UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE CONTADURIA Y ADMINISTRACION

SISTEMA DE CONTROL ESCOLAR DE LA FACULTAD DE ESTUDIOS SUPERIORES ZARAGOZA

Seminario de Investigación INFORMATICA

Que en opción al grado de:

LICENCIADO EN INFORMATICA

p r e s e n t a n

Orozco Núñez Gloria Alicia

Tejeda Alvarez Sergio

Asesor del Seminario:

L.A.E. Mario Novoa Gamas

CALIDAD ACADEMICA

FCA

México, D.F.

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

AGRADECIMIENTOS:

Al Dr. Francisco Alvarez Herrera por su apoyo y confianza, y muy en especial al LAE. Mario Novoa Gamas por todos sus consejos y experiencias transmitidas en el desarrollo de este trabajo.
Con mucho cariño a nuestros padres.

CONTENIDO

INDICE

La evolución que ha sufrido la tecnología y la evidente disminución en los costos de procesamiento y almacenamiento, han ocasionado que el principal desafío sea el desarrollo de sistemas eficientes con menores costos, por lo cual se han desarrollado herramientas y metodologías que contribuyan a cumplir esta meta.

En este sistema se utilizaron técnicas de desarrollo estructurado, las cuales hacen uso de diversas herramientas gráficas y textuales para generar modelos que ayudan a la comprensión del problema y a la mejor comunicación con el usuario.

En el capítulo I se presenta el análisis estructurado mediante el modelo esencial, el cual está integrado a su vez por el modelo ambiental y el modelo de comportamiento. En el modelo ambiental se definen las interrelaciones que tendrá el sistema con el resto del mundo, así como sus alcances y límites mediante la definición del objetivo, la elaboración del diagrama de contexto, la lista de eventos y la definición de los flujos. El detalle de cómo se llevan a cabo las tareas definidas en el modelo ambiental se presenta a través del modelo de comportamiento mediante la elaboración del diccionario de datos, los diagramas de flujo de datos, las miniespecificaciones y la definición de los archivos.

La transformación de los modelos presentados por el análisis en jerarquías de módulos que posteriormente se convertirán en los programas que integrarán el sistema, se presenta en el capítulo II, a través de la elaboración del diseño estructurado.

El capítulo III, muestra la definición de los estándares establecidos en la etapa de la programación, así como los diferentes tipos de pruebas realizadas al sistema.

Finalmente, el capítulo IV establece la definición de los requerimientos y pasos a seguir para la implantación del sistema.

El propósito fundamental del análisis estructurado es modelar las necesidades del usuario y sus políticas de funcionamiento a través de herramientas gráficas y textuales que faciliten la comprensión del sistema a los usuarios y permitan enfatizar los aspectos críticos del mismo.

" Se construyen modelos de sistemas por las siguientes razones:

1.- Enfocar características importantes de sistemas y minimizar los aspectos menos importantes.

2.- Para discutir cambios y correcciones a los requerimientos de los usuarios a bajos costos y riesgos mínimos.

3.- Para verificar que se entiende el ambiente del usuario y que se documentó de una manera clara para los diseñadores y programadores" [1]

El primer modelo que se presenta en la etapa de análisis es el "modelo esencial" el cual muestra lo que debe hacer el sistema para satisfacer adecuadamente las necesidades del usuario. Este modelo está integrado a su vez por el modelo ambiental y el modelo de comportamiento.

1.1 Modelo ambiental

Cualquier sistema tiene relación con el medio ambiente en el cual se sitúa. Esta interrelación dificulta la visión del sistema que se está desarrollando, por lo cual es

1 YOURDON, Edward, "MODERN STRUCTURED ANALYSIS", Prentice-Hall, USA. 1989, p. 150.

importante definir con claridad qué forma parte del sistema y qué forma parte del medio ambiente. Yourdon establece que "el modelo ambiental define la frontera entre el sistema y el resto del mundo"[2]. La definición de los límites del sistema comienza con la elaboración del objetivo a través de la redacción de las actividades que realizará el sistema.

1.1.1 Objetivo

El sistema de control escolar Zaragoza (CONEFESZ), manejará los dat os de los alumnos para su registro y reinscripción ordinaria y extraordinaria; asimismo, realizará cambios, actualizaciones a los historiales académicos, y proporcionará información a los alumnos, DGAE y departamentos internos que la requieran.

1.1.2 Lista de eventos

La descripción de los estímulos que recibe el sistema del medio ambiente y que provocan una reacción en él, se define como lista de eventos.

Cada uno de los eventos es analizado y clasificado como un evento de tipo flujo, temporal o de control. En el sistema de control escolar se identificaron los eventos de tipo flujo, como aquellos que proporcionan datos al sistema y que al ser captados hacen que se realice algún proceso, estos eventos están marcados con (F). Se identificaron además los eventos de tipo temporal, los cuales periódicamente proporcionan

2 YOURDON, Edward, "MODERN STRUCTURED ANALYSIS", Prentice-Hall, USA. 1989, p. 360

información a las entidades con las cuales se relaciona el sistema, estos eventos están marcados con (T). La siguiente lista muestra los eventos identificados durante el desarrollo de CONEFESZ.

1 *DGAE proporciona datos de alumnos de primer ingreso (F)*
2 *Jefatura de carrera proporciona información de grupos (F)*
3 *El alumno ingresa solicitud de inscripción extraordinaria, ordinaria o reinscripción (F)*
4 *El alumno ingresa solicitud de cambio (F)*
5 *DGAE requiere listas de grupos y profesores (T)*
6 *Jefatura de carrera requiere listas de alumnos por grupo (T)*
7 *DGAE proporciona calificaciones ordinarias y extraordinarias (F)*
8 *El alumno solicita reportes de avance académico (F)*

1.1.3 Diagrama de contexto

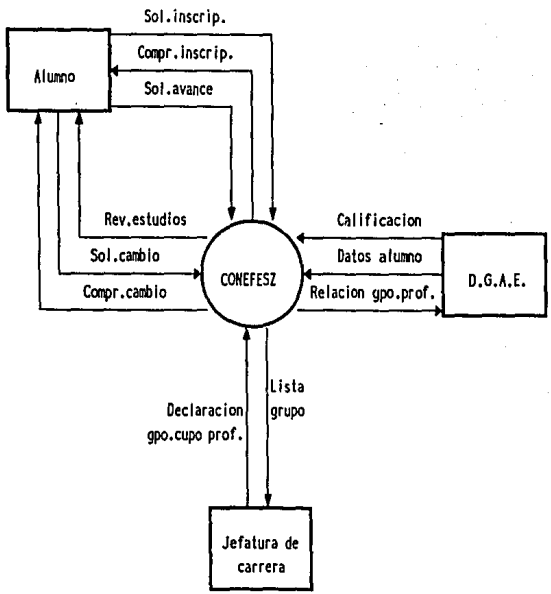
El diagrama de contexto como siguiente etapa del modelo ambiental "es un caso especial del diagrama de flujo de datos, en donde una sola burbuja representa todo el sistema"[3] .

Para poder cumplir con el objetivo planteado y así mismo realizar las funciones requeridas, es necesario que el sistema interactúe con otros sistemas u organizaciones, los cuales son llamados terminadores o entidades y están representadas en el diagrama

3 YOURDON, Edward, "MODERN STRUCTURED ANALYSIS", Prentice-Hall, USA. 1989, p. 374.

de contexto a través de rectángulos, dentro de los cuales se coloca el nombre de la entidad a la que representan. Los datos que proporciona o recibe de cada una de las entidades se muestran mediante flechas que representan a su vez los flujos de entrada y salida principales del sistema.

Diagrama de contexto



1.1.4 Flujos de datos

Los modelos gráficos representan una forma fácil de comunicación y comprensión entre el analista y el usuario, sin embargo las herramientas de modelado textuales dán una definición más detallada de las conexiones y tareas mostradas en el diagrama de contexto, el objetivo y la lista de eventos.

A continuación se especifica el nombre de cada uno de los flujos de datos de entrada y de salida que fueron identificados en el sistema, así como los datos que integran a cada uno de los flujos.

Calificacion: [4]

Cuenta
Asign
Grupo
Plantel
Periodo
Calif
Acta

Clave usuario:

Usuario

Compr.inscrip:

Plantel	Periodo
Carrera	Asign
Desc-Carr	Desc-Asign
Cuenta	Creditos
Alumno	Semestre
Ingreso	Grupo

4 El nombre de los flujos y datos no utiliza acentos, dentro del ambiente del usuario

Compr.cambio:

Cuenta
Alumno
Cambio
Desc-Cam
Origen
Destino
Autorizo
Fecha

Datos alumno:

Cuenta
Alumno
Plantel
Carrera
Ingreso
Nacion
Exalum
Sexo
Nacim

Declaracion gpo.cupo prof.:

Asign
Grupo
Cupo
RFC
RFC-Asis

Lista grupo:

Plantel
Desc-Plan
Carrera
Desc-Carr
Profesor
Asistente
Grupo
Asign
Desc-Asign
Cuenta
Alumno

Relacion gpo.prof

Plantel
Asign
Desc-Asign
Grupo
RFC
Profesor
RFC-Asis
Asistente

Sol.cambio:

Cuenta
Cambio
Origen
Destino
RFC

Rev.estudios:

Alumno	Calif
Ingreso	Periodo
Cuenta	Acta
Desc-Carr	Grupo
Desc-Plan	Num-Ord
Carrera	Num-Ext
Plantel	MB
Oblig-Ac	B
Oblig	S
Optat-Ac	NA
Optat	NP
Aprob	Reval
Reprob	Coval
Semestre	Aprob-Ord
Asign	Aprob-Ext
Creditos	Rep-Ord
Desc-Asign	Rep-Ext

Sol.avance:

Cuenta

Sol.incrip.:

Cuenta
Asign
Grupo
Tipo
Direccion
Tel

Tipo reporte:

Reporte

1.2 Modelo de comportamiento

Una vez definidos los alcances y límites, es necesario especificar el detalle de los datos que manejará el sistema, las actividades que realizará y el detalle de cada una de estas actividades a través del modelo de comportamiento.

"El modelo de comportamiento describe las actividades que el sistema requiere para que interactúe de manera exitosa con el medio ambiente" [5]

Para la elaboración del modelo de comportamiento, se utilizaron diversas herramientas, como lo son:

1.2.1 Diccionario de datos

El detalle de los datos que manejará el sistema, se lleva a cabo a través de la elaboración del diccionario de datos, el cual se presenta como "un listado organizado de todos los datos pertinentes al sistema, con definiciones precisas y rigurosas para que tanto el usuario como el analista tengan un entendimiento común" [6]

La definición de los datos se planteó en colaboración con los usuarios, ya que los nombres deben ser claros dentro del contexto de la universidad y compatibles a los estándares que utiliza la Dirección General de Administración Escolar (DGAE). En base a esto se establecieron las siguientes especificaciones:

5,6 YOURDON, Edward, "MODERN STRUCTURED ANALYSIS", Prentice-Hall, USA. 1989, p. 360, 212

= Está integrado por	|	Separador de opciones
+ Más	*Itálicas*	Comentario
{} Iteración	§	Espacio en blanco

[] Seleccionar una de las opciones

Diccionario de datos

Acta = {0-9}7

Folio del acta en que se encuentra la historia académica del alumno

Alumno = {[A-Z| a-z| 0-9| '| §]}32

Nombre del alumno, con formato: Apellido Paterno + Apellido Materno + Nombre(s)

Aprob = {0-9}2

Número de asignaturas acreditadas

Aprob-Ext = {0-9}2

Número de asignaturas acreditadas en extraordinario

Aprob-Ord = {0-9}2

Número de asignaturas acreditadas en ordinario

Asign = 4{0-9}4 *Clave de la Asignatura*

Asistente = {[A-Z| a-z| 0-9| '| §]}32

Nombre del profesor asistente con formato: Apellido Paterno + Apellido Materno + Nombre(s)

Autorizo = {[A-Z| a-z| 0-9| '| §]}32

Nombre de la persona que autoriza el cambio, con formato: Apellido Paterno + Apellido Materno + Nombre(s)

B = {0-9}2

Número de calificaciones B obtenidas

Calif = [MB| B| S| NA| NP| AC| CO| RE]

Calificación

Cambio = {0-9}3

Clave de cambio

Carrera = 2{0-9}2

Clave de la carrera

Coval = {0-9}2

Número de asignaturas covalidadas

Creditos = {0-9}3

Número de créditos que cubre la asignatura

Cuenta = 8{0-9}8

Clave de identificación del alumno

Cupo = {0-9}2
Número de alumnos que pueden inscribirse a un grupo en forma regular

Cupoi = {0-9}2
Número de alumnos que pueden inscribirse a un grupo en forma irregular

Desc-Asign = 1{[A-Z| a-z| 0-9| '| §]}25
Descripción de la asignatura

Desc-Cam = 1{[A-Z| a-z| 0-9| '| §]}20
Descripción del cambio

Desc-Carr = 1{[A-Z| a-z| 0-9| '| §]}30
Descripción de la carrera

Desc-Plan = 1{[A-Z| a-z| 0-9| '| §]}35
Descripción del plantel

Destino = {0-9}4
Clave del grupo, carrera o plantel al cual se quiere cambiar el alumno

Direccion = {[A-Z| a-z| 0-9| '| §]}40
Dirección del alumno

Duracion = {0-9}2
Duración de la carrera. Unidades: Años

Exalum = 2{0-9}2

Número que identifica la causa por la cual un alumno pasa a ser exalumno

Fecha = 6{0-9}6

Fecha en que se realizó el cambio, con formato: ddmmaa

Folio = {0-9}4

Número consecutivo que indica el lugar físico en que se encuentran los documentos del alumno

Grupo = 4{[A-Z| a-z| 0-9| '| §]}4

Clave del grupo, con formato numérico para grupos ordinarios y alfanumérico para extraordinarios.

Ingreso = 2{0-9}2

Año de ingreso del alumno a la FESZ

MB = {0-9}2

Número de calificaciones MB obtenidas

Morigen = 2{0-9}2

Medio de ingreso del alumno

NA = {0-9}2

Número de calificaciones NA obtenidas

Nacim = 6{0-9}6
Fecha de nacimiento del alumno, con formato: ddmmaa

Nacion = [0-9]
Clave que identifica la nacionalidad del alumno

NP = {0-9}2
Número de calificaciones NP obtenidas

Num-Ext = {0-9}2
Número de veces que se ha cursado una asignatura en forma extraordinaria

Num-Ord = {0-9}2
Número de veces que se ha cursado una asignatura en forma ordinaria

Oblig = {0-9}3
Número de créditos obligatorios de la carrera

Oblig-ac = {0-9}3
Número de créditos obligatorios acumulados

Optat = {0-9}3
Número de créditos optativos de la carrera

Optat-ac = {0-9}3
Número de créditos optativos acumulados

Origen = {0-9}4
Clave del grupo, carrera o plantel del cual se quiere cambiar el alumno

Periodo = 3{0-9}3
Periodo de inscripción a la asignatura compuesto por el año y el semestre.

Plan = 3{0-9}3
Clave del plan de estudios

Plantel = 3{0-9}3
Clave del plantel

Profesor = {[A-Z| a-z| 0-9| '| §]}32
Nombre del profesor, con formato: Apellido Paterno + Apellido Materno + Nombre(s)

Rep-Ext = {0-9}2
Número de asignaturas reprobadas en extraordinario

Rep-Ord = {0-9}2
Número de asignaturas reprobadas en ordinario

Reprob = {0-9}2
Número de asignaturas reprobadas

Reval = {0-9}2
Número de asignaturas revalidadas

RFC = 4{A-Z| a-z} + 6{0-9}6 + 2{A-Z| a-z}3
Clave del profesor

RFC-Asis = 4{A-Z| a-z} + 6{0-9}6 + 2{A-Z| a-z}3
Clave del profesor asistente

Reporte = *1{0-9}1*
Tipo de reporte que se desea imprimir

S {0-9}2
Número de calificaciones S obtenidas

Semestre = {0-9}2
Semestre en que se imparte la asignatura.

Seriacion = {0-9}4
Asignatura que debe aprobarse antes de cursar la asignatura actual

Sexo = [M| F]

Tel = 7{0-9}7
Teléfono del alumno

Tipo = 1{0-9}1
Tipo de solicitud ordinaria, reinscripción o extraordinaria

Usuario = {[A-Z| a-z| 0-9| '| §]}6
Clave de acceso especial que permite realizar cambios especiales

1.2.2 Diagramas de flujo de datos

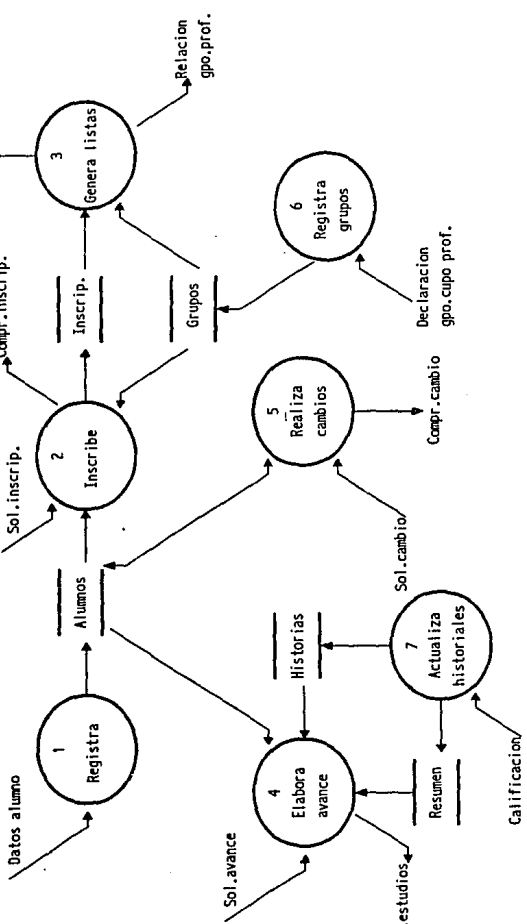
Los diagramas de flujo de datos describen los procesos que realiza el sistema, las entradas que recibe y las salidas que proporciona. "Es una herramienta que permite visualizar un sistema como una red de procesos funcionales, conectados entre sí por flujos y almacenamientos de datos".[7]

El punto de partida de los diagramas de flujo es la creación del nivel 0, el cual muestra una vista global del sistema a través de burbujas que representan las actividades identificadas en el modelo ambiental; flujos de datos que muestran el movimiento de datos y almacenamientos que representan los datos que el sistema debe conservar.

El detalle de cada uno de los procesos definidos en el nivel 0, se lleva a cabo a través de la elaboración de los niveles siguientes de los diagramas de flujo de datos.

Cada una de las burbujas de los diagramas es identificada mediante un número consecutivo, sin embargo dicha numeración no implica una secuencia lógica de actividades.

7 YOURDON, Edward, "MODERN STRUCTURED ANALYSIS", Prentice-Hall, USA. 1989, p. 390


Nivel 0
Datos alumno
1
Registra
Alumnos
Sol.inscrip.
2
Inscribe
Compr.inscrip.
Inscrip.
3
Genera listas
Lista grupo
Relacion gpo.prof.
Grupos
6
Registra grupos
Declaracion gpo.cupo prof.
5
Realiza cambios
Compr.cambio
Sol.cambio
Historias
7
Actualiza historiales
Resumen
Calificacion
Sol.avance
4
Elabora avance
Rev.estudios

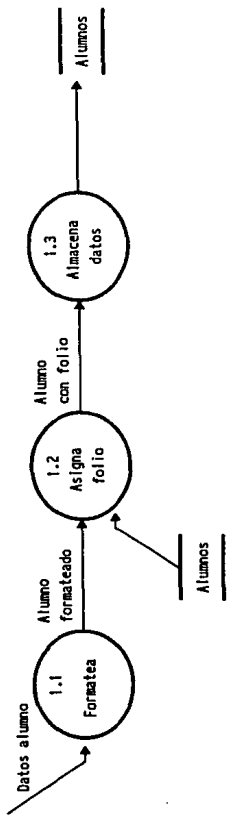


Figura 1

Regiistra

Figura 2

I n s c r i b e

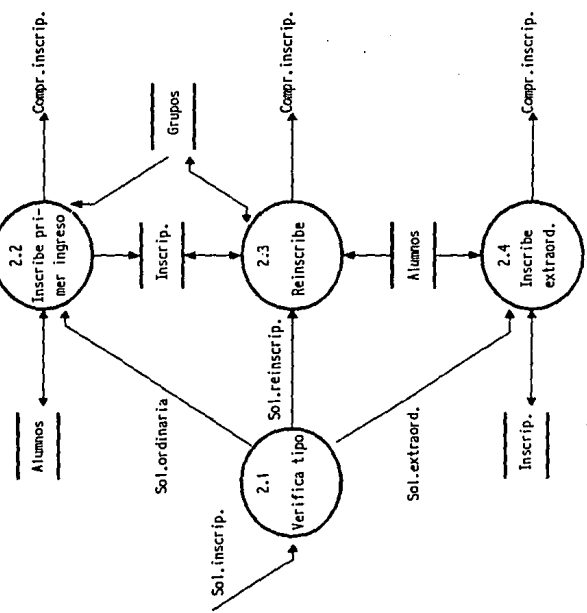
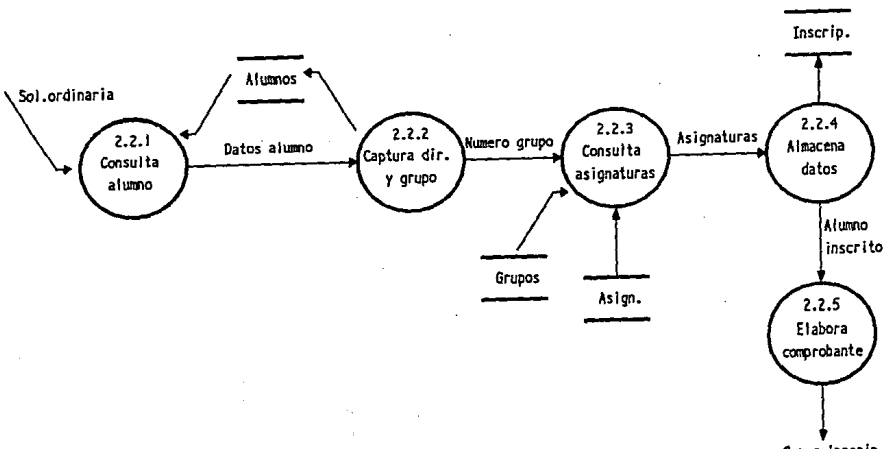


Figura 2.2

Inscribe primer ingreso



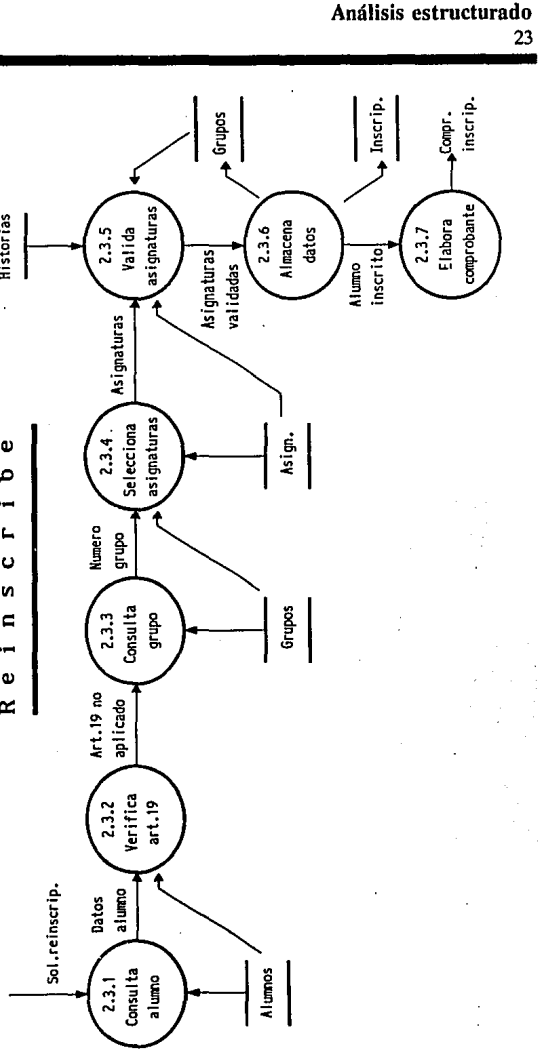


Figura 2.3

Reinscribe

Figura 2.3.5

Valida asignaturas
Historias
Asignatura
2.3.5.1
Valida
créditos
Créditos
validados
2.3.5.2
Verifica
cupo
Cupo
aceptado
2.3.5.3
Verifica
art.27
Art.27 no
aplicado
2.3.5.4
Verifica
calif.
Calif.
validada
2.3.5.5
Verifica
seriación
No seriada
o aprobada
Asign.
Grupos
Historias
Asign.
2.3.5.6
Verifica
inscrito
Alumno no
inscrito
Inscrip.
2.3.5.7
Acepta
asignatura
Asignaturas
validadas

Figura 2.4

Inscribe extraordinario

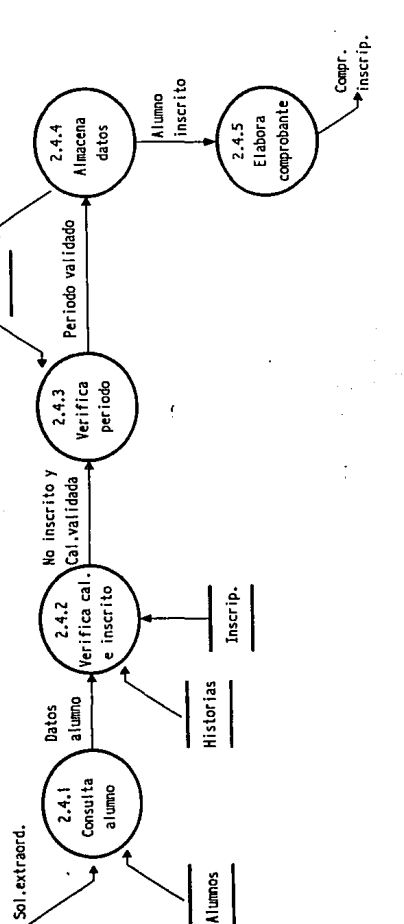
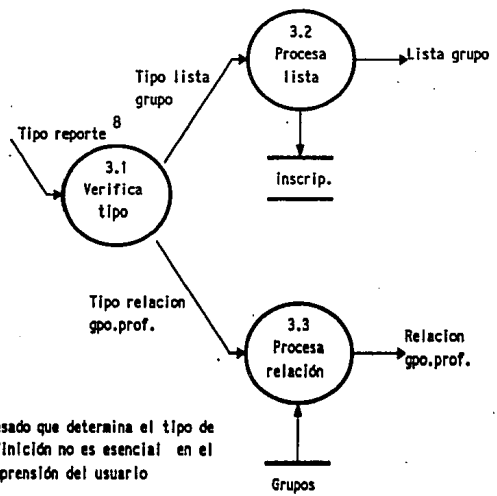


Figura 3

G e n e r a L i s t a s



8 Tipo reporte: Flujo ingresado que determina el tipo de reporte a imprimir; su definición no es esencial en el nivel anterior para la comprensión del usuario

Figura 3.2

P r o c e s a l i s t a

Figura 3.3

P r o c e s a r e l a c i o n

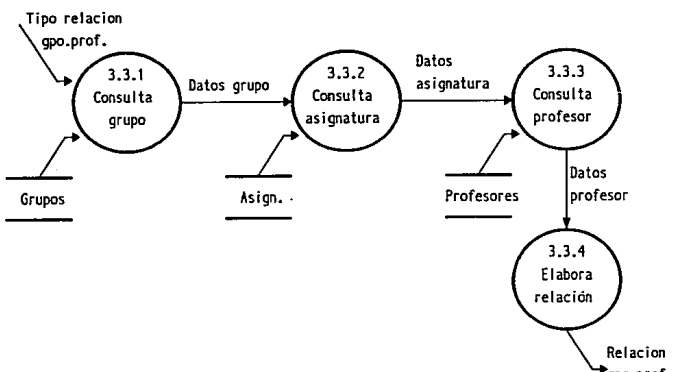


Figura 4

E l a b o r a a v a n c e

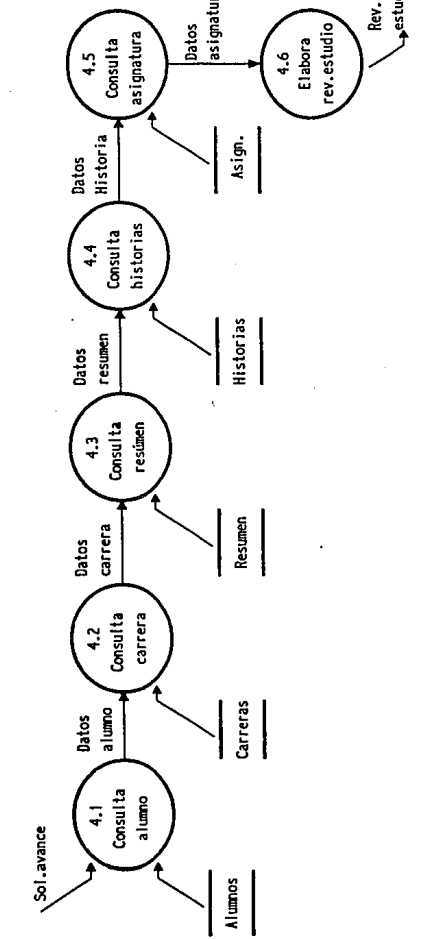


Figura 5

R e a l i z a c a m b i o s

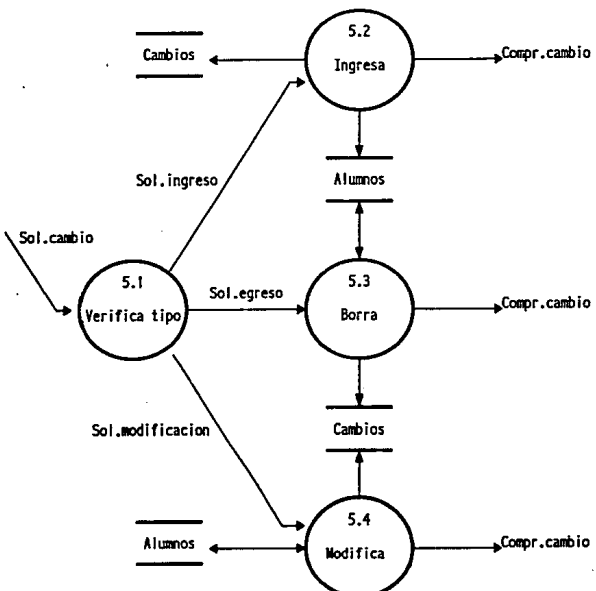
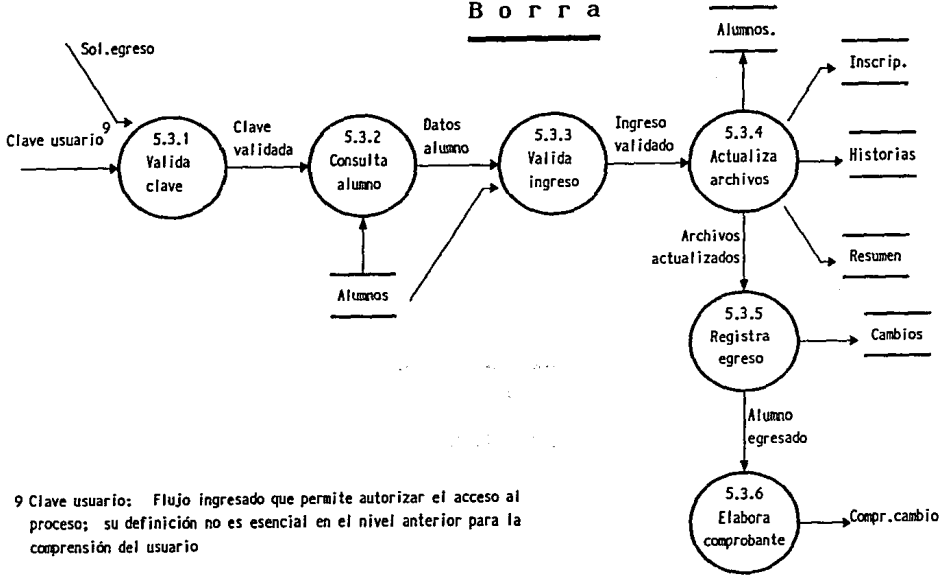


Figura 5.2

I n g r e s a

Figura 5.3

B o r r a



9 Clave usuario: Flujo ingresado que permite autorizar el acceso al proceso; su definición no es esencial en el nivel anterior para la comprensión del usuario

Figura 5.4

Modifica

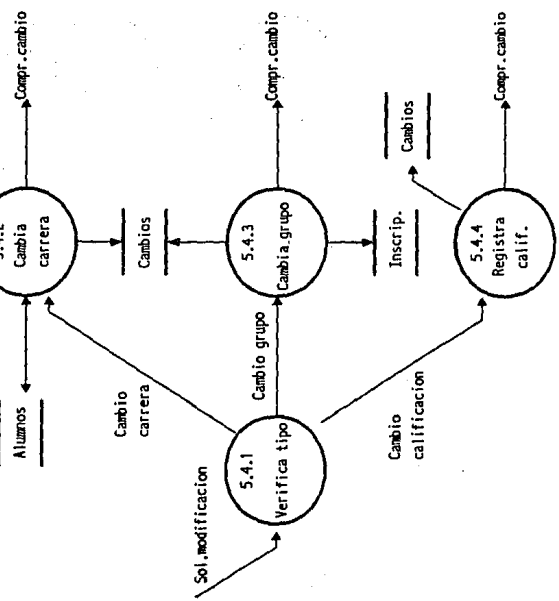


Figura 5.4.2

Cambia carrera

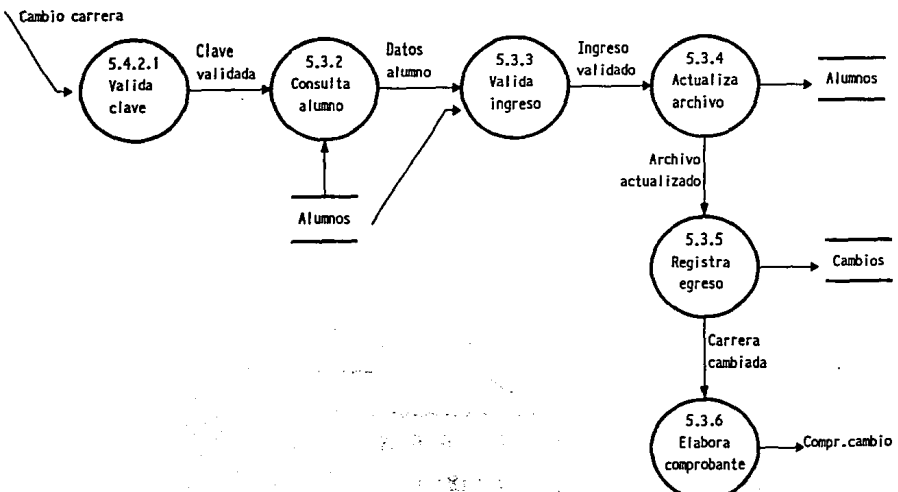


Figura 5.4.3

C a m b i a g r u p o

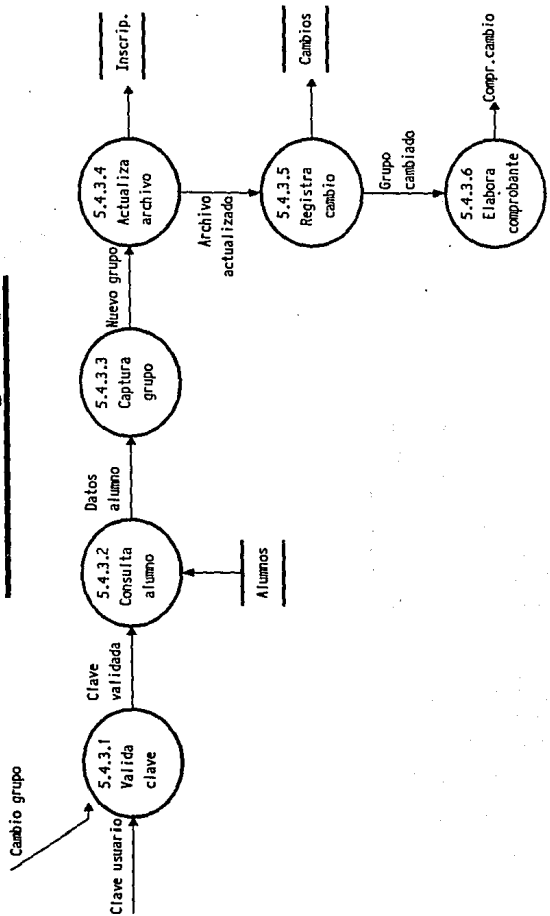


Figura 6

Registra grupos

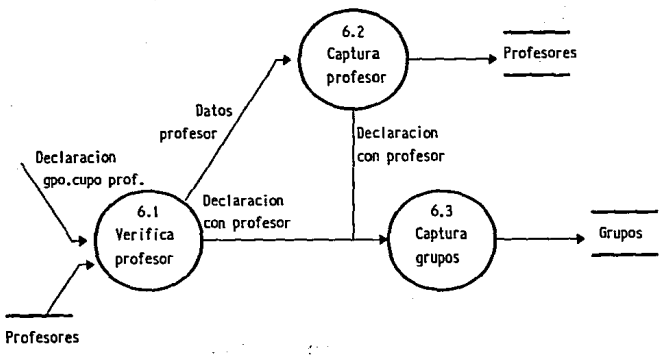
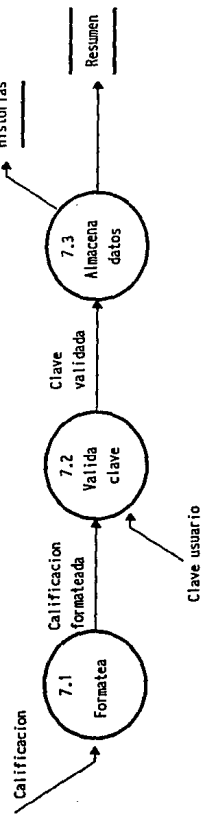


Figura 7

Actualiza historiales



1.2.3 Archivos

Paralelamente con la elaboración de los diagramas de flujo y el diccionario de datos en donde los flujos fueron identificados y cada uno de los datos fué descrito, se definen los archivos que utilizará el sistema, así como los índices necesarios para accesar los datos almacenados.

Con el fin de evitar redundancia e inconsistencia en los archivos definidos y eficientar los accesos, se llevó a cabo el proceso de normalización, tomando en cuenta los siguientes criterios:

"1a. Forma normal: Una relación es una tabla en donde en cada intersección de un renglón y una columna, sólo puede haber un valor.

2a. Forma normal: Todo atributo que no sea clave, debe ser completamente dependiente en forma funcional de la clave primaria.

3a. Forma normal: No existe ninguna dependencia transitiva entre los atributos que no sean clave"[10].

Para la presentación de los archivos definidos se utilizaron las siguientes convenciones:

MAYUSCULAS	Indican el nombre de un archivo
Sombreado	Indica que el campo forma parte de la llave primaria

10 ATRE, Shakuntala. "TECNICAS DE BASES DE DATOS" Trillas, México, 1989, p. 141

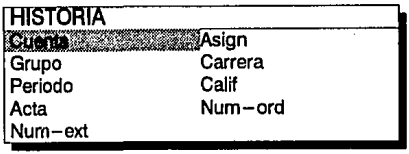
ALUMNO	
Cuenta	**Carrera**
Alumno	Ingreso
Nacion	Sexo
Nacim	Direccion
Tel	Folio
Exalum	Plantel
Morigen	

ASIGN	
Asign	**Carrera**
Desc–asign	Creditos
Semestre	Seriacion

CAMBIO	
Cuenta	**Cambio**
Fecha	Origen
Destino	Rfc

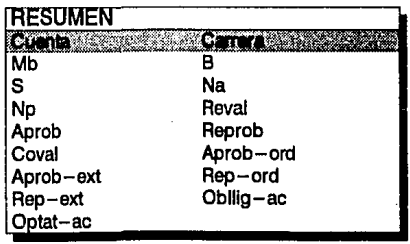
CARRERA	
Carrera	Desc–carrera
Duracion	Plan
Oblig	Optat

GRUPO	
Grupo	**Asign**
Carrera	Rfc
Cupo	Cupoi
Rfc–asis	


HISTORIA
Cuenta
Asign
Grupo
Carrera
Periodo
Calif
Acta
Num-ord
Num-ext


INSCRIP
Cuenta
Grupo
Asign
Tipo


PROF
Rfc
Profesor


RESUMEN
Cuenta
Carrera
Mb
B
S
Na
Np
Reval
Aprob
Reprob
Coval
Aprob-ord
Aprob-ext
Rep-ord
Rep-ext
Obllig-ac
Optat-ac


TIPO-CAMBIO
Cambio
Desc-cam

1.2.4 Miniespecificaciones

El detalle de cómo se llevan a cabo las actividades definidas en los diagramas de flujo de datos de nivel más bajo, se logra a través de la redacción de las miniespecificaciones.

Esta etapa se debe realizar sin tomar en cuenta el lenguaje de programación en el cual se desarrollará el sistema, y constituye junto con el diseño la base para el desarrollo de la etapa de la programación.

Para la redacción de las miniespecificaciones se utilizaron las siguientes convenciones:

/* */	Indica el nombre del proceso
Itálicas	Nombre de campo
v-<variable>	Nombre de variable de trabajo
MAYUSCULAS	Nombre de archivo
Define	Especifica las variables o reportes a utilizar
Mientras <condición> **Haz:** <instrucciones> **Fin-Mientras**	Evalúa la condición y si ésta se cumple, realiza las instrucciones especificadas hasta que la condición deje de cumplirse
Lee	Realiza una lectura de datos a un archivo
Inserta	Agrega caracteres a un campo
Graba	Agrega un registro a un archivo

En caso de:
<condicion-1>, **Haz:** <instrucciones>
<condicion-2>, **Haz:** <instrucciones> ...
<condicion-n>, **Haz:** <instrucciones>

Fin-En caso	Evalúa el valor de una variable y ejecuta las instrucciones especificadas de acuerdo a la condición que se cumple
Busca	Localiza un registro en un archivo
Donde	Establece condiciones de búsqueda

Si <condicion> **Entonces, Haz:**
<instrucciones>
Fin-Si
Sino
<instrucciones>

Fin-Si	Evalúa una condicion y si esta se cumple, realiza las instrucciones especificadas, en caso contrario, realiza las instrucciones especificadas después de la palabra Sino
Despliega	Manda un mensaje a la pantalla
Imprime	Realiza la impresión de un reporte
Asigna	Dá un valor a una variable

Repite
<instrucciones>

Hasta <condicion>	Realiza las instrucciones especificadas hasta que se cumpla la condición
Ordena	Clasifica un reporte o archivo

Miniespecificaciones

1 Registra

```
/* 1.1 Formatea */
Define archivo AUXILIAR
 Mientras haya datos alumnos, Haz:
   Lee siguientes datos alumno
   Inserta separadores de campo
   Graba registro en AUXILIAR
Fin-Mientras

/* 1.2 Almacena datos */
Mientras haya alumnos formateados en AUXILIAR, Haz:
   Lee siguiente registro de AUXILIAR
   Graba registro en ALUMNO
Fin-Mientras
```

2 Inscribe

/* 2.1 Verifica tipo */
Define variable v-tipo
Lee v-tipo
En caso de:
v-tipo = 1, **Haz:** [2.2 Inscribe primer ingreso]
v-tipo = 2, **Haz:** [2.3 Reinscribe]
v-tipo = 3, **Haz:** [2.4 Inscribe extraord]
v-tipo = **otro, Despliega** "Tipo de inscripción no válida"
Fin-En caso

2.2 Inscribe primer ingreso

Define reporte Compr.inscrip.
Define variables v-cuenta, v-grupo, v-direccion

/* 2.2.1 Consulta alumno */
Lee v-cuenta
Busca *cuenta* en ALUMNO **Donde** ALUMNO*cuenta* = v-cuenta
Si encuentra registro **Entonces, Haz:**
Lee registro
Graba registro en Compr.inscrip.
/* 2.2.2 Captura dir. y grupo */
Lee v-direccion, v-grupo
Graba v-direccion en ALUMNO*direccion*

```
   /* 2.2.3 Consulta asignatura */
   Mientras haya registros en GRUPO,
    Donde GRUPOgrupo = v-grupo, Haz:
      Lee siguiente registro de GRUPO
      Graba registro en Compr.inscrip.
      Busca desc-asign en ASIGN,
       Donde ASIGNasign = GRUPOasign

      /* 2.2.4 Almacena datos */
      Graba registro en INSCRIP
      Graba registro en Compr.inscrip.
   Fin-Mientras
Sino
   Despliega "Número de Cuenta erróneo"
Fin-Si
/* 2.2.5 Elabora comprobante */
Imprime Compr.inscrip.
```

2.3 Reinscribe

```
Define reporte Compr.inscrip.
Define variables v-cuenta, v-grupo, v-valida, v-asign, v-primer
Asigna verdad a v-primer
/* 2.3.1 Consulta alumno */
Lee v-cuenta
Busca cuenta en ALUMNO Donde ALUMNOcuenta = v-cuenta
Si encuentra registro Entonces, Haz:
```

```
/* 2.3.2 Verifica art.19 */
Lee carrera en ALUMNO
Busca duracion en CARRERA
 Donde CARRERAcarrera = ALUMNOcarrera
 Si fecha-actual <= CARRERAduracion * 1.5 +
 ALUMNOingreso Entonces, Haz:

   /* 2.3.3 Consulta grupo */
   Lee v-grupo
   Busca grupo en GRUPO Donde GRUPOgrupo = v-grupo
   Si encuentra registro Entonces Haz:

      /* 2.3.4 Selecciona asignaturas */
      Mientras haya registros en GRUPO, Donde GRUPOgrupo = v-grupo Haz:
         Lee siguiente registro de GRUPO
         Asigna GRUPOasign a v-asign
         Busca desc-asign en ASIGN, Donde ASIGNasign = GRUPOasign
         Haz: [2.3.5 Valida asignatura]
         Asigna falso a v-primer

         /* 2.3.6 Almacena datos */
         Si v-valida es verdadera Entonces Haz:
            Graba registro en INSCRIP
            Asigna GRUPOcupo - 1 a GRUPOcupo
            Graba registro en Compr.inscrip.
         Fin-Si
      Fin-Mientras
```

```
        /* 2.3.7 Elabora comprobante */
        Imprime comprobante
      Sino
        Despliega "Grupo no registrado"
   Sino
      Despliega "Alumno en artículo 19"
Sino
   Despliega "Número de cuenta no registrado"
```

2.3.5 Valida asignatura

```
/* 2.3.5.1 Valida créditos */
Define variables v-creditos, v-art27, v-seria
Si v-primer = verdad Entonces, Haz:
   Mientras haya registros en INSCRIP,
     Donde v-cuenta = INSCRIPcuenta, Haz:
       Busca creditos en ASIGN,
        Donde INSCRIPasign = ASIGNasign
       Asigna ASIGNcreditos + v-creditos a v-creditos
   Fin-Mientras
Fin-Si
Si v-creditos <= 60 Entonces, Haz:

   /* 2.3.5.2 Verifica cupo */
   Busca cupo en GRUPO,
     Donde v-grupo = GRUPOgrupo Y v-asign = GRUPOasign
   Si GRUPOcupo > 0 Entonces, Haz:
```

```
/* 2.3.5.3 Verifica artículo 27 */
Busca num-ord en HISTORIA
  Donde, v-cuenta = HISTORIAcuenta
  Y v-asig = HISTORIAasign
Si encuentra registro Entonces, Haz:
   Si HISTORIAnum-ord <= 2 Entonces, Haz:

      /* 2.3.5.4 Verifica calificación */
      Si HISTORIAcalif = "NA" O "NP" Entonces, Haz:
        Asigna verdad a v-art27
      Fin-Si
Sino
   Asigna verdad a v-art27
Fin-Si
Si v-art27 = verdad Entonces, Haz:

   /* 2.3.5.5 Valida Seriación */
   Lee seria en ASIGN
      Asigna ASIGNseria a v-seria
   Busca calif en HISTORIA Donde,
     v-cuenta = HISTORIAcuenta Y v-seria = HISTORIA.asign
   Si calif = "S" o "B" o "MB" Entonces, Haz:

      /* 2.3.5.6 Verifica inscrito */
      Busca cuenta en INSCRIP
        Donde, v-cuenta = INSCRIPcuenta
        Y v-asign = INSCRIPasign
      Si no encuentra registro Entonces, Haz:
```

```
            /* 2.3.5.7 Acepta asignatura */
            Asigna verdad a v-valida
            Asigna v-creditos + ASIGN.creditos a
               v-creditos
            Despliega "Asignatura aceptada "
         Sino
            Despliega "Alumno ya inscrito "
         Fin-Si
      Sino
         Despliega " La materia seriada no ha sido aprobada "
      Fin-Si
   Sino
      Despliega "Alumno en artículo 27 "
   Fin-Si
Sino
   Despliega " El grupo se encuentra saturado "
Fin-Si
Sino
   Despliega " No se pueden cursar más de 60 créditos por semestre "
Fin-Si
```

2.4 Inscribe extraordinario

```
/* 2.4.1 Consulta alumno */
Define reporte Compr.inscrip.
Define variables v-asig, v-periodo, v-grupo, v-primer, v-salir
Asigna falso a v-primer
```

```
Lee v-cuenta
Lee v-grupo
Repite
  Busca cuenta en ALUMNO Donde ALUMNOcuenta = v-cuenta
  Si encuentra registro Entonces, Haz:

    /* 2.4.2 Verifica cal. e inscrito */
    Lee v-asig
     Busca calif en HISTORIA Donde HISTORIAcuenta = v-cuenta
      Y HISTORIAasign = v-asign
    Si calif = "MB" O "B" O "S" Entonces, Haz:
       Busca cuenta en INSCRIP
        Donde INSCRIPcuenta = v-cuenta
        Y INSCRIPasign = v-asign
       Si encuentra registro Entonces, Haz:

          /* 2.4.3 Verifica periodo */
          Si v-primer = falso Entonces, Haz:
             Mientras haya registros en INSCRIP, Donde
              v-cuenta = INSCRIPcuenta, Haz:
                Si v-grupo = INSCRIPgrupo Entonces, Haz:
                   Asigna v-periodo + 1 a v-periodo
                Fin-Si
             Fin-Mientras
          Fin-Si
       Fin-Si
       Si v-periodo <= 2 Entonces, Haz:
```

```
        /* 2.4.4 Almacena datos */
        Asigna v-periodo + 1 a v-periodo
        Graba registro en INSCRIP
        Graba registro en Compr.inscrip.
      Sino
        Despliega "El alumno tiene más de 2 extraordinarios en el periodo actual"
    Sino
      Despliega "La asignatura ya ha sido aprobada"
  Sino
      Despliega "Número de cuenta erróneo"
Lee v-salir
Hasta v-salir = si

/* 2.4.5 */
Imprime Compr.inscrip.
```

3 Genera Listas

```
/* 3.1 Verifica tipo */
Define variable v-tipo
Lee v-tipo
En caso de:
   v-tipo = 1, Haz: [3.2 Procesa lista]
   v-tipo = 2, Haz: [3.3 Procesa relación]
   v-tipo = otro, Despliega "Tipo de reporte no válido"
Fin-En caso
```

3.2 Procesa lista

```
/* 3.2.1 Consulta grupo */
Define reporte Lista-grupo
Mientras haya registros en GRUPO Haz:
   Lee siguiente registro de GRUPO
   Graba registro en Lista-grupo

   /* 3.2.2 Consulta asignatura */
   Busca desc-asign en ASIGN Donde ASIGN.asign = GRUPO.asign Y
    GRUPOcarrera = ASIGN.carrera
   Graba desc-asign en Lista-grupo
```

```
/* 3.2.3 Consulta profesor */
Busca profesor en PROFESOR
 Donde PROFESOR.rfc = GRUPO.rfc
Graba profesor en Lista-grupo
Busca profesor en PROFESOR
 Donde PROFESOR.rfc = GRUPO.rfc-asis
Graba profesor en Lista-grupo

/* 3.2.4 Consulta inscripción */
Mientras haya registros en INSCRIP Donde
  INSCRIP.grupo = GRUPO.grupo Y GRUPO.asign = INSCRIP.asign
    Graba cuenta, carrera en Lista-grupo

    /* 3.2.5 Consulta alumno */
    Busca alumno en ALUMNO
     Donde ALUMNO.cuenta = INSCRIP.cuenta Y
     ALUMNO.carrera =INSCRIP.carrera
    Graba alumno en Lista-grupo
  Fin-Mientras
Fin-Mientras

/* 3.3.4 Procesa lista */
Ordena Lista-grupo
Imprime Lista-grupo
```

3.3 Procesa Relación

/* 3.3.1 Consulta grupo */
Define reporte Relacion-gpo.prof.
Mientras haya registros en GRUPO **Haz**:
  Lee siguiente registro de GRUPO
  Graba registro en Relacion-gpo.prof.

  /* 3.3.2 Consulta asignatura */
  Busca *desc-asign* en ASIGN
    Donde ASIGN.*asign* = GRUPO.*asign* Y
    GRUPO*carrera* = ASIGN.*carrera*
  Graba *desc-asign* en Relacion-gpo.prof.

  /* 3.3.3 Consulta profesor */
  Busca *profesor* en PROFESOR
    Donde PROFESOR.*rfc* = GRUPO.*rfc*
  Graba *profesor* en Relacion-gpo.prof.
  Busca *profesor* en PROFESOR
    Donde PROFESOR.*rfc* = GRUPO.*rfc-asis*
  Graba *profesor* en Relacion-gpo.prof.
Fin-Mientras

/* 3.3.4 Elabora relación */
Ordena Relacion-gpo.prof.
Imprime Relacion-gpo.prof.

4 Elabora avance

Define reporte Rev.estudios
Define variables v-cuenta, v-carrera

/* 4.1 Consulta alumno */
Lee v-cuenta, v-carrera
Busca *cuenta* en ALUMNO
  Donde ALUMNO*cuenta* = v-cuenta **Y**
  ALUMNO*carrera* = v-carrera
Si encuentra registro **Entonces, Haz:**
  Graba registro en Rev.estudios

  /* 4.2 Consulta carrera */
  Busca *carrera* en CARRERA
   Donde CARRERA*carrera* = ALUMNO*carrera*
  Graba carrera en Rev.estudios

  /* 4.3 Consulta resúmen */
  Busca *cuenta* en RESUMEN
   Donde RESUMEN*cuenta* = ALUMNO*cuenta* **Y**
   RESUMEN*carrera* = ALUMNO*carrera*
  Graba registro en Rev.estudios

```
/* 4.4 Consulta historias */
Mientras haya registros en HISTORIA
 Donde HISTORIAcuenta = ALUMNOcuenta
     Y HISTORIAcarrera = ALUMNOcarrera, Haz:
   Lee siguiente registro de HISTORIA
   Graba registro en Rev.estudios

   /* 4.5 Consuta asignatura */
   Busca desc-asign en ASIGN
    Donde ASIGNasign = HISTORIAasign
   Graba desc-asign en Rev.estudios
 Fin-Mientras

/* 4.6 Elabora rev.estudios */
Sino
   Despliega "Número de cuenta erróneo"
Fin-Sino
```

5 Realiza cambios

```
/* 5.1 Verifica tipo */
Define variable v-tipo
Lee v-tipo
En caso de:
   v-tipo = 1, Haz: [5.2 Ingresa]
   v-tipo = 2, Haz: [5.3 Borra]
   v-tipo = 3, Haz: [5.4 Modifica]
   v-tipo = otro, Despliega "Tipo de cambio no válido"
Fin-En caso
```

5.2.2 Ingresa alumno

```
Define reporte Compr.cambio
Define variables v-cuenta, v-grupo, v-primer, v-carrera
Asigna verdadero a v-primer

/* 5.2.2.1 Captura alumno */
Lee v-cuenta, v-carrera
Busca cuenta en ALUMNO Donde ALUMNOcuenta = v-cuenta
 Y ALUMNOcarrera = v-carrera
Si encuentra registro Entonces Haz:
   Despliega "El alumno ya ha sido dado de alta"
Sino
   Lee Sol.ingreso
   Graba registro en ALUMNO
```

/* 5.2.2.4 Almacena datos */
Graba registro en CAMBIOS
Graba registro en Compr.cambio

/* 5.2.2.5 Elabora comprobante */
Imprime Compr.cambio
Fin-Si

5.3 B o r r a

Define reporte Compr.cambio
Define variables v-clave, v-cuenta, v-destino, v-carrera

/* 5.3.1 Valida clave */
Lee v-clave
Si v-clave = "clave permitida" **Entonces, Haz:**

/* 5.3.2 Consulta alumno */
Lee v-cuenta, v-carrera, v-destino
Busca *cuenta* en ALUMNO
Donde ALUMNO*cuenta* = v-cuenta
Si encuentra registro **Entonces Haz:**

/* 5.3.3 Valida ingreso */
Si ALUMNO*ingreso* < ALUMNO*ingreso* +2 **Entonces, Haz:**

```
        /* 5.3.4 Actualiza archivos */
        Borra registro en ALUMNO Donde ALUMNOcuenta =
         v-cuenta Y Alumnocarrera = v-carrera
        Mientras haya registros en INSCRIP Donde
         INSCRIPcuenta = v-cuenta Y ALUMNOcuenta =
         v-cuenta Haz:
           Borra siguiente registro de INSCRIP
        Fin-Mientras
        Mientras haya registros en HISTORIA Donde
         HISTORIAcuenta = v-cuenta Haz:
            Borra siguiente registro de HISTORIA
        Fin-Mientras
        Borra registro en RESUMEN Donde RESUMENcuenta = v-cuenta Y
          ALUMNOcarrera = v-carrera
      Sino
        Despliega "El alumno no puede ser dado de baja"
    Sino
      Despliega "El Alumno no existe"

    /* 5.3.5 Registra egreso */
    Graba registro en CAMBIO
    Graba v-destino en CAMBIOdestino
    Graba registro en Compr.cambio
    /* 5.3.6 Elabora comprobante */
    Imprime Compr.cambio
Sino
    Despliega "Clave de acceso no válida"
Fin-Si
```

5.4 Modifica

```
/* 5.4.1 Verifica tipo */
Define variable v-tipo
Lee v-tipo
En caso de:
    v-tipo = 1, Haz: [5.4.2 Cambia carrera]
    v-tipo = 2, Haz: [5.4.3 Cambia grupo]
    v-tipo = 3, Haz: [5.4.4 Registra calif.]
    v-tipo = otro, Despliega "Tipo de cambio no válido"
Fin-En caso
```

5.4.4 Registra calif.

```
Define reporte Compr.cambio
Define variables v-cuenta, v-destino, v-asign, v-carrera
Lee v-cuenta, v-carrera
Busca cuenta en ALUMNO Donde ALUMNOcuenta = v-cuenta
 Y ALUMNOcarrera = v-carrera
Si encuentra registro Entonces Haz:
    Lee v-asign
    Busca calif en HISTORIA Donde HISTORIAcuenta = v-cuenta Y
     HISTORIAasign = v-asign
    Si encuentra registro Entonces, Haz:
        Asigna v-asign a CAMBIOorigen
        Lee v-destino
        Graba registro en CAMBIO
```

Sino

Despliega "La asignatura no ha sido cursada"

Sino

Despliega "El alumno no existe"

Fin-Si

5.4.2 Cambia carrera

Define reporte Compr.cambio

Define variables v-clave, v-cuenta, v-destino, v-carrera

/* 5.4.2.1 Valida clave */

Lee v-clave

Si v-clave = "clave permitida" **Entonces, Haz:**

/* 5.4.2.2 Consulta alumno */

Lee v-cuenta, v-carrera

Busca *cuenta* en ALUMNO **Donde** ALUMNO*cuenta* = v-cuenta

Y ALUMNO*carrera* = v-carrera

Si encuentra registro **Entonces Haz:**

/* 5.4.2.3 Valida ingreso */

Si ALUMNO*ingreso* < ALUMNO*ingreso* +2 **Entonces, Haz:**

/* 5.4.2.4 Actualiza archivos */

Lee v-destino

Graba v-destino en ALUMNO*carrera*

```
        /* 5.4.2.5 Registra cambio */
        Graba registro en CAMBIO

        /* Elabora comprobante */
        Imprime Compr.cambio
    Sino
        Despliega "Cambio de carrera no permitido"
  Sino
    Despliega "Número de cuenta erróneo"
Sino
  Despliega "Clave de acceso no válida"
Fin-Si
```

5.4.3 Cambia grupo

```
Define reporte Compr.cambio
Define variables v-clave, v-cuenta, v-destino

/* 5.4.3.1 Valida clave */
Lee v-clave
Si v-clave = "clave permitida" Entonces, Haz:

  /* 5.4.3.2 Consulta alumno */
  Lee v-cuenta
  Busca cuenta en ALUMNO Donde ALUMNOcuenta = v-cuenta
  Si encuentra registro Entonces Haz:
```

```
    /* 5.4.3.3 Captura grupo*/
       Lee v-destino

    /* 5.4.3.4 Actualiza archivo */
    Mientras haya registros en INSCRIP Donde INSCRIPcuenta
     = v-cuenta y INSCRIPgrupo = v-destino, Haz:
       Graba v-destino en INSCRIPgrupo
    Fin-Mientras

    /* 5.4.3.5 Registra cambio */
    Graba registro en CAMBIO
    Graba registro en Compr.cambio

    /* 5.3.4.6 Elabora comprobante */
    Imprime Compr.cambio
  Sino
    Despliega "Número de cuenta erróneo"
Sino
  Despliega "Clave de acceso no válida"
Fin-Si
```

6 Registra grupos

Define variables v-rfc, v-grupo, v-profesor, v-asign, v-cupo, v-rfc-asis
Mientras haya Declaraciones gpo.cupo prof. **Haz:**

/* 6.1 Verifica profesor */
Lee v-rfc-asis
Busca *rfc* en PROFESOR **Donde** PROFESOR*rfc* = v-rfc-asis
Si no encuentra registro **Entonces, Haz:**
Lee v-profesor
Graba registro en PROFESOR
Fin-Si
Lee v-rfc
Busca *rfc* en PROFESOR **Donde** PROFESOR*rfc* = v-rfc
Si no encuentra registro **Entonces, Haz:**
Lee v-profesor
Graba registro en PROFESOR
Fin-Si
Mientras haya grupos para profesor **Haz:**

/* 6.3 Captura grupos */
Lee v-grupo, v-asign, v-cupo
Busca registro en GRUPO **Donde** GRUPO*gnupo* = v-grupo **Y** GRUPO*asign* = v-asign **Y** **GRUPO***rfc* = v-rfc

```
    Si encuentra registro Entonces, Haz:
        Despliega "El profesor ya ha sido asignado a este  grupo y asign"
    Sino

        /* 6.3 Graba grupos */
        Graba registro en GRUPOS
    Fin-Si
  Fin-Mientras
Fin-Mientras
```

7 Actaliza historiales

Define archivo AUXILIAR
Define variable v-clave

/* 7.1 Formatea */
Mientras haya Calificaciones **Haz:**
  Lee siguiente calificacacion
  Inserta separador de campo
  Graba registro en AUXILIAR
Fin-Mientras

/* 7.2 Valida clave */
Lee v-clave
Si v-clave = "clave permitida" **Entonces, Haz:**

  /* 7.3 Almacena datos */
  Mientras haya calificaciones en AUXILIAR , **Haz:**
    Lee siguiente registro de AUXILIAR
    Graba registro en HISTORIA
    Graba registro en RESUMEN
  Fin-Mientras
Sino
  Despliega "Clave de acceso no válida"
Fin-*Si*

La actividad principal del diseño es convertir los modelos del análisis en una estructura modular jerárquica. La primera estructura del diseño se crea analizando el nivel 0 de los diagramas de flujo; de esta manera se define un módulo principal que será identificado con el nombre del sistema y será el encargado de mandar ejecutar los módulos que llevarán a cabo las actividades del sistema.

La definición del diseño continúa con el análisis de los siguientes niveles de los diagramas de flujo de datos para distinguir los flujos de transacción de los de transformación.

Los flujos de transacción son aquellos que muestran los diversos caminos que puede seguir el sistema; estos flujos son transformados en el diseño en un módulo principal también llamado módulo ejecutivo encargado de llevar a cabo alguno de los caminos existentes, cada uno de los cuales es definido a su vez como un módulo subordinado al módulo ejecutivo.

Los flujos de transformación se identifican en el diagrama de flujo de datos como una secuencia de procesos encargada de transformar las entradas que recibe el sistema en las salidas que emite. Para cada uno de los flujos de transformación se define un módulo ejecutivo encargado de mandar ejecutar el módulo que obtendrá los datos (flujos aferentes), el módulo que realizará las transformaciones (flujo de transformación), así como el módulo encargado de proporcionar las salidas requeridas (flujo eferente).

Lo anterior dá como resultado una estructura modular inicial, que debe posteriormente ser analizada y refinada para verificar que los módulos establecidos son

altamente cohesivos, es decir, "no fragmentar los procesos esenciales en módulos y no juntar procesos no relacionados"[11]; se debe además buscar el menor acoplamiento posible, el cual es definido como el "grado en el cual los módulos se interconectan o relacionan entre si"[12].

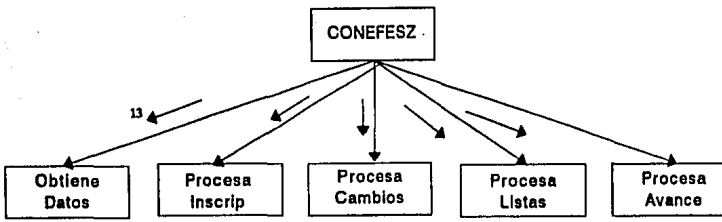
El número de procesos del nivel 0, no necesariamente debe de coincidir con el número de módulos definidos en la primera estructura del diseño, así pues los procesos de registra, registra grupo y actualiza historias fueron definidos dentro del módulo de obtiene datos ya que en todos los casos el sistema recibe los datos del exterior y ejecuta un proceso de carga de datos.

Para llevar a cabo la definición del diseño, se establecieron las siguientes especificaciones con el fin de facilitar el entendimiento de los diagramas.

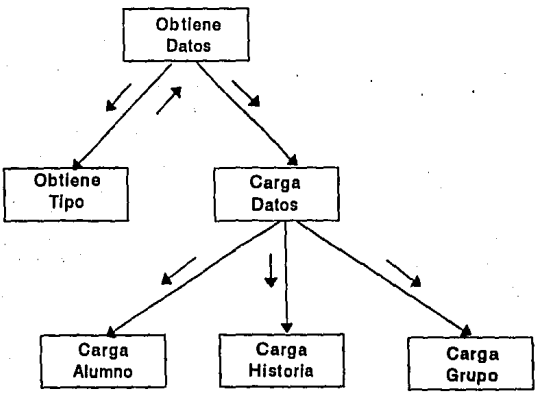
Actualiza	Modifica un registro de un archivo
Almacena	Agrega datos a un archivo
Borra	Elimina un registro de un archivo
Carga	Importa datos de un archivo en otro formato
Consulta	Lee datos de un archivo
Elabora	Proporciona reportes de salida
Formatea	Dá características específicas a un archivo de entrada
Genera	Realiza cálculos específicos
Lee	Realiza una lectura de datos de un archivo en otro formato
Obtiene	Realiza una lectura de datos de entrada
Valida	Realiza funciones de comparación

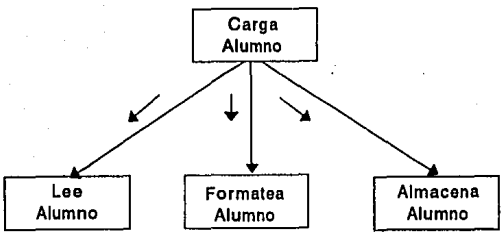
11,12 YOURDON, Edward, "MODERN STRUCTURED ANALYSIS", Prentice-Hall, USA. 1989, p. 465

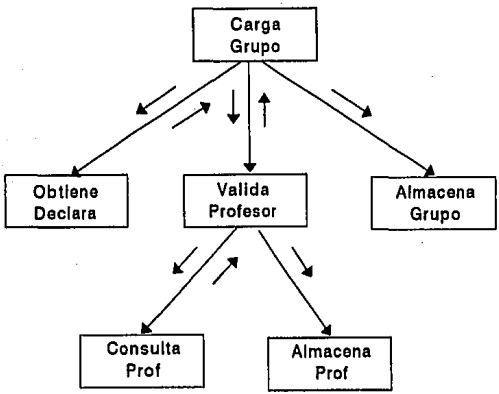
2.1 Definición de módulos

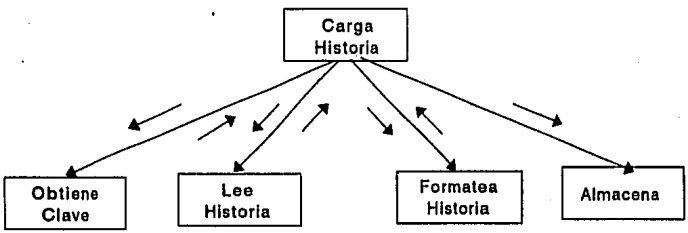


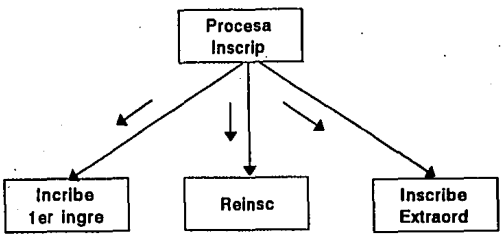
13 No se muestra el paso de parámetros entre módulos, debido a que su escritura no es necesaria para la comprensión de los mismos.

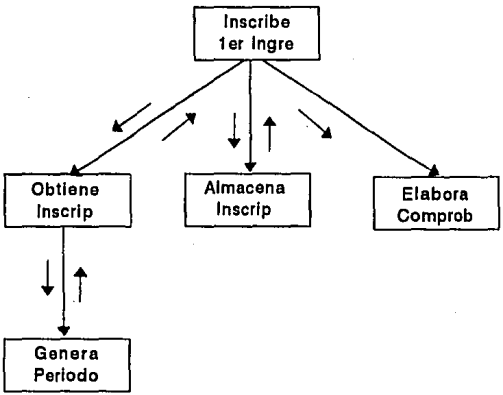

Obtiene Datos
Obtiene Tipo
Carga Datos
Carga Alumno
Carga Historia
Carga Grupo

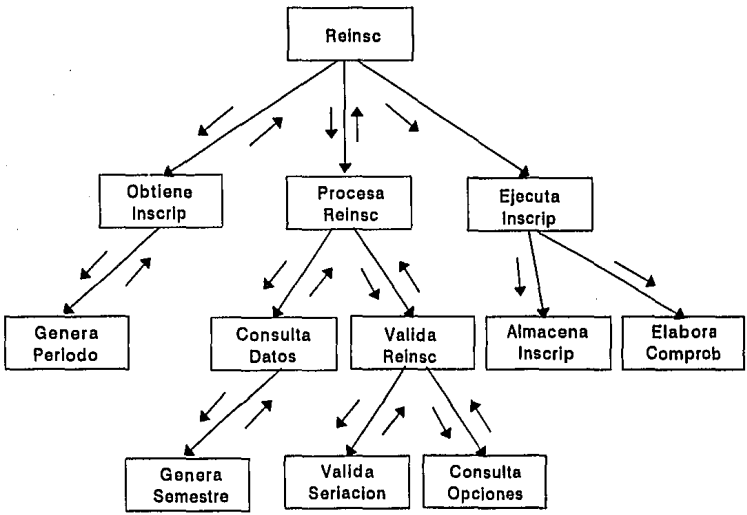

Carga
Alumno
Lee
Alumno
Formatea
Alumno
Almacena
Alumno

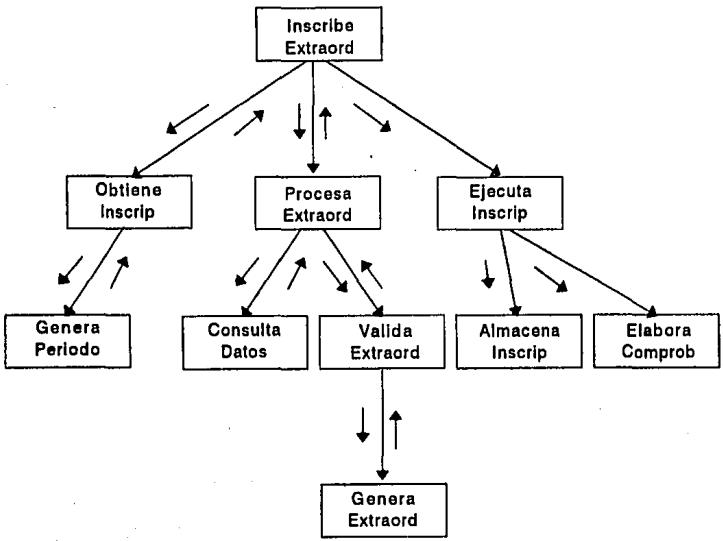

Carga Grupo
Obtiene Declara
Valida Profesor
Almacena Grupo
Consulta Prof
Almacena Prof

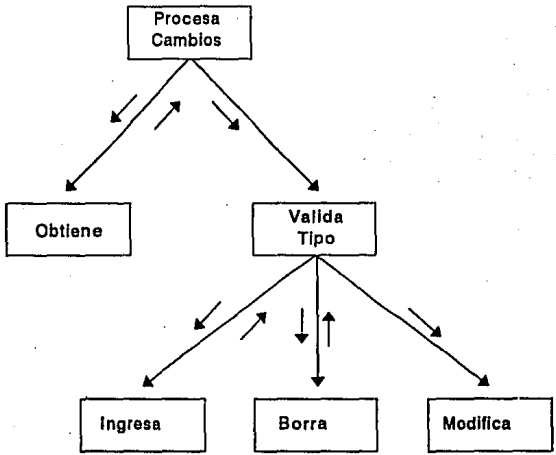

Carga Historia
Obtiene Clave
Lee Historia
Formatea Historia
Almacena

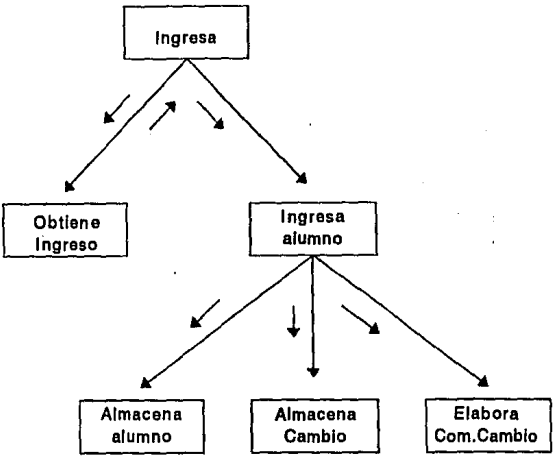

Procesa
Inscrip
Incribe
1er ingre
Reinsc
Inscribe
Extraord

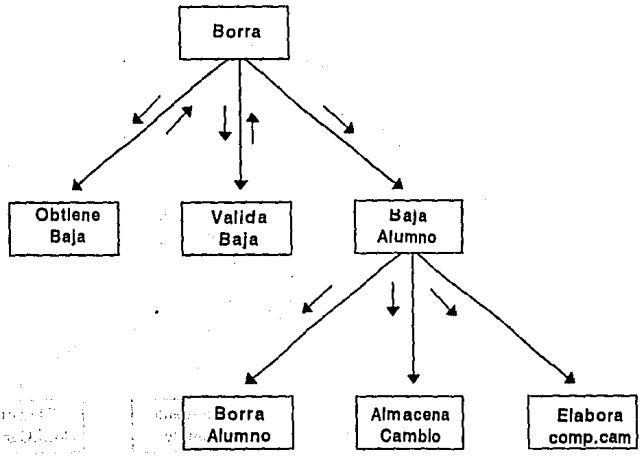

Inscribe
1er Ingre
Obtiene
Inscrip
Almacena
Inscrip
Elabora
Comprob
Genera
Periodo

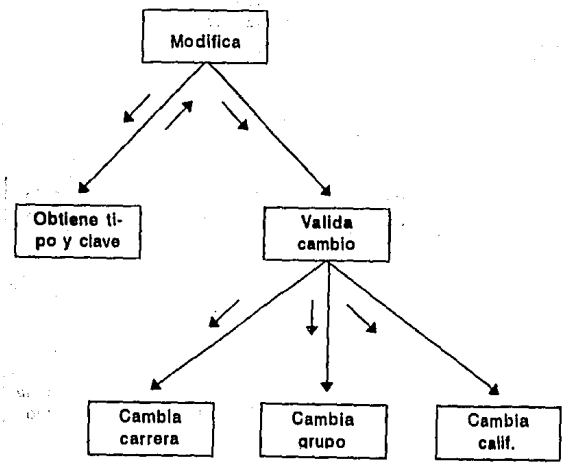

Reinsc
Obtiene Inscrip
Procesa Reinsc
Ejecuta Inscrip
Genera Periodo
Consulta Datos
Valida Reinsc
Almacena Inscrip
Elabora Comprob
Genera Semestre
Valida Seriacion
Consulta Opciones

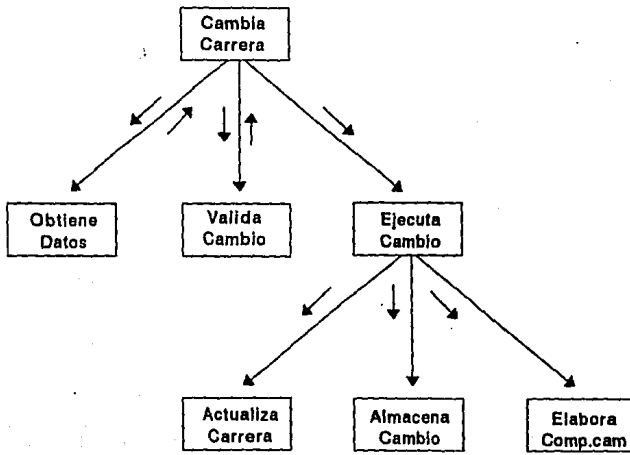

Inscribe Extraord
Obtiene Inscrip
Procesa Extraord
Ejecuta Inscrip
Genera Periodo
Consulta Datos
Valida Extraord
Almacena Inscrip
Elabora Comprob
Genera Extraord

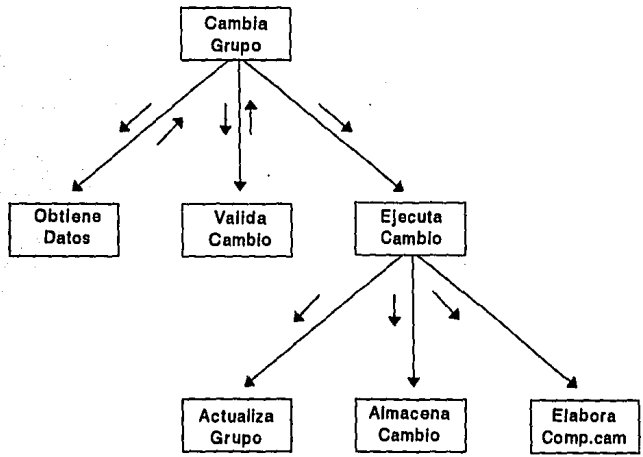

Procesa
Cambios
Obtiene
Valida
Tipo
Ingresa
Borra
Modifica

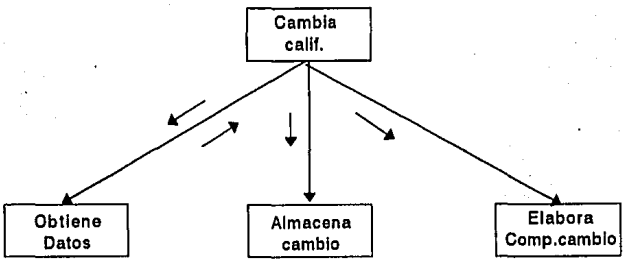


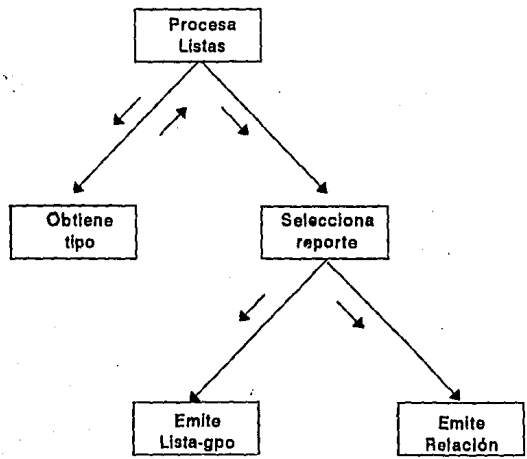

Borra
Obtiene Baja
Valida Baja
Baja Alumno
Borra Alumno
Almacena Cambio
Elabora comp.cam

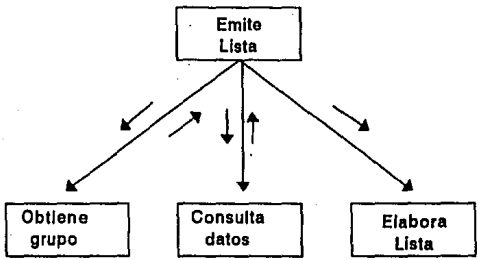

Modifica
Obtiene tipo y clave
Valida cambio
Cambia carrera
Cambia grupo
Cambia calif.

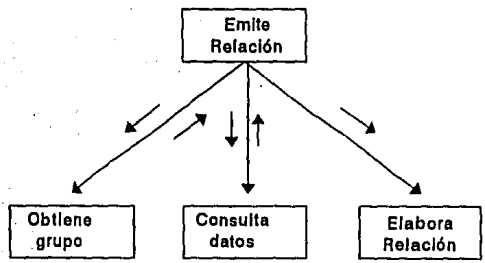

Cambia Carrera
Obtiene Datos
Valida Cambio
Ejecuta Cambio
Actualiza Carrera
Almacena Cambio
Elabora Comp.cam

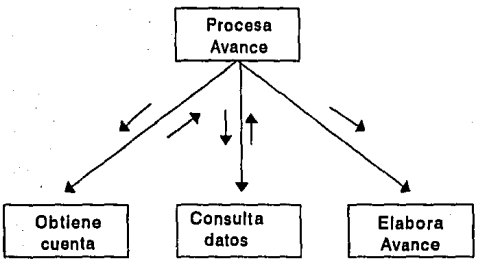

Cambia Grupo
Obtiene Datos
Valida Cambio
Ejecuta Cambio
Actualiza Grupo
Almacena Cambio
Elabora Comp.cam


Cambia
calif.
Obtiene
Datos
Almacena
cambio
Elabora
Comp.cambio


Procesa
Listas
Obtiene
tipo
Selecciona
reporte
Emite
Lista-gpo
Emite
Relación


Emite
Lista
Obtiene
grupo
Consulta
datos
Elabora
Lista


Emite
Relación
Obtiene
grupo
Consulta
datos
Elabora
Relación


Procesa Avance
Obtiene cuenta
Consulta datos
Elabora Avance

3.1 Programación estructurada

La siguiente fase en el desarrollo del sistema es la programación, la cual "involucra la escritura de instrucciones en algún lenguje de programación para implantar lo que el analista ha especificado y el diseñador ha organizado en módulos" [14]. En el desarrollo de CONEFESZ se utilizó el lenguaje de programación del manejador de base de datos Progress versión 6.0.

Para la elaboración de los programas fuentes se definieron los siguientes estándares de programación de manera que se facilitara la comprensión y el mantenimiento de los programas.

- Los nombres de programas inician con las dos primeras letras del módulo al que correspondan y terminan con la extensión *.p* .

- Los nombres de programas de menúes inician con la palabra menu, seguidos de un nombre nemotécnico y la terminación *.p* .

- El nombre y descripción del programa se colocan al inicio del mismo.

- La definición de variables se realiza después del nombre del programa.

- Los nombres de variables inician con *v-*, seguidos de un nombre nemotécnico.

14 YOURDON, Edward, "MODERN STRUCTURED ANALYSIS", Prentice-Hall, USA. 1989, p. 471

- Los nombres de variables de trabajo que referencian campos de archivos inician con *v-*, seguidos del nombre del campo.

- La definición de archivos de trabajo se realiza después de la definición de las variables.

- Los nombres de archivos de trabajo inician con *w-*, seguidos de un nombre nemotécnico.

- La definición de los formatos de pantalla y reportes se realiza después de la definición de los archivos de trabajo.

A medida que los programas fueron desarrollados la estructura del diseño fué modificándose, ya que en ocasiones, el programa tendía a ser muy grande o a duplicar código en diversos programas.

3.2 Pruebas

Es importante verificar que el sistema opera de acuerdo a las especificaciones establecidas, para lograr esto se realizaron los siguientes dos tipos de pruebas:

Pruebas funcionales: Se aplicó una estrategia de enfoque descendente, para lo cual se probó que los módulos de nivel más alto ejecutaran correctamente los módulos subordinados; posteriormente se probaron los módulos subordinados hasta llegar a los

módulos de nivel más bajo. "Este tipo de prueba conocida como TOP-DOWN es una estrategia que prueba los módulos de más alto nivel, antes de que hayan sido codificados los módulos de bajo nivel y posiblemente antes de que hayan sido diseñados." [15]

Para poner en práctica esta estrategia se comenzó con la programación de los módulos jerárquicamente más altos con el fin de presentarlos periódicamente con los directivos, para así programar los menúes, pantallas principales y formatos tal como ellos los solicitaban. Este proceso fué utilizado durante toda la etapa de la programación.

Al pasar de los módulos ejecutivos a los módulos subordinados, se realizaron casos de prueba generados a partir de las especificaciones o salidas deseadas en la etapa de análisis. Una vez obtenidos los resultados esperados, se presentó a los usuarios que operarían el sistema, los cuales dieron sus puntos de vista y se les pidió que generaran casos de prueba excepcionales que con su experiencia han observado.

Finalmente, se capacitó al personal y durante esta etapa se adicionaron los programas y mensajes de ayuda en línea. Esta última fase se efectuó con la finalidad de hacer más comprensible y amigable el sistema y preparar al personal para hacer una prueba en paralelo durante las inscripciones del período 94-1.

15 YOURDON, Edward, "MANAGING THE STRUCTURED TECHNIQUES", Prentice-Hall, USA. 1989, 4a. ed p. 67

Pruebas de desempeño: Para asegurar la eficiencia de CONEFESZ se aplicó este tipo de pruebas el cual tiene como objetivo "asegurar que el sistema pueda manejar el volúmen de datos y transacciones de entrada especificados en el modelo de implantación de usuario, y asegurar que tenga el tiempo de respuesta requerido"[16].

Habiendo terminado de programar los módulos subordinados se solicitaron los catálogos a la DGAE para cargarlos al sistema. Se hicieron pruebas con volúmenes de 4,000 registros en el catálogo de alumnos, 160,000 registros en el catálogo de historias, y tres estaciones de trabajo operando simultáneamente para observar los tiempos de respuesta. En esta etapa se realizaron modificaciones a los índices de los archivos para hacer más eficiente el desempeño del sistema.

Una vez probados los aspectos de funcionamiento y desempeño, se realizó una prueba en paralelo con la carrera de psicología para las inscripciones del período 94-1, las cuales se llevaron a cabo durante el mes de septiembre. Estos resultados fueron enviados a la DGAE para ser procesados y posteriormente recibir el diagnóstico correspondiente. En base a esta prueba CONEFESZ logró reducir en un 60% el tiempo empleado para las inscripciones con respecto al promedio observado en periodos anteriores.

16 YOURDON, Edward, "MODERN STRUCTURED ANALYSIS", Prentice-Hall, USA. 1989, p. 482

La fase final del desarrollo consiste en definir las condiciones de medio ambiente requeridas para que pueda ser ejecutado el sistema así como la adición de los programas en código objeto y la estructura de la base de datos y finalmente la carga de datos y catálogos básica.

4.1 Medio ambiente

CONEFESZ está diseñado para operar en modo monousuario o multiusuario bajo ambientes DOS y Novell 386, respectivamente; en ambos casos se requiere la presencia del PROGRESS RUNTIME V. 6.0, el cual requiere:

"- Espacio en disco duro de 13 MB.
- MS-DOS v. 3.1 o mayores
- 640 kb de RAM" [17]

Adicionalmente a lo anterior, para la ejecución del sistema en modo multiusuario, se requiere un servidor con procesador 80386 o 80486 corriendo bajo el sistema operativo Netware 3.11 de Novell o superior; memoria RAM mínima de 1 MB y la instalación del software NLM.

"La instalación del software NLM de PROGRESS, se debe de realizar en el directorio raíz de la red mapeado al directorio SYS:\ del servidor de archivos de la red. El archivo de configuración de PROGRESS (progress.cfg) debe recidir en el directorio \dlc" [18]

17, 18 PROGRESS Software Corporation. Reference Manual, U.S.A. p.p 10,15

4.2 Instalación

Para la instalación del sistema se generó el archivo instala.bat el cual realiza las siguientes instrucciones:

Crea el directorio PRODES,
Copia los programas objeto (archivos con extensión .R),
Copia los archivos de la base de datos (archivos con extensión .DB .BI y .LG)
Copia el archivo que ejecuta el sistema (archivo CONEFESZ.BAT)

4.3 Carga básica de datos

Para realizar la carga de los catálogos básicos y así arrancar la operación diaria del sistema, se debe seleccionar la opción Carga archivos del menú principal, el cual contempla las siguientes opciones:

- Alumnos - Historias - Grupos

La etapa de instalación del sistema puede llevarse a cabo en forma radical o como un proceso gradual en el cual "un grupo de usuarios van recibiendo manuales y entrenamiento y comenzando a usar el nuevo sistema"[19].

La instalación de CONEFESZ se llevó a cabo siguiendo un proceso gradual el cual, se inició con la inscripción en paralelo de una de las carreras, y la capacitación de un grupo de usuarios. Posteriormente se capacitó al resto del personal para la puesta en marcha del sistema y su aplicación a toda la población estudiantil.

19 YOURDON, Edward, "MODERN STRUCTURED ANALYSIS", Prentice-Hall, USA. 1989, p. 471

La importancia que tiene la aplicación de una metodología formal se ve reflejada en todo el ciclo de vida de un sistema; en las primeras etapas del desarrollo, nos ayuda a definir los límites y alcances, y nos abre grandes canales de comunicación con el usuario que facilitan la comprensión de sus necesidades, asimismo, en las últimas etapas nos auxilia en el desarrollo de un sistema bien estructurado y documentado. Sin embargo, no hay que perder de vista que una metodología solo proporciona herramientas que facilitan el desarrollo del sistema y enfatizan los puntos básicos que hay que tomar en cuenta para el éxito del objetivo planteado, por lo tanto estas herramientas pueden sufrir algunas adaptaciones que faciliten la actividad de desarrollo del sistema.

En los diagramas de flujo de datos, todas las entradas o salidas deben de corresponder con las del nivel anterior, sin embargo, partiendo del objetivo básico de un modelo, el cual es facilitar la comprensión y comunicación con el usuario, en algunas ocasiones omitimos entradas que no eran indispensables para el entendimiento de un diagrama y por el contrario dificultaban su comprensión. Por otro lado, la metodología muestra en los diagramas de flujo de datos, los archivos que utiliza cada proceso, sin que estos hayan sido definidos en su totalidad, ocasionando inconsistencias y redundancia de datos, por lo cual realizamos la normalización de los archivos conjuntamente con la definición de los dfd's.

Para la transformación de los modelos del análisis al diseño, la metodología define una secuencia de pasos independientes de la etapa anterior y que no toman en cuenta los aspectos lógicos del sistema, razón por la cual, la estructuración del diseño se realizó

tomando en consideración la lógica de los procesos definidos en los diagramas de flujo de datos y posteriormente se refinó tomando en consideración los aspectos de cohesión y acoplamiento que define la metodología. Por otro lado, para facilitar la comprensión de los diagramas de estructura, omitimos los nombres de pasos de parámetros debido a que estos se sobreentienden con los nombres de los procesos.

La etapa del diseño se presenta como una jerarquía de módulos, cada uno de los cuales se convertirá en un programa o en una parte del mismo. Al pasar a la etapa de programación y definir el código fuente, algunos programas crecieron demasiado debido a que en el diseño no se muestran los aspectos de presentación y toda la complejidad de los módulos, ocasionando que los programas tuvieran que ser reestructurados junto con los módulos del diseño generados anteriormente.

La metodología define en la etapa del análisis la realización de un estudio de factibilidad, que permita evaluar el costo del sistema contra los beneficios que aportará el mismo para determinar si es factible de ser desarrollado; sin embargo, para la realización de CONEFESZ, no se hizo este estudio debido a que la facultad contaba ya con los recursos necesarios, y el sistema manual que se tenía presentaba muchas deficiencias.

Debido a la limitación de los recursos de la facultad, para las etapas de análisis y diseño, no fué posible utilizar una herramienta CASE, que facilitara el desarrollo y corrección de los modelos gráficos, validara las especificaciones creadas para detectar ambigüedades o inconsistencias y permitiera incrementar la productividad de la etapa de la programación al generar código fuente de manera automatizada; habiendo carecido de una herramienta de este tipo, trabajamos en forma semimanual con graficadores muy limitados.

En lo que se refiere a los procesos que realiza el sistema, debido a que la DGAE envía las historias académicas a la FESZ, en fechas posteriores a las fechas de inscripción, el sistema no puede evaluar correctamente la seriación de las asignaturas, permitiendo realizar la inscripción de una materia aún cuando la asignatura seriada del semestre anterior no haya sido aprobada. Por otra parte, los criterios de depuración de archivos no se han terminado de definir correctamente y pueden crecer semestre a semestre, disminuyendo la eficiencia del sistema.

Con el resultado de las inscripciones del periodo 94-1, podemos afirmar que la confiabilidad de CONEFESZ es alta, ya que según el diagnóstico emitido por la DGAE, el porcentaje de eficiencia fué del 99.4; sin embargo consideramos que las ayudas en línea que tiene el sistema son deficientes y la presentación de los menúes y pantallas son monótonas y al usuario le cuesta trabajo ubicar en qué parte del sistema se encuentra.

Por la importancia de la información que maneja el sistema, CONEFESZ puede seguir creciendo para proporcionar estadísticas en forma automática, que faciliten la evaluación y dirección del plantel; asimismo, se pueden agregar nuevos módulos como lo son, el proceso de inscripción a exámenes profesionales, registro de tesis y servicio social, entre otros.

Finalmente, podemos afirmar que este trabajo nos permitió desarrollar habilidades básicas que debe tener toda persona que desarrolle sistemas, como lo son, capacidad para comprender las necesidades de los usuarios, habilidad en la coordinación de equipos de trabajo y seguridad en el trato con directivos.

- Yourdon, Edward. Modern structured analysis, Prentice Hall, 1a. ed, USA, 1989.

- Yourdon, Edward y Constantine, Larry. Structured Design, Prentice Hall, USA, 1989.

- Yourdon, Edward. Managing the structured techniques, Prentice Hall, 4a ed., USA, 1989

- Atre, Shakuntala. Técnicas de bases de datos, Trillas, México, 1989.

- Progress Software Corporation. Reference Manual, USA.

- Progress Software Corporation. User's Guide, USA.

ANEXOS

```
/*     <<<<<<<<< CONEFESZ >>>>>>>>>>>>     */


/* <MENUPRIN.P> menu principal*/

/* Definicion de variables  */
DEF VAR J AS INTEGER.
DEF VAR FES AS CHAR FORMAT "X(13)" INITIAL "FES ZARAGOZA".

/* Definición de opciones del menú principal */
DEF VAR MENU1 AS CHAR EXTENT 6 FORMAT "X(16)"
   INITIAL ["INSCRIPCIONES",
            "GENERA LISTAS",
            "ELABORA AVANCE",
            "REALIZA CAMBIOS",
            "CARGA ARCHIVOS",
            "UTILERIAS"].

/*  Definición de programas que se pueden elegir  */
DEF VAR programa AS CHAR EXTENT 6 FORMAT "X(11)"
   INITIAL ["MENUINSC.P",
            "MENULIST.P",
            "MENUAVAN.P",
            "CAMBIOS.P",
            "CARGA.P",


/*  Definición de formatos de pantalla  */
FORM HEADER
  FES AT 2 "MENU PRINCIPAL" AT 33 TODAY AT 69
  FORMAT "99-99-9999" WITH PAGE-TOP WIDTH 80 TITLE "C O N E F E S Z"
  CENTER FRAME TITULO ROW 1.

/*  Módulo principal  */
REPEAT:
   HIDE ALL.
   VIEW FRAME TITULO.
   PUT SCREEN ROW 5 COLUMN 37 COLOR MESSAGE  " MENU ".
   DISPLAY MENU1 WITH NO-LABELS FRAME MEN1 1 COLUMN CENTER ROW 6.
   MESSAGE
   "  |" + CHR(17) + CHR(196) + CHR(196) + CHR(217) +  "|
   SELECCIONAR OPCION  |" + CHR(24) + "|  ANTERIOR    |" + CHR(25)
   + "|  SIGUIENTE    | F4|  SALIR  ".
   CHOOSE FIELD MENU1 AUTO-RETURN WITH FRAME MEN1.
   IF CAN-DO("RETURN,PICK,GO",KEYFUNCTION(LASTKEY)) THEN
     DO J= 1 TO 6:
       IF FRAME-VALUE = MENU1[J] THEN LEAVE.
     END.
```

```
  ELSE DO:
    BELL.
    NEXT.
  END.
  HIDE MESSAGE.
  PAUSE 0.
  HIDE ALL.
  IF SEARCH(PROGRAMA[J]) = ? THEN
    MESSAGE "EL PROGRAMA:" PROGRAMA[J] "NO EXISTE".
  ELSE RUN VALUE(PROGRAMA[J]).
END.
/*QUIT*/

/*          << FIN CONEFESZ >>      */

**********************************************************************

/* <<<<<<<<<<<<<< PROCESA INSCRIP >>>>>>>>>>>>>>>  */

/* <MENUINSC.P> menu de inscripciones */

/*  Definición de variables  */
DEF VAR J AS INTEGER.
DEF VAR FES AS CHAR FORMAT "X(13)" INITIAL "FES ZARAGOZA".

/* Definición de opciones de menú de inscripciones */
DEF VAR MENU1 AS CHAR EXTENT 3 FORMAT "X(26)"
   INITIAL ["PRIMER INGRESO",
            "REINSCRIPCION",
            "INSCRIPCION EXTRAORDINARIA"].

/* Definición de los programas que se pueden ejecutar  */
DEF VAR PROGRAMA AS CHAR EXTENT 7 FORMAT "X(10)"
   INITIAL ["INSCRIP.P","REINSC.P","EXTRA.P"].

/* Definición de formatos de pantalla    */
FORM HEADER
  FES AT 2 "MENU DE INSCRIPCIONES" AT 30 TODAY AT 69
  FORMAT "99-99-9999" WITH PAGE-TOP WIDTH 80 TITLE "C O N E F E S Z"
  CENTER FRAME TITULO ROW 1.

REPEAT:
   HIDE ALL.
   VIEW FRAME TITULO.
   PUT SCREEN ROW 5 COLUMN 37 COLOR MESSAGE  " MENU ".
   DISPLAY MENU1 WITH NO-LABELS FRAME MEN1 1 COLUMN CENTER ROW 6.
   MESSAGE
```

```
" |" + CHR(17) + CHR(196) + CHR(196) + CHR(217) + "| SELECCIONAR OPCION
|" + CHR(24) + "| ANTERIOR |" + CHR(25) + "| SIGUIENTE |F4| SALIR ".
 CHOOSE FIELD MENU1 AUTO-RETURN WITH FRAME MEN1.
 IF CAN-DO("RETURN,PICK,GO",KEYFUNCTION(LASTKEY)) THEN
  DO J= 1 TO 3:
   IF FRAME-VALUE = MENU1[J] THEN LEAVE.
  END.
 ELSE DO:
  BELL.
  NEXT.
 END.
 HIDE MESSAGE.
 PAUSE 0.
 HIDE ALL.
 IF SEARCH(PROGRAMA[J]) = ? THEN
  MESSAGE "EL PROGRAMA:" PROGRAMA[J] "NO EXISTE".
 ELSE RUN VALUE(PROGRAMA[J]).
END.
/*QUIT*/

/*          << FIN PROCESA INSCRIP >>      */

**********************************************************************


/* <<<<<<<<<<<<<<<  INSCRIBE 1er. INGRESO  >>>>>>>>>>>>>> */

/*<INSCRIP.P> inscribe alumnos de primer ingreso*/

/*  Definición de variables  */
DEF NEW SHARED VAR VGPO AS CHA FORMAT "XXXX".
DEF NEW SHARED VAR VCARRERA LIKE CARRERA.CARRERA.
DEF NEW SHARED VAR VCTA LIKE ALUMNO.CUENTA.
DEF NEW SHARED VAR CREDI LIKE ASIGN.CREDITOS.
DEF NEW SHARED VAR VPLANTEL LIKE ALUMNO.PLANTEL.
DEF NEW SHARED VAR VALUMNO LIKE ALUMNO.ALUMNO.
DEF NEW SHARED VAR VINGRESO LIKE ALUMNO.INGRESO.
DEF NEW SHARED VAR VPERIOD AS CHA.
DEF NEW SHARED VAR VANIO AS INT.
DEF VAR IMPRESION AS CHA FORMAT "X".
DEF VAR EXISTE AS LOG.

/*  Definición de archivos de trabajo  */
DEF NEW SHARED WORKFILE VARCH
FIELD WCARRERA LIKE ALUMNO.CARRERA
FIELD WASIGN LIKE ASIGN.ASIGN
FIELD WDESC LIKE ASIGN.DESC-ASIGN
FIELD WCREDITOS LIKE ASIGN.CREDITOS
```

```
FIELD WSEMESTRE LIKE ASIGN.SEMESTRE
FIELD WGRUPO LIKE GRUPO.GRUPO.

/*  Definición de formatos de pantalla  */
FORM  "  " SKIP
    "UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO " AT 20 SKIP
    "CONTROL ESCOLAR FESZ ZARAGOZA" AT 24 SKIP
     WITH WIDTH 80 CENTERED FRAME TITULO.

FORM
  SKIP(1)
  VCTA LABEL "CUENTA" AT 15 SPACE(11) VCARRERA LABEL "CARRERA"
ALUMNO.ALUMNO
  ALUMNO.DIRECCION COLUMN-LABEL "DIRECCION"
  ALUMNO.TEL COLUMN-LABEL "TELEFONO" VGPO LABEL "GRUPO"
  WITH WIDTH 70 CENTERED SIDE-LABELS
  TITLE "INSCRIPCION DE ALUMNOS DE PRIMER INGRESO" FRAME ALU.


/*  <<<<<<<<<<<<  OBTIENE INSCRIP  >>>>>>>>>>>>>>>>  */


RUN PERIODO.P.   /*  (
VANIO = YEAR(TODAY) - 1900 + 1.

REPEAT:
 HIDE ALL.
 EXISTE = FALSE.
 HIDE MESSAGE.
 MESSAGE "F4 SALIR".
 VIEW FRAME TITULO.
 UPDATE VCTA help "NUMERO DE CUENTA DEL ALUMNO"
 VCARRERA HELP "CLAVE DE LA CARRERA" WITH FRAME ALU.
 FIND ALUMNO WHERE ALUMNO.CUENTA = VCTA AND
   ALUMNO.CARRERA = VCARRERA NO-ERROR.
 IF NOT AVAILABLE ALUMNO THEN DO:
   MESSAGE "CUENTA O CARRERA ERRONEA.FAVOR DE RECTIFICAR".
   PAUSE NO-MESSAGE.
   NEXT.
 END.
 ELSE DO:

  /*  valida que el alumno sea de primer ingreso  */
  IF VANIO > ALUMNO.INGRESO THEN DO:
   MESSAGE "EL ALUMNO NO ES DE PRIMER INGRESO".
   PAUSE NO-MESSAGE.
   NEXT.
  END.
  ELSE DO:
```

```
   /* actualiza la dirección del alumno */
   DISPLAY ALUMNO.ALUMNO WITH FRAME ALU.
   UPDATE DIRECCION TEL WITH FRAME ALU.
   REPEAT:
   /* solicita el grupo al alumno */
    UPDATE VGPO HELP "NUMERO DE GRUPO" WITH FRAME ALU.
    FOR EACH GRUPO WHERE GRUPO.GRUPO = VGPO AND GRUPO.CARRERA =
VCARRERA:
     EXISTE = TRUE.
    END.
    IF NOT EXISTE THEN DO:
     MESSAGE "NUMERO DE GRUPO ERRONEO".
     PAUSE NO-MESSAGE.
     NEXT.
    END.
    ELSE DO:
     HIDE MESSAGE.
     VCTA = ALUMNO.CUENTA.
     VPLANTEL = ALUMNO.PLANTEL.
     VALUMNO = ALUMNO.ALUMNO.
     VINGRESO = ALUMNO.INGRESO.
/*            << FIN OBTIENE INSCRIP >>          */

/*    <<<<<<<<<<<<<<  ALMACENA INSCRIP  >>>>>>>>>>>>>>>>>   */

    FOR EACH GRUPO WHERE VGPO = GRUPO.GRUPO AND GRUPO.CARRERA =
VCARRERA:
     DISPLAY GRUPO.ASIGN WITH CENTERED.
     CREATE INSCRIP.
     INSCRIP.CUENTA = ALUMNO.CUENTA.
     INSCRIP.GRUPO = GRUPO.GRUPO.
     INSCRIP.ASIGN = GRUPO.ASIGN.
     INSCRIP.CARRERA = ALUMNO.PLAN.
     INSCRIP.TIPO = 1.
     FIND ASIGN WHERE ASIGN.ASIGN = GRUPO.ASIGN AND
       ASIGN.CARRERA = VCARRERA NO-ERROR.
     CREDI = CREDI + ASIGN.CREDITOS.
     /* crea arcivo de trabajo para imprimir comprobante */
     CREATE VARCH.
      WCARRERA = VCARRERA.
      WASIGN = ASIGN.ASIGN.
      WDESC = ASIGN.DESC-ASIGN.
      WCREDITOS = ASIGN.CREDITOS.
      WSEMESTRE = ASIGN.SEMESTRE.
      WGRUPO = GRUPO.GRUPO.
    END.

/*            << FIN  ALMACENA INSCRIP   >>        */
```

```
    DISPLAY "DESEA ELABORACION DE COMPROBANTE?" WITH COLUMN 1
       OVERLAY ROW 20 FRAME IMPR.
    UPDATE IMPRESION WITH NO-LABELS FRAME IMPR.
    IF IMPRESION = "S" THEN RUN IMPRIME.P.
    LEAVE.
   END.
  END.
 END.
END.


/*        <<    FIN INSCRIBE 1er. INGRESO    >>        */

*****************************************************************

/*    <<<<<<<<<<<<  GENERA PERIODO  >>>>>>>>>>>    */
/*PERIODO.P*/
/*CALCULA EL PERIODO CORRESPONDIENTE PARA EL PROGRAMA DE EXTRA.P*/
DEF SHARED VAR VPERIOD AS CHAR.
DEF VAR ANO AS INT.

 ANO = YEAR(TODAY).
 ANO = ANO - 1900.
 VPERIOD = STRING(ANO).
 IF (MONTH(TODAY)>=8 AND MONTH(TODAY)<=12) OR (MONTH(TODAY)=1) THEN DO:
 VPERIOD = STRING(ANO + 1).
 VPERIOD = VPERIOD + "-1".
 END.
 IF (MONTH(TODAY)>=2 AND MONTH(TODAY)<=7) THEN VPERIOD = VPERIOD + "-2".


 /*        <<    FIN GENERA PERIODO    >>        */

*****************************************************************
/*    <<<<<<<<<<<<<<<  ELABORA COMPROB.  >>>>>>>>>>>>>>>    */

/*IMPRIME.P
 PROG. DE IMPRESION DE COMPROBANTES DE INSCRIPCION A EXAMEN
EXTRAORDINARIOS
            UTILIZADO POR EXTRA.P/*

DEF SHARED VAR CREDI LIKE ASIGN.CREDITOS.
DEF SHARED VAR VPLANTEL LIKE ALUMNO.PLANTEL.
DEF SHARED VAR VALUMNO LIKE ALUMNO.ALUMNO.
DEF SHARED VAR  VINGRESO LIKE ALUMNO.INGRESO.
DEF SHARED VAR VCTA LIKE ALUMNO.CUENTA.
DEF SHARED VAR VCARRERA LIKE ALUMNO.CARRERA.
DEF SHARED VAR VPERIOD AS CHA.
```

```
/* Datos del archivo temporal provenientes de extra.p */

DEF SHARED WORKFILE VARCH
  FIELD WCARRERA LIKE ALUMNO.CARRERA
  FIELD WASIGN LIKE ASIGN.ASIGN
  FIELD WDESC LIKE ASIGN.DESC-ASIGN
  FIELD WCREDITOS LIKE ASIGN.CREDITOS
  FIELD WSEMESTRE LIKE ASIGN.SEMESTRE
  FIELD WGRUPO LIKE GRUPO.GRUPO.

FORM HEADER
 TODAY
    "          UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO" SKIP
    "             DIRECCION GENERAL DE ADMINISTRACION ESCOLAR" SKIP
    "                    COMPROBANTE DE INSCRIPCION"
    SKIP WITH WIDTH 80 CENTERED FRAME E.

FORM
    VPLANTEL COLUMN-LABEL "PTL" AT 2
    VCARRERA COLUMN-LABEL "CAR" AT 6
    VCTA COLUMN-LABEL "N-CUENTA" AT 11
    VALUMNO COLUMN-LABEL "NOMBRE DEL ALUMNO" AT 22
    VINGRESO COLUMN-LABEL "ADEI" AT 56
    VPERIOD COLUMN-LABEL "PDO." AT 62
    WITH WIDTH 80 CENTERED FRAME E1.


  HIDE ALL.
  OUTPUT TO PRINTER PAGE-SIZE 66.
  VIEW FRAME E.

  /* Impresion de datos */

  DISPLAY VPLANTEL VCARRERA VCTA VALUMNO VINGRESO VPERIOD WITH FRAME E1.
  FOR EACH VARCH:
     DOWN WITH FRAME E2.
     DISPLAY  WCARRERA AT 4 WASIGN WDESC COLUMN-LABEL "NOMBRE DE
ASIGNATURA"
              WCREDITOS WSEMESTRE WGRUPO WITH WIDTH 80 CENTERED
              FRAME E2 DOWN.
  END.
  DISPLAY SKIP.
  DISPLAY " ----------------------------------------------------------------
  ".
  DISPLAY "TOTAL: " AT 43 CREDI AT 50 WITH NO-LABELS.
  DISPLAY SKIP(3).
  DISPLAY "COMPROBANTE SIN VALIDEZ OFICIAL".
  DISPLAY "FAVOR DE RECOGER SU TIRA DE MATERIAS EN LAS FECHAS PUBLICADAS".
  PAGE.
```

```
OUTPUT CLOSE.

/*  Borrado del archivo temporal de datos  */

FOR EACH VARCH:
   DELETE VARCH.
END. /* end for each */

/*          <<    FIN ELABORA COMPROB.    >>      */


**********************************************

/*    <<<<<<<<<<<<<<<<<    REINSC   >>>>>>>>>>>>>>  */

/*<REINSC.P> inscribe alumnos que han cursado mas de un semestre*/
/* Definición de variables  */
DEF VAR M AS INT.
DEF VAR SEMGPO AS INT FORMAT ">9".
DEF VAR ANIO AS INT.
DEF VAR FECHA AS DATE.
DEF VAR TIEMPO AS INT.
DEF VAR SEMALU AS INT.
DEF VAR IRREGU AS CHAR FORMAT "X(1)".
DEF VAR ART19 AS LOG.
DEF VAR MARCA AS CHAR FORMAT "X(1)". /*variable de marca de inscrito*/

/*variables compartidas con el procedimiento <IMPRIME.P>*/

DEF NEW SHARED VAR REG AS LOG FORMAT "S/N".
DEF NEW SHARED VAR VGPO AS CHA FORMAT 9999.
DEF NEW SHARED VAR VGPO2 LIKE GRUPO.GRUPO.
DEF NEW SHARED VAR VCARRERA LIKE CARRERA.CARRERA.
DEF NEW SHARED VAR VASIG AS INT.
DEF NEW SHARED VAR VCTA LIKE ALUMNO.CUENTA.
DEF NEW SHARED VAR CREDI LIKE ASIGN.CREDITOS.
DEF NEW SHARED VAR VPLANTEL LIKE ALUMNO.PLANTEL.
DEF NEW SHARED VAR VALUMNO LIKE ALUMNO.ALUMNO.
DEF NEW SHARED VAR VINGRESO LIKE ALUMNO.INGRESO.
DEF NEW SHARED VAR VPERIOD AS CHAR.
DEF NEW SHARED VAR BAN2 AS LOG.
DEF NEW SHARED VAR FLAG AS LOG.
DEF NEW SHARED VAR VANIO AS INT.
DEF NEW SHARED VAR SEMESTRE AS INT FORMAT ">9".
DEF NEW SHARED VAR CREDIT LIKE ASIGN.CREDITOS.
DEF NEW SHARED VAR CREDI2 LIKE ASIGN.CREDITOS.
DEF NEW SHARED VAR VALIDA AS LOG.
DEF NEW SHARED VAR PRIMERA AS LOG.  /*para la contabilizacion de creditos*/
```

```
DEF NEW SHARED VAR VSEM1 LIKE ASIGN.SEMESTRE.
DEF NEW SHARED VAR REGULAR AS LOG.  /*indica si el alumno es regular*/
DEF NEW SHARED VAR IMPRESION AS LOG. /*evita impresion si no hay inscripcion*/
DEF VAR EXISGRUP AS LOG.
DEF NEW SHARED VAR BAN AS LOG.

/*Archivo temporal para las asignaturas seleccionadas*/
DEF NEW SHARED WORKFILE VARCH
 FIELD WCARRERA LIKE ALUMNO.CARRERA
 FIELD WASIGN LIKE ASIGN.ASIGN
 FIELD WDESC LIKE ASIGN.DESC-ASIGN
 FIELD WCREDITOS LIKE ASIGN.CREDITOS
 FIELD WSEMESTRE LIKE ASIGN.SEMESTRE
 FIELD WGRUPO LIKE GRUPO.GRUPO.

DEF VAR NUMASIG AS INT.

/*  Definición de formatos de pantalla  */
FORM " " SKIP
    "UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO " AT 20 SKIP
    "CONTROL ESCOLAR FES ZARAGOZA" AT 24 SKIP
     WITH PAGE-TOP WIDTH 80 CENTERED FRAME TITULO.

FORM GRUPO.GRUPO MARCA NO-LABELS GRUPO.ASIGN WITH CENTERED TITLE
"ASIGNATURAS"
 FRAME ASIG-FRAME SCROLL 1 9 DOWN ATTR-SPACE.


/*     <<<<<<<<<<<<<<<     OBTIENE REINSC   >>>>>>>>>>>     */
REG = TRUE.
RUN PERIODO.P.  /* < GENERA PERIODO >  */
REPEAT:
 PRIMERA = TRUE.
 ART19 = FALSE.
 VIEW FRAME TITULO.
 IMPRESION = FALSE.
 CREDI = 0.
 MESSAGE "F4 SALIR".
 UPDATE VCTA HELP "NUMERO DE CUENTA DEL ALUMNO"
 VCARRERA HELP "CLAVE DE LA CARRERA" WITH WIDTH 70 CENTERED TITLE
          "REINSCRIPCION" FRAME ALU.
 FIND ALUMNO WHERE ALUMNO.CUENTA = VCTA AND ALUMNO.CARRERA =
VCARRERA NO-ERROR.
 IF NOT AVAILABLE ALUMNO THEN DO:
   MESSAGE "CUENTA O CARRERA ERRONEA.FAVOR DE RECTIFICAR".
   PAUSE NO-MESSAGE.
   NEXT.
 END. /*fin IF*/
```

```
ELSE DO:
 VCTA = ALUMNO.CUENTA.
 VPLANTEL = ALUMNO.PLANTEL.
 VALUMNO = ALUMNO.ALUMNO.
 VINGRESO = ALUMNO.INGRESO.
 DISPLAY ALUMNO INGRESO WITH FRAME ALU.

 RUN SEMESTRE.P.
 NUMASIG = 0.
 REPEAT:  /* solicita otro grupo */
  EXISGRUP = FALSE.
  UPDATE VGPO HELP "NUMERO DE GRUPO" LABEL "GRUPO" WITH FRAME ALU.
  FOR EACH GRUPO WHERE GRUPO.GRUPO = VGPO AND GRUPO.CARRERA =
VCARRERA:
  EXISGRUP = TRUE.
 END.
 IF NOT EXISGRUP THEN DO:
   MESSAGE "NUMERO DE GRUPO ERRONEO".
   PAUSE NO-MESSAGE.
   NEXT.
 END.
 ELSE DO:
  HIDE MESSAGE.
  SEMGPO = INTEGER(SUBSTRING(VGPO,2,1)).
  IF SEMESTRE > SEMGPO THEN REGULAR = FALSE.
  ELSE REGULAR = TRUE.
  IF REGULAR THEN IRREGU = "R".
  ELSE IRREGU = "I".
  DISP IRREGU NO-LABELS WITH FRAME ALU.
  UPDATE IRREGU VALIDATE (IRREGU = "R" OR IRREGU = "I","DEBE SER R o I")
  WITH FRAME ALU.
  IF IRREGU = "R" THEN REGULAR = TRUE.
  ELSE REGULAR = FALSE.
  FIND CARRERA WHERE ALUMNO.CARRERA = CARRERA.CARRERA.
  VCARRERA = CARRERA.CARRERA.
/*              << FIN OBTIENE REINSC >>        */

/*    <<<<<<<<<<<<<<<<<  PROCESA REINSC  >>>>>>>>>>>>>>  */

/*  ****  valida articulo 19  ****    */
  ANIO = year(today).
  ANIO = ANIO - 1900.
  IF CARRERA.DURACION = 8 THEN TIEMPO = 12.
  IF CARRERA.DURACION = 6 THEN TIEMPO = 9.
  IF CARRERA.DURACION = 9 THEN TIEMPO = 14.
  SEMALU = (ANIO - ALUMNO.INGRESO) * 2.
  IF (SUBSTRING(VPERIOD,4,1)) = "I" THEN SEMALU = SEMALU + 1.
  ELSE SEMALU = SEMALU + 2.
```

```
  IF TIEMPO >= SEMALU THEN ART19 = TRUE.
  ELSE ART19 = FALSE.
  IF ART19 THEN DO:

/*     <<<<<<<<<<<<<<<<<<<< CONSULTA DATOS >>>>>>>>>>>   */

/* localiza las asignaturas de que se compone
   un grupo utilizando el scroll  */
  HIDE FRAME ASIG-FRAME.
  clear frame asig-frame all.
  FOR EACH GRUPO USE-INDEX GR-ASIG WHERE VGPO = GRUPO.GRUPO
                  AND GRUPO.CARRERA = VCARRERA:
    FIND ASIGN WHERE GRUPO.ASIGN = ASIGN.ASIGN NO-ERROR.
    IF AVAILABLE ASIGN THEN DO:
    FIND INSCRIP WHERE INSCRIP.CUENTA = VCTA AND
    INSCRIP.ASIGN = GRUPO.ASIGN AND INSCRIP.CARRERA = VCARRERA
    AND INSCRIP.GRUPO = VGPO NO-ERROR. /*Verifica si ya existe la inscrip.*/
    IF AVAIL INSCRIP THEN MARCA = "*". /*Marca de inscrito a la materia*/
    ELSE MARCA = " ".
    DISPLAY GRUPO.GRUPO MARCA DESC-ASIG GRUPO.ASIGN WITH FRAME
ASIG-FRAME.
    DOWN WITH FRAME ASIG-FRAME.
    END. /*fin de IF*/
  END. /*fin FOR EACH*/

  REPEAT:
    HIDE MESSAGE.
    MESSAGE "|" + CHR(17) + CHR(196) + CHR(217) + "| SELECCIONA OPCION |"
    + CHR(24) + "| ANTERIOR |" + CHR(25) "| SIGUIENTE | F4 SALIR | F5 BORRAR".
    CHOOSE ROW GRUPO.ASIGN NO-ERROR WITH FRAME ASIG-FRAME.
    VASIG = INTEGER(FRAME-VALUE).
    COLOR DISPLAY NORMAL GRUPO.ASIGN WITH FRAME ASIG-FRAME.
    FIND ASIGN WHERE ASIGN.ASIGN = VASIG AND
    ASIGN.CARRERA = VCARRERA NO-ERROR.
    VSEM1 = SEMALU.
    FLAG = TRUE.
    IF LASTKEY = KEYCODE("F5") THEN DO: /*Se eliminara la inscripcion a*/
     MARCA = " ".                    /*la asignatura seleccionada*/
     DISP MARCA WITH FRAME ASIG-FRAME.
     IF CREDI <> 0 THEN
     CREDI = CREDI - ASIGN.CREDITOS. /*Actualizacion de los creditos*/
     FIND GRUPO WHERE GRUPO.GRUPO = VGPO AND GRUPO.ASIGN = VASIG AND
        GRUPO.CARRERA = VCARRERA NO-ERROR.
     IF REGULAR THEN GRUPO.CUPO = GRUPO.CUPO + 1.
     ELSE GRUPO.CUPOI = GRUPO.CUPOI + 1.
     FIND INSCRIP WHERE INSCRIP.CUENTA = VCTA AND INSCRIP.ASIGN = VASIG
        AND INSCRIP.CARRERA = VCARRERA NO-ERROR.
     IF AVAIL INSCRIP THEN DELETE INSCRIP.
```

```
     FOR EACH VARCH WHERE VARCH.WASIGN = VASIG:
      DELETE VARCH.
     END.
     NUMASIG = NUMASIG - 1.
     NEXT.
    END.
    IF KEYFUNCTION(LASTKEY) = "RETURN" THEN DO: /*se selecciono una asig*/

/*                 <<   FIN CONSULTA DATOS   >>                */
    RUN VALIDA.P. /*    < VALIDA REINSC >          */

/*                 <<  FIN PROCESA REINSC  >>                */

/*     <<<<<<<<<<<<<<  EJECUTA  REINSC   >>>>>>>>>>>>                */
     IF VALIDA THEN DO: /*corre si el resultado de <VALIDA.P> es verdadero*/
       MARCA = "*".
       NUMASIG = NUMASIG + 1.
       DISP MARCA WITH FRAME ASIG-FRAME.

       /*    <<<<<<<<<<<<<  ALMACENA INSCRIP  >>>>>>>>>>>   */
       CREATE INSCRIP. /*registra la inscripcion*/
       INSCRIP.CUENTA = VCTA.
       INSCRIP.GRUPO = VGPO.
       INSCRIP.ASIGN = VASIG.
       INSCRIP.TIPO = 1.
       INSCRIP.CARRERA = VCARRERA.
       FIND ASIGN WHERE ASIGN.ASIGN=VASIG AND
       ASIGN.CARRERA=VCARRERA NO-ERROR.
       CREATE VARCH.    /*llena archivo para elaboracion de comprobante*/
       WCARRERA = VCARRERA.
       WASIGN = VASIG.
       WDESC = ASIGN.DESC-ASIGN.
       WCREDITOS = ASIGN.CREDITOS.
       WSEMESTRE = ASIGN.SEMESTRE.
       WGRUPO = VGPO.
       primera = false.
     END. /*fin IF*/

     /*         <<  FIN A  ALMACENA INSCRIP    >>     */
    IF BAN THEN DO: /*Bandera que indica que la inscripcion fue a otro*/
     VGPO = VGPO2.  /*grupo con el programa HELPY.P*/
     HIDE ALL.
     HIDE MESSAGE.
     VIEW FRAME TITULO.
     VIEW FRAME ASIG-FRAME.
     ban = false.
     MESSAGE "ALUMNO INSCRITO.....".
    END.
```

```
    END.
    ELSE NEXT.
     PRIMERA = FALSE.
   END. /*fin REPEAT*/
  END. /*fin IF*/
  ELSE DISPLAY "ALUMNO EN ARTICULO 19" WITH 1 DOWN CENTERED.
END.
for each varch:
  display varch with centered title " COMPROBANTE DE INSCRIPCION ".
end.
END. /*fin REPEAT*/
PRIMERA = TRUE.
IF IMPRESION AND NUMASIG > 0 THEN RUN IMPRIME.P. /* < ELABORA COMPROB > */
END.
END.

/*          <<    FIN EJECUTA INSCRIP    >>          */


/*          <<         FIN REINSC        >>          */

**********************************************************
/*    <<<<<<<<<<<<<<<<    CALCULA SEMESTRE    >>>>>>>>>>>>>>  */

/*SEMESTRE.P*/
/* calcula el semestre en que debe ir un alumno*/

DEF SHARED VAR VPERIOD AS CHAR.
DEF SHARED VAR SEMESTRE AS INT FORMAT ">9".
DEF SHARED VAR VINGRESO LIKE ALUMNO.INGRESO.
DEF VAR NUMANIO AS INT.
DEF VAR ANIO AS CHAR.

ANIO = SUBSTRING(VPERIOD,1,2).
NUMANIO = INTEGER(ANIO).
SEMESTRE = (NUMANIO - VINGRESO) * 2.
IF (SUBSTRING (VPERIOD,4,1)) = "1" THEN SEMESTRE = SEMESTRE + 1.
IF (SUBSTRING (VPERIOD,4,1)) = "2" THEN SEMESTRE = SEMESTRE + 2.

/*              <<  FIN CALCULA SEMESTRE   >>          */


**********************************************************

/*    <<<<<<<<<<<<<<    VALIDA REINSC    >>>>>>>>>>>>  */
```

```
/*VALIDA.P  Valida la asignatura a la que se desea inscribir */

DEF NEW SHARED VAR VSERIACION LIKE ASIGN.SERIACION.
DEF SHARED VAR REG AS LOG FORMAT "S/N".
DEF SHARED VAR VCARRERA LIKE CARRERA.CARRERA.
DEF SHARED VAR VASIG AS INT.
DEF SHARED VAR VGPO AS CHA FORMAT "9999".
DEF SHARED VAR VGPO2 LIKE GRUPO.GRUPO.
DEF SHARED VAR CREDI LIKE ASIGN.CREDITOS.
DEF SHARED VAR CREDI2 LIKE ASIGN.CREDITOS.
DEF SHARED VAR VCTA LIKE ALUMNO.CUENTA.
DEF VAR CREDIT AS INT.
DEF VAR TOTCRED LIKE ASIGN.CREDITOS.
DEF SHARED VAR VALIDA AS LOG.
DEF SHARED VAR PRIMERA AS LOG.
DEF SHARED VAR SEMESTRE AS INT FORMAT ">9".
DEF SHARED VAR VSEM1 LIKE ASIGN.SEMESTRE.
DEF SHARED VAR REGULAR AS LOG.
DEF SHARED VAR IMPRESION AS LOG.
DEF VAR ART27 AS LOG.
DEF NEW SHARED VAR SERIADA AS LOG.
DEF VAR EXISTE AS LOG.
DEF SHARED VAR BAN AS LOG.
DEF SHARED VAR BAN2 AS LOG.
DEF VAR PASE AS CHA FORMAT "AAAA999999-XXX".
DEF VAR RESP AS CHAR FORMAT "X(1)".
DEF SHARED VAR FLAG AS LOG.
DEF VAR CONT AS INT.

DEF NEW SHARED VAR HA LIKE GRUPO.GRUPO.
DEF NEW SHARED VAR HD LIKE GRUPO.CUPO.
DEF NEW SHARED VAR HG LIKE GRUPO.CUPOI.


DEF NEW SHARED WORKFILE HGPO
 FIELD WGRUPO LIKE GRUPO.GRUPO
 FIELD WCUPO LIKE GRUPO.CUPO
 FIELD WCUPOI LIKE GRUPO.CUPOI.

VALIDA = FALSE.
SERIADA = TRUE.
ART27 = FALSE.


EXISTE = FALSE.
  /* CONTABILIZA EL NUMERO DE CREDITOS EN EL PERIODO DE INSCRIPCION
     VIGENTE SOLO EN LA INSCRIPCION A LA PRIMERA ASIGNATURA      */
IF PRIMERA THEN DO:
```

```
 FOR EACH INSCRIP WHERE INSCRIP.CUENTA = VCTA:
  /* FALTA EL PERIODO */
  IF INSCRIP.TIPO = 1 THEN DO:  /* el tipo 1 es para ordinarios*/
  FIND ASIGN WHERE INSCRIP.ASIGN = ASIGN.ASIGN
  AND ASIGN.CARRERA = VCARRERA NO-ERROR.
  IF AVAILABLE ASIGN THEN CREDI = CREDI + ASIGN.CREDITOS.
  END.
 END.
END.
IF VCARRERA = 512 THEN TOTCRED = 135.
ELSE TOTCRED = 60.
FIND ASIGN WHERE ASIGN.ASIGN = VASIG AND ASIGN.CARRERA = VCARRERA
NO-ERROR.
 CREDI = CREDI + ASIGN.CREDITOS.
 IF CREDI <= TOTCRED THEN DO:  /* DEFINIR EL MAXIMO PERMITIDO POR SEM */
  FIND GRUPO WHERE VGPO = GRUPO.GRUPO AND  VASIG = GRUPO.ASIGN
        AND GRUPO.CARRERA = VCARRERA NO-ERROR.
 IF AVAILABLE GRUPO THEN DO:
    FIND HISTORIA WHERE VCTA = HISTORIA.CUENTA
            AND  VASIG = HISTORIA.ASIGN NO-ERROR.
    IF AVAILABLE HISTORIA THEN DO:
      IF NUM-ORD < 2 AND (CALIF = "NA" OR CALIF = "NP") THEN ART27 = TRUE.
      IF CALIF = "MB" OR CALIF = "B" OR CALIF = "S" THEN DO:
       MESSAGE "EL ALUMNO YA APROBO LA MATERIA".
       CREDI = CREDI - ASIGN.CREDI.
       LEAVE.
      END.
    END.
    ELSE ART27 = TRUE.
    IF ART27 THEN DO:
     /* VALIDA LA SERIACION */
     FIND ASIGN WHERE ASIGN.ASIGN=VASIG AND
        ASIGN.CARRERA=VCARRERA NO-ERROR.
      IF ASIGN.SERIACION <> 0 THEN DO:
      VSERIACION = ASIGN.SERIACION.
      RUN SERIADA.P.  /*    < VERIFICA SERIACION >    */
      END.
     IF SERIADA THEN DO:
     FOR EACH INSCRIP WHERE VCTA = INSCRIP.CUENTA
       AND VASIG = INSCRIP.ASIGN:
       EXISTE = TRUE.
     END.
      IF NOT EXISTE  THEN DO:
       IF (GRUPO.CUPO > 0 AND REGULAR) OR (GRUPO.CUPOI > 0 AND NOT REGULAR)
       THEN DO:
         VALIDA = TRUE.
         IMPRESION = TRUE.
         MESSAGE "  ALUMNO INSCRITO.....  ".
         IF REGULAR THEN GRUPO.CUPO = GRUPO.CUPO - 1.
```

```
      ELSE GRUPO.CUPOI = GRUPO.CUPOI - 1.
     END.

     ELSE DO:
      VGPO2 = VGPO.
      MESSAGE " EL GRUPO ESTA SATURADO.  < F2 > AYUDA  < F4 > SALIR".
      CREDI = CREDI - ASIGN.CREDITOS.
      CONT = 0.
      FOR EACH GRUPO WHERE GRUPO.ASIGN = VASIG AND GRUPO.CARRERA =
VCARRERA:
       IF (GRUPO.CUPO > 0 AND REGULAR) OR (GRUPO.CUPOI > 0 AND NOT REGULAR)
       THEN DO:
       CREATE HGPO.
       WGRUPO = GRUPO.GRUPO.
       WCUPO = GRUPO.CUPO.
       WCUPOI = GRUPO.CUPOI.
       CONT = CONT + 1.
       END.
      END.
      UPDATE RESP WITH NO-LABELS NO-BOX NO-UNDERLINE
      EDITING:
       READKEY.
       IF LASTKEY = KEYCODE("F2") THEN DO:
       IF CONT > 0 THEN DO:
       RUN HELPY.P.    /*   <   EJECUTA OPCIONES  >   */
       IF BAN THEN DO:
        CREDI = CREDI + ASIGN.CREDITOS.
        VGPO = HA.
        FIND GRUPO WHERE GRUPO.GRUPO = VGPO AND GRUPO.ASIGN = VASIG AND
        GRUPO.CARRERA = VCARRERA NO-ERROR.
        VALIDA = TRUE.
        IMPRESION = TRUE.
        IF REGULAR THEN GRUPO.CUPO = GRUPO.CUPO - 1.
        ELSE GRUPO.CUPOI = GRUPO.CUPOI - 1.
       END.            /* end - if  ban     */
       LEAVE.
       END.
       ELSE DO:
       MESSAGE "TODOS LOS GRUPOS ESTAN SATURADOS".
       LEAVE.
       END.
       END.            /* end - if last key */
       ELSE  LEAVE.
      END.
      /* end - editing     */
     END.              /* end - else do     */


     END.
```

```
      ELSE DO:
      MESSAGE "EL ALUMNO YA HA SIDO INSCRITO EN ESTA MATERIA".
      CREDI = CREDI - ASIGN.CREDITOS.
      END.
     END.
      ELSE DO:
       MESSAGE COLOR BLINK "NO HA APROBADO LA MATERIA SERIADA" .
       CREDI = CREDI - ASIGN.CREDITOS.
      END.
    END.

    ELSE DO:
    MESSAGE " ALUMNO EN ARTICULO  27 ".
    CREDI = CREDI - ASIGN.CREDITOS.
   END.
  END.
 END.
ELSE DO:
 CREDI = CREDI - ASIGN.CREDITOS.
IF FLAG THEN DO:
 MESSAGE  " REBASA EL LIMITE DE CREDITOS. < F2 > AUTORIZACION".
 FLAG = FALSE.
 CREDI2 = CREDI.
 UPDATE RESP WITH NO-LABELS NO-BOX NO-UNDERLINE
 EDITING:
  READKEY.
  IF LASTKEY = KEYCODE("F2") THEN DO:
   UPDATE PASE BLANK VALIDATE(PASE = "TEAS691003111", "PASE ERRONEO")
   HELP "INTRODUZCA CLAVE DE ACCESO" AUTO-RETURN
   WITH CENTERED SIDE-LABELS TITLE "AUTORIZACION" FRAME A.
   CREDI = CREDI - 10.
   RUN VALIDA.P.
   CREDI = CREDI2.
   IF VALIDA THEN CREDI = CREDI + ASIGN.CREDITOS.
   HIDE FRAME A.
   LEAVE.
  END.
  ELSE LEAVE.
 END.
 END.
 ELSE DO:
 MESSAGE "REBASA EL LIMITE DE CREDITOS. PRESS BARRA ESPACIADORA".
 LEAVE.
 END.
END.

/*        <<     FIN VALIDA REINSC      >>       */

**************************************************************
```

```
/*     <<<<<<<<<<<<    EJECUTA OPCIONES   >>>>>>>>>>>>   */

/* Proporciona opciones de inscripción cuando un grupo está saturado */

DEF SHARED VAR HA LIKE GRUPO.GRUPO.
DEF SHARED VAR HD LIKE GRUPO.CUPO.
DEF SHARED VAR HG LIKE GRUPO.CUPOI.
DEF VAR K      AS INTE NO-UNDO.
DEF VAR I      AS INTE NO-UNDO.
DEF VAR CONTAD  AS INTE NO-UNDO.
DEF VAR RECACT  AS RECID.
DEF SHARED VAR BAN AS LOG.

DEF SHARED WORKFILE HGPO
 FIELD WGRUPO LIKE GRUPO.GRUPO
 FIELD WCUPO LIKE GRUPO.CUPO
 FIELD WCUPOI LIKE GRUPO.CUPOI.

ASSIGN
  CONTAD = 5
  K  = 0.

FORM
  HGPO.WGRUPO      COLUMN-LABEL "GRUPO" AT 2
  HGPO.WCUPO       COLUMN-LABEL "CUPO REG." AT 10
  HGPO.WCUPOI      COLUMN-LABEL "CUPO IRREG." AT 20
  WITH FRAME desfac SCROLL 1 overlay center
  5 DOWN ROW 10 TITLE " G R U P O S " attr-space.

BAN = FALSE.
CLEAR FRAME DESFAC ALL.
FIND FIRST HGPO.
DISP HGPO.WGRUPO HGPO.WCUPO HGPO.WCUPOI
    WITH FRAME DESFAC.
DO K = 1 TO CONTAD - 1:
 FIND NEXT HGPO NO-ERROR.
 IF NOT AVAILABLE HGPO THEN DO:
  FIND LAST HGPO.
  LEAVE.
 END.
 DOWN WITH FRAME DESFAC.
 DISP HGPO.WGRUPO HGPO.WCUPO HGPO.WCUPOI WITH FRAME DESFAC.
 PAUSE 0.
END.

_escoger:
DO WHILE TRUE ON ENDKEY UNDO, RETRY:
  MESSAGE
   "          ||F4||  SALIR          ||" + CHR(24) +
   "||  ANTERIOR          ||" + CHR(25) + "||  SIGUIENTE          ".
```

```
  DISP HGPO.WGRUPO HGPO.WCUPO HGPO.WCUPOI WITH FRAME DESFAC.
  COLOR DISPLAY MESSAGES HGPO.WGRUPO HGPO.WCUPO HGPO.WCUPOI WITH FRAME
desfac.
  readkey.
  COLOR DISPLAY NORMAL HGPO.WGRUPO HGPO.WCUPO HGPO.WCUPOI WITH FRAME
desfac.

 IF CAN-DO("RETURN,PICK",KEYFUNCTION(LASTKEY)) THEN DO:
  HA = HGPO.WGRUPO.
  HD = HGPO.WCUPO.
  HG = HGPO.WCUPOI.
  BAN = TRUE.
  leave _escoger.
 END.
 ELSE
 IF KEYFUNCTION(LASTKEY) = "CURSOR-DOWN" THEN DO:
  find next HGPO no-error.
   if not available HGPO then
     FIND LAST HGPO.
   else do:
    down with frame desfac.
    DISP HGPO.WGRUPO HGPO.WCUPO HGPO.WCUPOI WITH FRAME DESFAC.
    PAUSE 0.
   END.
 END.
 ELSE
 IF KEYFUNCTION(LASTKEY) = "CURSOR-UP" THEN DO:
  FIND PREV HGPO NO-ERROR.
  if not available HGPO then do:
    FIND FIRST HGPO.
  end.
  else do:
   up with frame desfac.
   DISP HGPO.WGRUPO HGPO.WCUPO HGPO.WCUPOI WITH FRAME DESFAC.
   PAUSE 0.
  end.
 END.
 ELSE
 IF KEYFUNCTION(LASTKEY) = "PAGE-DOWN" THEN do:
  DO K = 1 TO CONTAD - 1:
    FIND NEXT HGPO NO-ERROR.
    if not available HGPO then do:
       FIND LAST HGPO.
       leave.
    end.
    down with frame desfac.
    DISP HGPO.WGRUPO HGPO.WCUPO HGPO.WCUPOI WITH FRAME DESFAC.
    pause 0.
  END.
```

```
 END.
 ELSE
 IF KEYFUNCTION(LASTKEY) = "PAGE-UP" THEN do:
  DO K = 1 TO CONTAD - 1:
    FIND PREV HGPO NO-ERROR.
    if not available HGPO then do:
      FIND FIRST HGPO.
      leave.
    end.
    up with frame desfac.
    DISP HGPO.WGRUPO HGPO.WCUPO HGPO.WCUPOI WITH FRAME DESFAC.
    pause 0.
  END.
 end.
 ELSE
 IF KEYFUNCTION(LASTKEY) = "END-ERROR" THEN DO:
  LEAVE _escoger.
 END.
END.

HIDE FRAME desfac NO-PAUSE.
RETURN.

/*            <<  FIN EJECUTA OPCIONES  >>          */


**************************************************************
/*  <<<<<<<<<<<<<  VERIFICA SERIACION  >>>>>>>>>>>>>  */

/*SERIADA.P*/
DEF SHARED VAR VSERIACION LIKE ASIGN.SERIACION.
DEF SHARED VAR VCTA LIKE ALUMNO.CUENTA.
DEF SHARED VAR SERIADA AS LOG.
     FIND HISTORIA WHERE HISTORIA.CUENTA=VCTA
       AND HISTORIA.ASIGN=VSERIACION NO-ERROR.
     IF AVAILABLE HISTORIA THEN DO:
      IF CALIF = "NA" OR CALIF = "NP" THEN SERIADA = FALSE.
     END.
     ELSE SERIADA = FALSE.


/*          <<  FIN VERIFICA SERIACION >>            */


**********************************************************

/*     <<<<<<<<<<<<<<<<<  INSCRIBE EXTRAORD  >>>>>>>>>>>  */
```

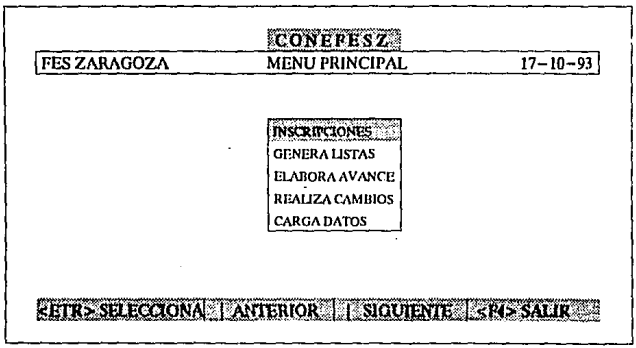
```
/*<EXTRA.P> Programa que realiza inscripciones a examen extraordinario*/

/* Variables compartidas con el programa Extra1.p */
DEF NEW SHARED VAR VGPO AS CHA FORMAT 9999.
DEF NEW SHARED VAR P AS CHA.
DEF NEW SHARED VAR IG LIKE INSCRIP.GRUPO.
/* Variable compartida con el programa Periodo.p */
DEF NEW SHARED VAR VPERIOD AS CHA.
/* Variables compartidas con el programa Imprime.p */
DEF NEW SHARED VAR CREDI LIKE ASIGN.CREDITOS.
DEF NEW SHARED VAR VPLANTEL LIKE ALUMNO.PLANTEL.
DEF NEW SHARED VAR VALUMNO LIKE ALUMNO.ALUMNO.
DEF NEW SHARED VAR VINGRESO LIKE ALUMNO.INGRESO.
DEF NEW SHARED VAR VCTA LIKE ALUMNO.CUENTA.
DEF NEW SHARED VAR VCARRERA LIKE ALUMNO.CARRERA.

/* Variables locales */
DEF VAR PASE AS CHA FORMAT "X(7)".
DEF VAR VASIGN AS INT.
DEF VAR VPERIODO AS INT.
DEF VAR VPERIODO1 AS INT.
DEF VAR VPRIMER AS LOG.
DEF VAR RESP AS CHA FORMAT "X".
DEF VAR IMPRESION AS LOG.
DEF VAR VGPO1 LIKE INSCRIP.GRUPO.

/* Definición de archivo de trabajo */
DEF NEW SHARED WORKFILE VARCH
 FIELD WCARRERA LIKE ALUMNO.CARRERA
 FIELD WASIGN LIKE ASIGN.ASIGN
 FIELD WDESC LIKE ASIGN.DESC-ASIGN
 FIELD WCREDITOS LIKE ASIGN.CREDITOS
 FIELD WSEMESTRE LIKE ASIGN.SEMESTRE
 FIELD WGRUPO LIKE GRUPO.GRUPO.

/* Definición de formatos de pantalla */
 FORM " " SKIP
     "UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO " AT 20 SKIP
     "CONTROL ESCOLAR FESZ ZARAGOZA" AT 24 SKIP
      WITH WIDTH 80 CENTERED FRAME TITULO.

/*    <<<<<<<<<<<<<<  OBTIENE EXTRAORD  >>>>>>>>>>>>>  */
RESP = "S".
PASE = "********".
VPRIMER = FALSE.
RUN PERIODO.P.

REPEAT:
 HIDE ALL.
```

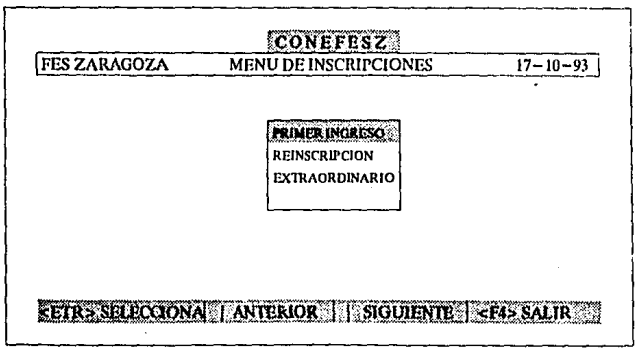
```
IMPRESION = FALSE.
IF RESP = "S" OR RESP = "s" THEN DO:
 VPERIODO = 0.
 VPERIODO1 = 0.
 CREDI = 0.
 VIEW FRAME TITULO.
 PROMPT-FOR ALUMNO.CARRERA WITH WIDTH 70 CENTERED ROW 6 TITLE
       "INSCRIPCION A EXAMEN EXTRAORDINARIO" FRAME ALU.
 VCARRERA = INPUT ALUMNO.CARRERA.
 PROMPT-FOR ALUMNO.CUENTA WITH FRAME ALU.
 VCTA = INPUT ALUMNO.CUENTA.
 FIND ALUMNO WHERE ALUMNO.CARRERA = VCARRERA AND ALUMNO.CUENTA =
VCTA
      NO-ERROR.
 IF NOT AVAILABLE ALUMNO THEN DO:
 MESSAGE "NUMERO DE CUENTA O CARRERA ERRONEOS. FAVOR DE RECTIFICAR".
  PAUSE NO-MESSAGE.
  NEXT.
 END.  /* End if */
 DISPLAY ALUMNO.ALUMNO COLUMN-LABEL "NOMBRE DEL ALUMNO" WITH FRAME
ALU.

 /* inician las validaciones para poder inscribir a extraordinario */

 /*            << FIN OBTIENE EXTRAORD >>     */

 /*  <<<<<<<<<<<<<<<    PROCESA EXTRAORD    >>>>>>>>>>>>>>>>
*/
  REPEAT:

    /*  <<<<<<<<<   CONSULTA DATOS    >>>>>>>>>>>>>>>   */
  PROMPT-FOR ASIGN.ASIGN WITH WIDTH 70 ROW 11 CENTERED
       TITLE "CLAVE DE LA ASIGNATURA" FRAME ALA OVERLAY.
  VASIGN = INPUT ASIGN.ASIGN.
  FIND ASIGN USING ASIGN NO-ERROR.
  IF NOT AVAILABLE ASIGN THEN DO:
   MESSAGE "ASIGNATURA INEXISTENTE FAVOR DE RECTIFICAR".
   PAUSE NO-MESSAGE.
   NEXT.
  END.  /* End if */
  DISPLAY ASIGN.DESC-ASIGN COLUMN-LABEL "NOMBRE DE ASIGNATURA"
       WITH FRAME ALA.
  FIND HISTORIA WHERE HISTORIA.CUENTA = VCTA AND
       HISTORIA.ASIGN = VASIGN NO-ERROR.
  IF NOT AVAILABLE HISTORIA THEN DO:
   MESSAGE "CUENTA O ASIGNATURA ERRONEAS. FAVOR DE RECTIFICAR".
   PAUSE NO-MESSAGE.
   NEXT.
    /*        << FIN CONSULTA DATOS >>      */
```

```
/*      <<<<<<<<<<<<<    VALIDA EXTRAORD    >>>>>>>>>>>>>      */

  END. /* End if */
  FIND INSCRIP WHERE INSCRIP.CUENTA = VCTA AND INSCRIP.CARRERA = VCARRERA
     NO-ERROR.
  IF AVAIL INSCRIP THEN DO:
   VGPO1 = INSCRIP.GRUPO.
   IF SUBSTRING(VGPO1,1,1) = "E" THEN DO:
    MESSAGE COLOR BLINK "ALUMNO YA INSCRITO A EXTRAORDINARIO".
    PAUSE NO-MESSAGE.
    NEXT.
   END. /* End if */
  END. /* End if */

  /* Verifica si la calificacion del alumno es reprobatoria */

  IF CALIF = "NA" OR CALIF = "NP" THEN DO:
   IF VPRIMER = FALSE THEN DO:

    /* Rutina de asignacion de grupo */

    RUN EXTRA1.P. /*  < CALCULA EXTRAORD >  */

    /* Verificacion de numero de examenes extraordinarios del alumno */

    FOR EACH INSCRIP WHERE VCTA = INSCRIP.CUENTA:
     IF VGPO EQ INSCRIP.GRUPO THEN DO:
      IF VGPO BEGINS "EA" OR VGPO BEGINS "EB" THEN VPERIODO = VPERIODO + 1.
      IF VGPO BEGINS "EC" OR VGPO BEGINS "ED" THEN VPERIODO1 = VPERIODO1 + 1.
     END.  /*End if*/
    END.  /*End for each */

   /* Verifica si en el periodo actual se pueden otorgar mas extraordianrios*/

   IF VGPO BEGINS "EB" AND VPERIODO > 2 OR VGPO BEGINS "ED" AND VPERIODO1 >
2
     THEN DO:
    MESSAGE " EL ALUMNO TIENE DOS EXTRAORDINARIOS EN EL PERIODO ACTUAL".
    UPDATE PASE BLANK WITH FRAME PSSWRD TITLE "CLAVE DE AUTORIZACION"
    CENTERED SIDE-LABELS ROW 18.

    /* Validacion de la clave de autorizacion */

    IF PASE EQ "LUISMI" THEN DO:
     VPERIODO = 0.
     VPERIODO1 = 0.
    END. /* End if */
    ELSE DO:
     MESSAGE "CLAVE ERRONEA, INSCRIPCION DENEGADA. FAVOR DE RECTIFICAR".
```

```
    MESSAGE "<ESC> PARA SALIR".
    PAUSE NO-MESSAGE.
    NEXT.
   END. /* End else */
  END. /* End if */
/*          <<   FIN VALIDA EXTRAORD    >>        */

/*          <<   FIN PROCESA EXTRAORD   >>        */

/*  <<<<<<<<<<<<<  EJECUTA  EXTRAORD  >>>>>>>>>>>  */
  IF VPERIODO < 2 AND VPERIODO1 < 2 THEN DO:

  /* Realizacion de la inscripcion a extraordinario */

/*  <<<<<<<<<<<<<  ALMACENA INSCRIP  >>>>>>>>>>>>>>>  */
   CREATE INSCRIP.
   INSCRIP.CUENTA = VCTA.
   INSCRIP.ASIGN = VASIGN.
   INSCRIP.GRUPO = VGPO.
   DISPLAY "                ALUMNO INSCRITO A EXTRAORDINARIO"
        WITH WIDTH 70 ROW 17 FRAME AL CENTERED OVERLAY.
   MESSAGE "<ESC> PARA SALIR".
   CREDI = CREDI + ASIGN.CREDITOS.
   /* Asignacion de datos para la rutina de impresion */
   CREATE VARCH.
   WCARRERA = VCARRERA.
   WASIGN = VASIGN.
   WDESC = ASIGN.DESC-ASIGN.
   WCREDITOS = ASIGN.CREDITOS.
   WSEMESTRE = ASIGN.SEMESTRE.
   WGRUPO = INSCRIP.GRUPO.
   IMPRESION = TRUE.

   END.  /* End if */
   /*          <<    FIN ALMACENA INSCRIP  >>       */

   ELSE
    MESSAGE "EL ALUMNO TIENE DOS EXTRAORDINARIOS EN EL PERIODO ACTUAL
     <ESC> PARA SALIR".
    END. /* End else */
   END. /* End if  */
  END. /* End if  */
 UPDATE  RESP  LABEL  "DESEA  REGISTRAR  OTRO  ALUMNO  A  EXAMEN
EXTRAORDINARIO"
              WITH CENTERED OVERLAY SIDE-LABELS.
  VPLANTEL = ALUMNO.PLANTEL.
  VALUMNO = ALUMNO.ALUMNO.
  VINGRESO = ALUMNO.INGRESO.
```

```
  /* Impresion del comprobante de inscripcion a examen extraordinario */

  IF IMPRESION THEN RUN IMPRIME.P.
 END.  /* End if */
 ELSE LEAVE.
END.  /* End repeat */

/*            <<      FIN EJECUTA EXTRAORD    >>        */


**********************************************************

/*   <<<<<<<<<<<<<<<<   CALCULA EXTRAORD   >>>>>>>>>>>>>>>  */

/*<EXTRA1.P> Asignacion de grupos para examen extraordinario */

DEF SHARED VAR VGPO AS CHA FORMAT 9999.
DEF SHARED VAR P AS CHA.
DEF SHARED VAR IG LIKE INSCRIP.GRUPO.

P = STRING(YEAR(TODAY) - 1900).
IF (MONTH(TODAY)>=10) AND (MONTH(TODAY)<=12) THEN DO:
 IG = "EA" + P.
 P = P + "-1".
END.
IF (MONTH(TODAY)>=1) AND (MONTH(TODAY)<=3) THEN DO:
 IG = "EB" + P.
 P = P + "-1".
END.
IF (MONTH(TODAY)>=4) AND (MONTH(TODAY)<=6) THEN DO:
 IG = "EC" + P.
 P = P + "-2".
END.
IF (MONTH(TODAY)>=7) AND (MONTH(TODAY)<=9) THEN DO:
 IG = "ED" + P.
 P = P + "-2".
END.
VGPO = IG.


/*            <<   FIN CALCULA EXTRAORD   >>        */

*****************************************
```


CONEFESZ
FES ZARAGOZA
MENU PRINCIPAL
17-10-93
INSCRIPCIONES
GENERA LISTAS
ELABORA AVANCE
REALIZA CAMBIOS
CARGA DATOS
<ETR> SELECCIONA
ANTERIOR
SIGUIENTE
<F4> SALIR


CONEFESZ
FES ZARAGOZA
MENU DE INSCRIPCIONES
17-10-93
PRIMER INGRESO
REINSCRIPCION
EXTRAORDINARIO
<ETR> SELECCIONA
ANTERIOR
SIGUIENTE
<F4> SALIR

CONEFESZ

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
CONTROL ESCOLAR FES ZARAGOZA

INSCRIPCION DE ALUMNOS DE PRIMER INGRESO

CUENTA: - CARRERA:
ALUMNO:
DIRECCION:
CP: TEL: GRUPO:

<F4> SALIR
NUMERO DE CUENTA DEL ALUMNO

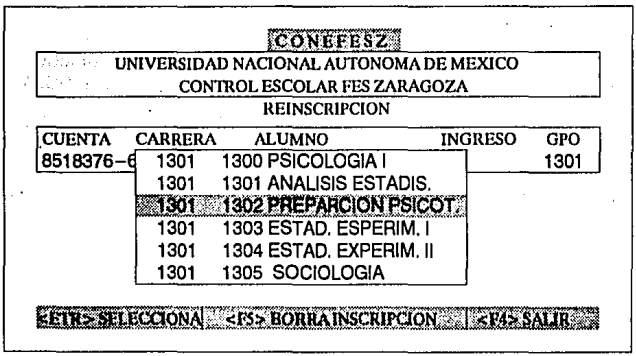
CONEFESZ

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
CONTROL ESCOLAR FES ZARAGOZA

REINSCRIPCION

CUENTA: - CARRERA: ALUMNO: INGRESO GPO:

<F4> SALIR
NUMERO DE CUENTA DEL ALUMNO


CONEFESZ
UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
CONTROL ESCOLAR FES ZARAGOZA
REINSCRIPCION
CUENTA CARRERA ALUMNO INGRESO GPO
8518376-6
1301 1300 PSICOLOGIA I 1301
1301 1301 ANALISIS ESTADIS.
1301 1302 PREPARCION PSICOT.
1301 1303 ESTAD. ESPERIM. I
1301 1304 ESTAD. EXPERIM. II
1301 1305 SOCIOLOGIA
<ETR> SELECCIONA <F5> BORRA INSCRIPCION <F4> SALIR

COMPROBANTE DE INSCRIPCION

17–10–93		UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO FACULTAD DE ESTUDIOS SUPERIORES ZARAGOZA COMPROBANTE DE INSCRIPCION			
Plan.	Carr	Cuenta	Nombre del alumno	Ingreso	Periodo

Carrera	Asing	Nombre de la asignatura	Creditos	Semestre	Grupo
		TOTAL DE CREDITOS			

COMPROBANTE PROVISIONAL

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
FESZ ZARAGOZA
LISTA DE ALUMNOS
(519) LIC. EN PSICOLOGIA

GRUPO: 1301 ALUMNOS REGULARES

No	CAR	CLV	CUENTA	NOMBRE DEL ALUMNO
01	21	519	8828668-4	ABASCAL CA\AS MARTHA LAURA
02	21	519	9037236-0	ALAVEZ MEJIA ISABEL ESTHER
03	21	519	9017685-2	ALCANTARA MORENO HUGO
04	21	519	8918611-6	ALMANZA TREJO SANDRA AIDA
05	21	519	9017707-9	APARICIO CRISOSTOMO GEORGINA
06	21	519	9023793-3	CALVO CARREON ROSARIO ANGELICA
07	21	519	9016703-8	CASTILLO CAMPOS SOCORRO ELENA
08	21	519	8639130-8	CASTRO PIMENTEL ANTONIO IVAN
09	21	519	8916573-5	CELIS ESPINOSA ROBERTO
10	21	519	8517639-3	COSS DIAZ ANA BERTHA
11	21	519	9019026-5	CRUZ RAMIREZ PATRICIA
12	21	519	8923674-7	DIAZ GONZALEZ ROCIO
13	21	519	8917772-5	FERNANDEZ ORTIZ CLAUDIA IVETT
14	21	519	9022844-9	FIGUEROA PASTRANA BLANCA ESTELA
15	21	519	8819457-6	FLORES CRUZ ALFREDO
16	21	519	9019185-3	GARCIA HERNANDEZ ADRIANA ISABEL
17	21	519	8937913-2	GONZALEZ ARREGUIN KARLA LIZBETH
18	21	519	8818294-0	GONZALEZ MATA CAYETANO
19	21	519	9018100-1	HERNANDEZ VELAZQUEZ ANGELICA
20	21	519	8918020-0	LOPEZ GARCIA RAUL
21	21	519	8710391-9	LOPEZ PALACIOS RODRIGO MACARIO
22	21	519	9354060-7	LOPEZ SALAS JUAN LUIS
23	21	519	9019471-3	MANJARREZ TREJO GISELA
24	21	519	9018209-1	MARIN VARGAS ELVIA
25	21	519	9354031-5	MARTINEZ ALVAREZ SILVIA IVETH
26	21	519	9259705-5	MELO LOPEZ WOOLWORTH FILADELPHIA
27	21	519	9017152-7	MENDOZA AGUILAR MA DEL CONSUELO
28	21	519	9020964-6	NAVARRO MONCADA GERARDO
29	21	519	8829199-8	NOGUEDA VALDES LILIANA