTES A LA BASE DE DATOS 5 10
4 CAPTURA DE BASES DE DATOS 10 10
5 REALIZACION DE PRUEBAS DE CODIGO 8 9
6 AJUSTES A LA PROGRAMACION 8 47
7 ELABORACION DEL DICCIONARIO DE DATOS 16 19
8 PRUEBAS FINALES 6 4
9 IMPRESION Y REDACCION DE TEXTO REQUERIDO 7 9
10 CONCLUSIONES 5 9
11 INTEGRACION DE ANEXOS 4 3
TIEMPO ESTIMADO
TIEMPO REAL


ACTIVIDADES
DURACION DIAS NATURALES
EST.
REAL
JUNIO
JULIO
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 1 2 3 4 5 6 7 8 9 10
V. PROGRAMACION
1 DETERMINAR EL LENGUAJE DE PROGRAMACION A UTILIZAR 6 17
2 PROGRAMACION
2.1 PROGRAMACION DEL MENU 20 16
2.2 PROGRAMACION DE LAS PANTALLAS 20 20
2.3 PROGRAMACION DE LOS DIALOGOS 40 54
2.4 PROGRAMACION DE LAS CONSULTAS 20 20
2.5 PROGRAMACION DE LOS REPORTES 30 30
3 BASE DE DATOS
3.1 GENERAR LAS TABLAS EN LA BASE DE DATOS 10 20
3.2 AJUSTES A LA BASE DE DATOS 5 10
4 CAPTURA DE BASES DE DATOS 10 10
5 REALIIACION DE PRUEBAS DE CODIGO 8 9
6 AJUSTES A LA PROGRAMACION 8 47
7 ELABORACION DEL DICCIONARIO DE DATOS 16 19
8 PRUEBAS FINALES 6 4
9 IMPRESION Y REDACCION DE TEXTO REQUERIDO 7 9
10 CONCLUSIONES 5 9
11 INTEGRACION DE ANEXOS 4 3
TIEMPO ESTIMADO
TIEMPO REAL

	ACTIVIDADES	DURACION DIAS NATURALES EST.	REAL
	V. PROGRAMACION		
1	DETERMINAR EL LENGUAJE DE PROGRAMACION A UTILIZAR	6	17
2	PROGRAMACION		
	2.1 PROGRAMACION DEL MENU	20	16
	2.2 PROGRAMACION DE LAS PANTALLAS	20	20
	2.3 PROGRAMACION DE LOS DIALOGOS	40	54
	2.4 PROGRAMACION DE LAS CONSULTAS	20	20
	2.5 PROGRAMACION DE LOS REPORTES	30	30
3	BASE DE DATOS		
	3.1 GENERAR LAS TABLAS EN LA BASE DE DATOS	10	20
	3.2 AJUSTES A LA BASE DE DATOS	5	10
4	CAPTURA DE BASES DE DATOS	10	10
5	REALIZACION DE PRUEBAS DE CODIGO	8	9
6	AJUSTES A LA PROGRAMACION	8	47
7	ELABORACION DEL DICCIONARID DE DATOS	16	19
8	PRUEBAS FINALES	6	4
9	IMPRESION Y REDACCION DE TEXTO REQUERIDO	7	9
10	CONCLUSIONES	5	9
11	INTEGRACION DE ANEXOS	4	3

JULIO 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

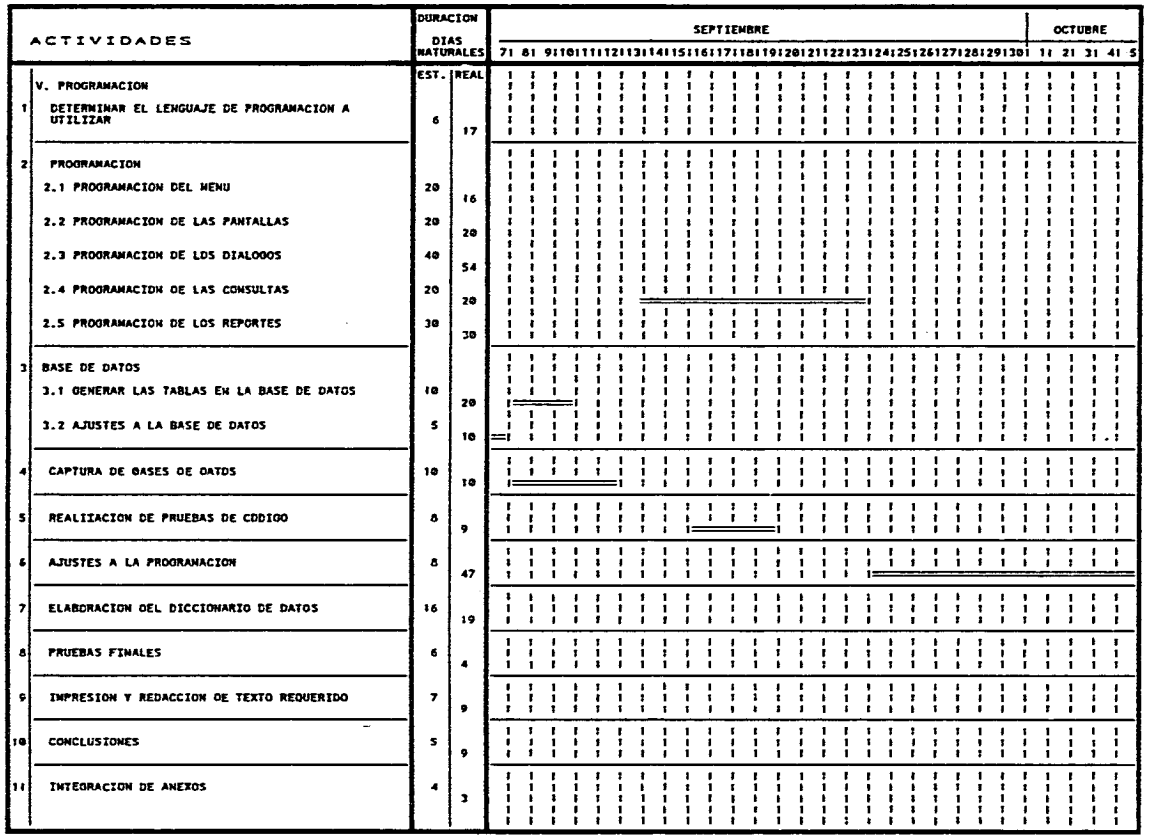
AGOSTO 1 2 3 4 5 6 7 8

TIEMPO ESTIMADO
TIEMPO REAL

	ACTIVIDADES	DURACION DIAS NATURALES EST.	REAL
	V. PROGRAMACION		
1	DETERMINAR EL LENGUAJE DE PROGRAMACION A UTILIZAR	6	17
2	PROGRAMACION		
	2.1 PROGRAMACION DEL MENU	20	16
	2.2 PROGRAMACION DE LAS PANTALLAS	20	20
	2.3 PROGRAMACION DE LOS DIALOGOS	40	54
	2.4 PROGRAMACION DE LAS CONSULTAS	20	28
	2.5 PROGRAMACION DE LOS REPORTES	30	30
3	BASE DE DATOS		
	3.1 GENERAR LAS TABLAS EN LA BASE DE DATOS	10	20
	3.2 AJUSTES A LA BASE DE DATOS	5	10
4	CAPTURA DE BASES DE DATOS	10	10
5	REALIZACION DE PRUEBAS DE CODIGO	8	9
6	AJUSTES A LA PROGRAMACION	8	47
7	ELABORACION DEL DICCIONARIO DE DATOS	16	19
8	PRUEBAS FINALES	6	4
9	IMPRESION Y REDACCION DE TEXTO REQUERIDO	7	9
10	CONCLUSIONES	5	9
11	INTEGRACION DE ANEXOS	4	3

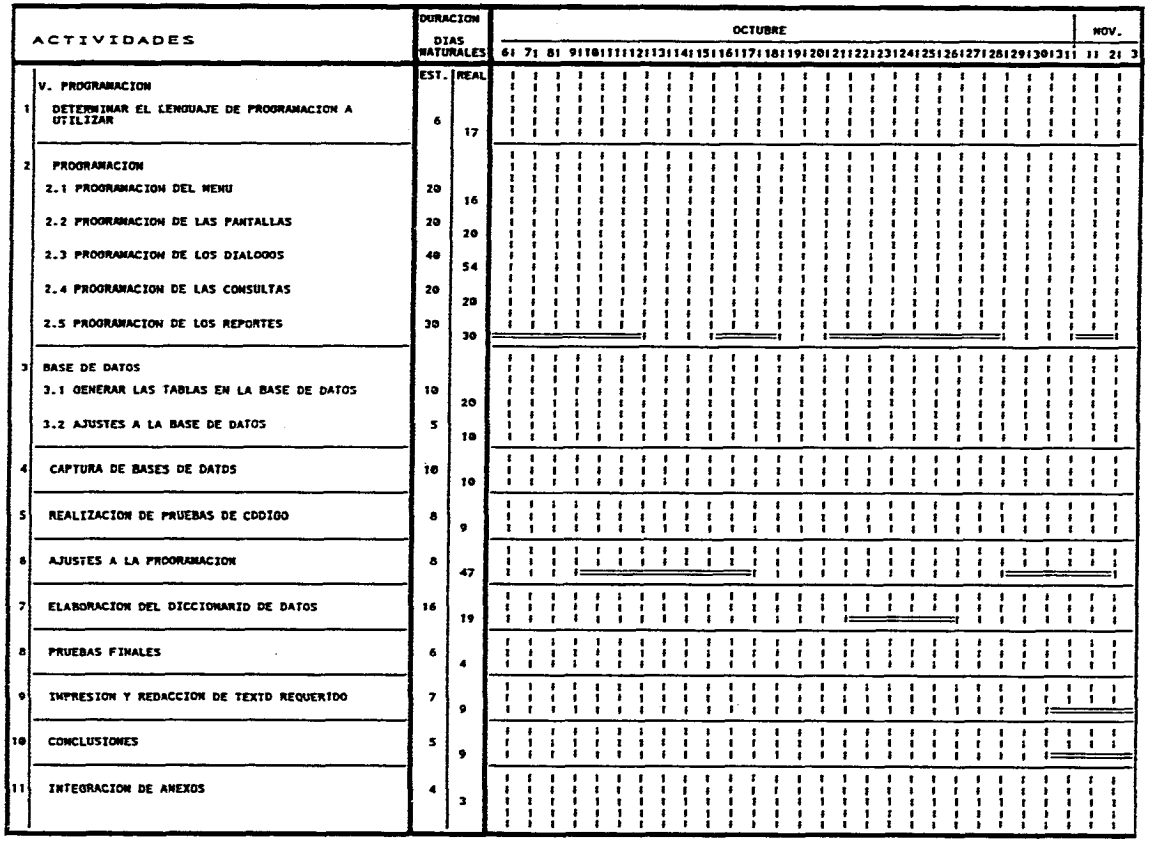
TIEMPO ESTIMADO
TIEMPO REAL

	ACTIVIDADES	DURACION DIAS NATURALES EST.	REAL
	V. PROGRAMACION		
1	DETERMINAR EL LENGUAJE DE PROGRAMACION A UTILIZAR	6	17
2	PROGRAMACION		
	2.1 PROGRAMACION DEL MENU	20	16
	2.2 PROGRAMACION DE LAS PANTALLAS	20	20
	2.3 PROGRAMACION DE LOS DIALOGOS	40	54
	2.4 PROGRAMACION DE LAS CONSULTAS	20	20
	2.5 PROGRAMACION DE LOS REPORTES	30	30
3	BASE DE DATOS		
	3.1 GENERAR LAS TABLAS EN LA BASE DE DATOS	10	20
	3.2 AJUSTES A LA BASE DE DATOS	5	10
4	CAPTURA DE BASES DE DATOS	10	10
5	REALIIACION DE PRUEBAS DE CODIGO	8	9
6	AJUSTES A LA PROGRAMACION	8	47
7	ELABORACION DEL DICCIONARIO DE DATOS	16	19
8	PRUEBAS FINALES	6	4
9	IMPRESION Y REDACCION DE TEXTO REQUERIDO	7	9
10	CONCLUSIONES	5	9
11	INTEGRACION DE ANEXOS	4	3



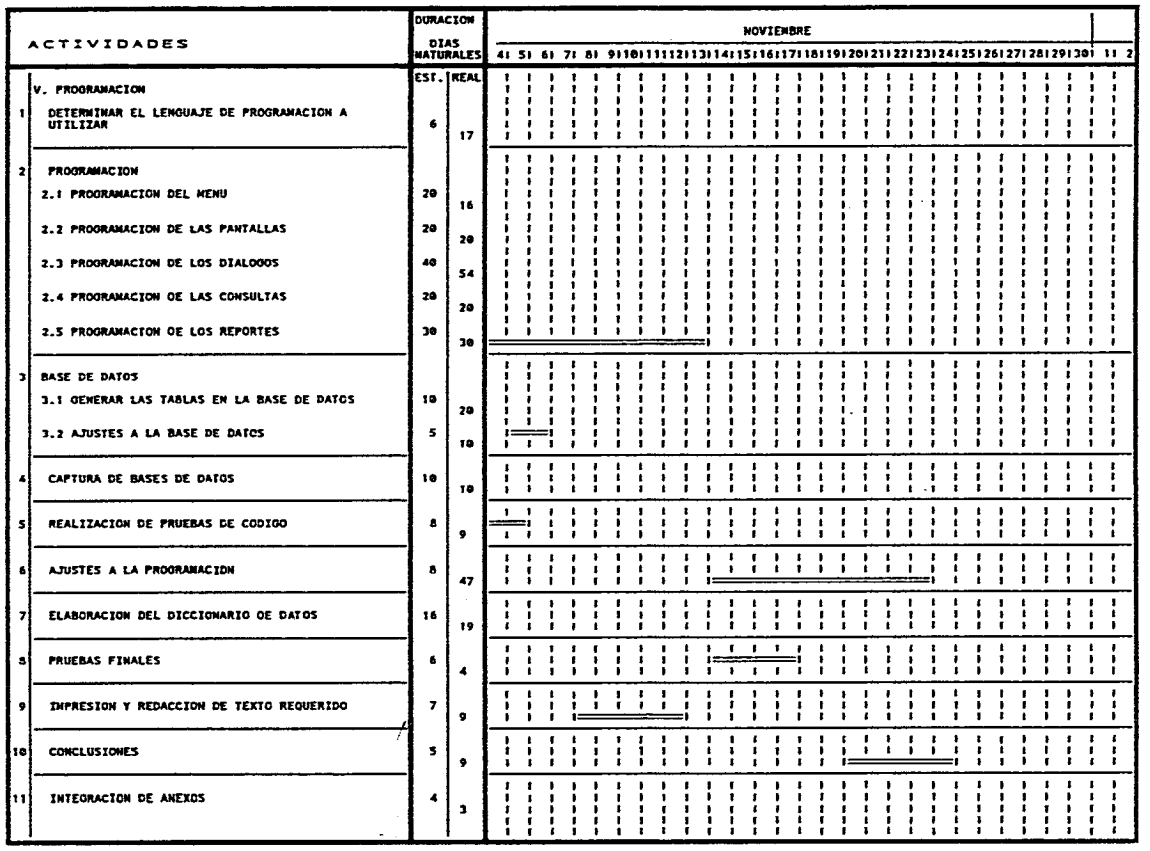
TIEMPO ESTIMADO
TIEMPO REAL

	ACTIVIDADES	DURACION DIAS NATURALES EST.	DURACION DIAS NATURALES REAL
	V. PROGRAMACION		
1	DETERMINAR EL LENGUAJE DE PROGRAMACION A UTILIZAR	6	17
2	PROGRAMACION		
	2.1 PROGRAMACION DEL MENU	20	16
	2.2 PROGRAMACION DE LAS PANTALLAS	20	20
	2.3 PROGRAMACION DE LOS DIALOGOS	40	54
	2.4 PROGRAMACION DE LAS CONSULTAS	20	20
	2.5 PROGRAMACION DE LOS REPORTES	30	30
3	BASE DE DATOS		
	3.1 GENERAR LAS TABLAS EN LA BASE DE DATOS	10	20
	3.2 AJUSTES A LA BASE DE DATOS	5	10
4	CAPTURA DE BASES DE DATOS	10	10
5	REALIZACION DE PRUEBAS DE CODIGO	8	9
6	AJUSTES A LA PROGRAMACION	8	47
7	ELABORACION DEL DICCIONARID DE DATOS	16	19
8	PRUEBAS FINALES	6	4
9	IMPRESION Y REDACCION DE TEXTO REQUERIDO	7	9
10	CONCLUSIONES	5	9
11	INTEGRACION DE ANEXOS	4	3



TIEMPO ESTIMADO
TIEMPO REAL

	ACTIVIDADES	DURACION DIAS NATURALES ESTIMADO	DURACION DIAS NATURALES REAL
	V. PROGRAMACION		
1	DETERMINAR EL LENGUAJE DE PROGRAMACION A UTILIZAR	6	17
2	PROGRAMACION		
	2.1 PROGRAMACION DEL MENU	20	16
	2.2 PROGRAMACION DE LAS PANTALLAS	20	20
	2.3 PROGRAMACION DE LOS DIALOGOS	40	54
	2.4 PROGRAMACION DE LAS CONSULTAS	20	20
	2.5 PROGRAMACION DE LOS REPORTES	30	30
3	BASE DE DATOS		
	3.1 GENERAR LAS TABLAS EN LA BASE DE DATOS	10	20
	3.2 AJUSTES A LA BASE DE DATOS	5	10
4	CAPTURA DE BASES DE DATOS	10	10
5	REALIZACION DE PRUEBAS DE CODIGO	8	9
6	AJUSTES A LA PROGRAMACION	8	47
7	ELABORACION DEL DICCIONARIO DE DATOS	16	19
8	PRUEBAS FINALES	6	4
9	IMPRESION Y REDACCION DE TEXTO REQUERIDO	7	9
10	CONCLUSIONES	5	9
11	INTEGRACION DE ANEXOS	4	3



TIEMPO ESTIMADO
TIEMPO REAL

VI. CONCLUSIONES

Como resultado de la elaboración del presente trabajo concluyo que si es posible elaborar un sistema que controle y agilice los movimientos de personal, tal como se plantea en los requerimientos del capítulo I, y apegándose a los lineamientos y normatividad vigente. Aunado a lo anterior la realización del prototipo demostró que si es posible aplicar metodologías orientadas a objetos para el desarrollo de sistemas de control administrativo. El desarrollo de este trabajo se apegó a la metodología orientada a objetos de Coad y Yourdon, resultando que esta metodología obliga a comprender y plasmar las necesidades reales de los usuarios, haciendo especial énfasis en el entendimiento del problema, asentando así las bases para un diseño que cubra sus requerimientos,

Otro aspecto importante de la metodología aplicada es la notación que introducen los autores para la representación de modelos durante el análisis y diseño ya que ésta resulta bastante clara y amigable.

Por otra parte dicha metodología propone en la fase de análisis la

generación de diagramas de estado, los cuales reflejan los cambios de un objeto a través del tiempo o después de un evento, lo cual parece una herencia de la metodología estructurada de Yourdon, que además no tiene aplicación posterior dentro de su metodología orientada a objetos, por lo que tal vez resultaría conveniente retomarlos durante el diseño, en mi opinión, durante el diseño del componente del manejador de tareas, de manera tal que proporcionaría mayores elementos de especificidad al programador para elaborar el código.

Durante el diseño del componente del administrador de datos la metodología señala determinar las necesidades de almacenamiento para procesar la información que cubra los requerimientos del usuario, sin embargo, ya que se considera que el prototipo resultante puede ser aplicado en diversas dependencias, se requeriría para cada una de ellas realizar un cálculo específico, dependiendo principalmente de la cantidad de plazas y trabajadores que cada una de ellas tenga, por lo que en caso de retomar este trabajo se sugiere adicionar elementos de capacidad de almacenamiento para aplicarlo al caso particular.

El presente trabajo tomó dos años para su conclusión, tiempo que personalmente considero suficiente para adoptar y aplicar una

metodología que cambie la visión de desarrollar sistemas con metodologías estructuradas a la visión de desarrollo de sistemas orientados a objetos, resultando este esfuerzo en el principal factor de retraso en el programa de trabajo planteado al iniciar esta tesis, lo cual se refleja en el capítulo de control del proyecto.

La implementación del sistema se realizó en un ambiente orientado a eventos lo cual resultó de gran utilidad para familiarizarse con los lenguajes orientados a objetos ya que rompe con el paradigma de pensar y desarrollar sistemas de manera estructurada.

El plantearse un programa de trabajo para controlar el proyecto resultó de gran utilidad para medir y controlar las actividades que debían realizarse para concluir el trabajo además de evitar desviaciones provocadas por costumbres en el desarrollo de sistemas estructurados y controlar retrasos.

El sistema que propone la presente tesis podría verse enriquecido si se pensara en una base de datos centralizada con información de todos los trabajadores universitarios, ya que el sistema desarrollado trabaja con una base de datos local, con información de trabajadores que laboran en una sola dependencia, para lo cual habría que realizar

modificaciones a partir del componente del manejador de tareas del diseño que se enfocaran básicamente a la distribución y replicación de los datos.

BIBLIOGRAFÍA

1.- Borland C++ 3.1. Object-Oriented Programming
Ted Faison
Ed. Sams Publishing, 2ª edición 1992 (a Division of Prentice Hall Computer Publishing)

2.- Ingeniería de Software, un enfoque práctico.
Roger S. Pressman
Mc. Graw Hill, 3ª Edición
Traducción Carlos Cervigón Ruckaüer/ Luis Hernández Yañez
Madrid.

3.- Instructivo de ejercicio y Catálogo Presupuestal.
UNAM, publicación anual.

4.- Manual de borland C++
Chris H. Pappas. William H. Murray III.
Serie Mc. Graw Hill

5.- Programación avanzada de gráficos en C para Windows
Lee Adams
Serie Mc. Graw Hill de Informática

6.- Programming Windows 3.1.
Charles Petzold
Ed. Microsoft Press, 3ª edición 1992

7.- Object-Oriented Programming. The Software Development Revolution.
Varhol, Peter D.
Ed. Computer Technology Research Corp. 1ª Edición 1992.

8.- Object Oriented Requirements Analysis and Logical Design. (A Software Engineering Approach)
Donald G. Firesmith
Ed. WILEY, USA

9.- SQL Language User´s Manual.
NetWare SQL.
Novell Database Products. Ed. 1992.

10.- SYBASE SQL Server. Transact-SQL User´s Guide.
Server Publications Group
Sybase Inc. 1993

11.- Tecnicas de Bases de Datos. Estructuración en diseño y administración.
Atre, Shakuntala
Ed. Trillas 1988.

ANEXO A. PROGRAMACIÓN

La programación del presente sistema se realizó en Borland C++ versión 4.0 para ambiente Windows, la interfase gráfica para el usuario se codificó utilizando la herramienta Workshop que provee Borland y el manejador de base de datos utilizado es SYBASE.

A continuación se presenta el código fuente del programa principal y algunas funciones, el código para generar la pantalla de Nuevo Ingreso, la definición de los campos para dicha pantalla, y el archivo de cabecera donde están definidas algunas de las estructuras de datos utilizadas para manejar la información de la base de datos.

Cabe señalar que el código para recuperar, actualizar o borrar registros de la base de datos se realiza a través de instrucciones de Structured Query Language (SQL) que se incuentra imbuído en el código de las funciones de los diálogos.

El código fuente, como podrá apreciarse en la reproducción que de él se incluye, contiene comentarios que indican lo que cada trozo de código realiza, lo cual proporcionará al lector una idea del funcionamiento del sistema.

Programa principal:

```
#include "tesis1.h"
#include "tesis1.rh"

#define msNombreDeLaClase "Movimientos"

int iBaseDeDatosAbierta = 0;

long FAR PASCAL wlpfnDespachadorDeMensajes(HWND, UINT, UINT, LONG);

int PASCAL WinMain(HANDLE whAplicacion, HANDLE whAplicacionPrevia, LPSTR
wsLineaDeComando, int iCommandShow)
{
    HWND    whVentana;
    MSG     wmMensaje;
    WNDCLASS wwClaseVentana;

    if (!whAplicacionPrevia)
    {
       wwClaseVentana.style         = CS_HREDRAW | CS_VREDRAW;
       wwClaseVentana.lpfnWndProc   = wlpfnDespachadorDeMensajes;
       wwClaseVentana.cbClsExtra    = 0;
       wwClaseVentana.cbWndExtra    = 0;
       wwClaseVentana.hInstance     = whAplicacion;
       wwClaseVentana.hIcon         = LoadIcon(NULL,IDI_APPLICATION);
       wwClaseVentana.hCursor       = LoadCursor(NULL,IDC_ARROW);
       wwClaseVentana.hbrBackground = GetStockObject(WHITE_BRUSH);
       wwClaseVentana.lpszMenuName  = "MenuMovimientos";
```

```
        wwClaseVentana.lpszClassName = msNombreDeLaClase;
        RegisterClass(&wwClaseVentana);
    }

    whVentana = CreateWindow(msNombreDeLaClase, "Movimientos al Personal Universitario",
                             WS_OVERLAPPEDWINDOW,
                             CW_USEDEFAULT, CW_USEDEFAULT,
                             CW_USEDEFAULT, CW_USEDEFAULT,
                             NULL, NULL,
                             whAplicacion, NULL);

    ShowWindow(whVentana, iCommandShow);
    UpdateWindow(whVentana);

    while (GetMessage(&wmMensaje, NULL, 0, 0))
    {
        TranslateMessage(&wmMensaje);
        DispatchMessage(&wmMensaje);
    }

    return wmMensaje.wParam;
}

long FAR PASCAL wlpfnDespachadorDeMensajes(HWND whVentana, UINT wuMensaje, UINT
wuParametro, LONG
wlParametro)
{
  int     basura;
  char    acBasura[10];
  HANDLE  whLib;
  FARPROC lpfnLeeNombreDeUsuario;
```

```
switch(wuMensaje)
{
 static FARPROC aoCajaDeDialago;
  static HANDLE whInstanciaManejaVentana;

  case WM_CREATE:
     whInstanciaManejaVentana = ((LPCREATESTRUCT) wlParametro)->hInstance;

     return 0L;

  case WM_COMMAND:
     {
       switch( wuParametro)
        {
        // Los siguientes CASE corrresponde a una opción del menú.
        // Cada opción direcciona el apuntador a la función que responde al evento,
        // llama a la caja de diálogo que le corresponde y después libera al apuntador

        case CM_ABRIRSESION:

            if(iConectarseALaBaseDeDatos("sa","sip123","tesis1","DgpSqlServer","dbtesis"))
              return 0L;
            iBaseDeDatosAbierta = 1;
              return 0L;

        case CM_CERRARSESION:
            if (iBaseDeDatosAbierta)
              dbexit();
            iBaseDeDatosAbierta = 0;
              return 0L;

        case CM_SALIR:
```

```
    SendMessage(whVentana,WM_CLOSE,0,0L);
      return 0L;

case CM_NUEVOINGRESO:
    aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)NuevoIngreso,
        whInstanciaManejaVentana); // la función NuevoIngreso se programa en nueing.c
  DialogBox(whInstanciaManejaVentana,"DialogoNuevoIngreso",
  whVentana,aoCajaDeDialago);
   FreeProcInstance((FARPROC)aoCajaDeDialago);
      return 0L;

case CM_REINGRESO:
    aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)Reingreso,
     whInstanciaManejaVentana);
  DialogBox(whInstanciaManejaVentana, "DialogoReingreso", whVentana, aoCajaDeDialago);
   FreeProcInstance((FARPROC)aoCajaDeDialago);
      return 0L;

case CM_OTRONOMBRAMIENTO:
    aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)OtroNombramiento,
     whInstanciaManejaVentana);
  DialogBox(whInstanciaManejaVentana,  "DialogoOtroNombramiento",  whVentana,
  aoCajaDeDialago);
    FreeProcInstance((FARPROC)aoCajaDeDialago);
      return 0L;

case CM_RECLASIFICACION:
    aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)Reclasificacion,
     whInstanciaManejaVentana);
  DialogBox(whInstanciaManejaVentana,  "DialogoReclasificacion",  whVentana,
  aoCajaDeDialago);
    FreeProcInstance((FARPROC)aoCajaDeDialago);
```

```
        return 0L;

case CM_LICENCIAS:
    aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)Licencia,
     whInstanciaManejaVentana);
    DialogBox(whInstanciaManejaVentana,"DialogoLicencia",whVentana,aoCajaDeDialago);
              FreeProcInstance((FARPROC)aoCajaDeDialago);
        return 0L;

case CM_BAJA:
  aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)Baja,
  whInstanciaManejaVentana);
    DialogBox(whInstanciaManejaVentana, "DialogoBaja", whVentana, aoCajaDeDialago);
    FreeProcInstance((FARPROC)aoCajaDeDialago);
        return 0L;

case CM_CONSULTARFC:
    aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)ConsultaRfc,
     whInstanciaManejaVentana);
  DialogBox(whInstanciaManejaVentana, "DialogoConsultaRFC", whVentana,
  aoCajaDeDialago);
    FreeProcInstance((FARPROC)aoCajaDeDialago);
        return 0L;

case CM_CONSULTAPLAZA:
    aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)ConsultaPlaza,
     whInstanciaManejaVentana);
  DialogBox(whInstanciaManejaVentana, "DialogoConsultaPlaza", whVentana,
  aoCajaDeDialago);
    FreeProcInstance((FARPROC)aoCajaDeDialago);
        return 0L;
```

```
            case CM_CONSULTACATEGORIA:
                aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)ConsultaCategoria,
                 whInstanciaManejaVentana);
              DialogBox(whInstanciaManejaVentana,   "DialogoConsultaCategoria",   whVentana,
              aoCajaDeDialago);
                FreeProcInstance((FARPROC)aoCajaDeDialago);
                  return 0L;

            case CM_CONSPRESUPUESTO:
                aoCajaDeDialago = (DLGPROC)MakeProcInstance((FARPROC)ConsultaPresupuesto,
                 whInstanciaManejaVentana);
                DialogBox(whInstanciaManejaVentana, "DialogoConsultaPresupuesto", whVentana,
                 aoCajaDeDialago);
                FreeProcInstance((FARPROC)aoCajaDeDialago);
                  return 0L;

        }
        return 0L;
}

case WM_DESTROY:
  if (iBaseDeDatosAbierta)
    dbexit();
  PostQuitMessage(0);
   return 0L;

default:
  return DefWindowProc(whVentana, wuMensaje, wuParametro, wlParametro);
}
}
```

```
/*****************************************************************************
Programa de la función Nuevo Ingreso:

#include <windows.h>
#include <string.h>
#include "tesis1.h"

#include "tesis1.rh"

#define IDC_DESHABILITA_COMBO_CATEGO 2101

struct CatalogoCategorias                    *VarCatalogoCategorias;
struct CatalogoEstadosCiviles                 *VarCatalogoEstadosCiviles;
struct CatalogoNacionalidades                  *VarCatalogoNacionalidades;
struct CatalogoDependencias                    *VarCatalogoDependencias;
struct PlazasVacantes                        *VarPlazasVacantes;
struct CatalogoEscolaridades                  *VarCatalogoEscolaridades;

int iNumPlazasVacantes,
  iNumCategorias,
  iNumEstadosCiviles,
  iNumNacionalidades,
  iNumEscolaridades,
  iNumDependencias;


BOOL FAR PASCAL NuevoIngreso(HWND WhManejadorDeLaVentana, UINT iMensaje, WORD wParam,
DWORD lParam)
{
 char acInstruccion[600];
```

```
DBCHAR VarDePasoClave[13];
DBCHAR VarDePasoAux[6];
DBCHAR VarDePasoDesc[201];

RETCODE return_code;
int  iErrorDeSQL;
int  i=0;
float afTempSueldo;
char NombreCompleto[19];

DBCHAR  acTempNumPlaza[13];
DBCHAR  acTempHoras[10];
DBCHAR acApellidoPaterno[30];
DBCHAR acApellidoMaterno[30];
DBCHAR acNombre[30];
DBCHAR acdNacimiento[9];
DBCHAR acDomicilio[100];
DBCHAR acdSolicitud[9];
DBCHAR acdIngreso[9];
DBCHAR acTelefono[13];
DBCHAR acExperiencia[3];
DBCHAR ActaNacim;
DBCHAR PermisoSRE;
DBCHAR Estudios;
DBINT  aiConsTHistoria;

static DBCHAR acRFCUSU[14];
static DBCHAR acRFCDB[14];
static int dwSeleccionCategoria;
static int dwSeleccionPlaza;
static int dwSeleccionNacionalidad;
static int dwSeleccionEscolaridad;
```

```
static int dwSeleccionDependencia;
static int dwSeleccionEstadoCivil;



switch(iMensaje)
{
  case WM_INITDIALOG:
     SetFocus(GetDlgItem(WhManejadorDeLaVentana,IDC_RFCNI));
     EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_VERIFICANI),TRUE);
     EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_ACEPTARNI),FALSE);
     EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_CANCELARNI),FALSE);
     EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_TERMINARNI),TRUE);

     iErrorDeSQL = iEjecutaLaInstruccionSQL("select cCategoria, vcDescCatego from CCategorias");
     if(iErrorDeSQL)
     {
       MessageBox(NULL, "Error al recuperar datos de CCategoria", "ERROR DE CONEXION",
MB_ICONSTOP);
       return FALSE;
     }

     iNumCategorias = 0;
     while ((return_code = dbresults(dbproc)) != NO_MORE_RESULTS)
       if (return_code = = SUCCEED)
       {
         dbbind(dbproc,1,NTBSTRINGBIND,(DBINT)0,(char unsigned far *)VarDePasoClave);
         dbbind(dbproc,2,NTBSTRINGBIND,(DBINT)0,(char unsigned far *)VarDePasoDesc);

       for(; dbnextrow(dbproc) != NO_MORE_ROWS && iNumCategorias <
       miNumMaximoDeCategorias;
         iNumCategorias + + )
```

```
    {
        VarCatalogoCategorias = realloc(VarCatalogoCategorias,(iNumCategorias+1)*sizeof(struct
              CatalogoCategorias));
        strcpy((char*)VarCatalogoCategorias[iNumCategorias].acClave,(char*)VarDePasoClave);
        strcpy((char *)VarCatalogoCategorias[iNumCategorias].acDescCategoria,(char
        * ) V a r D e P a s o D e s c ) ;
        SendDlgItemMessage(WhManejadorDeLaVentana,IDC_CATEGORIANI,CB_ADDSTRING,
        0,(long)(char
        *)VarDePasoDesc);
    }
  }

// ******** Lectura de la BD Estados Civiles *********
iNumEstadosCiviles = 0;
 iErrorDeSQL = iEjecutaLaInstruccionSQL("select cEstadoCivil, vcDescEdoCivil from
 CEstadosCiviles");
 if(iErrorDeSQL)
 {
  MessageBox(NULL, "Error al recuperar datos de CEstadosCiviles", "ERROR DE CONEXION",
 MB_ICONSTOP);
 return FALSE;
 }

while ((return_code = dbresults(dbproc)) != NO_MORE_RESULTS)
 if (return_code == SUCCEED)
  {
       if ((dbbind(dbproc,1,NTBSTRINGBIND,(DBINT)0,(char unsigned far *)VarDePasoClave))
       == FAIL)
      return FALSE;
       if((dbbind(dbproc,2,NTBSTRINGBIND,(DBINT)0,(charunsignedfar *)VarDePasoDesc)) ==
       FAIL)
      return FALSE;
```

```
    for(; dbnextrow(dbproc) != NO_MORE_ROWS && iNumEstadosCiviles <
     miNumMaximoDeEstadosCiviles; iNumEstadosCiviles++)
      {
       VarCatalogoEstadosCiviles =realloc(VarCatalogoEstadosCiviles, (iNumEstadosCiviles+1)
       * sizeof(struct CatalogoEstadosCiviles));
       strcpy((char    *)VarCatalogoEstadosCiviles[iNumEstadosCiviles].acClave,(char
       *)VarDePasoClave);
       strcpy((char *)VarCatalogoEstadosCiviles[iNumEstadosCiviles].acDescEstadoCivil,(char
        *)VarDePasoDesc);

       SendDlgItemMessage(WhManejadorDeLaVentana,IDC_ESTADOCIVILNI,CB_ADDSTRING
       ,0,(long)
        (char *)VarDePasoDesc);
      }
  }

// ******** Lectura de la BD Nacionalidades *********
iNumNacionalidades = 0;
 iErrorDeSQL  =  iEjecutaLaInstruccionSQL("select  cNacionalidad,  vcDescNacional  from
 CNacionalidad");
if(iErrorDeSQL)
 {
 MessageBox(NULL, "Error al recuperar datos de CNacionalidad", "ERROR DE CONEXION",
 MB_ICONSTOP);
 return FALSE;
 }

while ((return_code = dbresults(dbproc)) != NO_MORE_RESULTS)
 if (return_code = = SUCCEED)
  {
   if ((dbbind(dbproc,1,STRINGBIND,(DBINT)0,(char unsigned far *)VarDePasoClave)) = = FAIL)
```

```
        return FALSE;
         if((dbbind(dbproc,2,NTBSTRINGBIND,(DBINT)0,(charunsignedfar*)VarDePasoDesc)) = =
         FAIL)
        return FALSE;

     for(; dbnextrow(dbproc) != NO_MORE_ROWS && iNumNacionalidades <
      miNumMaximoDeNacionalidades; iNumNacionalidades + +)
        {
         VarCatalogoNacionalidades =realloc (VarCatalogoNacionalidades, (iNumNacionalidades+1)
         * sizeof(struct CatalogoNacionalidades));
         strcpy((char  *)VarCatalogoNacionalidades[iNumNacionalidades].acClave,(char
         *)VarDePasoClave);
         strcpy((char*)VarCatalogoNacionalidades[iNumNacionalidades].acDescNacionalidad,(char
          *)VarDePasoDesc);
         SendDlgItemMessage(WhManejadorDeLaVentana,IDC_NACIONALIDADNI,CB_ADDSTRI
         NG,0,(long)(char *)VarDePasoDesc);
        }
   }

// ******** Lectura de BD Escolaridad **********
iNumEscolaridades=0;
iErrorDeSQL = iEjecutaLaInstruccionSQL("select cClaveEscol, vcDescEscol from CEscolaridad");
if(iErrorDeSQL)
 {
 MessageBox(NULL, "Error al recuperar datos de CEscolaridad", "ERROR DE CONEXION",
 MB_ICONSTOP);
  return FALSE;
 }

while ((return_code = dbresults(dbproc)) != NO_MORE_RESULTS)
 if (return_code = = SUCCEED)
  {
```

```
        if((dbbind(dbproc,1,NTBSTRINGBIND,(DBINT)0,(charunsigned far *)VarDePasoClave)) == FAIL)
              return FALSE;
        if((dbbind(dbproc,2,NTBSTRINGBIND,(DBINT)0,(char unsigned far *)VarDePasoDesc)) == FAIL)
              return FALSE;

        for(;dbnextrow(dbproc) != NO_MORE_ROWS && iNumEscolaridades <
        miNumMaximoDeEscolaridades;
            iNumEscolaridades++)
              {
               VarCatalogoEscolaridades = realloc (VarCatalogoEscolaridades, (iNumEscolaridades+1)
               *sizeof(struct CatalogoEscolaridades));
               strcpy((char  *)VarCatalogoEscolaridades[iNumEscolaridades].acClave,(char
               *)VarDePasoClave);
               strcpy((char *)VarCatalogoEscolaridades[iNumEscolaridades].acDescEscolaridad,(char
                *)VarDePasoDesc);

               SendDlgItemMessage(WhManejadorDeLaVentana,IDC_ESCOLARIDADNI,CB_ADDSTRIN
               G,0,(long )(char *)VarDePasoDesc);
              }
         }

      iNumDependencias = 0;
      iErrorDeSQL = iEjecutaLaInstruccionSQL("select cDependencia, cSubDep, vcNombreDep from
CDependencias
       where cDependencia = '413' ");
      if(iErrorDeSQL)
      {
       MessageBox(NULL, "Error al recuperar datos de CDependencias", "ERROR DE CONEXION",
       MB_ICONSTOP);
       return FALSE;
      }
```

```
  while ((return_code = dbresults(dbproc)) != NO_MORE_RESULTS)
    if (return_code == SUCCEED)
      {
        dbbind(dbproc,1,NTBSTRINGBIND,(DBINT)0,(char unsigned far *)VarDePasoClave);
        dbbind(dbproc,2,NTBSTRINGBIND,(DBINT)0,(char unsigned far *)VarDePasoAux);
        dbbind(dbproc,3,NTBSTRINGBIND,(DBINT)0,(char unsigned far *)VarDePasoDesc);
            for(;dbnextrow(dbproc) != NO_MORE_ROWS && iNumDependencias <
            miNumMaximoDeDependencias; iNumDependencias++)
           {
            VarCatalogoDependencias = realloc (VarCatalogoDependencias, (iNumDependencias+1)
            * sizeof(struct CatalogoDependencias));
            strcpy((char *)VarCatalogoDependencias[iNumDependencias].acClaveDep,(char
            *)VarDePasoClave);
            strcpy((char *)VarCatalogoDependencias[iNumDependencias].acClaveSub,(char
            *)VarDePasoAux);
            strcpy((char *)VarCatalogoDependencias[iNumDependencias].acDescDependencia,(char
             *)VarDePasoDesc);

            SendDlgItemMessage(WhManejadorDeLaVentana,IDC_DEPENDENCIANI,CB_ADDSTRIN
            G,0,(long )(char *)VarDePasoDesc);
           }
      }
  return TRUE; // Fin del WM_INITDIALOG


case WM_COMMAND:
  switch(wParam)
  {
    case IDC_DESHABILITA_COMBO_CATEGO:
        dwSeleccionCategoria = -1;
        dwSeleccionPlaza = -1;
        dwSeleccionNacionalidad = -1;
```

```
            dwSeleccionEscolaridad = -1;
            dwSeleccionDependencia = -1;
            dwSeleccionEstadoCivil = -1;

            // limpia los campos edit y  check
            SetDlgItemText(WhManejadorDeLaVentana,IDC_RFCNI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_NOMBRENI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_APELLIDOPATERNONI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_APELLIDOMATERNONI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_FECHANACIMIENTONI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_DOMICILIONI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_TELEFONONI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_SUELDONI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_HORASNI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_FECHASOLICITUDNI,"");
            SetDlgItemText(WhManejadorDeLaVentana,IDC_FECHAINGRESONI,"");

            // los campos combo box
            SendDlgItemMessage(WhManejadorDeLaVentana, IDC_CATEGORIANI, CB_SETCURSEL, -1,
0L);
            SendDlgItemMessage(WhManejadorDeLaVentana,IDC_ESCOLARIDADNI,CB_SETCURSEL,-1,
0L);
            SendDlgItemMessage(WhManejadorDeLaVentana,IDC_ESTADOCIVILNI,CB_SETCURSEL,-1,
0L);
            SendDlgItemMessage(WhManejadorDeLaVentana,IDC_DEPENDENCIANI,CB_SETCURSEL,-1,
0L);
            SendDlgItemMessage(WhManejadorDeLaVentana, IDC_NACIONALIDADNI, CB_SETCURSEL,
-1, 0L);

            // los campos check
           SendDlgItemMessage(WhManejadorDeLaVentana,IDC_ACTANACNI,BM_SETCHECK,FALSE,0L);
```

```
SendDlgItemMessage(WhManejadorDeLaVentana,IDC_PERMISOSRENI,BM_SETCHECK,FALSE,0L);
        SendDlgItemMessage(WhManejadorDeLaVentana,IDC_ESTUDIOSNI,BM_SETCHECK,
FALSE,0L);

        // limpia la combo box de Plazas
        SendDlgItemMessage(WhManejadorDeLaVentana, IDC_NUMPLAZANI, CB_RESETCONTENT,
0, 0L);

        // deshabilita los botones correspondientes
        EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_VERIFICANI),TRUE);
        EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_ACEPTARNI),FALSE);
        EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_CANCELARNI),FALSE);
        SetFocus(GetDlgItem(WhManejadorDeLaVentana,IDC_RFCNI));
        return 0L;

      // Se presiona el botón VERIFICAR
      case IDC_VERIFICANI:
        //  Primero ejecuta un select a trabajador para verificar que
        //  No exista su registro en la base de datos
        //  Si existe lo rechaza, notificando al usuario el "reingreso"
        //  Si no existe permite capturar los demás campos
        //  EndDialog (WhManejadorDeLaVentana, TRUE);
        GetDlgItemText(WhManejadorDeLaVentana,IDC_RFCNI,acRFCUSU,14);
        wsprintf(acInstruccion,"select cRfcTrab"
                                        " from CTrabajadores"
                                        " where cRfcTrab = '%s'", acRFCUSU);
        iErrorDeSQL = iEjecutaLaInstruccionSQL(acInstruccion);
        if(iErrorDeSQL)
          return;

        while ((return_code = dbresults(dbproc)) != NO_MORE_RESULTS)
          if (return_code == SUCCEED)
```

```
{
  dbbind(dbproc,1,STRINGBIND,(DBINT)0,(char unsigned far *)acRFCDB);
  while (dbnextrow(dbproc) != NO_MORE_ROWS);

  if (strcmp(acRFCUSU,acRFCDB) == 0)
   {
     MessageBeep(-1);
   MessageBox(NULL, "El RFC del trabajador ya existe en la base de datos,\nprobablemente
   deba tramitar REINGRESO u OTRO NOMBRAMIENTO", "RFC existente!", MB_ICONSTOP);
     EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_CANCELARNI),TRUE);
     EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_VERIFICANI),FALSE);
   }
   else
   {
     EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_VERIFICANI),FALSE);
     EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_ACEPTARNI),TRUE);
     EnableWindow(GetDlgItem(WhManejadorDeLaVentana,IDC_CANCELARNI),TRUE);
     SetFocus(GetDlgItem(WhManejadorDeLaVentana,IDC_APELLIDOPATERNONI));
   }
}
i=0;
return 0L;


case IDC_CATEGORIANI:
    if(HIWORD(lParam) == CBN_SELCHANGE)
    {
      /* Se deja en la variable 'dwSeleccionCategoría' la opción seleccionada. */
      dwSeleccionCategoria = SendDlgItemMessage(WhManejadorDeLaVentana,
       IDC_CATEGORIANI, CB_GETCURSEL, 0, 0L);

SendDlgItemMessage(WhManejadorDeLaVentana,IDC_NUMPLAZANI,CB_RESETCONTENT,0,
```

```
                VarPlazasVacantes[i].acNumPlaza);

SendDlgItemMessage(WhManejadorDeLaVentana,IDC_NUMPLAZANI,CB_RESETCONTENT,0,
                0L);
                iNumPlazasVacantes = 0;

                wsprintf(acInstruccion,"select cNumPlaza, cHorasPlaza, fSueldoPlaza"
                                       " from CPlazas "
                                        " where cCategoria = '%s' and "
                                    " cEstatusPlaza = 'Vacante'",
                                     VarCatalogoCategorias [dwSeleccionCategoria].acClave);

                iErrorDeSQL = iEjecutaLaInstruccionSQL(acInstruccion);
                if(iErrorDeSQL)
                  return;

                while ((return_code = dbresults(dbproc)) != NO_MORE_RESULTS)
                  if (return_code == SUCCEED)
                  {
                      if  ((dbbind(dbproc,1,NTBSTRINGBIND,(DBINT)0,(char  unsigned  far
*)acTempNumPlaza)) == FAIL)
                    return;
                      if  ((dbbind(dbproc,2,NTBSTRINGBIND,(DBINT)0,(char  unsigned  far
*)acTempHoras)) == FAIL)
                    return;
                    if ((dbbind(dbproc,3,FLT4BIND,(DBINT)0,&afTempSueldo)) == FAIL)
                    return;

                    for(; dbnextrow(dbproc) != NO_MORE_ROWS; iNumPlazasVacantes++)
                     {
                    VarPlazasVacantes =
                    realloc(VarPlazasVacantes,(iNumPlazasVacantes+1)*sizeof(struct  PlazasVacantes));
```

```
            strcpy((char   *)   VarPlazasVacantes[iNumPlazasVacantes].acNumPlaza,
            (char*)acTempNumPlaza);
            strcpy((char     *)VarPlazasVacantes[iNumPlazasVacantes].acHoras,(char
            *)acTempHoras);
            wsprintf(VarPlazasVacantes[iNumPlazasVacantes].acSueldo,"N$
              %6.2ld",afTempSueldo);
            SendDlgItemMessage(WhManejadorDeLaVentana,IDC_NUMPLAZANI,CB_ADDST
            RING,0,VarPlazasVacantes[iNumPlazasVacantes].acNumPlaza);
            }

          if (iNumPlazasVacantes = =0)
            {
            MessageBeep(-1);
            MessageBox(NULL,"Por el momento no existen plazas vacantes\npara la categoría
             seleccionada", "No hay vacantes.", MB_ICONSTOP);
            return;
            }
          } // fin del if(return_code = =SUCCEED)
    } // fin del if(HIWORD..
return(TRUE);

case IDC_NUMPLAZANI: // implementar con doble click del mouse
    if(HIWORD(lParam) = = CBN_SELCHANGE)
    {
       /* Se deja en la variable 'dwSeleccionPlaza' la opción seleccionada. */
     dwSeleccionPlaza = SendDlgItemMessage(WhManejadorDeLaVentana, IDC_NUMPLAZANI,
        CB_GETCURSEL, 0, 0L);

       /* Las horas de la plaza  seleccionada se escribe sobre la ventana de */
       /* edición destinada para ello. */

    SetDlgItemText(WhManejadorDeLaVentana,IDC_HORASNI,VarPlazasVacantes[dwSeleccionPlaza]
```

```
        .acHoras);

        /* El sueldo de la plaza  seleccionada se escribe sobre la ventana de */
        /* edición destinada para ello. */

     SetDlgItemText(WhManejadorDeLaVentana,IDC_SUELDONI,VarPlazasVacantes[dwSelec
     cionPlaza].acSueldo);
     }
    return 0L;

case IDC_ESTADOCIVILNI: // implementar con doble click del mouse
    if(HIWORD(lParam) == CBN_SELCHANGE)
    {
        /* Se deja en la variable 'dwSeleccionEstadoCivil' la opción seleccionada. */
        dwSeleccionEstadoCivil = SendDlgItemMessage(WhManejadorDeLaVentana,
         IDC_ESTADOCIVILNI, CB_GETCURSEL, 0, 0L);
    }
    return 0L;

case IDC_ESCOLARIDADNI: // implementar con doble click del mouse
    if(HIWORD(lParam) == CBN_SELCHANGE)
    {
        /* Se deja en la variable 'dwSeleccionEscolaridad' la opción seleccionada. */
        dwSeleccionEscolaridad = SendDlgItemMessage(WhManejadorDeLaVentana,
         IDC_ESCOLARIDADNI, CB_GETCURSEL, 0, 0L);
    }
    return 0L;

case IDC_DEPENDENCIANI: // implementar con doble click del mouse
    if(HIWORD(lParam) == CBN_SELCHANGE)
    {
        /* Se deja en la variable 'dwSeleccionDependencia' la opción seleccionada. */
```

```
            dwSeleccionDependencia = SendDlgItemMessage(WhManejadorDeLaVentana,
             IDC_DEPENDENCIANI, CB_GETCURSEL, 0, 0L);
          }
          return 0L;

        case IDC_NACIONALIDADNI: // implementar con doble click del mouse
          if(HIWORD(lParam) == CBN_SELCHANGE)
          {
            /* Se deja en la variable 'dwSeleccionNacionalidad' la opción seleccionada. */
            dwSeleccionNacionalidad = SendDlgItemMessage(WhManejadorDeLaVentana,
             IDC_NACIONALIDADNI, CB_GETCURSEL, 0, 0L);
          }
          return 0L;

    // Se presiona el botón ACEPTAR
    case IDC_ACEPTARNI:
      GetDlgItemText(WhManejadorDeLaVentana,IDC_APELLIDOPATERNONI,acApellidoPaterno,7);
      GetDlgItemText(WhManejadorDeLaVentana,IDC_APELLIDOMATERNONI,acApellidoMaterno,7);
      GetDlgItemText(WhManejadorDeLaVentana,IDC_NOMBRENI,acNombre,7);
      GetDlgItemText(WhManejadorDeLaVentana,IDC_FECHANACIMIENTONI,acdNacimiento,9);
      GetDlgItemText(WhManejadorDeLaVentana,IDC_TELEFONONI,acTelefono,13);
      GetDlgItemText(WhManejadorDeLaVentana,IDC_FECHASOLICITUDNI,acdSolicitud,9);
      GetDlgItemText(WhManejadorDeLaVentana,IDC_FECHAINGRESONI,acdIngreso,9);
      GetDlgItemText(WhManejadorDeLaVentana,IDC_EXPERIENCIANI,acExperiencia,3);
        ActaNacim = SendDlgItemMessage(WhManejadorDeLaVentana, IDC_ACTANACNI,
        BM_GETCHECK,0,0L);
      PermisoSRE = SendDlgItemMessage(WhManejadorDeLaVentana,
       IDC_PERMISOSRENI,BM_GETCHECK,0,0L);
        Estudios = SendDlgItemMessage(WhManejadorDeLaVentana, IDC_ESTUDIOSNI,
        BM_GETCHECK, 0,0L);
      GetDlgItemText(WhManejadorDeLaVentana,IDC_DOMICILIONI,acDomicilio,100);
      if (!ActaNacim)
```

```
            {
             MessageBeep(-1);
             MessageBox(NULL, "Falta documento de acta de nacimiento,\nel trámite no puede
             procesarse.",
              "Documentos Faltantes", MB_ICONSTOP);
             return;
            }

          if (!Estudios)
             {
              MessageBeep(-1);
              MessageBox(NULL, "Faltan documentos de escolaridad,\nel trámite no puede procesarse.",
               "Documentos Faltantes", MB_ICONSTOP);
              return;
             }

                                                                                    i      f
(strcmp(VarCatalogoNacionalidades[dwSeleccionNacionalidad].acDescNacionalidad,"Mexicana")!=0)
            {
             if(!PermisoSRE)
                {
                MessageBeep(-1);
                MessageBox(NULL, "Falta el permiso de la S.R.E.,\nel trámite no puede procesarse.",
                 "Documentos Faltantes", MB_ICONSTOP);
                return;
                }
            }

          //aqui se hace un INSERT a la BD en CTrabajadores con la información
          //primero se pone en una sola variable el nombre completo
       wsprintf(NombreCompleto,"%s   %s   %s",(char   *)acApellidoPaterno,(char
       *)acApellidoMaterno,(char
```

```
    *)acNombre);
wsprintf(acInstruccion,"insert into CTrabajadores (cRfcTrab, vcNombreTrab, dNacimientoTrab,
cNacionalidad, cClaveEscol,cEstadoCivil,vcDomicilioTrab, cEstatus) "
       " values ('%s','%s','%s','%s','%s','%s','%s','Vigente')",
          (char *)acRFCUSU,
          (char *)NombreCompleto,
          (char *)acdNacimiento,
          (char *)VarCatalogoNacionalidades[dwSeleccionNacionalidad].acClave,
          (char *)VarCatalogoEscolaridades[dwSeleccionEscolaridad].acClave,
          (char *)VarCatalogoEstadosCiviles[dwSeleccionEstadoCivil].acClave,
          (char *)acDomicilio);
   // Realiza el insert a la base de datos...
   iErrorDeSQL = iEjecutaLaInstruccionSQL(acInstruccion);
   if(iErrorDeSQL)
     {
MessageBox(NULL, "Error al ejecutar el Insert en CTrabajadores", "ERROR.", MB_ICONSTOP);
     return;
     }

// obtiene el último registro del trabajador en THistoria
 wsprintf(acInstruccion,"select iConsecutivoHist"
                                        " from THistoria"
                                        " where cRfcTrab = '%s'",(char *)acRFCUSU);
 iErrorDeSQL = iEjecutaLaInstruccionSQL(acInstruccion);
 if(iErrorDeSQL)
  return;
 aiConsTHistoria = 0;
 while ((return_code = dbresults(dbproc)) != NO_MORE_RESULTS)
  if (return_code == SUCCEED)
  {
    dbbind(dbproc,1,INTBIND,(DBINT)0,&aiConsTHistoria);
    for (;dbnextrow(dbproc) != NO_MORE_ROWS;);
```

```
        }
        // se construye la instrucción para el Insert en THistoria
        wsprintf(acInstruccion,"insert into THistoria (iConsecutivoHist,cRfcTrab,cDepHist,cNumPlaza,
         cCategoHist,dInicioHist) "
              "values (%d,'%s','%s','%s','%s','%s')",
                  (int)(aiConsTHistoria + 1),
                  (char *)acRFCUSU,
                  (char *)VarCatalogoDependencias[dwSeleccionDependencia].acClaveDep,
                  (char *)VarPlazasVacantes[dwSeleccionPlaza].acNumPlaza,
                  (char *)VarCatalogoCategorias[dwSeleccionCategoria].acClave,
                  (char *)acdIngreso);

        //  Realiza el insert a la base de datos en THistoria.
        iErrorDeSQL = iEjecutaLaInstruccionSQL(acInstruccion);
        if(iErrorDeSQL)
          {
          MessageBox(NULL, "Error al ejecutar el Insert en THistoria", "ERROR.", MB_ICONSTOP);
          return;
          }

        //****************************
    // Falta hacer un UPDATE a la BD CPlazas para modificar el estatus de la plaza.
          wsprintf(acInstruccion,"update CPlazas set cEstatusPlaza = 'Ocupada' "
              " where cNumPlaza = '%s'",VarPlazasVacantes[dwSeleccionPlaza].acNumPlaza );
          iErrorDeSQL = iEjecutaLaInstruccionSQL(acInstruccion);
          if(iErrorDeSQL)
           {
           MessageBox(NULL, "Error al ejecutar el Update", "ERROR de instrucci¾nSQL.",
MB_ICONSTOP);
           return;
           }
```

```
        // después envía un mensaje de registro grabado y limpia los campos.
      MessageBox(NULL, "Registro grabado en la base de datos\nNuevo Ingreso concluído.", "Registro
      de datos",  MB_ICONASTERISK);

      SendMessage(WhManejadorDeLaVentana,WM_COMMAND,IDC_DESHABILITA_COMBO_CATE
      GO,0L);
         return(TRUE);

      // Se presiona el botón CANCELAR o la tecla ESC
      case IDC_CANCELARNI:
        // Se deberán poner en blanco los campos de captura excepto
        // los que se tomaron de la BD (categoría, plaza y nivel
        // lo hace a través de DESHABILITA_COMBO_CATEGO

      SendMessage(WhManejadorDeLaVentana,WM_COMMAND,IDC_DESHABILITA_COMBO_CATE
      GO,0L);
        return 0L;

      // Se presiona el botón TERMNAR
      case IDC_TERMINARNI:
        // se cierra el diálogo
        EndDialog(WhManejadorDeLaVentana, FALSE);
        return 0L;

     } // Final del switch
     break;

  default:
     return (FALSE);

} // Fin del switch iMensaje
return(TRUE);
```

```
}

/*****************************************************************************
Archivo de cabecera:

#define DBMSWIN
#include <windows.h>
#include <string.h>
#include <SQLfront.H>
#include <SQLdb.H>
#include <stdio.h>
#include <stdlib.h>


/* definición de la estrucutra de datos para almacenar las plazas vacantes */

#define miTamNumPlaza 12
#define miTamHoras    8
#define miTamSueldo   18
#define miNumMaximoPlazasVacantes 30
struct PlazasVacantes
{
 DBCHAR  acNumPlaza[miTamNumPlaza + 1];
 DBCHAR  acHoras[miTamHoras + 1];
 DBCHAR  acSueldo[miTamSueldo + 1];
};

/* definición de la estrucutra de datos para almacenar el catálogo de categorías */
/*#define miTamCveCategoria   6
#define miTamCategoria    100
#define miTamSueldoCatego  9 */
#define miNumMaximoDeCategorias 50
```

```
#define miTamCveCategoria   12
#define miTamCategoria    150
#define miTamSueldoCatego  18
struct CatalogoCategorias
{
 DBCHAR acClave[miTamCveCategoria + 1];
 DBCHAR acDescCategoria[miTamCategoria + 1];
 DBCHAR acSueldoCat[miTamSueldoCatego + 1];
};

/* definición de la estrucutra de datos para almacenar el catálogo de estados civiles */
/*#define miTamCveEdoCivil  2
#define miTamEdoCivil    12 */
#define miNumMaximoDeEstadosCiviles 6
#define miTamCveEdoCivil  4
#define miTamEdoCivil    24
struct CatalogoEstadosCiviles
{
 DBCHAR acClave[miTamCveEdoCivil + 1];
 DBCHAR acDescEstadoCivil[miTamEdoCivil + 1];
};

/* definición de la estrucutra de datos para almacenar el catálogo de Nacionalidades */
/*#define miTamCveNacionalidad  2
#define miTamNacionalidad    12 */
#define miNumMaximoDeNacionalidades 10
#define miTamCveNacionalidad  4
#define miTamNacionalidad    24
struct CatalogoNacionalidades
{
 DBCHAR acClave[miTamCveNacionalidad + 1];
 DBCHAR acDescNacionalidad[miTamNacionalidad + 1];
```

```
};

/* definición de la estrucutra de datos para almacenar el catálogo de Escolaridades */
/*#define miTamCveEscolaridad 2
#define miTamEscolaridad   18 */
#define miNumMaximoDeEscolaridades 20
#define miTamCveEscolaridad 4
#define miTamEscolaridad   36
struct CatalogoEscolaridades
{
 DBCHAR acClave[miTamCveEscolaridad + 1];
 DBCHAR acDescEscolaridad[miTamEscolaridad + 1];
};

/* definición de la estrucutra de datos para almacenar el catálogo de Dependencias */
#define miNumMaximoDeDependencias 5
#define miTamCveDependencia     6
#define miTamCveSubDependencia  4
#define miTamDependencia       150
struct CatalogoDependencias
{
 DBCHAR acClaveDep[miTamCveDependencia + 1];
 DBCHAR acClaveSub[miTamCveSubDependencia + 1];
 DBCHAR acDescDependencia[miTamDependencia + 1];
};

extern struct PlazasVacantes          *VarPlazasVacantes;
extern struct CatalogoCategorias      *VarCatalogoCategorias;
extern struct CatalogoEstadosCiviles  *VarCatalogoEstadosCiviles;
extern struct CatalogoNacionalidades  *VarCatalogoNacionalidades;
extern struct CatalogoEscolaridades   *VarCatalogoEscolaridades;
extern struct CatalogoDependencias    *VarCatalogoDependencias;
```

```
//extern struct CatalogoEscolaridades  VarCatalogoEscolaridades[miNumMaximoDeEscolaridades];

extern int  iNumPlazasVacantes,
       iNumCategorias,
       iNumEstadosCiviles,
       iNumNacionalidades,
       iNumEscolaridades,
       iNumDependencias;


extern LOGINREC  *login;
extern DBPROCESS *dbproc;
extern DBCHAR acRFC[];
extern DBCHAR acCategoria[];
extern DBCHAR acSueldo[];
extern DBCHAR acDependencia[];

int iConectarseALaBaseDeDatos(char *, char *, char *, char *, char *);
int iEjecutaLaInstruccionSQL(char *);

BOOL FAR PASCAL ConsultaRfc(HWND , UINT , WORD , DWORD);

BOOL FAR PASCAL NuevoIngreso(HWND , UINT , WORD , DWORD);

BOOL FAR PASCAL Reingreso(HWND, UINT, WORD, DWORD );

BOOL FAR PASCAL OtroNombramiento(HWND, UINT, WORD, DWORD );

BOOL FAR PASCAL Reclasificacion(HWND, UINT, WORD, DWORD );

BOOL FAR PASCAL Licencia(HWND, UINT, WORD, DWORD );
```

```
BOOL FAR PASCAL Baja(HWND, UINT, WORD, DWORD );

BOOL FAR PASCAL ConsultaPlaza(HWND, UINT, WORD, DWORD );

BOOL FAR PASCAL ConsultaCategoria(HWND, UINT, WORD, DWORD );

BOOL FAR PASCAL ConsultaPresupuesto(HWND, UINT, WORD, DWORD );

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

// Definiciones para el dialogo 'Nuevo Ingreso'
 // Definiciones para los campos de captura
 #define IDC_APELLIDOPATERNONI          301
 #define IDC_APELLIDOMATERNONI          302
 #define IDC_NOMBRENI                   303
 #define IDC_FECHANACIMIENTONI          304
 #define IDC_ESCOLARIDADNI               305
 #define IDC_ESTADOCIVILNI              306
 #define IDC_DOMICILIONI                307
 #define IDC_TELEFONONI                 308
 #define IDC_DEPENDENCIANI              309
 #define IDC_NUMPLAZANI                 310
 #define IDC_CATEGORIANI                311
 #define IDC_SUELDONI                   312
 #define IDC_HORASNI                    313
 #define IDC_RFCNI                      314
 #define IDC_FECHASOLICITUDNI           315
 #define IDC_FECHAINGRESONI             316
 #define IDC_NACIONALIDADNI             317
 #define IDC_EXPERIENCIANI              318
```

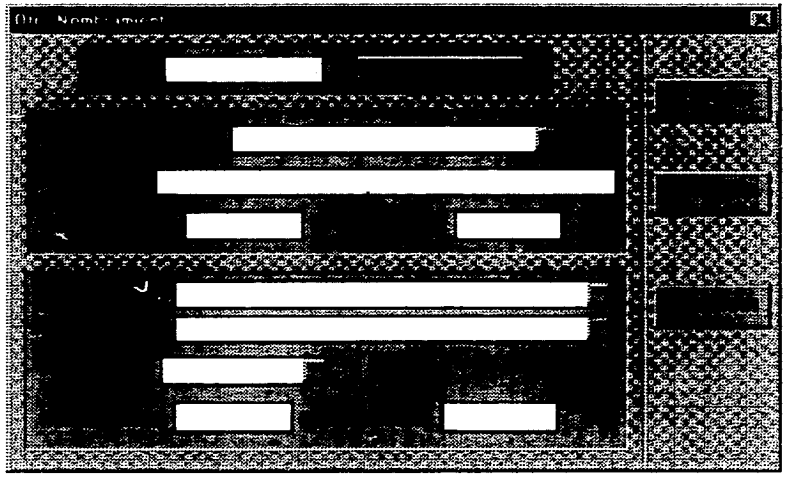
```
// Definiciones para los botones
#define IDC_ACEPTARNI                320
#define IDC_CANCELARNI               321
#define IDC_TERMINARNI               322
#define IDC_VERIFICARNI              323



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

Código del diálogo de Nuevo Ingreso
#include "tesis1.rh"


DialogoNuevoIngreso DIALOG 16, 17, 371, 296
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION | WS_SYSMENU
CLASS "bordlg"
CAPTION "Nuevo ingreso"
FONT 8, "MS Sans Serif"
{
 CONTROL "", -1, "BorShade", BSS_VDIP | BSS_LEFT | WS_CHILD | WS_VISIBLE, 325, 3, 1, 290
 LTEXT "Apellido paterno:", -1, 7, 41, 54, 10
 EDITTEXT IDC_APELLIDOPATERNONI, 66, 37, 190, 13, ES_AUTOHSCROLL | ES_UPPERCASE |
WS_BORDER |
 WS_TABSTOP
 LTEXT "Apellido materno:", -1, 5, 55, 56, 13
 EDITTEXT IDC_APELLIDOMATERNONI, 66, 53, 190, 13, ES_AUTOHSCROLL | ES_UPPERCASE |
WS_BORDER |
 WS_TABSTOP
 LTEXT "Nombre(s):", -1, 24, 70, 37, 11
 EDITTEXT IDC_NOMBRENI, 66, 69, 190, 13, ES_AUTOHSCROLL | ES_UPPERCASE | WS_BORDER
```

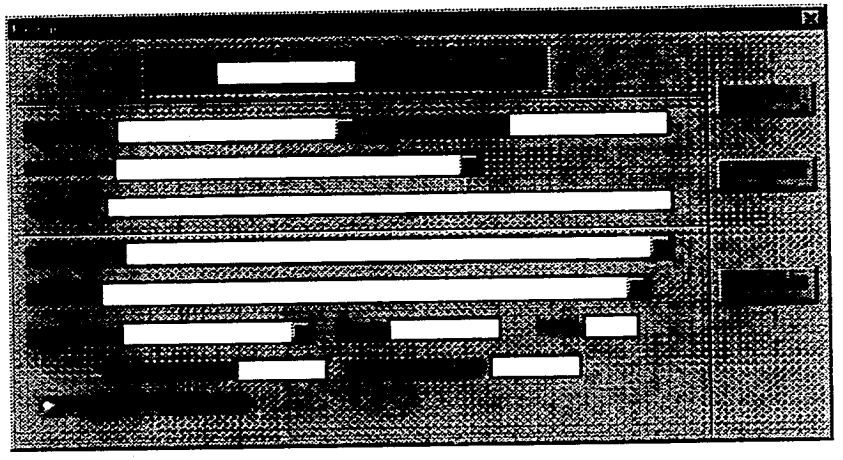
```
|
 WS_TABSTOP
  RTEXT "Fecha de nacimiento:", -1, 1, 89, 68, 12
  EDITTEXT IDC_FECHANACIMIENTONI, 74, 87, 51, 12
  LTEXT "Escolaridad:", -1, 3, 104, 42, 10
  COMBOBOX IDC_ESCOLARIDADNI, 44, 103, 164, 52, CBS_DROPDOWNLIST | WS_TABSTOP
  LTEXT "Estado civil:", -1, 152, 121, 39, 9
  COMBOBOX IDC_ESTADOCIVILNI, 193, 120, 110, 37, CBS_DROPDOWNLIST | WS_TABSTOP
  RTEXT "Domicilio:", -1, 7, 138, 31, 10
  EDITTEXT IDC_DOMICILIONI, 42, 136, 261, 12, ES_AUTOHSCROLL | WS_BORDER | WS_TABSTOP
  LTEXT "Teléfono:", -1, 18, 120, 32, 10
  EDITTEXT IDC_TELEFONONI, 50, 119, 53, 11
  LTEXT "Dependencia:", -1, 3, 164, 45, 11
  COMBOBOX IDC_DEPENDENCIANI, 49, 163, 261, 37, CBS_DROPDOWNLIST | WS_TABSTOP
  RTEXT "Plaza No:", -1, 220, 184, 31, 10
  LTEXT "Categoría:", -1, 2, 183, 37, 9
  LTEXT "Sueldo:", -1, 2, 204, 25, 8
  EDITTEXT IDC_SUELDONI, 27, 202, 51, 12, ES_RIGHT | ES_AUTOVSCROLL | WS_BORDER |
WS_TABSTOP
  LTEXT "Horas:", -1, 83, 204, 22, 9
  EDITTEXT IDC_HORASNI, 104, 201, 25, 13, ES_RIGHT | WS_BORDER | WS_TABSTOP
  LTEXT "RFC:", -1, 71, 10, 16, 9
  EDITTEXT IDC_RFCNI, 89, 9, 65, 12, ES_UPPERCASE | WS_BORDER | WS_TABSTOP
  CONTROL "", -1, "BorShade", BSS_HDIP | BSS_LEFT | WS_CHILD | WS_VISIBLE, 2, 153, 324, 2
  LTEXT "Fecha solicitud:", -1, 134, 203, 52, 9
  EDITTEXT IDC_FECHASOLICITUDNI, 183, 202, 42, 12
  LTEXT "Fecha Ingreso:", -1, 231, 204, 49, 8
  EDITTEXT IDC_FECHAINGRESONI, 279, 203, 42, 12
  PUSHBUTTON "ACEPTAR", IDC_ACEPTARNI, 327, 21, 41, 19
  PUSHBUTTON "CANCELAR", IDC_CANCELARNI, 327, 58, 41, 20
  PUSHBUTTON "TERMINAR", IDC_TERMINARNI, 327, 190, 41, 20
  LTEXT "Nacionalidad:", -1, 136, 89, 45, 8
```

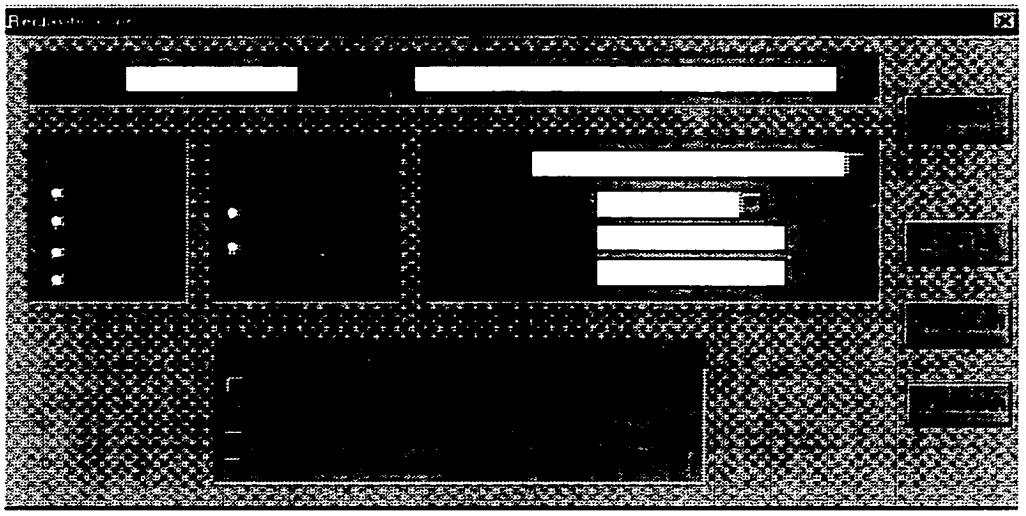
```
 COMBOBOX IDC_NACIONALIDADNI, 181, 88, 123, 39, CBS_DROPDOWNLIST | WS_TABSTOP
 LTEXT "Experiencia:", -1, 216, 105, 43, 9
 EDITTEXT IDC_EXPERIENCIANI, 259, 104, 24, 12
 LTEXT "años", -1, 285, 106, 19, 9
 CONTROL "", -1, "BorShade", BSS_HDIP | BSS_LEFT | WS_CHILD | WS_VISIBLE, 1, 223, 324, 3
 LTEXT "Requisitos Documentales:", -1, 14, 229, 84, 8
 CONTROL "Permiso de la Secretaría de Relaciones Exteriores", IDC_PERMISOSRENI, "BorCheck",
BS_AUTOCHECKBOX | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 84, 258, 167, 10
 CONTROL "Comprobante de estudios o equivalencia", IDC_ESTUDIOSNI, "BorCheck",
BS_AUTOCHECKBOX |
WS_CHILD | WS_VISIBLE | WS_TABSTOP, 84, 269, 165, 10
 CONTROL "", -1, "BorShade", BSS_GROUP | BSS_CAPTION | BSS_LEFT | WS_CHILD | WS_VISIBLE,
81, 245, 173,
39
 COMBOBOX IDC_CATEGORIANI, 35, 182, 181, 38, CBS_DROPDOWNLIST | WS_TABSTOP
 CONTROL "123456", IDC_NUMPLAZANI, "COMBOBOX", CBS_DROPDOWNLIST | WS_CHILD |
WS_VISIBLE |
WS_TABSTOP, 252, 183, 58, 33
 CONTROL "", -1, "BorShade", BSS_HDIP | BSS_LEFT | WS_CHILD | WS_VISIBLE, 1, 29, 311, 5
 CONTROL "", -1, "BorShade", BSS_GROUP | BSS_CAPTION | BSS_LEFT | WS_CHILD | WS_VISIBLE,
66, 5, 178,
20
 PUSHBUTTON "VERIFICAR", IDC_VERIFICANI, 160, 9, 76, 13
 CONTROL "Acta de Nacimiento", IDC_ACTANACNI, "BorCheck", BS_AUTOCHECKBOX | WS_CHILD
| WS_VISIBLE
 | WS_TABSTOP, 84, 248, 97, 9
}
```

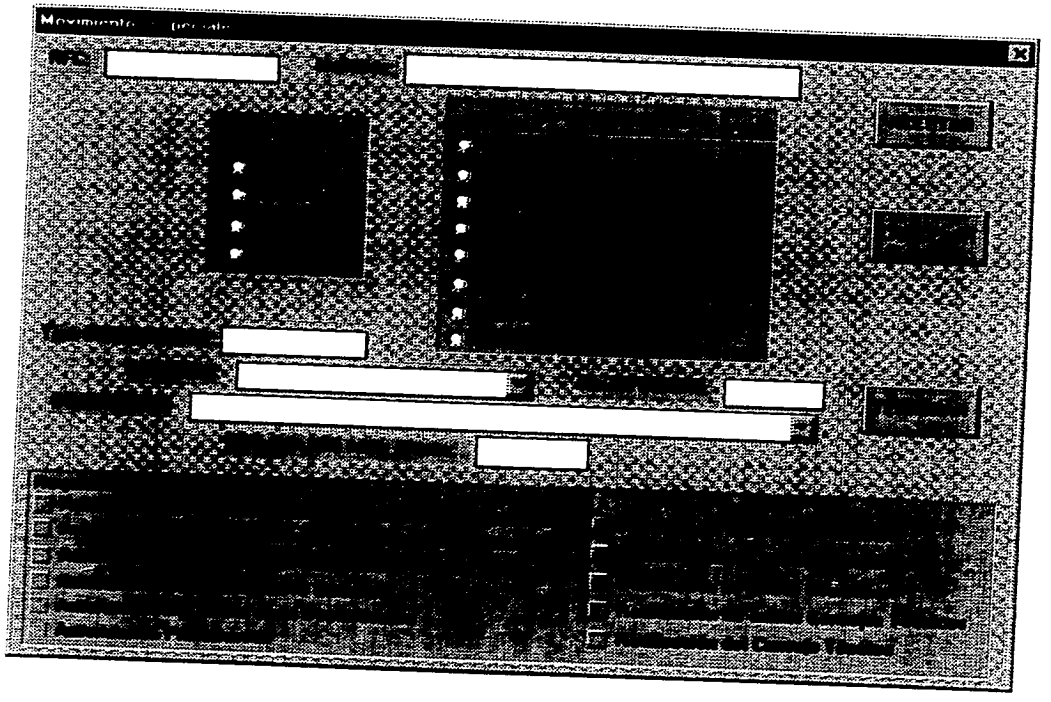
ANEXO B. PANTALLAS Y REPORTES DEL SISTEMA

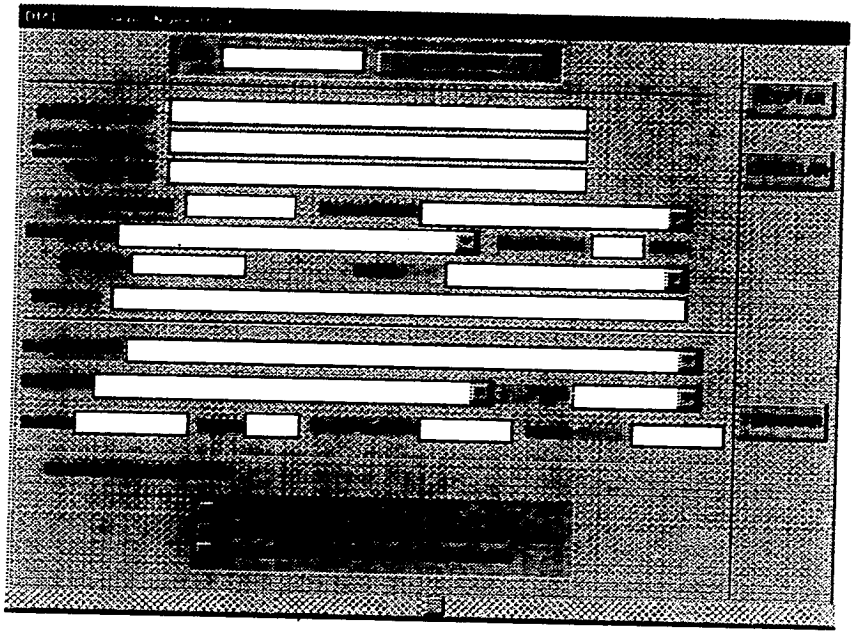
A continuación se presentan las pantallas de interfase de usuario, (pantallas de captura de datos y de consulta) así como de los reportes que emite el sistema desarrollado para la presente tesis : Sistema Automatizado de Movimientos al Organigrama de la UNAM.

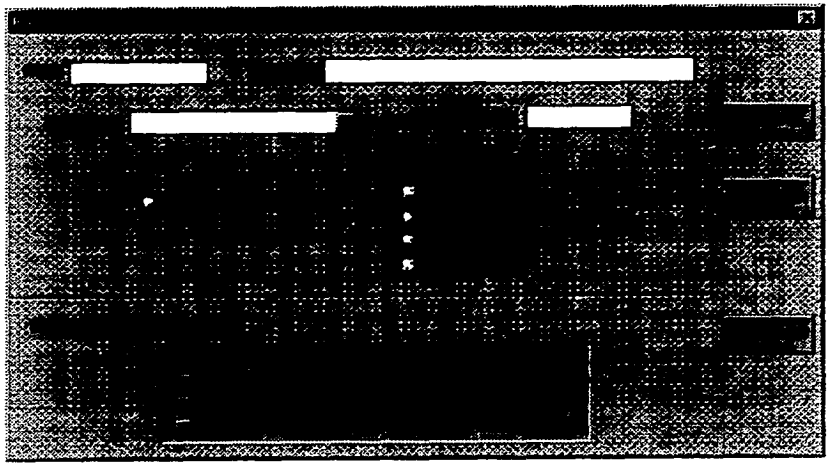


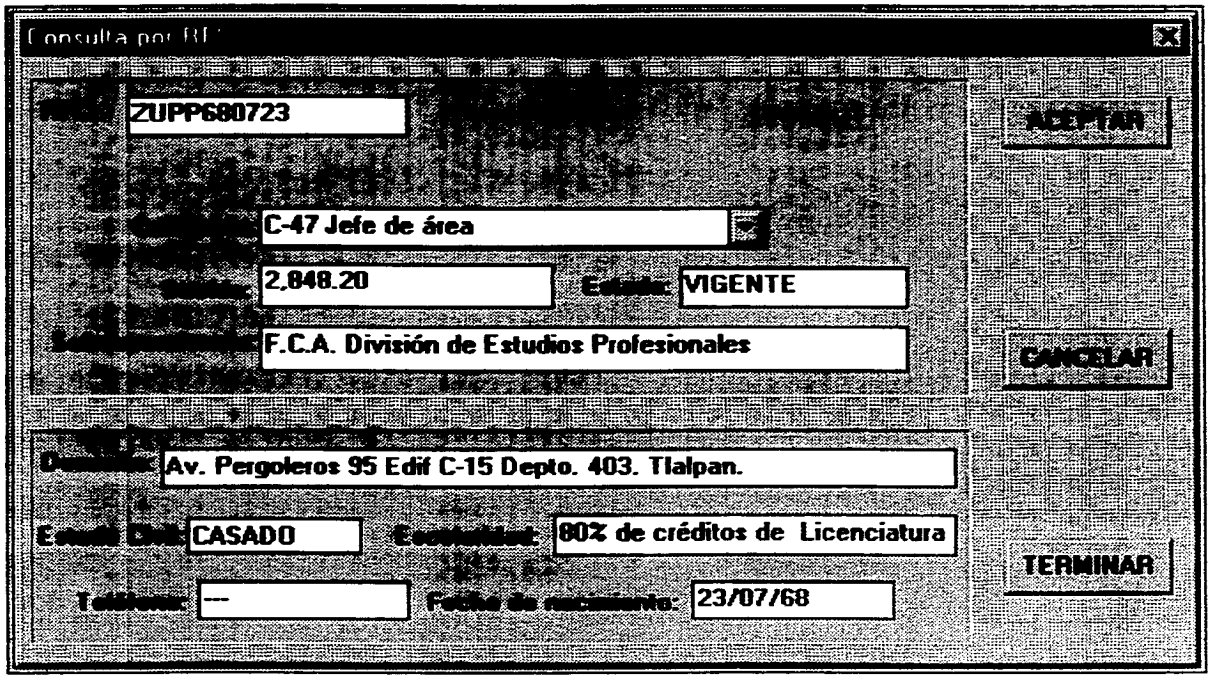




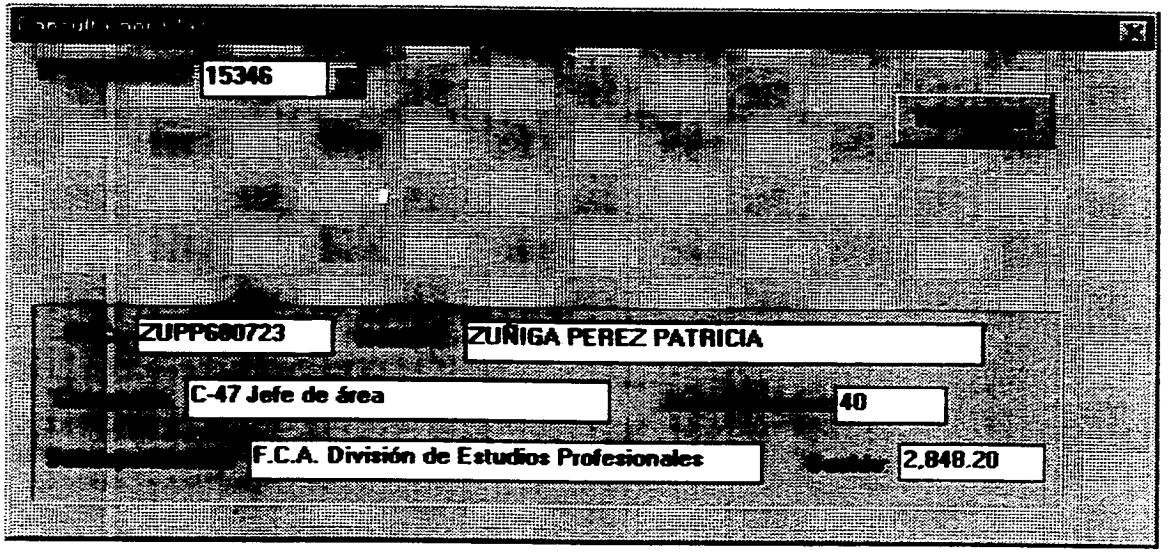


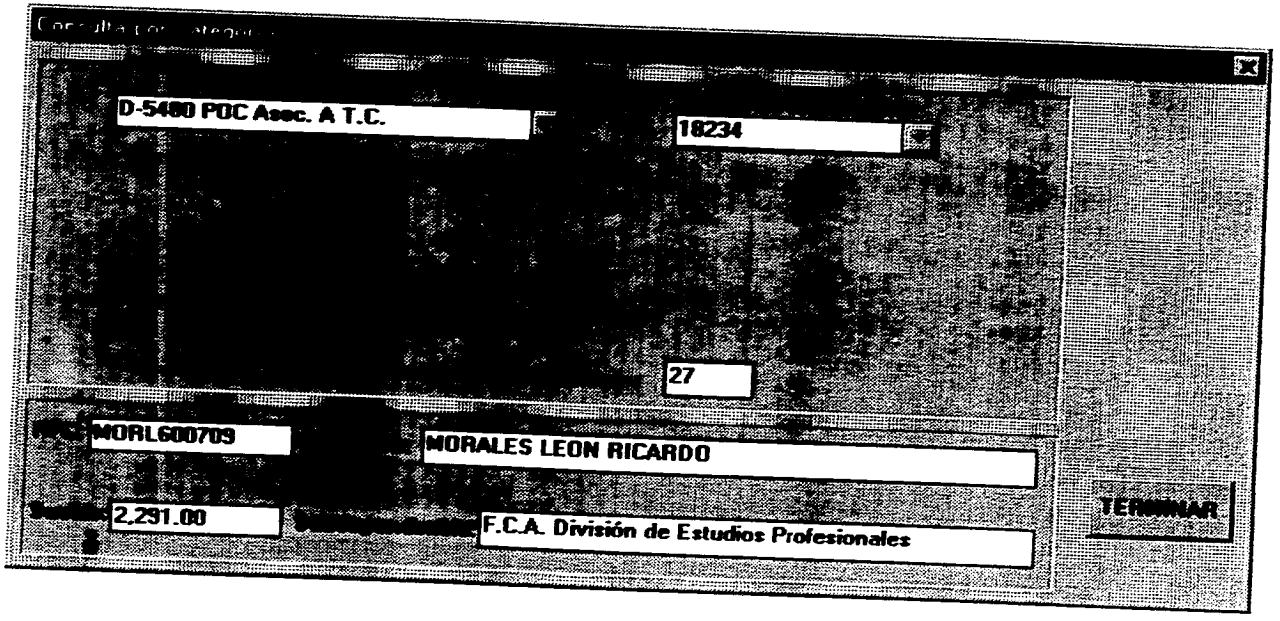





Consulta por RF
ZUPP680723
ACEPTAR
C-47 Jefe de área
2,848.20
Estado:
VIGENTE
F.C.A. División de Estudios Profesionales
CANCELAR
Av. Pergoleros 95 Edif C-15 Depto. 403. Tlalpan.
CASADO
80% de créditos de Licenciatura
TERMINAR

23/07/68


15346
ZUPP680723
ZUÑIGA PEREZ PATRICIA
C-47 Jefe de área
40
F.C.A. División de Estudios Profesionales
2,848.20


D-5480 POC Asoc. A T.C.
18234
27
MORL600709
MORALES LEON RICARDO
2,291.00
F.C.A. División de Estudios Profesionales
TERMINAR

VACANTES ACADEMICAS

PLAZA	CATEGORIA
09405	D-5144 POC Asoc. A M.T.
16034	D-6160 POC Tit. A M.T.
23031	D-6489 POC Tit. A T.C.
28953	D-6489 POC Tit. A T.C.
27833	D-6489 POC Tit. A T.C.
27945	D-6489 POC Tit. A T.C.
31045	D-6489 POC Tit. A T.C.
33952	D-6489 POC Tit. A T.C
40367	D-7544 TAO Aux B T.C.
40935	D-7544 TAO Aux B T.C.
50115	D-8682 TAO Asoc C T.C.
50429	D-8682 TAO Asoc C T.C.

PLAZA	CATEGORIA	RFC
01783	D-5480 POC Asoc. A T.C.	PALE491004
03596	D-5480 POC Asoc. A T.C.	DOJV581010
08267	D-5480 POC Asoc. A T.C.	QUHA580830
12457	D-5480 POC Asoc. A T.C.	DOML601206
17036	D-5480 POC Asoc. A T.C.	SAPI660710
18234	D-5480 POC Asoc. A T.C.	MORL600709
18723	D-5480 POC Asoc. A T.C.	JICC501205
19289	D-5480 POC Asoc. A T.C.	MACE691025
19824	D-5480 POC Asoc. A T.C.	PUAJ620915
20349	D-5480 POC Asoc. A T.C.	BALR601119
34592	D-5480 POC Asoc. A T.C.	ROAM340706
34790	D-5480 POC Asoc. A T.C.	COMA479521
45608	D-5480 POC Asoc. A T.C.	SALA390629
45689	D-5480 POC Asoc. A T.C.	SARA520420
50738	D-5480 POC Asoc. A T.C.	SAMJ270412
60302	D-5480 POC Asoc. A T.C.	CAJM680502