UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERIA

SISTEMA DE INFORMACIÓN PARA PROGRAMAS DE POSGRADO DE LA FACULTAD DE FILOSOFÍA Y LETRAS

T E S I S

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A N

DORA NOHEMI LEDEZMA LÓPEZ

LUIS JESÚS GÓMEZ RIVERA

DIRECTOR DE TESIS: ING. JOSÉ ARTURO ORIGEL COUTIÑO

MÉXICO D.F. 2000

A mis Padres:

Gracias por la ayuda moral y económica durante toda mi carrera y por haberme apoyado cada día en mis decisiones.

A la Universidad Nacional Autónoma de México

Por abrirme sus puertas y prestar sus instalaciones durante este tiempo de estudio.

Dora Ledezma

A la Universidad Nacional Autónoma de México por todo lo que me enseñó y significa para mí.

Por ser un ejemplo de tenacidad, este trabajo se debe en gran parte a ustedes. Gracias Papá, Mamá y Ale por todo su apoyo que nunca olvidaré.

Luis Gómez

Índice

Introducción

En México, el Consejo Nacional de Ciencia y Tecnología (CONACYT) es el organismo gubernarnental encargado de coordinar y promover el desarrollo científico y tecnológico. Sus objetivos, de acuerdo al Programa de Ciencia y Tecnología 1995-2000 son:

1. Federalizar la actividad científica y tecnológica. (Descentralización).
2. Mejorar y ampliar la formación de recursos humanos altamente calificados.
3. Articular la actividad científica del país con las corrientes mundiales del conocimiento.
4. Contribuir al entendimiento de la realidad de los problemas nacionales en las diversas áreas de la investigación.
5. Elevar y promover la capacidad técnica de los productores del país, para atender las demandas de bienestar de la población.

Por estas razones, el CONACYT es el encargado de proporcionar becas y dar seguimiento a los planes de Posgrado en la Universidad Nacional Autónoma de México (UNAM). Debido a esto, año con año realiza una serie de evaluaciones y seguimientos a cada Facultad e Instituto de la Universidad.

Con motivo de estas evaluaciones y seguimientos, CONACYT hizo entrega de un sistema de captura de toda la información que requiere, que se llama "Padrón de Programas de Posgrado de Excelencia para Ciencia y Tecnología".

En este Sistema se capturan los datos de los becarios, profesores, egresados, universidades, programas de intercambio y otros. Con base en estos datos,

CONACYT, de acuerdo a sus facultades, decide si se sigue proporcionando apoyo a los programas de posgrado de la Facultad de Filosofía y Letras.

La Facultad de Filosofía y Letras cuenta con toda la información que pide CONACYT. El problema de la Facultad de Filosofía y Letras radica esencialmente en dos puntos.

1. La información se encuentra dispersa en diversas áreas.
2. Es capturada en diversos formatos, sin que exista un estándar.

Es por estos motivos, que en la División de Estudios de Posgrado de la Facultad de Filosofía y Letras se debe capturar dos veces la información; la primera es en sus propios formatos y la segunda en el sistema de captura que proporciona el CONACYT.

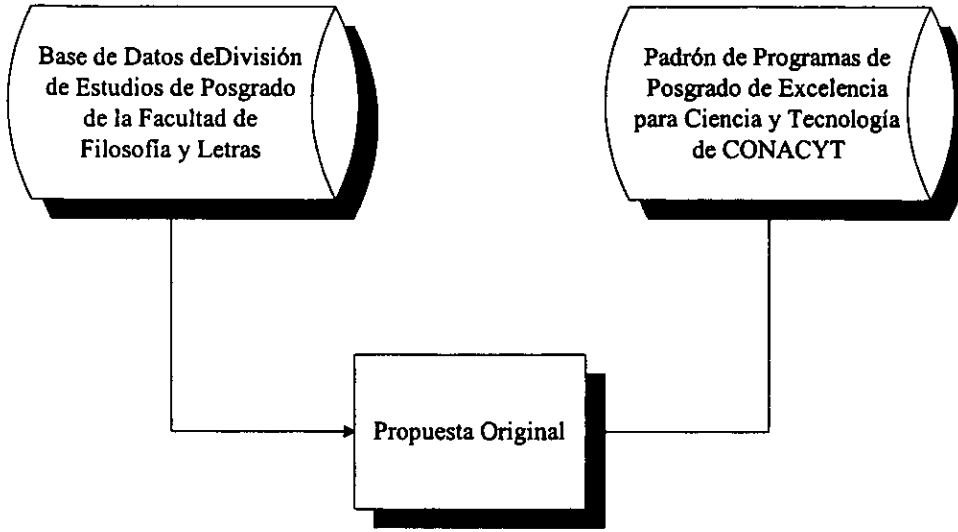
Por estas razones es que pretendemos facilitar su tarea y con la propuesta, resolver la mayoría de sus problemas.

Antecedentes y Problemática

La propuesta que nosotros tenemos para la División de Estudios de Posgrado de la Facultad de Filosofía y Letras, radica en diseñar un Sistema, donde con los datos dispersos que tienen (estos datos se encuentran en cada una de las coordinaciones de esa Facultad), por medio de un "puente" de programación, evitarles el que ellos capturen en el Sistema "Padrón de Programas de Posgrado de Excelencia para Ciencia y Tecnología" que les proporciona CONACYT.

El diseño que nosotros proponemos se basa en generar, a partir de la información de la División de Estudios de Posgrado de la Facultad de Filosofía y

Letras, un esquema similar a la base de datos de CONACYT, tal y como se muestra en la siguiente figura:



A continuación describimos la manera en que realizamos el análisis al Padrón de Programas de Posgrado de Excelencia para Ciencia y Tecnología de CONACYT.

Este análisis lo hicimos utilizando Ingeniería Inversa, la cual consiste en que a partir de un producto ya finalizado, se va desmenuzando lo más posible hasta llegar a la primera etapa del producto.

Lo primero que hicimos, fue conocer el Padrón de Programas de Posgrado de Excelencia para Ciencia y Tecnología, para esto introdujimos datos reales y posteriormente por cada campo a capturar le fuimos ingresando un número consecutivo, para identificar al momento de desglosar la base de datos, a qué tabla y campo corresponden cada uno de estos datos.

El siguiente paso consistió en abrir la base de datos y generar un diccionario de datos considerando cada tabla y a su vez obtener una descripción de cada campo.

Después se identificaron las llaves foráneas y primarias de cada tabla, para que a partir de esto, se genere un esquema de entidad-relación de toda la base de datos, cuya finalidad es obtener la relación que existe en cada tabla y la manera en que se lleva a cabo el descargo de la información a la base de datos.

Precisamente en este punto fue donde nos encontramos con problemas realmente serios. Notamos que hay diversos errores en esta Base de Datos:

1. En cada tabla existe un campo llamado Consec, el cual genera un número consecutivo que hace único a cada registro. El primer problema radica en que este número no está controlado adecuadamente, ¿qué significa esto?. Significa que realmente este consecutivo no existe como tal. Nos sucedió que al analizar los datos introducidos, en las tablas no se halla una continuidad en los consecutivos, esto es que se están realizando una especie de saltos entre estos, p.e. se tienen los consecutivos desde el 1 hasta 22, y continua el 24, 27 y 29.
2. No aparece un identificador, fuera de este consecutivo en cada tabla, que nos permita tener un registro único en toda la Base de Datos. Si este consecutivo no se genera adecuadamente y a esto le añadimos que en un momento determinado puede cambiarse o sustituir fácilmente, no nos da en lo más mínimo seguridad ni integridad en la información.
3. La información que se introduce, no es grabada en su totalidad, lo cual presenta un grave problema, ya que pone en riesgo la información.

Debido a estos errores y limitantes que tuvimos con el sistema, y una vez planteada la problemática a la gente de la División de Estudios de Posgrado de la

Facultad de Filosofía y Letras, buscamos diversas soluciones. Las soluciones que proponemos son las siguientes:

1. Diseñar un nuevo sistema. La idea de realizar esto consiste en que debido a la problemática existente (explicada anteriormente), lo mejor es realizar un nuevo sistema, partiendo desde cero e iniciando con un nuevo análisis de la problemática existente.
2. Diseño parcial a un nuevo sistema. Esta idea se nos antoja viable debido a que en ocasiones anteriores se nos había planteado la importancia que para la División de Estudios de Posgrado de la Facultad de Filosofía y Letras tiene el mantener al corriente toda la información de Profesores, Alumnos, Planes de estudio y Programas de la propia División.
3. Diseño de un nuevo sistema en colaboración con CONACYT. Sin lugar a dudas, es la mejor opción de las que se han planteado. Sin embargo fue una tarea realmente difícil, por no decir imposible, que implicaba el tener contacto continuo y directo con el personal de Informática de CONACYT. Se intentó un primer acercamiento, por medio de la gente de la División de Estudios de Posgrado de la Facultad de Filosofía y Letras, el cual no prosperó, argumentando la gente de CONACYT que no disponían del tiempo suficiente para realizar alguna reunión. El segundo intento fue vía Correo electrónico, que de igual manera, fue inútil. Finalmente el tercer y último intento de contacto lo hicimos vía telefónica, teniendo como respuesta una negativa rotunda a la idea presentada.

Con estos resultados y con la presión del tiempo, llegamos al acuerdo con la gente de la División de Estudios de Posgrado de la Facultad de Filosofía y Letras, de ejecutar la tarea de la primera propuesta sugerida.

Metodología de Diseño

En un principio pensamos en utilizar la Ingeniería de Software Asistida por Computadora, mejor conocida como CASE (Computer Aided Software Engineering), la cual es una serie de herramientas y técnicas que se utiliza para el desarrollo de los sistemas. No pudimos hacer uso de esta metodología, debido a que las herramientas CASE que nosotros tenemos (EasyCASE y EasyER) no se encuentran completas, ya que son versiones Demo y nunca conseguimos herramienta alguna completa, como era el caso de Power Developer, una de las herramientas líderes en el desarrollo CASE.

Otra de la Metodologías más utilizadas es el Análisis Estructurado, que es precisamente en el que trabajaremos para el desarrollo de nuestra tesis. El Análisis Estructurado, a su vez tiene diversas corrientes como son las de Yurdon, De Marco, Sarson, etc. Nosotros nos basaremos en el análisis estructurado propuesto por Edward Yourdon[1]. Esto no significa que seguiremos al pie de la letra todos los pasos, ya que también incluiremos lo que la experiencia laboral nos ha ido enseñando.

Queremos dejar en claro, que las Metodologías de Diseño, no son las mismas que el Diseño de Bases de Datos.

[1] "Análisis Moderno Estructurado", Edward Yourdon., Prentice Hall

Objetivo

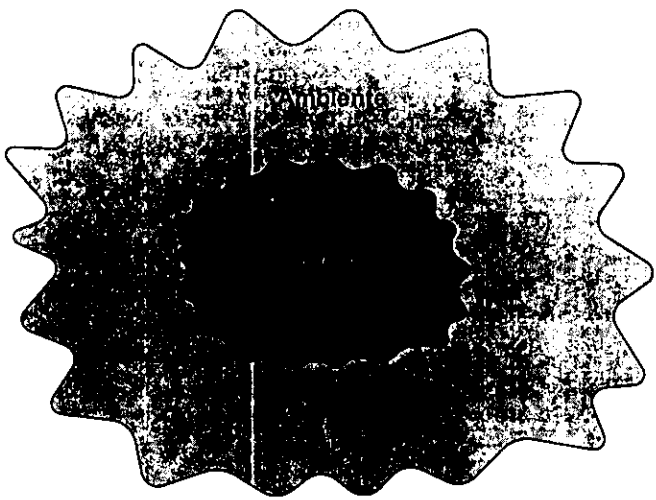
La presente tesis tiene como objetivo diseñar y desarrollar un sistema que permita a la División de Estudios de Posgrado de la Facultad de Filosofía y Letras, llevar de manera correcta el Padrón de Programas de Posgrado que pide CONACYT cada año, evitando realizar capturas innecesarias y redundantes. Aplicando los conocimientos adquiridos durante la estancia en esta facultad. Con la finalidad de brindar una herramienta que sea útil, oportuna y práctica en su uso.

Análisis

1.1 Modelado

Una de las labores más difíciles de llevar a cabo en el análisis de un sistema, es el saber identificar qué "cosas" son las que van a formar parte del sistema y que "cosas" no.

Se recomienda que lo primero que se debe hacer, es definir un *Modelo Ambiental*, en el cual se determina qué es lo que forma parte del *sistema* y qué forma al *ambiente*, o sea, la parte que no tiene algo que ver con el sistema. Es muy importante conocer también cual es la información que entra al sistema desde el exterior y que sale del sistema al exterior.



El sistema que construiremos es racional y tiene un objetivo; este producirá salidas como *respuesta* a algún *estímulo* en el "ambiente".

Para poder definir el modelo de ambiente es necesario declarar tres componentes:

1. Declaración de propósitos.
2. Diagrama de contexto.
3. Lista de acontecimientos.

1. Declaración de propósitos

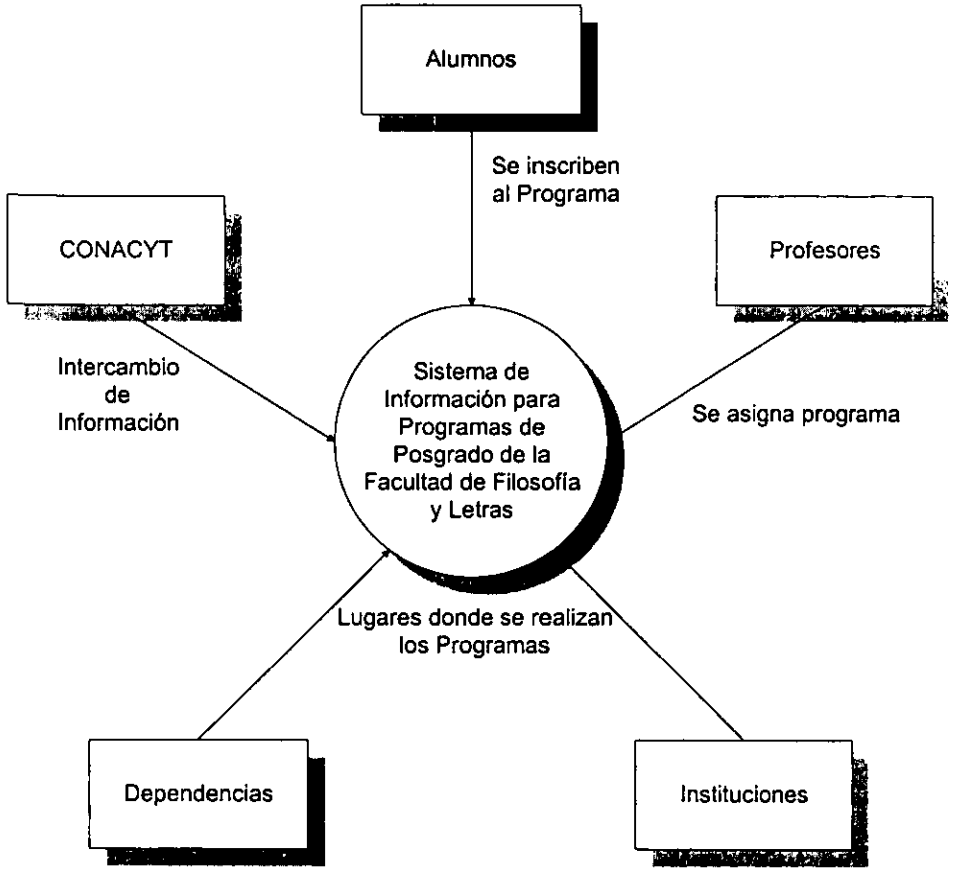
Es aquí donde debemos declarar de manera breve y concisa el propósito del sistema, tal y como lo hacemos a continuación:

> Reunir toda la información que se tiene en la División de Estudios de Posgrado de la Facultad de Filosofía y Letras en una sola base de datos, la cual debe ser amigable para los usuarios y pueda proporcionar el reporte anual que requiere CONACYT, así como generar un histórico.

2. Diagrama de Contexto

Este diagrama nos representa al sistema por medio de una sola burbuja. El diagrama de contexto enfatiza varias características importantes del sistema.

Nuestro diagrama de contexto es el mostrado a continuación:



3. Lista de Acontecimientos

La lista de acontecimientos es una narrativa de estímulos que ocurren en el mundo exterior a los cuales el sistema debe responder.

Los acontecimientos del sistema son listados a continuación y en el mismo orden en que se van presentando.

I. Introducción de los Datos Generales de la Institución y Dependencia.

II. Introducción de los Datos del Programa.

III. Introducción del Plan de Estudios.

IV. Introducción de los Datos de la Investigación.

V. Introducción del Presupuesto.

VI. Cédula informativa de Alumnos.

VII. Cédula Informativa de Profesores.

VIII. Introducción del Acervo Bibliográfico.

IX. Evaluación al Programa.

Enseguida describimos estos acontecimientos de una manera un poco más amplia, sin llegar a ser totalmente detallada:

I. Introducción de los Datos Generales de la Institución y Dependencia

En este acontecimiento es requerida la dirección completa de la Institución y Dependencia en donde se llevará a cabo el Programa, así como número telefónico y número de fax.

II. Introducción de los Datos del Programa.

Aquí se solicitan los Datos particulares del Programa, algunos de ellos son el Nombre del Programa, Fechas de aprobación e inicio, Metas, Objetivos, Antecedentes, Apoyo, Importancia, etc.

III. Introducción del Plan de Estudios.

Este acontecimiento nos muestra un perfil del Plan de estudios, ya que se requiere información como los Requisitos de Ingreso, de Permanencia, de Egreso y de Cambio, los Fundamentos académicos al Programa, las Actividades Académicas, Tipo de Inscripción, Número de estudiantes, Duración del programa, Perfiles del egresado, Calendario académico y otros.

IV. Introducción de los Datos de la Investigación.

Este acontecimiento se divide en Áreas y Lineas de Investigación, En el primero se asignan las áreas principales de investigación. Y en el segundo, las líneas y disciplinas secundarias, indicando la especialidad de cada una.

V. Introducción del Presupuesto.

El Presupuesto se requiere para indicar los tipos de recurso que se van a asignar al programa. Estos recursos pueden ser Institucionales, donde se indica la Institución o Dependencia que los asigna, tipo de recurso y las medidas que se especifican para el apoyo. También pueden ser Recursos específicos, donde se solicita el monto y el objetivo para dicho programa.

VI. Cédula informativa de Alumnos.

Junto con la Cédula Informativa de los Profesores, este acontecimiento es uno de los más extensos, ya que son requeridos los datos del Alumno, estos son Nombre y datos personales, el tiempo de dedicación, su procedencia, instituciones de vinculación, tipo y tiempo de beca, sus datos de tesis, etc.

VII. Cédula Informativa de Profesores.

De los Profesores se requieren también algunos datos personales, así como las participaciones que se hayan tenido en publicaciones, los proyectos de investigación con los que se encuentran vinculados, los estudios realizados y demás participaciones académicas.

VIII. Introducción del Acervo Bibliográfico.

Aquí se registran las publicaciones que se tienen disponibles para ciertas áreas de investigación, las suscripciones vigentes a publicaciones nacionales y extranjeras e información sobre estas.

IX. Evaluación al Programa.

La evaluación es un suceso realmente pequeño, en donde sólo se indica si la evaluación es externa o interna, el periodo de fechas en que se realiza, los resultados de la evaluación y las acciones que son realizadas.

1.2 Diagrama de Flujo de Datos (DFD)

El Diagrama de Flujo de Datos (DFD) es la forma gráfica en que podemos modelar y analizar nuestro sistema. Nos permite visualizarlo como una red interconectada entre sí. El DFD permite plasmar y analizar de manera sencilla las funciones del sistema.

Al Diagrama de Flujo de Datos también se le conoce con otros nombres, los cuales mencionaremos para tenerlos presente por si llegase el caso de que existiera confusión con el mismo DFD.

- Carta de burbujas
- Diagrama de burbujas
- Diagrama de flujo de trabajo
- Modelo de proceso
- Modelo de función

Los DFD son herramientas ampliamente utilizadas en el desarrollo de sistemas.

Existen diferentes notaciones para representar los DFD. Nosotros utilizaremos la recomendada por Yourdon, la cual se compone de tres elementos:

1. El Proceso. También es conocido como *Función* o *Transformación*. El Proceso se define gráficamente como un círculo y nos dice como una o más entradas se transforman en salidas. Dentro del Proceso se debe indicar una descripción breve de lo que este hace.

2. El Flujo. Este se representa por medio de una flecha. El Flujo nos indica la forma y dirección en que se mueve el Proceso. Realmente El Flujo es un dato en proceso.

3. El Almacén. Este es representado por un rectángulo, ya sea redondeado en las esquinas o normal. El Almacén modela los datos en reposo, como pueden ser *archivos* o *tablas* de bases de datos.

A partir de la información descrita anteriormente podemos modelar nuestro sistema por medio de un Diagrama de Flujo de Datos.

Diagrama de Flujo de Datos – Diagrama General

El diagrama que se muestra a continuación, es el Diagrama de Flujo de Datos General, en el cual hacemos uso de los elementos antes mencionados para describir su comportamiento y mostrar desde un concepto (en el siguiente nivel de particular) más amplio nuestro sistema.

Diagrama de Flujo de Datos

Diagrama General

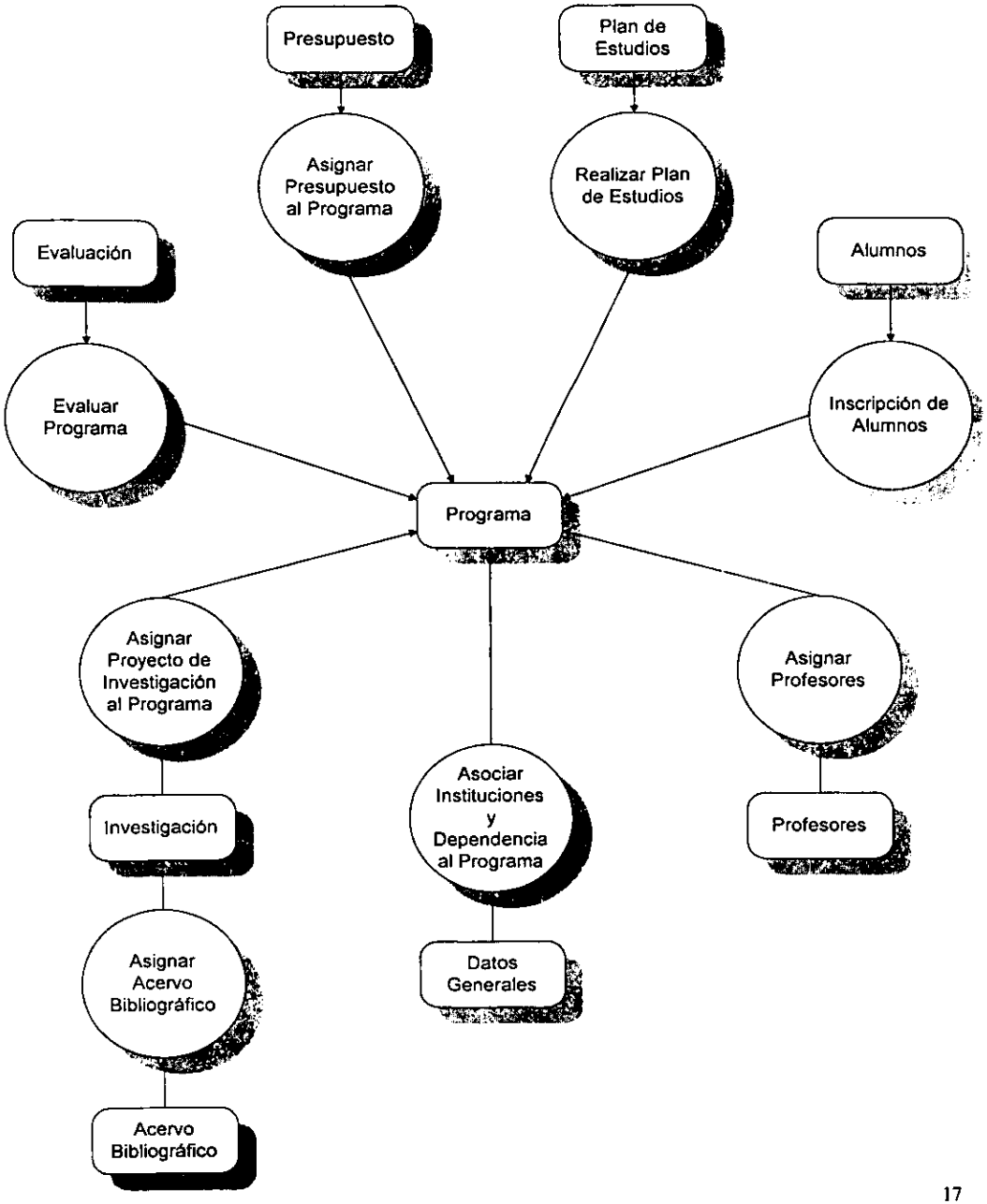
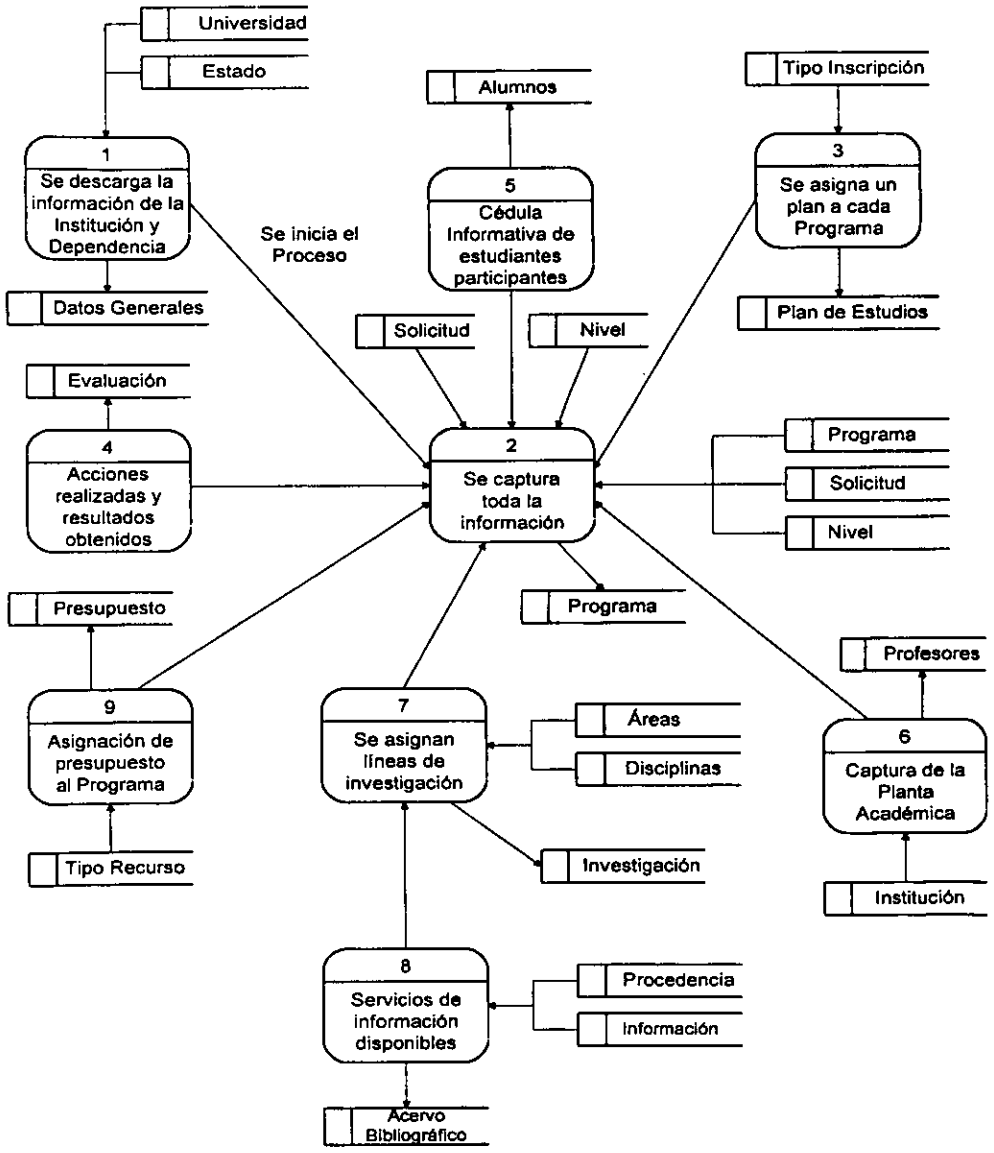


Diagrama de Flujo de Datos – Diagrama Cero

El siguiente diagrama es el Diagrama de Flujo de Datos Final. Este DFD nos va a permitir ver de una manera global, el comportamiento que va a tener nuestro sistema.



Como se puede ver, en el DFD General, se observan unas nuevas entidades numeradas. Estas entidades son utilizadas para representar cada uno de los procesos que se van a ejecutar durante la vida de este sistema. A su vez cada entidad ha sido numerada, para poder llevar a cabo el análisis sistemático y desglosado de cada uno de los sub-DFD que se van a ir generando.

El siguiente paso del análisis consiste en analizar a detalle cada una de las entidades que aparecen en el diagrama anterior. Los nuevos diagramas que se generen también se llaman DFD, con la diferencia de que cada uno de ellos se va a su vez re-numerando de acuerdo a la entidad a la que pertenezca, p.e. en el caso de los Datos Generales, sus DFD se va a llamar Diagrama Uno.

Diagrama de Flujo de Datos – Diagrama Uno

Precisamente es a continuación donde mostraremos el DFD que se genera de la entidad 1.

Este "Diagrama Uno" presenta el flujo de los Datos Generales, los cuales son creados a partir de la información de las Instituciones y las Dependencias, los cuales son llenados con los catálogos de los nombres de las Universidades, de los Estados y la Ciudad.

Como podemos ver, los datos que se generan dentro del Diagrama Uno – Datos Generales, son necesarios para alimentar a los datos del Programa

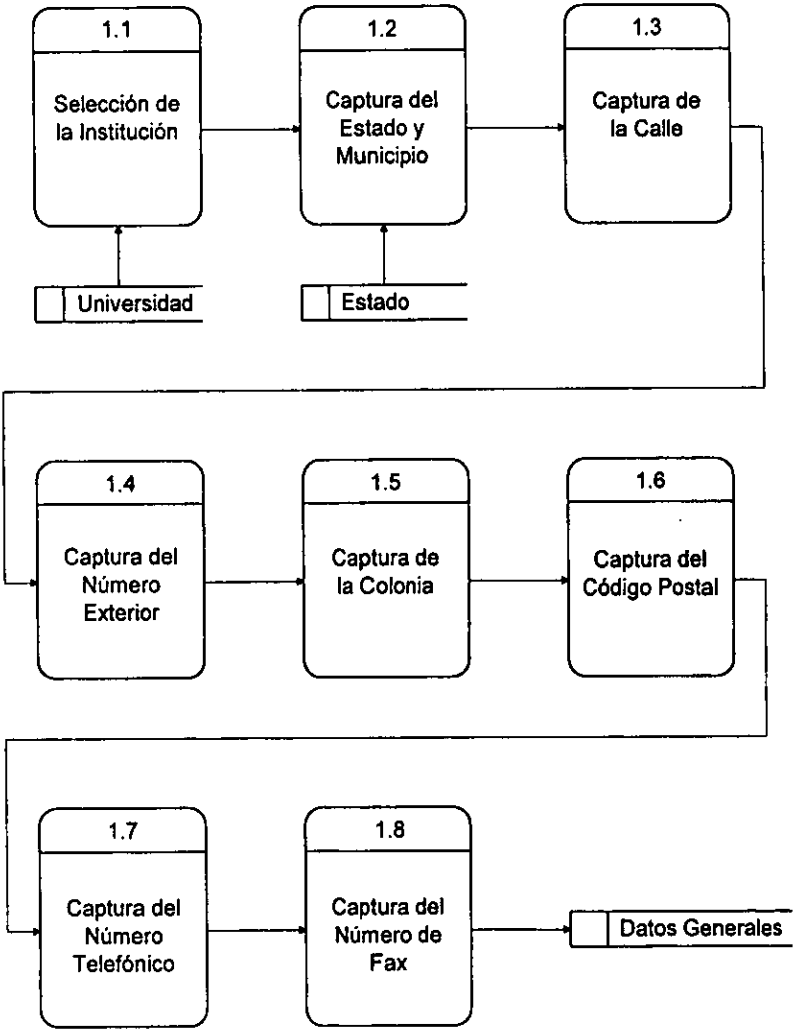

1.1
Selección de la Institución
1.2
Captura del Estado y Municipio
1.3
Captura de la Calle
Universidad
Estado
1.4
Captura del Número Exterior
1.5
Captura de la Colonia
1.6
Captura del Código Postal
1.7
Captura del Número Telefónico
1.8
Captura del Número de Fax
Datos Generales

Diagrama de Flujo de Datos – Diagrama Dos

Este diagrama nos muestra que para poder capturar la información del Programa, requerimos los datos que provienen de los catálogos de Nombre de Programa, el Tipo de Solicitud y el Nivel.

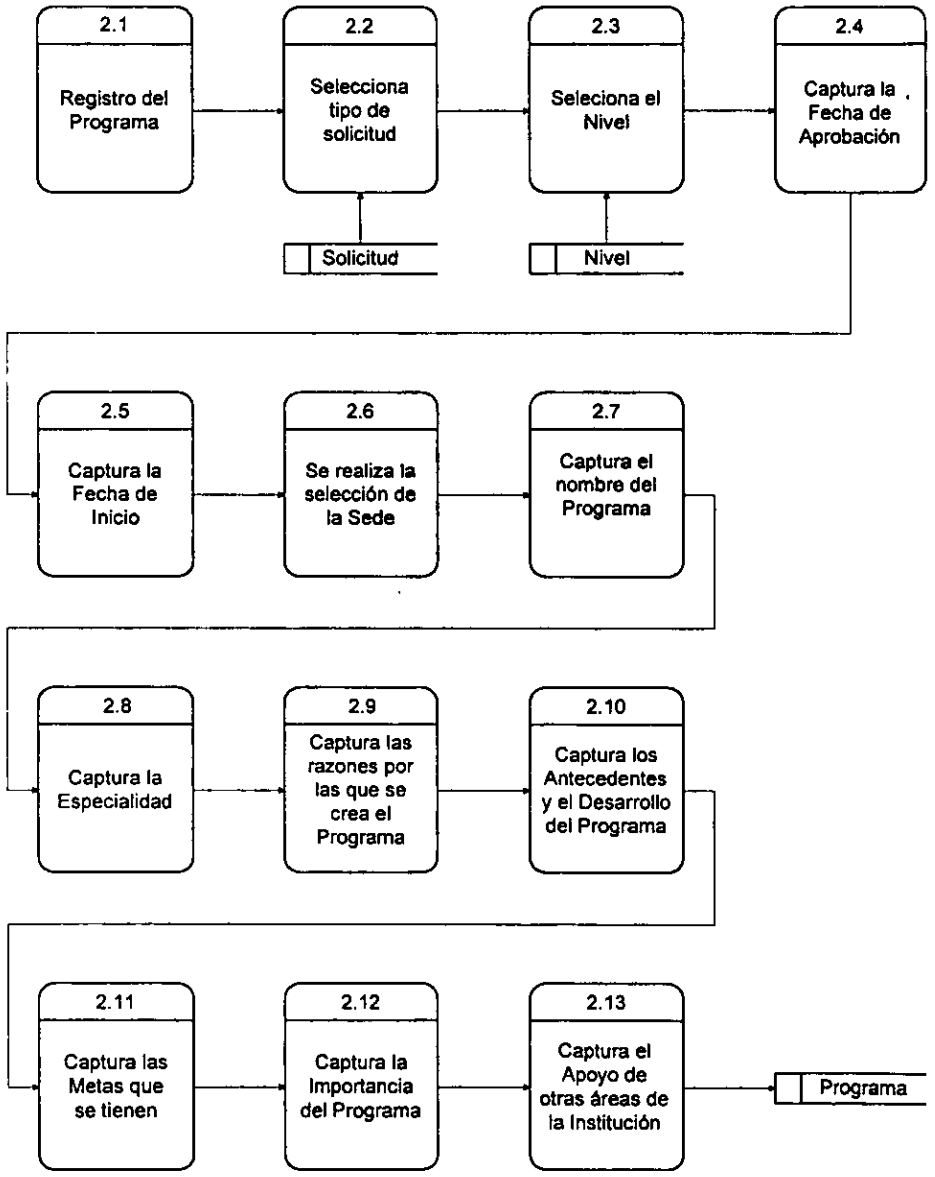


Diagrama de Flujo de Datos – Diagrama Tres

Este diagrama es muy sencillo, como podemos ver, los Datos generales del Plan de Estudios requieren información del Tipo de Inscripción y de la relación directa que se tiene con el Programa.

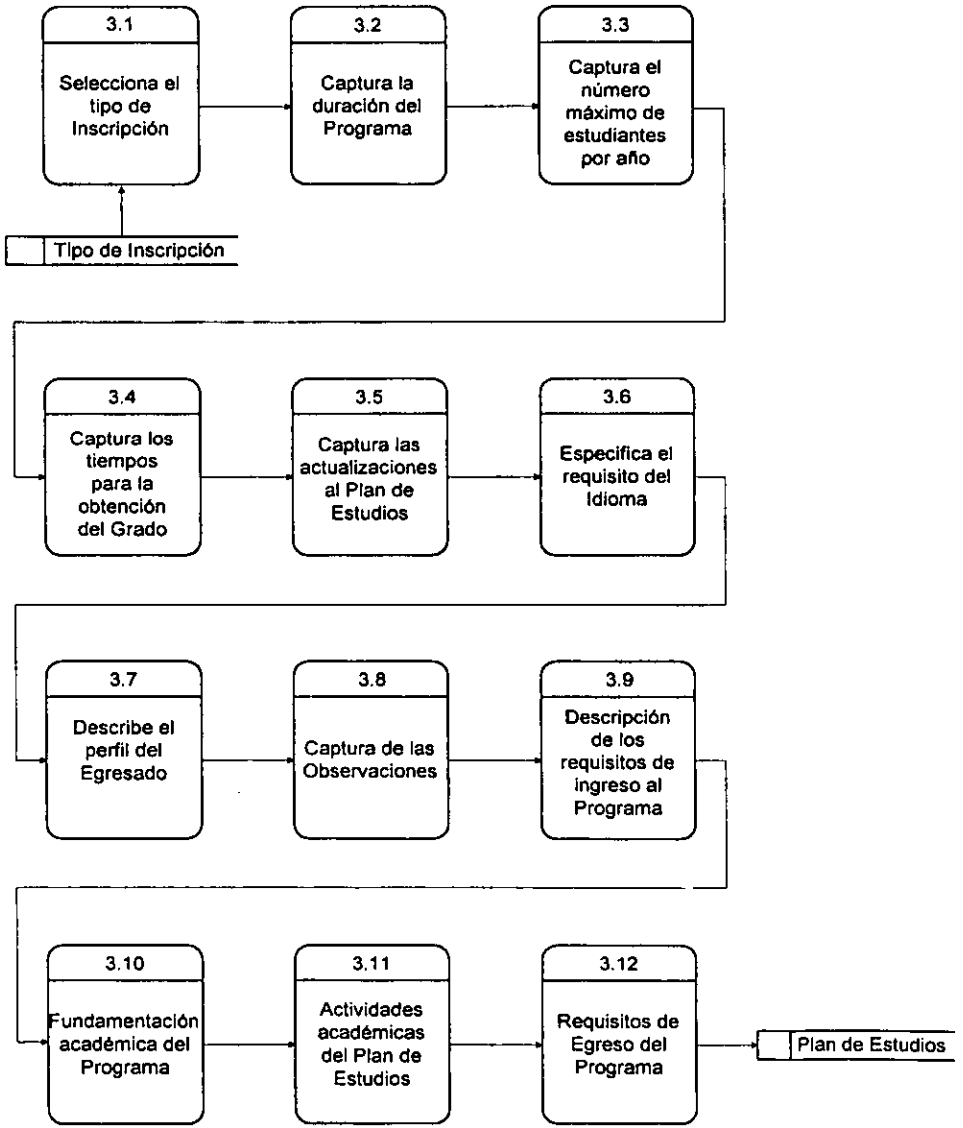


Diagrama de Flujo de Datos – Diagrama Cuatro

Este diagrama nos muestra la relación que se tiene entre la Evaluación y el Programa mismo.

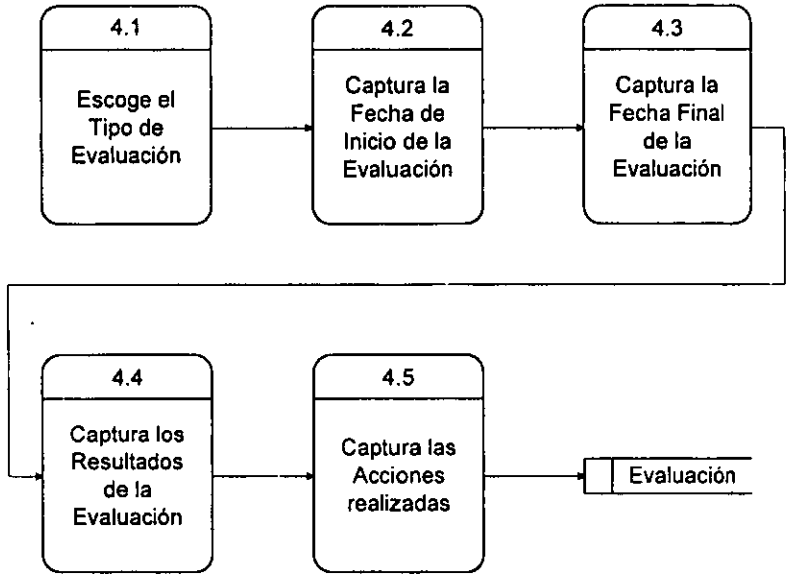
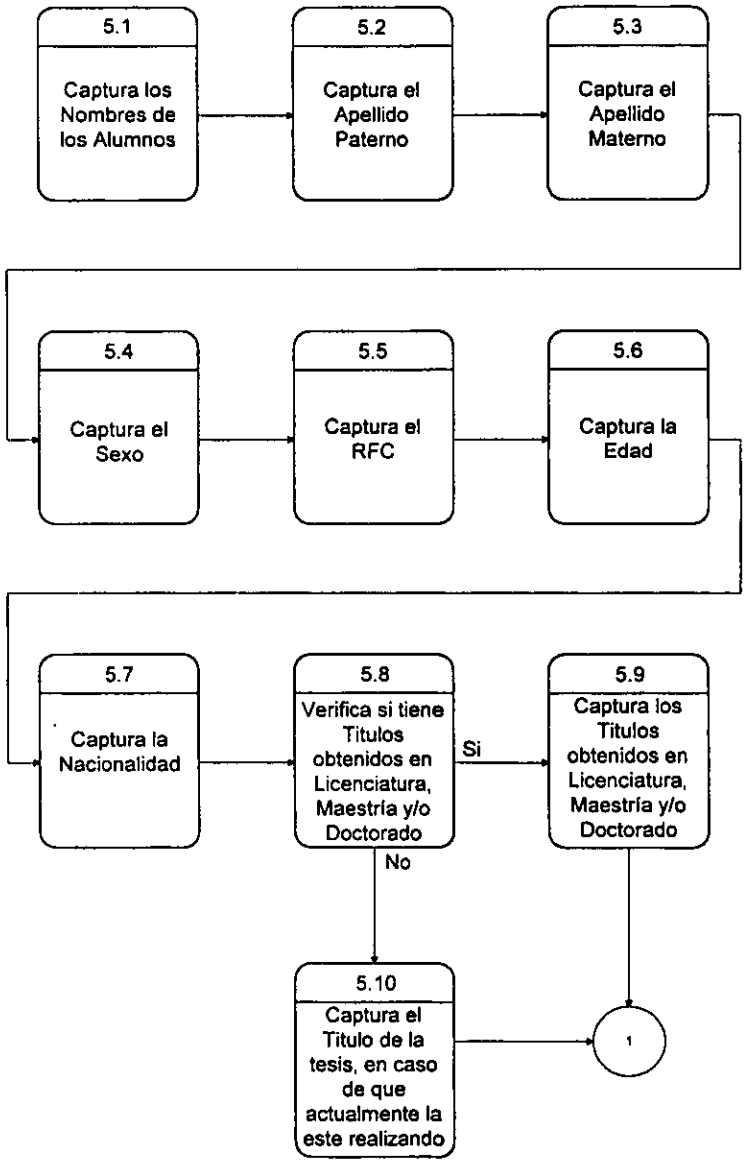


Diagrama de Flujo de Datos – Diagrama Cinco

Este diagrama nos muestra el panorama general que tiene la información de los Alumnos.



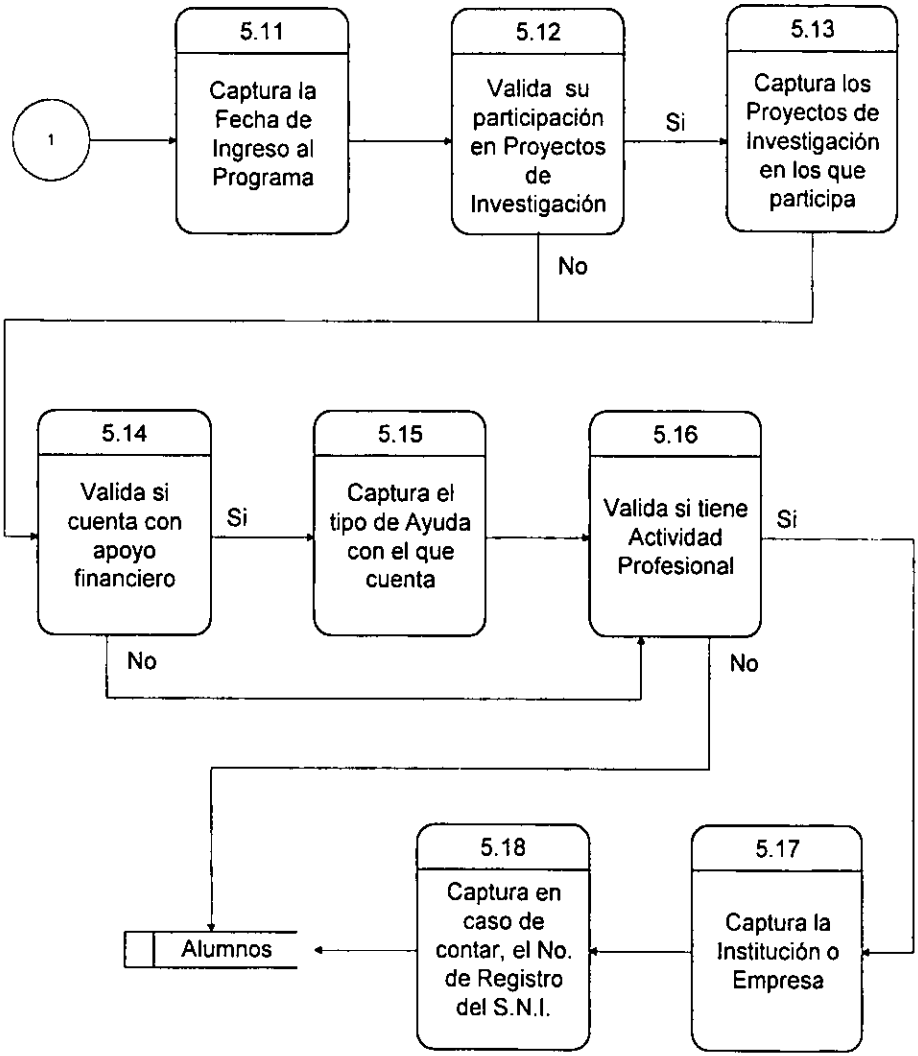

1
5.11
Captura la Fecha de Ingreso al Programa
5.12
Valida su participación en Proyectos de Investigación
Si
5.13
Captura los Proyectos de Investigación en los que participa
No
5.14
Valida si cuenta con apoyo financiero
Si
5.15
Captura el tipo de Ayuda con el que cuenta
5.16
Valida si tiene Actividad Profesional
Si
No
No
Alumnos
5.18
Captura en caso de contar, el No. de Registro del S.N.I.
5.17
Captura la Institución o Empresa

Diagrama de Flujo de Datos – Diagrama Cinco.Nueve.Uno

Debido a que el proceso 5.9 es aún más amplio y detallado, incluimos a continuación el diagrama 5.9.1 a detalle para su mejor entendimiento.

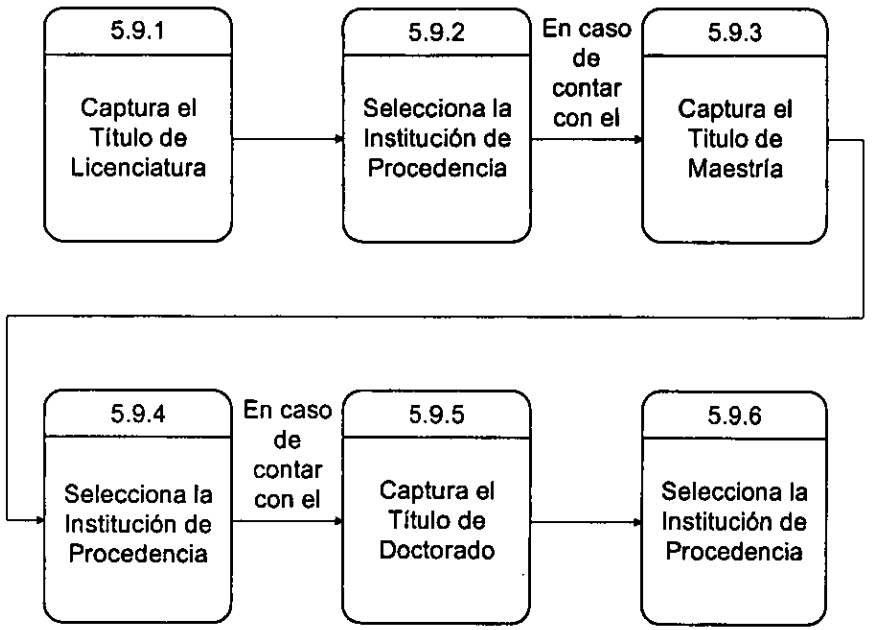
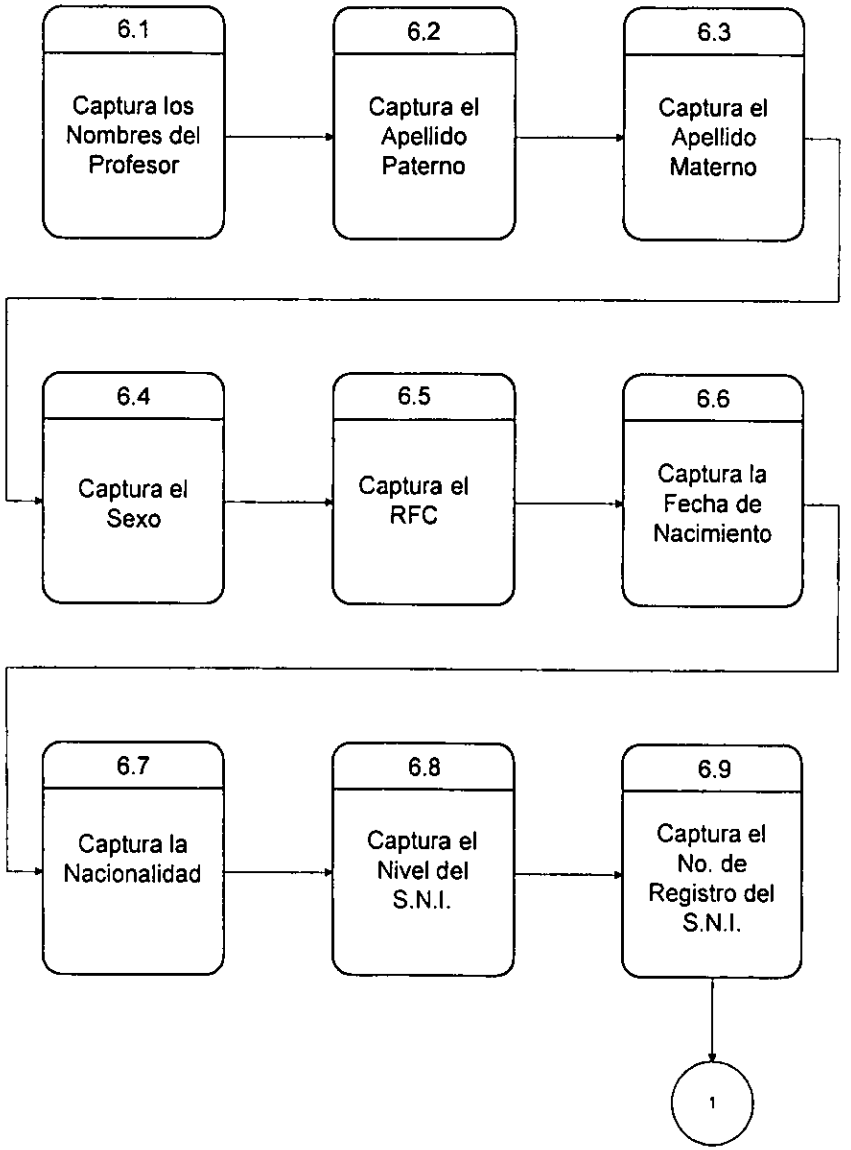
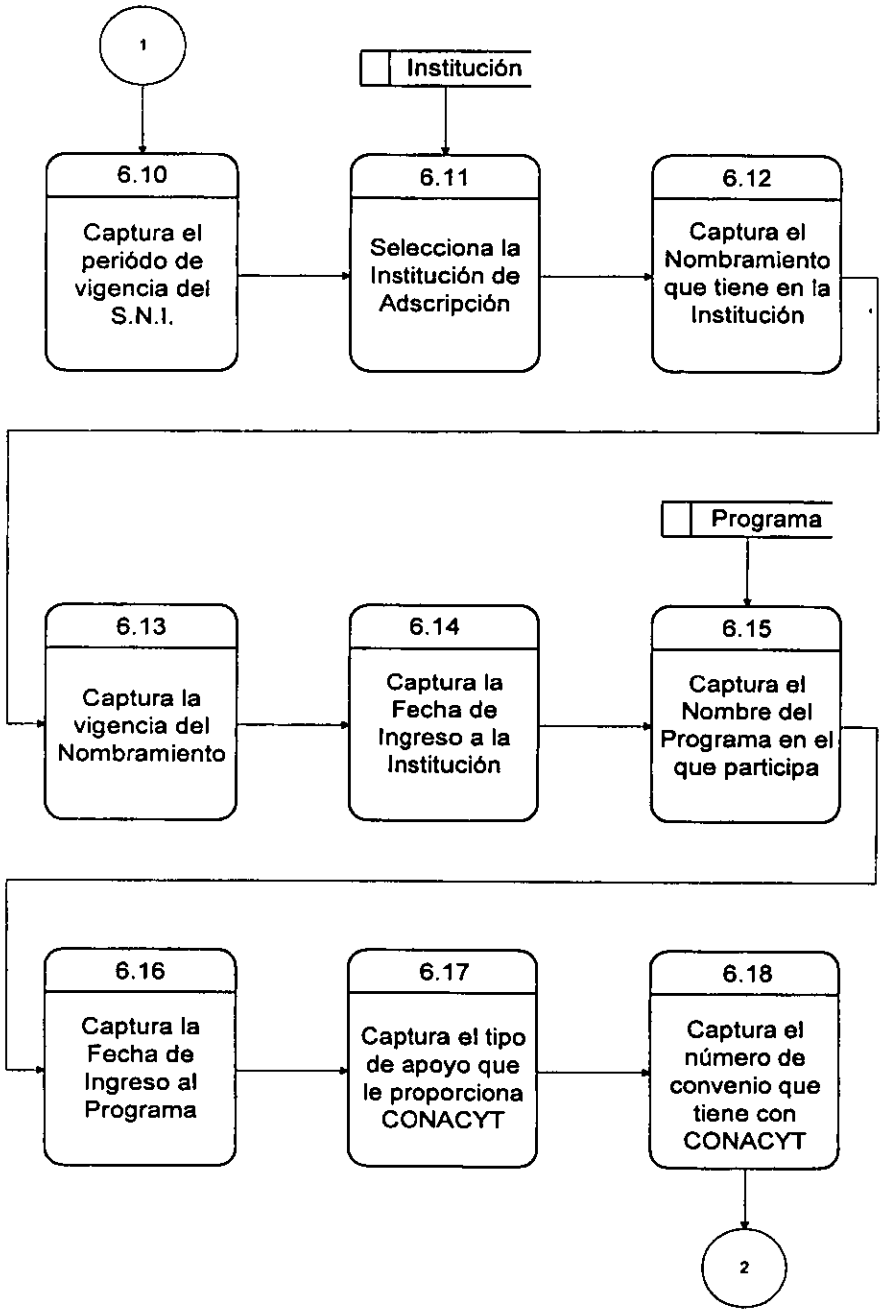


Diagrama de Flujo de Datos – Diagrama Seis

Este diagrama nos muestra de forma general la forma en que la información de los profesores se relaciona con la Institución y el Programa.




1
Institución
6.10
Captura el periódo de vigencia del S.N.I.
6.11
Selecciona la Institución de Adscripción
6.12
Captura el Nombramiento que tiene en la Institución
Programa
6.13
Captura la vigencia del Nombramiento
6.14
Captura la Fecha de Ingreso a la Institución
6.15
Captura el Nombre del Programa en el que participa
6.16
Captura la Fecha de Ingreso al Programa
6.17
Captura el tipo de apoyo que le proporciona CONACYT
6.18
Captura el número de convenio que tiene con CONACYT
2

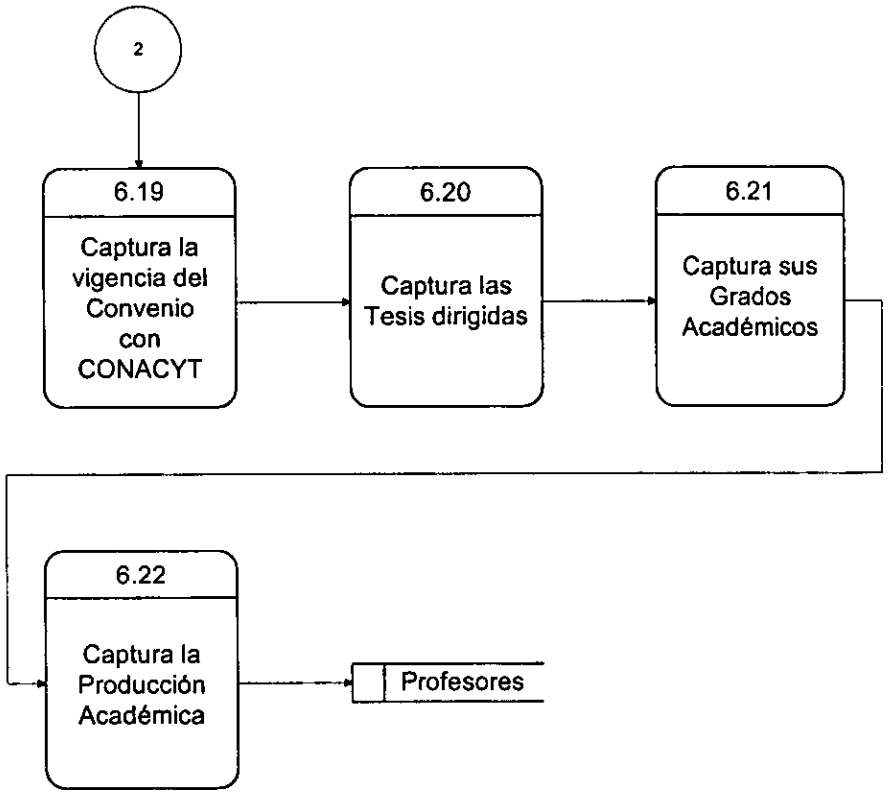

2
6.19
Captura la vigencia del Convenio con CONACYT
6.20
Captura las Tesis dirigidas
6.21
Captura sus Grados Académicos
6.22
Captura la Producción Académica
Profesores

Diagrama de Flujo de Datos – Diagrama Seis.Veintiuno

La información referente a la Captura de los Grados Académicos de los Profesores se muestra en el sub-proceso 6.21, mostrado a continuación.

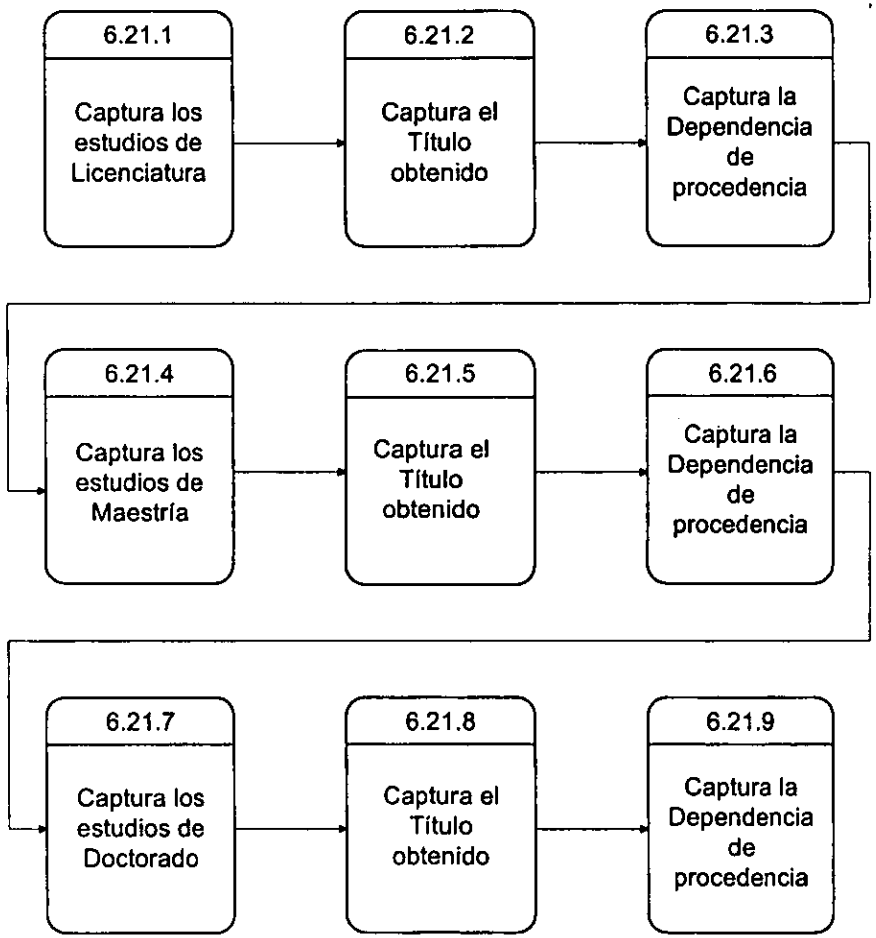


Diagrama de Flujo de Datos – Diagrama Seis.Veintidós

El detalle del proceso de la Captura de la Producción Académica de los profesores se muestra enseguida en el sub-proceso 6.22.

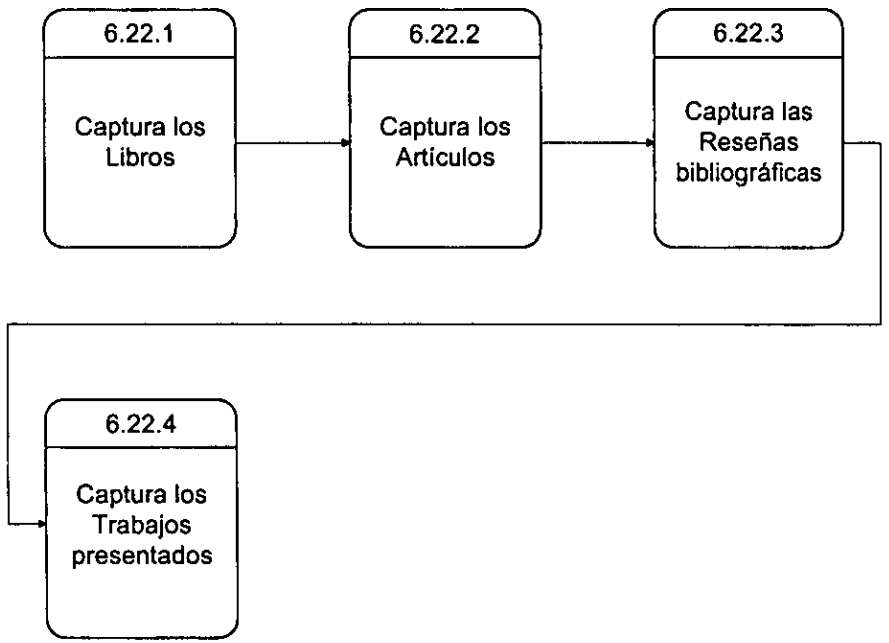


Diagrama de Flujo de Datos – Diagrama Siete

Este diagrama, a su vez se divide en dos partes, Área y Disciplina, las cuales requieren de su propio catálogo.

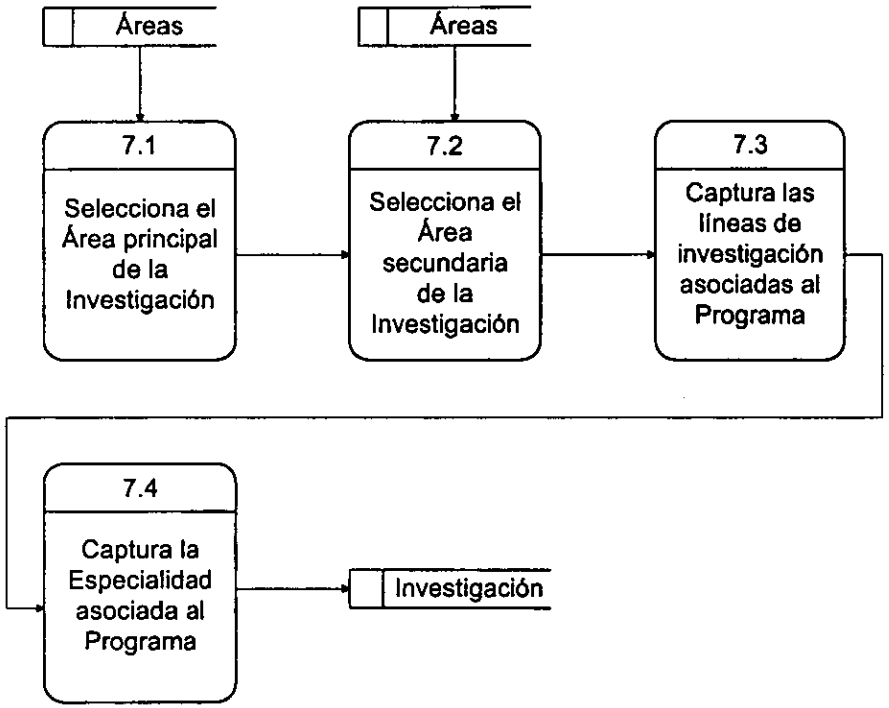


Diagrama de Flujo de Datos –Diagrama Ocho

Este diagrama muestra que para los Datos Generales del Acervo Bibliográfico se requiere de la información que se encuentra en los catálogos de los Servicios de Información y del lugar de Procedencia.

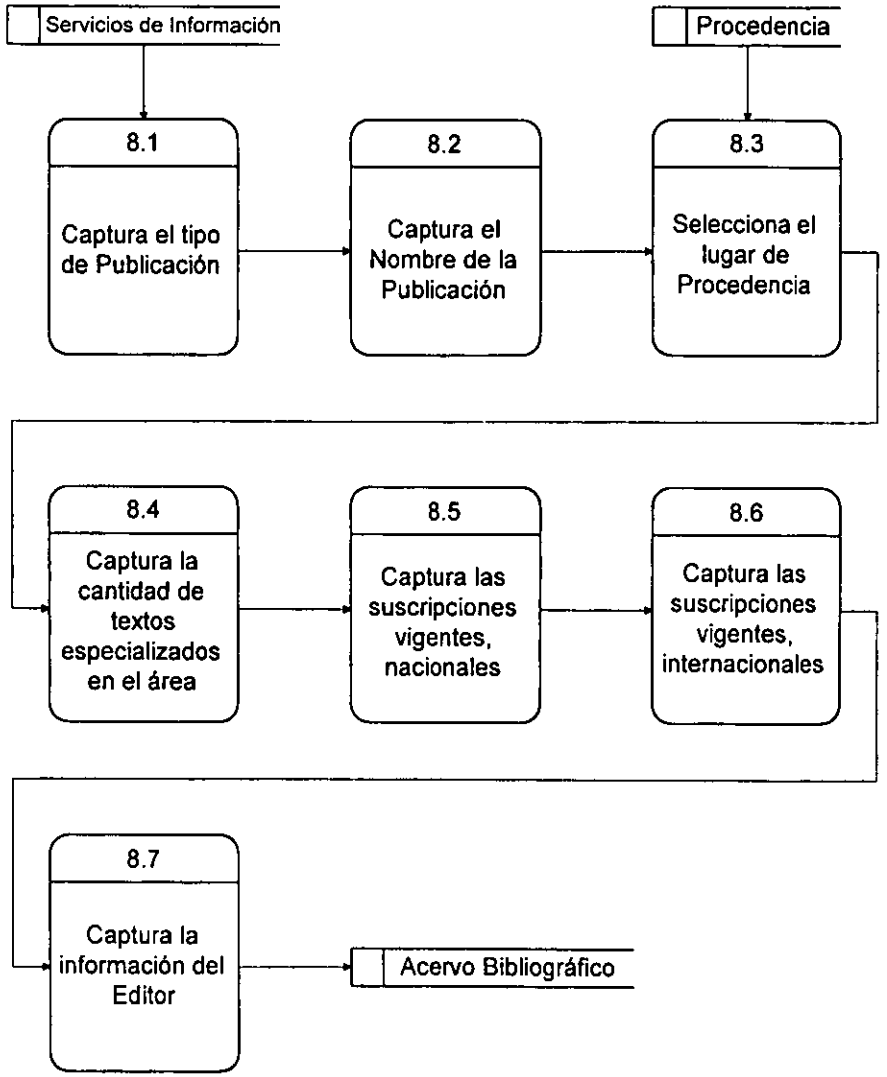
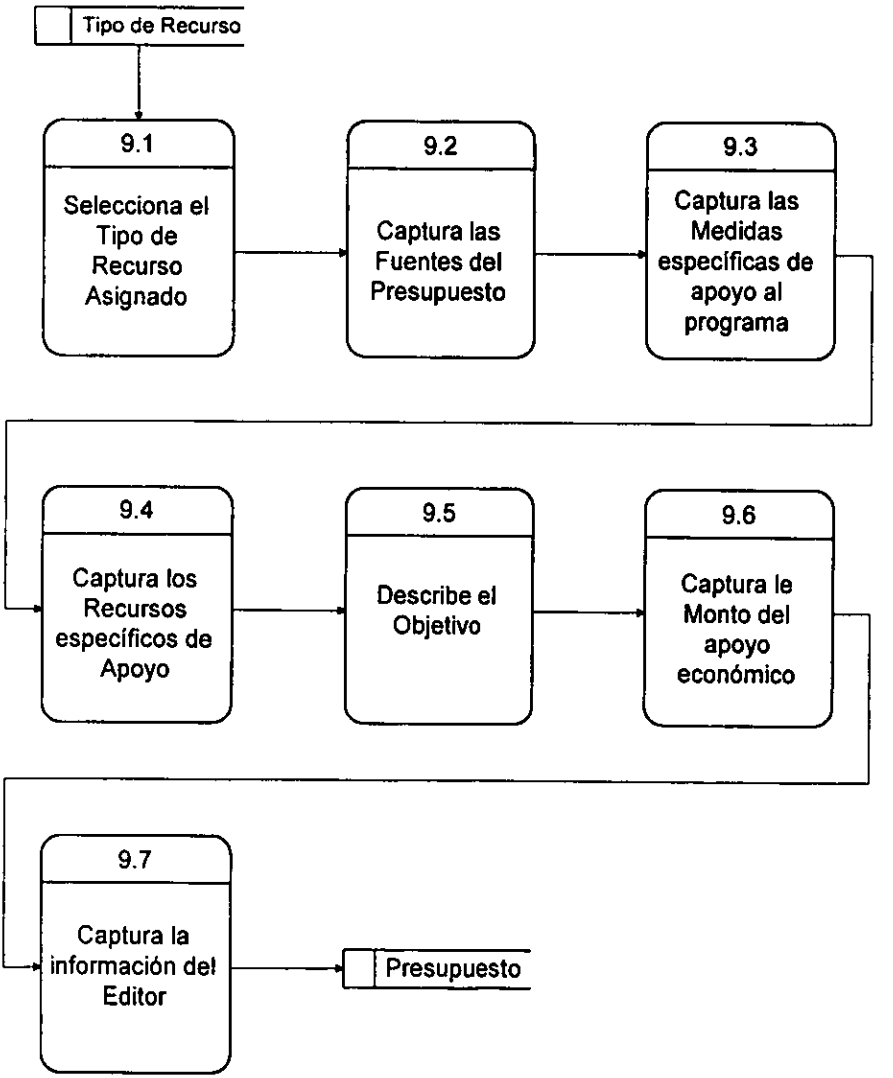


Diagrama de Flujo de Datos – Diagrama Nueve

En este diagrama son requeridos los datos de la institución que proporciona los recursos y el Tipo de recurso que se asigna al programa. También podemos observar la salida de información que envía al Programa.



Diseño

El diseño es el siguiente paso a seguir, contando con la información obtenida del análisis. Es en esta parte donde transformaremos la información recabada durante los análisis realizados en cada uno de nuestros puntos anteriores.

Para esto, es necesario hacer algunos comentarios y presentar ciertos fundamentos teóricos que son importantes durante el diseño.

El Diseño es el proceso de invención de la solución a un problema. El diseño se inicia con el prototipado (estructuración de un modelo rudimentario) ya que gran parte del prototipado es un tipo de diseño. El modelo de un prototipo es generalmente mucho menos complicado que el del producto final, tanto en los contenidos como en el proceso utilizado para construirlo. El prototipado también incluye pruebas de disponibilidad, ya que es imperioso el conocimiento de las herramientas con las que contamos para llevar a cabo el diseño.

El diseño de la arquitectura es un diseño a grandes rasgos, inventando y especificando las principales interfaces de los sistemas que interactúan en el sistema completo. El diseño minucioso especifica las interfaces detalladas de los objetos que interactúan dentro del sistema y sus interacciones con los otros objetos.

2.1 Diseño de Bases de Datos

El diseño de las Bases de Datos, requiere identificar perfectamente todas las exigencias de información de los usuarios y representarlas, en modelos muy bien definidos. Para llevar a cabo esto, se necesita observar con gran detenimiento la

naturaleza de las condiciones de los usuarios y el significado preciso de la representación de los mismos.

Una base de datos es una colección de elementos interrelacionados que pueden procesarse por uno o más sistemas de aplicación.

Un modelo es una sencilla representación de la realidad que presenta solo los detalles que son relevantes.

Este modelo se realiza a su vez en tres niveles, que son:

a) Metodología del diseño
b) Esquema de base de datos
c) Estado actual de la realidad

En el primer nivel, se dice que el estado en curso de una base de datos en particular es un modelo de la realidad porque es un registro de hechos seleccionados sobre la realidad misma.

El segundo nivel describe una estructura de la base de datos como un modelo de un conjunto de modelos. El esquema modela un enorme rango de estados de la base de datos, definiendo aquellas características que todos estos estados tienen en común.

El tercer nivel describe las reglas que pueden ser utilizadas en la formulación de un esquema.

Normalmente se habla del modelo conceptual de datos, como una metodología para la creación de esquemas de bases de datos para situaciones particulares de aplicación. Estos esquemas son en sí mismos, modelos que proveen una estructura lógica para capturar hechos sobre una porción particular de la

realidad. Cuando estos hechos son capturados y registrados en la base de datos, entonces la base de datos es en sí misma un modelo del estado actual de la realidad.

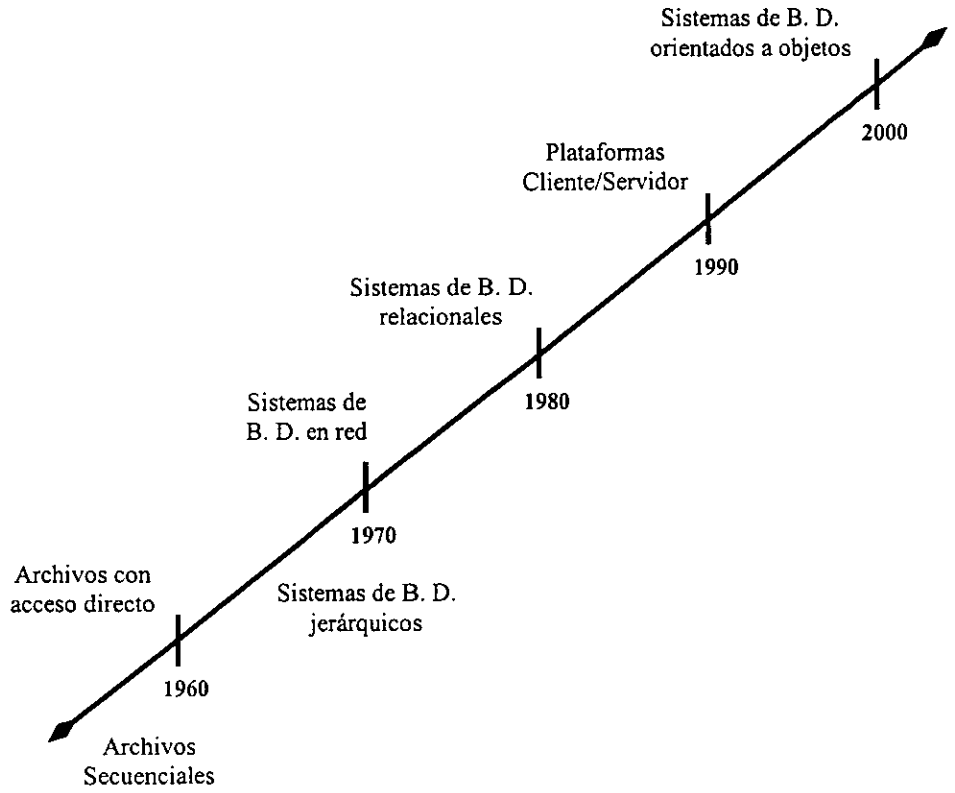
A continuación presentamos un esquema con los tres niveles del modelado.

Nivel del modelo	Ejemplo del modelo	Construcciones Típicas
Metodología del diseño	Entidad-relación, orientado a objetos, etc.	Objetos, relaciones, tablas, columnas.
Esquema de la base de datos	Esquema de la base de datos	Persona, nombre, dirección, claves, etc.
Estado actual de la realidad	Base de datos	Insurgentes Sur 1735, San Ángel Inn

Con el paso de los años y el avance de la tecnología, las Bases de Datos también han ido evolucionando. Parte de esta evolución se ha reflejado en la flexibilidad que se tiene para realizar ciertos cambios cuando estos sean necesarios en una organización.

Las primeras Bases de Datos, eran simples archivos que permitían una "sencilla manipulación" de datos. Este concepto ha ido evolucionando, de tal forma que ahora nos permite un verdadero manejo de la información, con niveles de seguridad y confiabilidad verdaderamente serios.

La siguiente figura nos muestra como ha sido la evolución de las Bases de Datos.



El modelado de bases de datos se puede realizar con distintas técnicas, siguiendo los tres principios arriba mencionados. Algunas de las técnicas que han existido y que pueden utilizarse son:

- *Sistemas orientados a los archivos o sistema de procesamiento de datos.*- Un sistema automatizado para procesar los datos de los registros de una organización.
- *Sistema del modelo jerárquico.*- Los primeros sistemas de bases de datos introducidos a mediados de los sesenta estaban basados en el modelo jerárquico, un modelo de datos que indique que todas las interrelaciones pueden estructurarse como jerarquías. Soportan el acceso a varios registros relacionados con un registro simple.
- *Sistema del modelo en red.*- Una interrelación de datos en la cual un registro puede estar subordinado a registros de más de un archivo. Soportan las interrelaciones jerárquicas y no jerárquicas en redes entre los datos.
- *Sistemas de base de datos relacionales.*- El modelo relacional se introdujo entre la comunidad de bases de datos en 1970, soportan todas las interrelaciones lógicas entre los datos. El acceso a los datos es lógico, independiente de las técnicas de instrumentación física.
- *Plataforma Cliente/Servidor.*- Red local que consiste en computadoras clientes que reciben servicios de una computadora servidor. La computadora servidor ofrece servicios de base de datos a las máquinas clientes.

El modelo que nosotros usamos para el diseño de nuestra base de datos es el Modelo Relacional, también conocido como Modelo de Entidad-Relación.

El modelo entidad–relación se introdujo en 1976 y ha tenido mucha aceptación en el diseño de base de datos. Ayuda al analista con tres conceptos semánticos principales: entidades, relaciones y atributos.

- *Entidades.* Son objetos distintos en una organización.
- Relaciones. Son interacciones significativas entre objetos.
- Atributos. Describen las entidades y las relaciones.

En el modelo entidad-relación se consideran las entidades como abstractas, pero significan cosas que existen dentro de la organización. Esas cosas se modelan como entidades que se pueden describir con atributos, también pueden interactuar con otras en cualquier número de relaciones.

Las relaciones también se tratan como objetos abstractos. Cada relación describe una interacción entre una entidad y otra, la relación se une con líneas a las entidades que participan en ella y las líneas tienen nombres diferentes a otros nombres de las líneas que emanan de la relación.

Los valores de los atributos de una relación describen los efectos o los métodos de interacción entre entidades.

El modelo entidad-relación, también se puede extender a una base de datos del tipo Cliente/Servidor.

Inicialmente los servidores se instalaron para controlar la impresión y el acceso a los archivos, esto creó los Servidores de impresión y los Servidores de archivos. Hoy, sin embargo, la mayoría de servidores son servidores de bases de datos a los clientes. Así un cliente que está ejecutando un proceso de una aplicación y necesita una consulta a la base de datos, envía la petición al servidor de la base datos y esté le devolverá los datos solicitados, el programa del cliente también puede enviar datos al servidor con la solicitud de actualizar la base de datos y el servidor efectuará esta actualización.

- Servidor. Es una computadora que suministra a los clientes servicios tales como bases de datos, conexiones a red o grandes controladores de disco. Los servidores pueden ser grandes computadoras, minicomputadoras, estaciones de trabajo o incluso una PC. Más de un servidor puede estar involucrado en suministrar servicios a los clientes.
- Clientes. Es una computadora o una estación de trabajo conectado a una red y que se utiliza para accesar a los recursos de la red.

- Servidor de base de datos. Programa que corre en un servidor para dar servicios de bases de datos a las estaciones de trabajo (clientes).

La potencia de la plataforma cliente/servidor descansa en el concepto división de funciones.

2.2 Interbase

La base de datos que construiremos, la desarrollaremos en Interbase. Este es un manejador tan potente como Informix, Oracle, SQL Server o Sybase, cuya propiedad pertenece a Inprise-Borland.

InterBase permite a los desarrolladores trabajar en un ambiente Cliente/Servidor o Stand Alone, sin la necesidad de comprar un servidor adicional.

Una base de datos en Interbase es creada utilizando lenguaje SQL, las cuales pueden dividirse en dos categorías: oraciones del lenguaje de definición de datos (DDL por sus siglas en ingles) y las oraciones de lenguaje de manipulación de datos (DML por sus siglas en inglés).

Las estructuras subyacentes de la base de datos (tablas, vistas e índices) son creadas con oraciones DDL. Colectivamente los objetos definidos con oraciones DDL se conocen como *metadatos*. La *definición de datos* es el proceso de crear, modificar y borrar metadatos. Inversamente, las oraciones DML son usadas para popularizar la base de datos con datos y para manipular los datos almacenados existentes en las estructuras previamente definidas con oraciones DDL.

Las oraciones DDL que crean metadatos inician con la frase CREATE, las oraciones que modifican los metadatos inician con la frase ALTER, y las oraciones

que borran los metadatos inician con la frase DROP. Algunas definiciones básicas son:

- CREATE DATABASE. Crea una base de datos.
- CREATE TABLE. Crea una tabla.
- ALTER TABLE. Altera una tabla.
- DROP TABLE. Vacía una tabla.

Basado en los requerimientos previamente definidos, es necesario identificar los objetos que necesitan estar en la base de datos, entidades y atributos. Una *entidad* es un tipo de persona, objeto o cosa que necesita ser descrita en la base de datos. Puede ser un objeto con una existencia física, como una persona, un carro o un empleado o puede ser tal vez un objeto con una existencia conceptual como una compañía, un trabajo o un proyecto. Cada entidad tiene propiedades llamadas *atributos.*

En una base de datos, los objetos que representan a una entidad se llaman *tablas*, la cual es una matriz bidimensional de renglones y columnas. Cada columna en una tabla representa un atributo. Cada renglón en la tabla representa una *instancia* específica de la entidad. Después de identificar las entidades y atributos, se crea el modelo de datos, el cual sirve como un diseño lógico para crear la base de datos.

El modelo de datos de las entidades y atributos de las tablas y columnas puede ser presentado mediante un diagrama de la base de datos (se detallará posteriormente).

Una de las tareas del diseño de una base de datos es proveer un camino que identifique únicamente cada ocurrencia o instancia de una entidad, así que el sistema puede recuperar cualquier renglón en una tabla. Los valores especificados en la llave primaria de la tabla distinguen los renglones de una y otra. Una Llave Primaria

(PRIMARY KEY) o Única (UNIQUE) asegura que los valores introducidos en la columna o conjunto de columnas son únicas en cada renglón. Si se intenta insertar un valor en una Llave Primaria o Única que ya existe en otro renglón de la misma columna, InterBase previene la operación y envía un mensaje de error.

Cuando se diseña una tabla, es necesario desarrollar un conjunto de reglas para cada tabla que estabilice y asegure la integridad de los datos. Estas reglas son:

- *Especificar el tipo de datos.* Una vez que se ha escogido un atributo dado como una columna en una tabla, se puede escoger un tipo de datos para el atributo. El tipo de datos define el conjunto de datos validos que la columna puede contener. Los tipos de datos también determinan cuales operaciones pueden ser desempeñadas en los datos, y define los requerimientos de espacio en disco para cada valor. Las categorías generales de los tipos de datos SQL incluyen:
    - Datos de tipo Caracter.
    - Datos numéricos del tipo Entero.
    - Datos del tipo flotante y decimal.
    - Datos del tipo Date para especificar fechas y horas.
    - Datos del tipo Blob para especificar datos binarios no estructurados, como gráficos y voz digitalizada.
- *Escoger un conjunto de caracteres internacionales.* Cuando se crea una base de datos se puede especificar un conjunto de caracteres por default. El uso de estos caracteres es para los interesados en proveer una base de datos para uso internacional. Aquí se especifica el tipo de caracteres que se van a usar de acuerdo a estándares, p.e. el ISO8859_1 soporta todos los lenguajes europeos.

- *Crear una columna basada en un dominio.* Cuando varias tablas en una base de datos contienen columnas con la misma definición y tipos de datos, se puede crear una definición de dominio y almacenarlos en una base de datos.

Los usuarios que crean tablas pueden hacer referencia al dominio para definir los atributos localmente.

- *Poner valores por default y estados nulos.* Se pueden poner valores por default opcionales. Estos valores pueden ser nulos o no nulos.

- *Definición de integridades limitadas.* Son reglas que gobiernan las relaciones de columna-tabla y tabla-tabla y validan las entradas de los datos. Ellas miden todas las operaciones de acceso a la base de datos y son automáticamente mantenidas por el sistema. Las integridades pueden ser aplicadas a una tabla entera o a una columna individual.

- *Definición de chequeos limitados.* Conjuntamente con la prevención de duplicación de valores usando llaves Únicas y Primarias, se puede especificar otro tipo de validación de entrada de datos. Un CHECK pone una condición o requerimiento en los valores de los datos en una columna a la vez que el dato es introducido.

Después de tener las tablas, columnas y llaves definidas, se debe observar el diseño como una totalidad y analizarla usando la normalización en orden de encontrar errores lógicos. La normalización involucra partir las tablas más grandes en unas más pequeñas, en orden al grupo de datos unidos a su naturaleza relacionada.

Cuando una base de datos esta diseñada utilizando métodos apropiados para la normalización, los datos relacionados a otros datos no necesitan ser almacenados en mas de un lugar. Las ventajas de almacenar los datos en un solo lugar son:

- Los datos son fáciles de actualizar y de borrar.
- Cuando cada dato es almacenado en una locación y accesado por referencia, la posibilidad del error debido a la existencia de duplicados es reducida.

- Debido a que el dato es almacenado solo una vez, la posibilidad para introducir datos inconsistentes es reducida.

En general, los procesos de normalización incluyen:

- Eliminación de grupos repetidos.
  Cuando un campo en una columna dada contiene mas de un valor en una ocurrencia de la llave primaria, entonces el grupo de datos se le conoce como *grupo repetido*. Esta es una violación a la primera forma normal, la cual no permite atributos multi-valores.
- Remoción parcial de columnas dependientes.
  Otro importante paso en el proceso de la normalización es remover cualquier columna no-llave que sea dependiente en solamente parte de una llave compuesta. Se dice que cada columna tiene una *llave de dependencia parcial*. Las columnas no llaves proveen información sobre el sujeto, pero no solamente la define.
- Remoción transitiva de columnas dependientes.
  El tercer paso en la normalización es remover cualquier columna no-llave que dependa de otra columna no-llave. Cada columna no-llave debe ser un solo hecho sobre la columna de la llave primaria.

2.2.1 Base de Datos Contra Modelo de Datos

Es importante distinguir entre la descripción de una base de datos y una base de datos misma. La descripción de la base de datos se llama *modelo de datos* y es creada durante el tiempo de diseño. El modelo es una plataforma para diseñar tablas y columnas. El modelo de datos describe la estructura lógica de la base de datos incluyendo los objetos o entidades, los tipos de datos, las operaciones de usuario, las relaciones entre objetos y las integridades.

En un modelo de base de datos relacional, las decisiones sobre los diseños lógicos son completamente independientes de la estructura física de la base de datos. Esto permite una gran flexibilidad:

- No es necesario definir las rutas de acceso físico entre los objetos de los datos a la hora del diseño, así es que se puede "preguntar" a la base de datos sobre cualquier relación lógica que exista en ella.
- La estructura lógica que describe la base de datos no es afectada por los cambios en la estructura de almacenaje subyacente. Se puede fácilmente transportar una base de datos relacional a una diferente plataforma porque los mecanismos de acceso a la base de datos definidas por el modelo de datos permanezcan a los mismos sin considerar como los datos están almacenados.
- La estructura lógica de la base de datos es también independiente de lo que el usuario final observa. El usuario final puede crear una versión personalizada de las tablas de la base de datos subyacente por medio de *vistas*. Una vista despliega un subconjunto de los datos a un usuario o grupo dado. Las vistas pueden ser utilizadas para ocultar datos importantes, o para filtrar la salida de los datos.

2.3 Lenguaje SQL

Para poder visualizar la información de nuestra base de datos es necesario realizar consultas por medio de un lenguaje especial. Los tres lenguajes más importantes son SQL (Structured Query Language, que significa Lenguaje de Consulta Estructurado), QBE (Query by Example, que significa Consulta por Ejemplos) y QUEL (Query Language, que significa Lenguaje de Consulta). El lenguaje que utilizaremos es SQL.

SQL fue el resultado de un proyecto de investigación de IBM, este proyecto incluyó el desarrollo de un sistema de base de datos relacional. Al final de los setenta SQL pasó a formar parte del dominio público y estuvo primero disponible como lenguaje para un sistema comercial de Oracle Corporation. En 1986 fue aprobado el primer estándar ANSI para SQL.

SQL es un lenguaje estándar para sistemas relacionales y ha quedado como el único lenguaje relacional de base de datos que es ANSI estándar. Además de las facilidades de consulta, incluye la definición de tablas, la actualización y definición de vistas y el otorgamiento de privilegios.

SQL contiene una gran variedad de capacidades de manipulación de datos, tanto para consulta como para actualización de una base de datos. Estas capacidades dependen sólo de la estructura lógica de la base de datos, no de su estructura física, consistente con los requisitos del modelo relacional.

2.4 Aplicaciones Cliente/Servidor

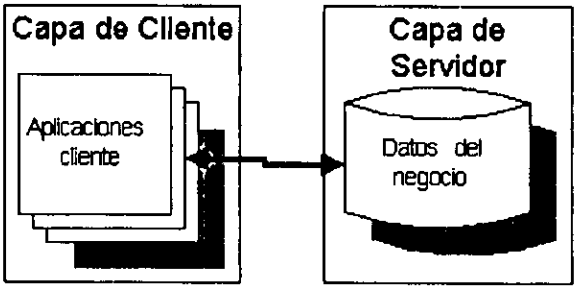
Una plataforma cliente/servidor es una red de computadoras en donde una o más ofrecen servicios, precisamente de ahí viene el nombre de servidor. El resto de las computadoras, son los clientes. Cada tipo de máquina se optimiza para que lleve a cabo sus funciones específicas dentro del sistema de la red. De esta manera se optimizan los servicios para brindar servicios a muchos clientes. Los servicios de bases de datos suelen ser los más frecuentes. Por otro lado, los sistemas clientes se optimizan para la entrada y presentación de los datos. Éstos incluyen normalmente interfaces gráficas de usuario (GUI) que brindan medios orientados al usuario para facilitar la entrada de datos y para dar formato y mostrar los datos de salida.

Toda aplicación trata de reflejar parte del mundo real, para automatizar tareas que de otro modo serían llevadas a cabo de modo más ineficiente, o que

simplemente no podrían realizarse. Para ello es necesario que cada aplicación refleje las restricciones que existen, de modo que nunca sea posible llevar a cabo acciones no válidas. Para poder llevar a cabo esto, es necesario seguir ciertas reglas, las cuales fueron descritas con anterioridad (en el apartado dedicado a InterBase).

En realidad, la información puede ser manipulada por muchos programas distintos, lo cual hace más difícil garantizar que todos respetan las reglas, especialmente si las aplicaciones corren en diversas máquinas, bajo distintos sistemas operativos, y están desarrolladas con distintos lenguajes y herramientas.

En un esquema Cliente/Servidor clásico existen dos capas, el cliente y el servidor. Este último está ubicado normalmente en otra máquina y suele ser un manejador de bases de datos, como Informix, Oracle, SQL Server, InterBase, etc., aunque puede ser una base de datos más pequeña como Paradox, Access, dBase, etc., que accesamos directamente desde la aplicación cliente, tal y como se muestra en la siguiente imagen.



Los mejores manejadores de base de datos relacionales, proporcionan soporte para implementar en ellos el uso de llaves primarias, integridad referencial, triggers, etc.

Suponiendo que tengamos la información en un manejador de bases de datos potente, podremos despreocuparnos de llevar a cabo la codificación de numerosas validaciones en nuestras aplicaciones; así, si en la base de datos creamos una regla de integridad referencial que indica que todo Programa pertenece a un Alumno, el manejador de base de datos rechazará cualquier intento de almacenar un pedido en el que se nos haya olvidado indicar el mismo. Cualquier aplicación que acceda a esta base de datos se beneficiará de esta y otras validaciones automáticamente, sin tener que añadir ni una línea de código.

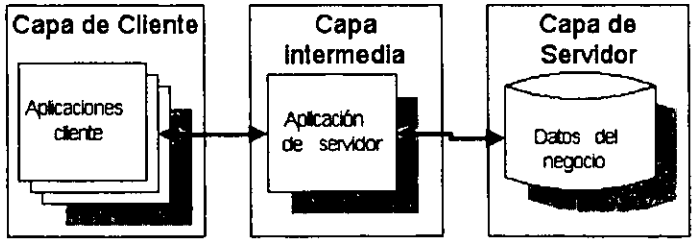
Si se está utilizando una base de datos menos potente, como dBase o Access, no tenemos entonces la misma suerte. Casi todas las reglas deberán implementarse dentro de los programas que accedan a la base de datos. Si los programas que acceden a la base de datos son varios, garantizar que en todos ellos se respetan todas las reglas, lo cual puede llegar a ser muy difícil y engorroso, especialmente si se desarrollan con distintas herramientas.

La necesidad de implementar las reglas dentro de las aplicaciones cliente puede surgir también utilizando manejadores de bases de datos más potentes. En primer lugar, las bases de datos relacionales son cada vez más potentes, pero no todas las reglas pueden reflejarse en ellas: por ejemplo, las reglas de flujo son bastante difíciles de implementar dentro de la base de datos, y suelen ser las aplicaciones cliente las que controlan que la información siga una ruta válida a través del sistema. Sin embargo, muchas de las reglas pueden reflejarse adecuadamente al nivel de la base de datos con estos manejadores.

El problema se agrava cuando la información se encuentra en distintas bases de datos, administradas por distintos manejadores, digamos InterBase y Oracle: si por ejemplo, en InterBase almacenamos la información sobre Plan de Estudios y en la base de datos Oracle almacenamos otra información, como la del Programa, ¿cómo reflejamos que ciertos Planes de Estudio corresponden a un Programa, y garantizamos que no podamos relacionar un Plan de Estudios con un Programa?.

Evidentemente, aquí no hay manera de establecer una regla de integridad referencial entre tablas almacenadas en dos bases de datos distintas y correspondientes a distintos manejadores de base de datos. De nuevo, la solución al problema es implementar el chequeo en cada aplicación cliente, comprobando que exista el Plan de Estudios en la tabla de InterBase antes de referenciarla en la tabla de Programa de Oracle.

Ya que parece que de cualquier modo seremos nosotros mismos los encargados de obligar a que se cumplan algunas reglas, puede ser conveniente encontrar la manera de centralizar la administración de estas reglas en un único lugar, de modo que todo el código necesario no se haya de duplicar en cada una de las aplicaciones. La solución puede ser crear una aplicación que se encargue de llevar a cabo estas tareas, de modo que todos los clientes pidan o envíen información a la misma, no al administrador de base de datos en el servidor: a éste solo accederá la nueva aplicación, que conforma una nueva capa dentro de un sistema Cliente/Servidor, la capa intermedia o middle-tier, con lo que nuestro sistema ha pasado de ser un sistema Cliente/Servidor convencional a ser un sistema con tres capas (three-tiered). Conviene apuntar que pueden haber varias de estas aplicaciones, que llamaremos servidores de aplicación, lo que permite distribuir la carga de trabajo.



Hemos hablado hasta ahora de la capa intermedia como si se tratara de una aplicación cualquiera, pero esto no es así. En los esquemas Cliente/Servidor

tradicionales de dos capas, suele ser el manejador de bases de datos el que proporciona la conectividad, así como capacidades tan fundamentales como el soporte de transacciones.

La introducción de una capa intermedia puede romper con esto, al ser necesario un modo de comunicar las aplicaciones cliente con la aplicación que lleva a cabo las labores de la capa intermedia, siendo ahora ésta la que se aprovecha de las capacidades de conectividad, el soporte de transacciones y las distintas capacidades que proporciona el manejador de base de datos. Solucionar todos los problemas de conectividad, etc., no es tarea fácil, y lógicamente debería utilizarse uno o más productos que solucionen algunos de estos problemas, basados en DCOM, CORBA y tecnologías similares: este es el caso de MIDAS/OleEnterprise, de Inprise, que proporciona protección contra la caída de servidores, conectividad e importación de algunas reglas a los clientes (aunque no soporta transacciones distribuidas), y de Microsoft Transaction Server (que proporciona conectividad y transacciones distribuidas, pero no protección contra caídas de los servidores).

La decisión de dónde ubicar una determinada regla dentro de una arquitectura Cliente/Servidor de tres capas puede simplificarse mucho si se atiende al tipo de regla de que se trata, utilizando la clasificación introducida más arriba.

Las reglas del modelo de datos especifican los valores válidos de cada atributo de las diversas entidades que se almacenan, lo que simplificando es lo mismo que decir los valores válidos para cada campo de cada tabla. Estas reglas deben, a ser posible, reforzarse en el servidor: esto se hace en primer lugar escogiendo correctamente el tipo de los campos de cada tabla (no almacenar una fecha en un campo de tipo cadena si nuestra base de datos dispone de campos del tipo fecha), y donde el servidor lo soporte, mediante restricciones (por ejemplo, en Interbase es posible especificar mediante CHECK diversas condiciones que debe verificar un dato para poder almacenarse en un campo), etc. El hacer esto así proporcionará mayor robustez a la base de datos.

Como complemento de esto, también se deben implementar estas validaciones al nivel del cliente, por una simple razón: evitar trabajo y esperas innecesarias a los usuarios. Lo mejor sería no tener que llevar a cabo ningún tipo de programación, y que las reglas del modelo de datos se pudieran importar del servidor al cliente, haciéndose en el mismo gran parte de los controles necesarios para garantizar la validez de la información introducida por el usuario: esto no es muy difícil, dado que son chequeos relativamente sencillos, al afectar solo a un campo, y de hecho algunos productos, como OleEnterprise de Inprise, incorporan la posibilidad de importar estas reglas del servidor.

Por lo que respecta a las reglas de relación, el lugar más adecuado para implementarlas es, sin lugar a dudas, el servidor. La mayor parte de los manejadores de base de datos proporcionan integridad referencial y los mecanismos necesarios para implementar fácilmente estas reglas, y esto proporciona una robustez enorme a la base de datos. Además, el hecho de que los datos necesarios para verificar si se respetan las relaciones, residan en la misma base de datos/máquina hace que estas verificaciones sean muy rápidas, mientras que si el chequeo se hace en el cliente se incrementará el tráfico de red. El problema lo encontraremos si el negocio está distribuido en varias bases de datos: si ese es el caso, será necesario implementar alguna de estas reglas en la capa intermedia, y que ésta resida lo más cerca posible de las bases de datos, de ser posible en la misma máquina o red local.

Las reglas de derivación pueden variar mucho en complejidad: la información derivada más simple, puede calcularse a partir de otros campos del mismo registro. Dado que no se requiere información adicional, y en general toda la información de un registro viaja a la vez al cliente, calcular esta información en el mismo es trivial, y no resulta peligroso. Si tenemos los datos distribuidos en diversas bases de datos, ésta será la única solución para implementar muchas de las reglas de derivación.

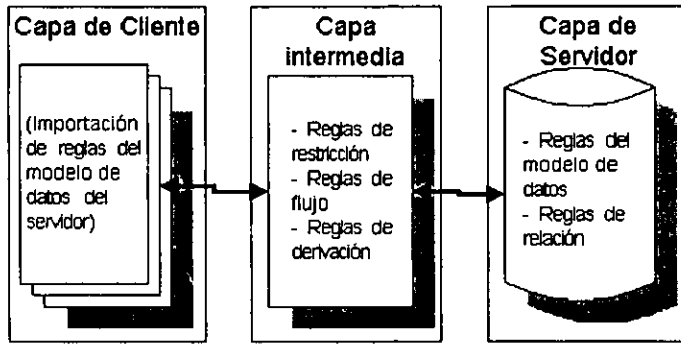
Por otro lado, las reglas más sencillas pueden también implementarse en la capa intermedia, de modo que tengamos las reglas centralizadas en una única aplicación, no dispersas por diversas aplicaciones, sin importar lo sencillas que sean.

Las reglas de restricción deben implementarse en el servidor, o en la capa intermedia. Dado que estas reglas contemplan restricciones en los datos que dependen casi siempre de información presente en varias tablas, llevar a cabo el control en el cliente puede implicar cierto tráfico de red, por lo que es más conveniente situar la implementación de la regla más cerca de los datos. El lugar ideal podría ser el servidor, pero aquí nos encontramos con las limitaciones del SQL, y con el posible problema del acceso a diversas bases de datos, por lo que ubicar estas reglas en la capa intermedia puede ser de nuevo una buena solución.

Las reglas de flujo son excelentes candidatas a ser implementadas en la capa intermedia, dado que su complejidad suele ser bastante grande, lo que las hace inmanejables por un manejador de bases de datos.

No hay una única posibilidad a la hora de distribuir un esquema Cliente/Servidor. Sin embargo, sí hay ciertas pautas que se pueden tener en cuenta a la hora de tomar una decisión, en general, lo más recomendable suele ser implementar todas las reglas al modelo de datos y las relaciones en el servidor, dado que los modernos servidores suelen llevar a cabo estas tareas muy eficazmente.

Si tuviésemos que acceder a varias bases de datos, la mejor solución suele ser implementarlas en la capa intermedia: habríamos de migrar aquí algunas de las reglas que de otro modo irían al servidor, especialmente las de relación. Si bien el tráfico en la red se incrementará al utilizar una capa intermedia, este puede quedar aliviado haciendo que ésta resida en la misma máquina que el servidor de datos, o al menos dentro de la misma red local. La siguiente figura muestra gráficamente la ubicación recomendable de las reglas dependiendo de su tipo (para acceso a una única base de datos).



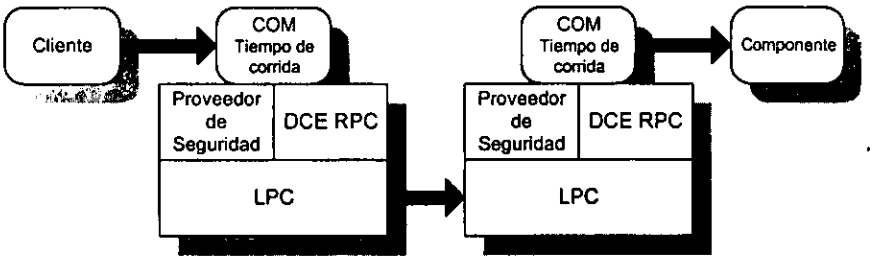
Por último, vale la pena resaltar la conveniencia de implementar las reglas del modelo de datos también en el cliente, para hacer fluida la interacción con el usuario, siendo lo ideal importarlas dinámicamente del servidor de base de datos. Por lo demás, la implementación de reglas en las aplicaciones cliente puede dar lugar a muchos problemas, tanto de velocidad como de portabilidad y fiabilidad, al tener que reflejar una y otra vez las mismas reglas en distintas aplicaciones, quizá desarrolladas con distintos lenguajes y ejecutándose bajo distintos sistemas operativos.

2.5 DCOM

DCOM (Distributed Common Object Model: Modelo de Objeto Común Distribuido) es una extensión de COM, conviene primero hablar sobre COM.

Las aplicaciones que funcionan con Windows 95 y Windows NT tienen espacios de memoria protegidos. Esto asegura que una aplicación no pueda acceder a la memoria ocupada por otra aplicación. Aunque en general este es un modelo adecuado, existen razones para proporcionar un nivel de comunicación entre

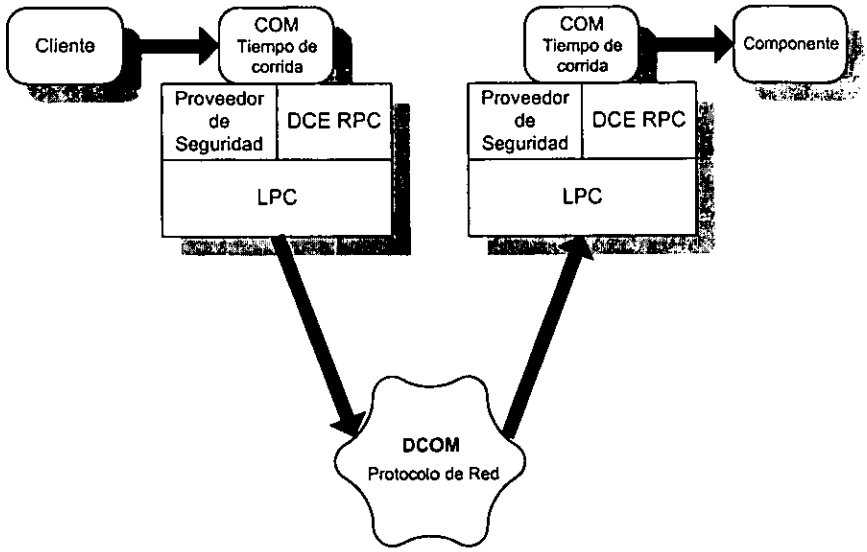
aplicaciones. Este nivel de comunicación es proporcionado en Windows 95 y Windows NT mediante COM. .



COM ofrece el protocolo que permite interactuar con otra en la misma máquina. La siguiente imagen muestra la arquitectura COM.

Ahora que COM ha permitido la posibilidad de comunicarse con otra aplicación que funcione en la misma máquina, el siguiente paso es proporcionar la capacidad de comunicarse con una aplicación funcionando en una máquina distinta dentro de la red. Esto es esencialmente lo que ofrece DCOM.

La siguiente imagen representa la arquitectura DCOM.



DCOM ofrece a una organización, muchos beneficios que van más allá de proporcionar simplemente la habilidad de crear aplicaciones en red.

Si una aplicación está programada en Delphi, puede activarse y utilizarse desde una aplicación como Visual Basic. Esto evita que los desarrolladores se queden bloqueados utilizando una sola herramienta, lo que permite seleccionar la herramienta más adecuada para el trabajo.

DCOM utiliza un contador de referencia para determinar cuantos clientes están conectados a un determinado servidor. Así, cuando un nuevo cliente se añade a la lista, el contador se incrementa y, a la inversa, cuando un cliente se desconecta del servidor de aplicaciones, la referencia disminuye. Cuando el contador de referencia llega a cero, la aplicación se cierra. Cuando un servidor de aplicaciones ya

no está en funcionamiento, DCOM lanzará la aplicación y establecerá el contador de referencia de la manera más adecuada.

Como DCOM está basada en sencillos protocolos TCP/IP y RPC, los servidores de aplicación pueden llamarse desde cualquier punto de Internet utilizando TCP/IP básico o cualquier otro protocolo basado en TCP (como PPTP – Point to Point Tunneling Protocol) o usuarios HTTP de Internet. Soportando el nivel más bajo de interoperabilidad de red, la comunicación en red de DCOM es transparente para las aplicaciones. Esto ofrece muchas opciones de red distintas para el desarrollador para que tome la decisión de elegir la mejor conectividad para la aplicación.

2.6 Midas

Ya tenemos el manejador de base de datos que vamos a utilizar (InterBase), ahora lo que necesitamos es la tecnología para desarrollar las aplicaciones Cliente/Servidor. Independientemente al lenguaje de programación, requeriremos de este software.

Utilizaremos un producto de Inprise llamado Midas (Multi-tier Distributed Application Services Suite), el cual es proporcionado en los lenguajes de desarrollo de Inprise, como es el caso de Delphi 3.0. Midas ofrece un completo conjunto de componentes avanzados, servicios y tecnologías para desarrollar aplicaciones Cliente/Servidor Multi-Nivel.

A medida que la infraestructura informática actual se amplía, se descentraliza más y se hace en general más compleja, las organizaciones deben buscar tecnologías que proporcionen soluciones inmediatas. Microsoft ha hecho importantes progresos con DCOM y promete mejorarlo aún más con nuevos desarrollos en el futuro. No obstante, con MIDAS, Inprise está proporcionando las soluciones

avanzadas para crear, desplegar y mantener aplicaciones distribuidas dentro de su entorno.

Inprise ha sido muy conocido por sus compiladores de gran calidad y sus entornos RAD (Rapid Application Development). No obstante, una cosa es desarrollar con una herramienta RAD para crear aplicaciones sencillas, y otra es tener RAD para sistemas distribuidos. Con MIDAS, Inprise ofrece RAD para sistemas distribuidos.

- El Business ObjectBroker ofrece equilibrio de cargas y seguridad frente a los fallos sin codificación o esfuerzo adicional.
- El Remote DataBroker facilita la creación de objetos comerciales de nivel medio manteniendo al mismo tiempo una gran facilidad y flexibilidad para distribuir los datos a aplicaciones cliente.
- El ConstraintBroker mejora los costos de mantenimiento, centralizando la integridad de los datos para aplicaciones cliente y proporcionando al mismo tiempo mecanismos sencillos para hacer cumplir estas reglas básicas en el cliente para reducir el tráfico de red.

Midas ofrece la distribución de objetos con las siguientes características:

- Alta disponibilidad de Servidores con seguridad contra fallas.
- Equipo dinámico de cargas.
- Conjunto de Datos Distribuidos y Procesos de Transacciones.
- Aplicaciones Cliente reducidas.
- Propagación automática de regla de integridad de Bases de Datos.
- Conectividad con Bases de Datos de alta velocidad.

Midas puede trabajar con protocolos TCP/IP (que son los más ampliamente utilizados) por medio de Sockets o DCOM.

Todas sus funciones han sido diseñadas para funcionar individualmente o junto con otras, y soportar sistemas operativos y estándares del mercado.

Midas incluye las siguientes tecnologías:

- Remote DataBroker.
- Business ObjectBroker.
- ConstraintBroker.

Remote DataBroker

Ofrece a los desarrolladores la posibilidad de crear aplicaciones de base de datos distribuidas. Estas aplicaciones tienen las siguientes características:

- Se crean en varios niveles con un nivel de cliente, un nivel medio y un nivel de base de datos.
- El nivel medio puede consistir en muchas máquinas físicas y muchos servidores de aplicaciones diferentes.
- Las aplicaciones desarrolladas para el cliente son "aplicaciones reducidas de cliente".

El término "aplicación reducida de cliente" puede definirse de dos maneras. Una es que la tecnología del lado del cliente utiliza los recursos mínimos en la máquina del cliente. En el caso del Remote DataBroker, el cliente solo requiere un DLL de 140 Kb. Esta, no necesita instalación o configuración manual, ya que puede configurarse automáticamente con las posibilidades de distribución de Delphi Web. La segunda es que la aplicación cliente no tiene reglas comerciales incorporadas, el cliente es simplemente un GUI (Interface Gráfico de Usuario) de los datos. La única codificación que está dirigida a la interface del cliente es a efectos de presentación.

Utilizando Midas y Delphi, los desarrolladores pueden crear una aplicación de cliente reducida que soporte ambas definiciones. Todas las reglas comerciales están

codificadas en la aplicación de nivel medio y, utilizando la tecnología Delphi Package, los recursos utilizados en el cliente son muy pequeños.

Una aplicación de cliente reducida proporciona diversos beneficios. Requiere poca o ninguna instalación, configuración y mantenimiento. La mayoría de las veces, el software que necesita instalarse, configurarse y mantener es el software de conectividad de cliente de base de datos. La tecnología DataBroker omite este paso para los clientes y únicamente es necesario utilizar software de conectividad con la base de datos en el servidor de aplicaciones, reduciendo la instalación, configuración y mantenimiento a una sola máquina.

En su forma más básica, el cliente realiza una petición DCOM al servidor. El servidor procesa la petición recogiendo los datos e incluyéndolos en un DataPacket. El DataPacket es el protocolo empleado para transferir los datos del servidor al cliente. Una vez que el Remote DataBroker ha recogido los datos, el resultado se envía de vuelta al cliente.

Busines ObjectBroker

Uno de los beneficios de un entorno multinivel es la capacidad de distribuir el proceso a lo largo de varios servidores. Con DCOM estándar, el cliente debe reconocer el nombre del servidor cuando se intente conectar con el servidor de aplicación. Esto crea dos posibles problemas:

1. ¿Qué ocurre si la máquina no funciona correctamente?.
2. ¿Qué ocurre si el servidor de aplicación ya está sobrecargado con otras peticiones de usuario?.

Busines ObjectBroker y OLEnterprise pueden proporcionar equilibrio de cargas y seguridad frente a los fallos a cualquier servidor de automatización OLE (Servidor COM) o servidor DCE/RPC. Pueden hacerlo independientemente de la

herramienta de desarrollo utilizada para crear el servidor de aplicación, siempre que el servidor de aplicación soporte automatización OLE o sea un servidor DCE/RPC.

Hay cuatro componentes principales en esta tecnología:

1. Business ObjectBroker
2. Object Factory
3. Object Agent
4. Object Explorer

Business ObjectBroker proporciona el registro global en el que el servidor de aplicación realizará sus entradas de registro. Este es el lugar en el que la aplicación cliente mirará para obtener una conexión con uno de los servidores de aplicación de la red. Devolverá a una de las máquinas de servidor que tenga la aplicación requerida instalada y registrada al cliente que realiza la petición.

Object Factory es utilizado por el servidor. Es un servidor RPC y un cliente de automatización OLE. Actúa como un cliente de automatización OLE proxy, representando al cliente remoto. Cuando se lleva a cabo una petición OLE remota, la petición se pasa como llamada de procedimiento remoto desde el Object Factory. El Object Factory reenvía la petición original OLE de la parte de la aplicación cliente remota. Cuando se realiza una petición remota para un servidor RPC, la petición no va a través del Object Factory. En lugar de ello, el Object Agent y el objeto RPC se comunican directamente.

Object Agent es utilizado por el cliente. Es un servidor de automatización OLE y un cliente RPC. Actúa como servidor de automatización OLE proxy, representando al servidor remoto. Cuando se lleva a cabo una petición remota, esta se pasa al Object Agent. Si la petición es para un objeto OLE remoto, el Object Agent empaqueta la petición y utiliza el RPC para adelantar la petición al Object Factory en la máquina servidor. Si la petición es para un objeto RPC remoto, el Object Agent lee la

definición de la interface desde el registro y crea dinámicamente una petición RPC, que envía al servidor.

Object Explorer es una utilidad de visualización que permite a los usuarios explorar la empresa y visualizar dinámicamente objetos y servicios. Proporciona también posibilidades de importación y exportación que permiten a los clientes compartir y reutilizar objetos. El desarrollador del objeto utiliza el Object Explorer para exportar o editar un objeto. Puede buscar en registros locales, remotos y globales. Cuando un objeto es seleccionado, el Explorer importa automáticamente toda la información requerida para permitir acceso al objeto y lo carga en el registro local de usuario.

ConstraintBroker

Uno de los principales dogmas de un buen diseño de bases de datos es que las reglas de integridad de los datos, deben estar centralizadas en la base de datos. Centralizando las reglas de validación, los desarrolladores tenemos asegurado que sea cual sea la aplicación de cliente que esté actualizando la base de datos, los datos se validarán correctamente. Aunque esto asegura la integridad de los datos, no proporciona la solución completa. Específicamente, si una aplicación tuviera que tomar una decisión en la base de datos que no fuera válida, se producirían los siguientes eventos:

1. La aplicación cliente enviaría los datos al servidor.
2. El servidor iniciaría una transacción.
3. Posteriormente intentaría realizar la actualización.
4. La actualización fallaría debido a los datos no válidos.
5. La transacción sería devuelta.
6. El servidor enviaría un mensaje de error a la aplicación del cliente.
7. El usuario final sería entonces requerido para corregir los datos.
8. El proceso entero iniciaría de nuevo.

El ConstraintBroker funciona en colaboración con el Remote DataBroker. El ConstraintBroker proporciona al desarrollador un método sencillo y automático de distribuir reglas de integridad con los Datos. La naturaleza dinámica de la propagación de reglas sitúa al desarrollador de modo que tenga fácilmente una nueva parte de la aplicación. Por ejemplo, cuando una regla cambia en la base de datos, el usuario necesita simplemente actualizar el Data Dictionary.

Desarrollo

Es en este capítulo, donde vamos a aplicar la teoría vista en el capítulo anterior.

Con la etapa de análisis terminada, y con las bases ya antes mencionadas, proseguiremos a iniciar el desarrollo, primero de nuestra base de datos y posteriormente del sistema de explotación de la información.

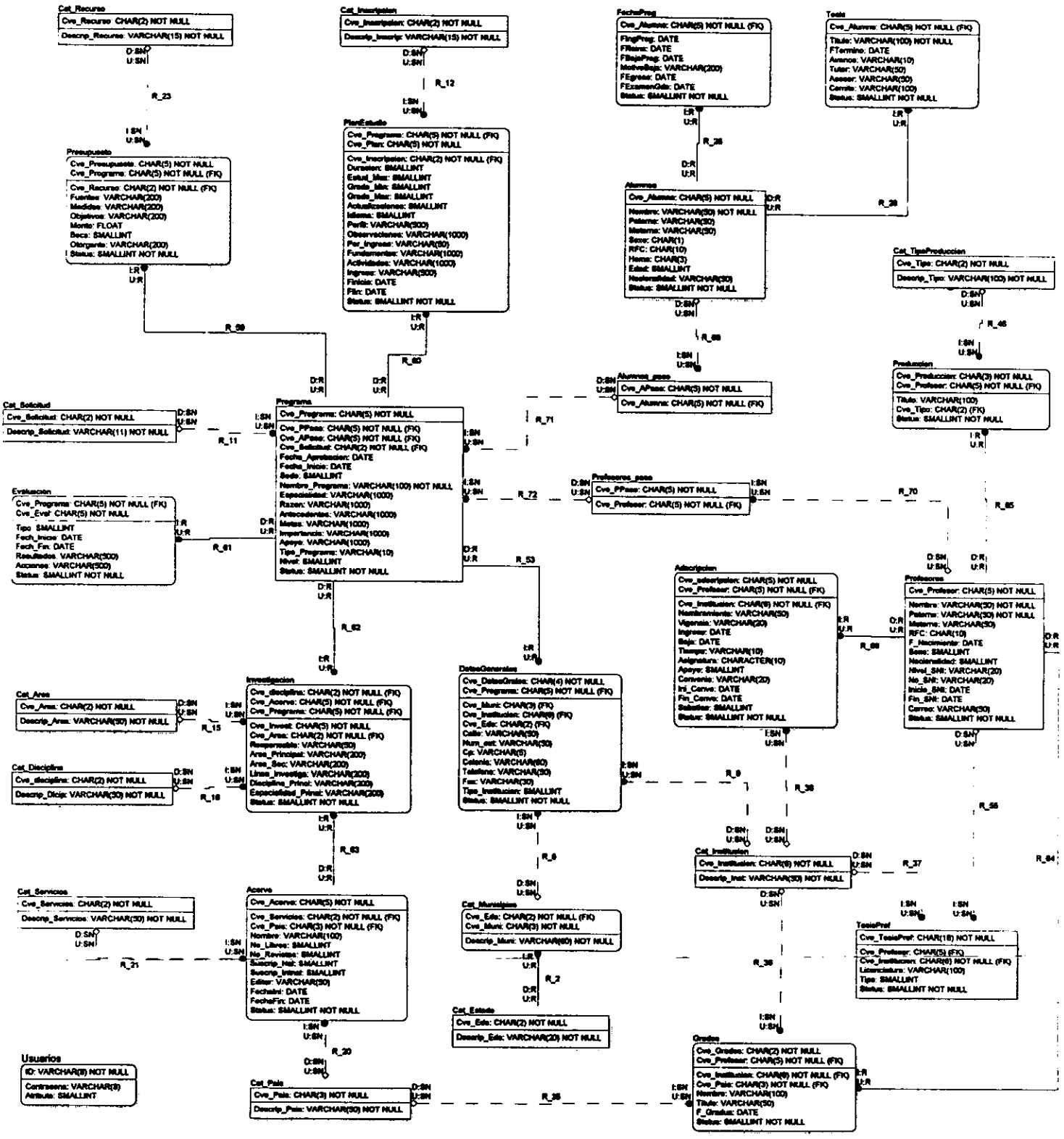
3.1 Base de Datos

En el capítulo 2 (Análisis), en el apartado 2.2 Diagrama de Flujo de Datos (DFD), nos dedicamos a analizar por medio de diagramas de flujo, la problemática que se nos presenta. Llegando finalmente a obtener el comportamiento de etapa del proceso plasmada en sus propios Diagramas de Flujo.

Con el resultado de este análisis, obtuvimos la información suficiente para modelar la base de datos de nuestra aplicación.

La base de datos con la cual vamos a trabajar está diseñada con el concepto Entidad-Relación. Esta base de datos la hemos modelado en Erwin/ERX versión 3.0 que nos permite realizar un modelaje muy profesional. El mismo es capaz de generar a partir del modelo, la base de datos y la documentación.

A continuación mostramos el esquema de la base de datos generado en Erwin.


Cat_Recurso
Cve_Recurso: CHAR(2) NOT NULL
Descrip_Recurso: VARCHAR(15) NOT NULL
Cat_Inscripcion
Cve_Inscripcion: CHAR(2) NOT NULL
Descrip_Inscrip: VARCHAR(15) NOT NULL
FechaProg
Cve_Alumno: CHAR(5) NOT NULL (FK)
FIngProg: DATE
FBaja: DATE
FBajaProg: DATE
MotivoBaja: VARCHAR(200)
FEgreso: DATE
FExamenGds: DATE
Status: SMALLINT NOT NULL
Tesis
Cve_Alumno: CHAR(5) NOT NULL (FK)
Titulo: VARCHAR(100) NOT NULL
FTermino: DATE
Asesor: VARCHAR(10)
Tutor: VARCHAR(50)
Asesor: VARCHAR(50)
Consis: VARCHAR(100)
Status: SMALLINT NOT NULL
PlanEstudio
Cve_Programa: CHAR(5) NOT NULL (FK)
Cve_Plan: CHAR(5) NOT NULL
Cve_Inscripcion: CHAR(2) NOT NULL (FK)
Duracion: SMALLINT
Estud_Max: SMALLINT
Grado_Min: SMALLINT
Grado_Max: SMALLINT
Actualizaciones: SMALLINT
Idioma: SMALLINT
Perfil: VARCHAR(500)
Observaciones: VARCHAR(1000)
Per_Ingreso: VARCHAR(50)
Fundamentos: VARCHAR(1000)
Actividades: VARCHAR(1000)
Ingreso: VARCHAR(500)
FInicio: DATE
FFin: DATE
Status: SMALLINT NOT NULL
Presupuesto
Cve_Presupuesto: CHAR(5) NOT NULL
Cve_Programa: CHAR(5) NOT NULL (FK)
Cve_Recurso: CHAR(2) NOT NULL (FK)
Fuentes: VARCHAR(200)
Medidas: VARCHAR(200)
Objetivos: VARCHAR(200)
Monto: FLOAT
Beca: SMALLINT
Otorgante: VARCHAR(200)
Status: SMALLINT NOT NULL
Alumnos
Cve_Alumno: CHAR(5) NOT NULL
Nombre: VARCHAR(50) NOT NULL
Paterno: VARCHAR(30)
Materno: VARCHAR(30)
Sexo: CHAR(1)
RFC: CHAR(10)
Homo: CHAR(3)
Edad: SMALLINT
Nacionalidad: VARCHAR(30)
Status: SMALLINT NOT NULL
Cat_TipoProduccion
Cve_Tipo: CHAR(2) NOT NULL
Descrip_Tipo: VARCHAR(100) NOT NULL
Produccion
Cve_Produccion: CHAR(3) NOT NULL
Cve_Profesor: CHAR(5) NOT NULL (FK)
Titulo: VARCHAR(100)
Cve_Tipo: CHAR(2) (FK)
Status: SMALLINT NOT NULL
Cat_Solicitud
Cve_Solicitud: CHAR(2) NOT NULL
Descrip_Solicitud: VARCHAR(11) NOT NULL
Programa
Cve_Programa: CHAR(5) NOT NULL
Cve_PPase: CHAR(5) NOT NULL (FK)
Cve_APase: CHAR(5) NOT NULL (FK)
Cve_Solicitud: CHAR(2) NOT NULL (FK)
Fecha_Aprobacion: DATE
Fecha_Inicio: DATE
Sede: SMALLINT
Nombre_Programa: VARCHAR(100) NOT NULL
Especialidad: VARCHAR(1000)
Razon: VARCHAR(1000)
Antecedentes: VARCHAR(1000)
Metas: VARCHAR(1000)
Importancia: VARCHAR(1000)
Apoyo: VARCHAR(1000)
Tipo_Programa: VARCHAR(10)
Nivel: SMALLINT
Status: SMALLINT NOT NULL
Alumnos_pase
Cve_APase: CHAR(5) NOT NULL
Cve_Alumno: CHAR(5) NOT NULL (FK)
Profesores_pase
Cve_PPase: CHAR(5) NOT NULL
Cve_Profesor: CHAR(5) NOT NULL (FK)
Evaluacion
Cve_Programa: CHAR(5) NOT NULL (FK)
Cve_Eval: CHAR(5) NOT NULL
Tipo: SMALLINT
Fech_Inicio: DATE
Fech_Fin: DATE
Resultados: VARCHAR(500)
Acciones: VARCHAR(500)
Status: SMALLINT NOT NULL
Adscripcion
Cve_adscripcion: CHAR(5) NOT NULL
Cve_Profesor: CHAR(5) NOT NULL (FK)
Cve_Institucion: CHAR(6) NOT NULL (FK)
Nombramiento: VARCHAR(50)
Vigencia: VARCHAR(20)
Ingreso: DATE
Baja: DATE
Tiempo: VARCHAR(10)
Asignatura: CHARACTER(10)
Apoyo: SMALLINT
Convenio: VARCHAR(20)
Ini_Convo: DATE
Fin_Convo: DATE
Sabatico: SMALLINT
Status: SMALLINT NOT NULL
Profesores
Cve_Profesor: CHAR(5) NOT NULL
Nombre: VARCHAR(30) NOT NULL
Paterno: VARCHAR(30) NOT NULL
Materno: VARCHAR(30)
RFC: CHAR(10)
F_Nacimiento: DATE
Sexo: SMALLINT
Nacionalidad: SMALLINT
Nivel_SNI: VARCHAR(20)
No_SNI: VARCHAR(20)
Inicio_SNI: DATE
Fin_SNI: DATE
Correo: VARCHAR(50)
Status: SMALLINT NOT NULL
Cat_Area
Cve_Area: CHAR(2) NOT NULL
Descrip_Area: VARCHAR(50) NOT NULL
Investigacion
Cve_disciplina: CHAR(2) NOT NULL (FK)
Cve_Acervo: CHAR(5) NOT NULL (FK)
Cve_Programa: CHAR(5) NOT NULL (FK)
Cve_Invest: CHAR(5) NOT NULL
Cve_Area: CHAR(2) NOT NULL (FK)
Responsable: VARCHAR(50)
Area_Principal: VARCHAR(200)
Area_Sec: VARCHAR(200)
Linea_Investiga: VARCHAR(200)
Disciplina_Princ: VARCHAR(200)
Especialidad_Princ: VARCHAR(200)
Status: SMALLINT NOT NULL
DatosGenerales
Cve_DatosGrales: CHAR(4) NOT NULL
Cve_Programa: CHAR(5) NOT NULL (FK)
Cve_Muni: CHAR(3) (FK)
Cve_Institucion: CHAR(6) (FK)
Cve_Edo: CHAR(2) (FK)
Calle: VARCHAR(50)
Num_ext: VARCHAR(50)
Cp: VARCHAR(5)
Colonia: VARCHAR(60)
Telefono: VARCHAR(50)
Fax: VARCHAR(20)
Tipo_Institucion: SMALLINT
Status: SMALLINT NOT NULL
Cat_Disciplina
Cve_disciplina: CHAR(2) NOT NULL
Descrip_Disip: VARCHAR(30) NOT NULL
Cat_Institucion
Cve_Institucion: CHAR(6) NOT NULL
Descrip_Inst: VARCHAR(30) NOT NULL
Cat_Servicios
Cve_Servicios: CHAR(2) NOT NULL
Descrip_Servicios: VARCHAR(50) NOT NULL
Acervo
Cve_Acervo: CHAR(5) NOT NULL
Cve_Servicios: CHAR(2) NOT NULL (FK)
Cve_Pais: CHAR(3) NOT NULL (FK)
Nombre: VARCHAR(100)
No_Libros: SMALLINT
No_Revistas: SMALLINT
Suscrip_Nal: SMALLINT
Suscrip_Intnal: SMALLINT
Editor: VARCHAR(50)
Fechalnl: DATE
FechaFin: DATE
Status: SMALLINT NOT NULL
Cat_Municipios
Cve_Edo: CHAR(2) NOT NULL (FK)
Cve_Muni: CHAR(3) NOT NULL
Descrip_Muni: VARCHAR(60) NOT NULL
Cat_Estado
Cve_Edo: CHAR(2) NOT NULL
Descrip_Edo: VARCHAR(20) NOT NULL
TesisProf
Cve_TesisProf: CHAR(18) NOT NULL
Cve_Profesor: CHAR(5) (FK)
Cve_Institucion: CHAR(6) NOT NULL (FK)
Licenciatura: VARCHAR(100)
Tipo: SMALLINT
Status: SMALLINT NOT NULL
Usuarios
ID: VARCHAR(8) NOT NULL
Contrasena: VARCHAR(8)
Attibute: SMALLINT
Cat_Pais
Cve_Pais: CHAR(3) NOT NULL
Descrip_Pais: VARCHAR(50) NOT NULL
Grados
Cve_Grados: CHAR(2) NOT NULL
Cve_Profesor: CHAR(5) NOT NULL (FK)
Cve_Institucion: CHAR(6) NOT NULL (FK)
Cve_Pais: CHAR(3) NOT NULL (FK)
Nombre: VARCHAR(100)
Titulo: VARCHAR(50)
F_Gradus: DATE
Status: SMALLINT NOT NULL

3.2 Diccionario de Datos

El diccionario de Datos es un listado organizado de todos los datos pertinentes del sistema, definido de manera precisa, para que tanto el usuario como el analista tengan un entendimiento común de todas las entradas, salidas y almacenes.

La construcción de un diccionario de datos, es una labor muy tediosa y larga, pero a su vez, es también una de las más importantes, ya que sin este no se puede tener una precisión por no saber el significado de los términos empleados.

Una de las ventajas que nos ofrece Erwin es la generación de la documentación. Esta se genera a partir de una serie de comentarios y anotaciones que vamos realizando conforme se va modelando la base de datos.

Se pueden generar varios tipos de reportes, pero los que creemos más importantes son:

1. Reporte por Entidades
2. Reporte por Llaves primarias
3. Reporte por Entidad, descripción

Estos reportes son los mismos que utilizaremos como Diccionario de Datos, ya que cumplen con esa finalidad, que es la generación de un diccionario que nos permita tener toda la información de las entidades, tales como características, utilidad y tipo.

A continuación mostramos la información completa de cada uno de los reportes.

Reporte por Entidades

Entity Name : **Acervo**
Entity Definition: Lleva los registros de las publicaciones disponibles para la investigación.

Entity Name : **Adscripcion**
Entity Definition: Institución de adscripción del profesorado.

Entity Name : **Alumnos**
Entity Definition: Almacena los datos personales de los Alumnos.

Entity Name : **Alumnos_paso**
Entity Definition: Tabla que sirve de paso entre el Programa y Alumnos.

Entity Name : **Cat_Area**
Entity Definition: Catálogo de las áreas de investigación.

Entity Name : **Cat_Disciplina**
Entity Definition: Disciplina utilizada en la investigación.

Entity Name : **Cat_Estado**
Entity Definition: Catálogo de Estados.

Entity Name : **Cat_Inscripcion**
Entity Definition: Tipo de Inscripción.

Entity Name : **Cat_Institucion**
Entity Definition: Institución educativa de procedencia.

Entity Name : **Cat_Municipios**
Entity Definition: Catálogo de Municipios.

Entity Name : **Cat_Pais**
Entity Definition: País de origen.

Entity Name : **Cat_Recurso**
Entity Definition: Tipo de recurso asignado al presupuesto.

Entity Name : **Cat_Servicios**
Entity Definition: Servicios de información bibliográfica disponibles para la Investigación.

Entity Name : **Cat_Solicitud**
Entity Definition: Tipo de Solicitud del Programa

Entity Name : **Cat_TipoProduccion**
Entity Definition: Tipo de Producción

Entity Name : **DatosGenerales**
Entity Definition: Información general del programa.

Entity Name : **Evaluacion**
Entity Definition: Evaluación aplicada al Programa.

Entity Name : **FechaProg**
Entity Definition: Son las fechas correspondientes al programa en el que está inscrito el Alumno.

Entity Name : **Grados**
Entity Definition: Grados académicos de los profesores.

Entity Name : **Investigacion**
Entity Definition: Datos de la investigación del programa.

Entity Name : **PlanEstudio**
Entity Definition: Información sobre el Plan de Estudios.

Entity Name : **Presupuesto**
Entity Definition: Presupuesto asignado al Programa.

Entity Name : **Produccion**
Entity Definition: Publicaciones realizadas por el profesorado.

Entity Name : **Profesores**
Entity Definition: Información de los profesores.

Entity Name : **Profesores_paso**
Entity Definition: Tabla que sirve de paso entre el Programa y. Profesores.

Entity Name : **Programa**
Entity Definition: Información general del Programa.

Entity Name : **Tesis**
Entity Definition: Describe los datos fundamentales de la Tesis del Alumno.

Entity Name : **TesisProf**
Entity Definition: Información sobre las Tesis de los profesores

Reporte por Llaves primarias

Entity Name: **Acervo**
Entity Primary Key Attribute Name

Cve_Acervo

Entity Name: **Adscripcion**
Entity Primary Key Attribute Name

Cve_adscripcion
Cve_Profesor

Entity Name: **Alumnos**
Entity Primary Key Attribute Name

Cve_Alumno

Entity Name: **Alumnos_paso**
Entity Primary Key Attribute Name

Cve_Programa

Entity Name: **Cat_Area**
Entity Primary Key Attribute Name

Cve_Area

Entity Name: **Cat_Disciplina**
Entity Primary Key Attribute Name

Cve_disciplina

Entity Name: **Cat_Estado**
Entity Primary Key Attribute Name

Cve_Edo

Entity Name: **Cat_Inscripcion**
Entity Primary Key Attribute Name

Cve_Inscripcion

Entity Name: **Cat_Institucion**
Entity Primary Key Attribute Name

Cve_Institucion

Entity Name: **Cat_Municipios**
Entity Primary Key Attribute Name

Cve_Edo
Cve_Muni

Entity Name: **Cat_Pais**
Entity Primary Key Attribute Name

Cve_Pais

Entity Name: **Cat_Recurso**
Entity Primary Key Attribute Name

Cve_Recurso

Entity Name: **Cat_Servicios**
Entity Primary Key Attribute Name

Cve_Servicios

Entity Name: **Cat_Solicitud**
Entity Primary Key Attribute Name

Cve_Solicitud

Entity Name: **Cat_TipoProduccion**
Entity Primary Key Attribute Name

Cve_Tipo

Entity Name: **DatosGenerales**
Entity Primary Key Attribute Name

Cve_DatosGrales
Cve_Programa

Entity Name: **Evaluacion**
Entity Primary Key Attribute Name

Cve_Programa
Cve_Eval

Entity Name: **FechaProg**
Entity Primary Key Attribute Name

Cve_Alumno

Entity Name: **Grados**
Entity Primary Key Attribute Name

Cve_Grados
Cve_Profesor

Entity Name: **Investigacion**
Entity Primary Key Attribute Name

Cve_Acervo
Cve_Invest
Cve_Programa

Entity Name: **PlanEstudio**
Entity Primary Key Attribute Name

Cve_Programa
Cve_Plan

Entity Name: **Presupuesto**
Entity Primary Key Attribute Name

Cve_Presupuesto
Cve_Programa

Entity Name: **Produccion**
Entity Primary Key Attribute Name

Cve_Produccion
Cve_Profesor

Entity Name: **Profesores**
Entity Primary Key Attribute Name

Cve_Profesor

Entity Name: **Profesores_paso**
Entity Primary Key Attribute Name

Cve_Programa

Entity Name: **Programa**
Entity Primary Key Attribute Name

Cve_Programa

Entity Name: **Tesis**
Entity Primary Key Attribute Name

Cve_Alumno

Entity Name: **TesisProf**
Entity Primary Key Attribute Name

Cve_TesisProf

Reporte por Entidad, Descripción

Entity Name : **Acervo**
Entity Definition : Lleva los registros de las publicaciones disponibles para la investigación.

Entity Attribute Name	Entity Attribute Definition
Suscrip_Intnal	Suscripciones vigentes en revistas internacionales
Suscrip_Nal	Suscripciones vigentes en revistas nacionales
No_Revistas	Número de revistas especializadas en el área
Editor	Nombre de la casa editora
Fecha_Inicio	Fecha de inicio de la suscripción
Fecha_Fin	Fecha de finalización de la suscripción
Cve_Servicios	Clave de los Servicios de Información
Cve_Acervo	Clave única del Acervo Bibliográfico
No_Libros	Número de libros especializados en el área
Cve_Pais	Clave del País
Nombre	Nombre del material bibliográfico

Entity Table Column Name	Entity Table Column Datatype
Suscrip_Intnal	SMALLINT
Suscrip_Nal	SMALLINT
No_Revistas	SMALLINT
Editor	VARCHAR(50)
FechaIni	DATE
FechaFin	DATE
Cve_Servicios	CHAR(2)
Cve_Acervo	CHAR(5)
No_Libros	SMALLINT
Cve_Pais	CHAR(3)
Nombre	VARCHAR(100)

Entity Name : **Adscripcion**
Entity Definition : Institucion de adscripción del profesorado

Entity Attribute Name	Entity Attribute Definition
Apoyo	Tipo de apoyo que recibe de Conacyt: Cátedra=0, Repatriación=1 y Ninguno=2
Asignatura	Asignatura a su cargo
Tiempo	Tiempo que permanece en la Institución
Fin_Conve	Fecha de finalización del convenio
Convenio	Número de convenio
Ini_Conve	Fecha de inicio del convenio
Sabatico	Tipo de estancia: Año sabático=0, Estancia postdoctoral=1 y Otro=2
Cve_Profesor	Clave del Profesor
Cve_Institucion	Clave de la Institución educativa
Cve_adscripcion	Clave de la Adscripción
Baja	Fecha de baja de la Institución
Ingreso	Fecha de ingreso a la Institución
Nombramiento	Nombramiento que tiene el profesor en la Institución
Vigencia	Vigencia del nombramiento

Entity Table Column Name	Entity Table Column Datatype
Apoyo	SMALLINT
Asignatura	CHARACTER(10)
Tiempo	VARCHAR(10)
Fin_Conve	DATE
Convenio	VARCHAR(20)
Ini_Conve	DATE
Sabatico	SMALLINT
Cve_Profesor	CHAR(5)
Cve_Institucion	CHAR(6)
Cve_adscripcion	CHAR(5)
Baja	DATE
Ingreso	DATE
Nombramiento	VARCHAR(50)
Vigencia	VARCHAR(20)

Entity Name : **Alumnos**
Entity Definition : Almacena los datos personales de los Alumnos

Entity Attribute Name	Entity Attribute Definition
Cve_Alumno	Clave del alumno
Nombre/2	Nombre
Paterno	Apellido paterno
Materno	Apellido materno
Sexo	Sexo
RFC	R.F.C.
Homo	Homoclave
Edad	Edad
Nacionalidad	Nacionalidad: Mexicana=0 y extranjera=1

Entity Table Column Name	Entity Table Column Datatype
Cve_Alumno	CHAR(5)
Nombre	VARCHAR(50)
Paterno	VARCHAR(50)
Materno	VARCHAR(50)
Sexo	CHAR(1)
RFC	CHAR(10)
Homo	CHAR(3)
Edad	SMALLINT
Nacionalidad	VARCHAR(50)

Entity Name : **Alumnos_paso**
Entity Definition : Tabla que sirve de paso entre el Programa y Alumnos

Entity Attribute Name	Entity Attribute Definition
Cve_Alumno	Clave del Alumno
Cve_APaso	Clave propia de la tabla
Cve_Programa	Clave del Programa

Entity Table Column Name	Entity Table Column Datatype
Cve_Alumno	CHAR(5)
Cve_APaso	CHAR(5)
Cve_Programa	CHAR(5)

Entity Name : **Cat_Area**
Entity Definition : Catálogo de las áreas de investigación

Entity Attribute Name	Entity Attribute Definition
Cve_Area	Clave del Área de Investigación
Descrip_Area	Área de Investigación del Programa

Entity Table Column Name	Entity Table Column Datatype
Cve_Area	CHAR(2)
Descrip_Area	VARCHAR(50)

Entity Name : **Cat_Disciplina**
Entity Definition : Disciplina utilizada en la investigación

Entity Attribute Name	Entity Attribute Definition
Cve_disciplina	Clave de la Disciplina de Investigación
Descrip_Dicip	Disciplina de Investigación del Programa

Entity Table Column Name	Entity Table Column Datatype
Cve_disciplina	CHAR(2)
Descrip_Dicip	VARCHAR(50)

Entity Name : **Cat_Estado**
Entity Definition : Catálogo de Estados

Entity Attribute Name	Entity Attribute Definition
Cve_Edo	Clave del Estado
Descrip_Edo	Descripción del Estado

Entity Table Column Name	Entity Table Column Datatype
Cve_Edo	CHAR(2)
Descrip_Edo	VARCHAR(20)

Entity Name : **Cat_Inscripcion**
Entity Definition : Tipo de Inscripción

Entity Attribute Name	Entity Attribute Definition
Cve_Inscripcion	Clave única para el Tipo de Inscripción
Descrip_Inscrip	Describe el tipo de Inscripción

Entity Table Column Name	Entity Table Column Datatype
Cve_Inscripcion	CHAR(2)
Descrip_Inscrip	VARCHAR(15)

Entity Name : **Cat_Institucion**
Entity Definition : Instutución educativa de procedencia

Entity Attribute Name	Entity Attribute Definition
Cve_Institucion	Clave de la Institución
Descrip_Inst	Nombre de la Institución educativa

Entity Table Column Name	Entity Table Column Datatype
Cve_Institucion	CHAR(6)
Descrip_Inst	VARCHAR(50)

Entity Name : **Cat_Municipios**
Entity Definition : Catálogo de Municipios

Entity Attribute Name	Entity Attribute Definition
Cve_Edo	Clave del Estado
Cve_Muni	Clave del Municipio
Descrip_Muni	Nombre del Municipio

Entity Table Column Name	Entity Table Column Datatype
Cve_Edo	CHAR(2)
Cve_Muni	CHAR(3)
Descrip_Muni	VARCHAR(60)

Entity Name : **Cat_Pais**
Entity Definition : Pais de origen

Entity Attribute Name	Entity Attribute Definition
Cve_Pais	Clave del Pais
Descrip_Pais	Nombre del Pais

Entity Table Column Name	Entity Table Column Datatype
Cve_Pais	CHAR(3)
Descrip_Pais	VARCHAR(50)

Entity Name : **Cat_Recurso**
Entity Definition : Tipo de recurso asignado al presupuesto

Entity Attribute Name	Entity Attribute Definition
Cve_Recurso	Clave del Tipo de Recurso
Descrip_Recurso	Tipo de Recurso asignado

Entity Table Column Name	Entity Table Column Datatype
Cve_Recurso	CHAR(2)
Descrip_Recurso	VARCHAR(15)

Entity Name : **Cat_Servicios**
Entity Definition : Servicios de información bibliográfica disponibles para la Investigación

Entity Attribute Name	Entity Attribute Definition
Cve_Servicios	Clave de los Servicios de Investigación
Descrip_Servicios	Servicios de información disponibles

Entity Table Column Name	Entity Table Column Datatype
Cve_Servicios	CHAR(2)
Descrip_Servicios	VARCHAR(50)

Entity Name : **Cat_Solicitud**
Entity Definition : Tipo de Solicitud del Programa

Entity Attribute Name	Entity Attribute Definition
Cve_Solicitud	Clave de la Solicitud
Descrip_Solicitud	Tipo de solicitud del Programa

Entity Table Column Name	Entity Table Column Datatype
Cve_Solicitud	CHAR(2)
Descrip_Solicitud	VARCHAR(11)

Entity Name : **DatosGenerales**
Entity Definition : Información general del programa

Entity Attribute Name	Entity Attribute Definition
Colonia	Colonia donde se localiza la Institución
Cp	Código Postal de la Colonia
Num_ext	Número Exterior de la Institución
Telefono	Número telefónico de la Institución
Fax	Número de fax de la Institución
Tipo_Institucion	Tipo de Entidad: Institución=0, Dependencia=1
Cve_Programa	Clave del Programa
Cve_DatosGrales	Clave de los Datos Generales del Programa
Calle	Nombre de la Calle donde se encuentra la Institución
Cve_Edo	Clave del Estado
Cve_Muni	Clave del Municipio
Cve_Institucion	Clave de la Institución

Entity Table Column Name	Entity Table Column Datatype
Colonia	VARCHAR(60)
Cp	VARCHAR(5)
Num_ext	VARCHAR(50)
Telefono	VARCHAR(50)
Fax	VARCHAR(30)
Tipo_Institucion	SMALLINT
Cve_Programa	CHAR(5)
Cve_DatosGrales	CHAR(4)
Calle	VARCHAR(50)
Cve_Edo	CHAR(2)
Cve_Muni	CHAR(3)
Cve_Institucion	CHAR(6)

Entity Name : **Evaluacion**
Entity Definition : Evaluación aplicada al Programa

Entity Attribute Name	Entity Attribute Definition
Fech_Fin	Fecha en la que finaliza la evaluación
Fech_Inicio	Fecha en que inició la evaluación
Resultados	Resultados obtenidos en la evaluación
Acciones	Acciones realizadas posteriormente a la evaluación
Cve_Programa	Clave del Programa
Tipo	Tipo de evaluación: Interna=0 y externa=1
Cve_Eval	Clave de la Evaluación

Entity Table Column Name	Entity Table Column Datatype
Fech_Fin	DATE
Fech_Inicio	DATE
Resultados	VARCHAR(500)
Acciones	VARCHAR(500)
Cve_Programa	CHAR(5)
Tipo	SMALLINT
Cve_Eval	CHAR(5)

Entity Name : **FechaProg**
Entity Definition : Son las fechas correspondientes al programa en el que está inscrito el Alumno

Entity Attribute Name	Entity Attribute Definition
Cve_Alumno	Clave del Alumno
FIngProg	Fecha de ingreso al programa
FReins	Fecha de reinscripción
FBajaProg	Fecha de baja del programa
MotivoBaja	Motivo de la baja
FEgreso	Fecha de egreso
FExamenGdo	Fecha del examen para la obtención del grado

Entity Table Column Name	Entity Table Column Datatype
Cve_Alumno	CHAR(5)
FIngProg	DATE
FReins	DATE
FBajaProg	DATE
MotivoBaja	VARCHAR(200)
FEgreso	DATE
FExamenGdo	DATE

Entity Name : **Grados**
Entity Definition : Grados académicos de los profesores

Entity Attribute Name	Entity Attribute Definition
Cve_Grados	Clave del Grado
Cve_Profesor	Clave del Profesor
Cve_Institucion	Clave de la institución
Cve_Pais	Clave del País
Nombre	Nombre del estudio realizado
Titulo	Titulo del grado obtenido
F_Gradua	Fecha de graduación

Entity Table Column Name	Entity Table Column Datatype
Cve_Grados	CHAR(2)
Cve_Profesor	CHAR(5)
Cve_Institucion	CHAR(6)
Cve_Pais	CHAR(3)
Nombre	VARCHAR(100)
Titulo	VARCHAR(50)
F_Gradua	DATE

Entity Name : **Investigacion**
Entity Definition : Datos de la investigación del programa

Entity Attribute Name	Entity Attribute Definition
Area_Sec	Área secundaria de la investigación
Area_Principal	Área principal de la investigación
Responsable	Responsable de la investigación
Linea_Investiga	Líneas de investigación asociadas al Programa
Disciplina_Princi	Disciplina principal de investigación
Especialidad_Princi	Especialidad principal de la investigación
Cve_Invest	Clave de la Investigación del Programa
Cve_Acervo	Clave del Acervo
Cve_Area	Clave del área de investigación
Cve_Programa	Clave del Programa
Cve_disciplina	Clave de la disciplina de investigación

Entity Table Column Name	Entity Table Column Datatype
Area_Sec	VARCHAR(200)
Area_Principal	VARCHAR(200)
Responsable	VARCHAR(50)
Linea_Investiga	VARCHAR(200)
Disciplina_Princi	VARCHAR(200)
Especialidad_Princi	VARCHAR(200)
Cve_Invest	CHAR(5)
Cve_Acervo	CHAR(5)
Cve_Area	CHAR(2)
Cve_Programa	CHAR(5)
Cve_disciplina	CHAR(2)

Entity Name : **PlanEstudio**
Entity Definition : Información sobre el Plan de Estudios

Entity Attribute Name	Entity Attribute Definition
Observaciones	Observaciones
Per_Ingreso	Periodos de ingreso
Perfil	Perfil del egresado
Idioma	Requisitos de idioma: Si=0, No=1
Finicio	Fecha de Inicio del Plan de estudio
Fundamentos	Fundamentación académica
Ingreso	Requisitos de ingreso al programa
Actividades	Actividades académicas del Plan de Estudios
Ffin	Fecha de Inicio del Plan de estudio
Cve_Plan	Clave del Plan de Estudios
Cve_Inscripcion	Clave de Inscripción
Cve_Programa	Clave del Programa
Actualizaciones	Número de actualizaciones al Plan de Estudios
Grado_Max	Tiempo máximo para la obtención del grado (en meses)
Duracion	Duración del Programa en meses
Grado_Min	Tiempo mínimo para la obtención del grado (en meses)
Estud_Max	Número máximo de estudiantes por año

Entity Table Column Name	Entity Table Column Datatype
Observaciones	VARCHAR(1000)
Per_Ingreso	VARCHAR(50)
Perfil	VARCHAR(500)
Idioma	SMALLINT
Finicio	DATE
Fundamentos	VARCHAR(1000)
Ingreso	VARCHAR(500)
Actividades	VARCHAR(1000)
Ffin	DATE
Cve_Plan	CHAR(5)
Cve_Inscripcion	CHAR(2)
Cve_Programa	CHAR(5)
Actualizaciones	SMALLINT
Grado_Max	SMALLINT
Duracion	SMALLINT
Grado_Min	SMALLINT
Estud_Max	SMALLINT

Entity Name : **Presupuesto**
Entity Definition : Presupuesto asignado al Programa

Entity Attribute Name	Entity Attribute Definition
Objetivos	Objetivos para el uso del presupuesto
Medidas	Medidas específicas de apoyo al Programa
Monto	Monto del presupuesto
Beca	Tipo de recurso obtenido por medio de una beca
Otorgante	Información del otorgante del recurso
Cve_Presupuesto	Clave del Presupuesto
Fuentes	Fuentes del presupuesto
Cve_Programa	Clave del Programa
Cve_Recurso	Clave del recurso

Entity Table Column Name	Entity Table Column Datatype
Objetivos	VARCHAR(200)
Medidas	VARCHAR(200)
Monto	FLOAT
Beca	SMALLINT
Otorgante	VARCHAR(200)
Cve_Presupuesto	CHAR(5)
Fuentes	VARCHAR(200)
Cve_Programa	CHAR(5)
Cve_Recurso	CHAR(2)

Entity Name : **Produccion**
Entity Definition : Publicaciones realizadas por el profesorado

Entity Attribute Name	Entity Attribute Definition
Articulos	Titulo de la producción bibliográfica
Cve_Tipo	Clave del tipo
Cve_Profesor	Clave del Profesor
Cve_Produccion	Clave de la producción

Entity Table Column Name	Entity Table Column Datatype
Titulo	VARCHAR(100)
Cve_Tipo	CHAR(2)
Cve_Profesor	CHAR(5)
Cve_Produccion	CHAR(3)

Entity Name : **Profesores**
Entity Definition : Información de los profesores

Entity Attribute Name	Entity Attribute Definition
Cve_Profesor	Clave del Profesor
Nombre	Nombre
Paterno	Apellido paterno
Materno	Apellido Materno
RFC	R.F.C.
F_Nacimiento	Fecha de nacimiento
Sexo	Sexo
Nacionalidad	Nacionalidad
Nivel_SNI	Nivel del Sistema Nacional de Investigadores
No_SNI	Número del Sistema Nacional de Investigadores
Inicio_SNI	Fecha de ingreso al Sistema Nacional de Investigadores
Fin_SNI	Fecha de finalización en el Sistema Nacional de Investigadores
Correo	Correo electrónico

Entity Table Column Name	Entity Table Column Datatype
Cve_Profesor	CHAR(5)
Nombre	VARCHAR(50)
Paterno	VARCHAR(50)
Materno	VARCHAR(50)
RFC	CHAR(10)
F_Nacimiento	DATE
Sexo	SMALLINT
Nacionalidad	SMALLINT
Nivel_SNI	VARCHAR(20)
No_SNI	VARCHAR(20)
Inicio_SNI	DATE
Fin_SNI	DATE
Correo	VARCHAR(50)

Entity Name : **Profesores_paso**
Entity Definition : Tabla que sirve de paso entre el Programa y Profesores

Entity Attribute Name	Entity Attribute Definition
Cve_Profesor	Clave del profesor
Cve_PPaso	Clave propia de la tabla
Cve_Programa	Clave del Programa

Entity Table Column Name	Entity Table Column Datatype
Cve_Profesor	CHAR(5)
Cve_PPaso	CHAR(5)
Cve_Programa	CHAR(5)

Entity Name : **Programa**
Entity Definition : Información general del Programa

Entity Attribute Name	Entity Attribute Definition
Metas	Metas existentes para el Programa a Corto, mediano y largo plazo.
Antecedentes	Antecedentes y desarrollo del Programa.
Razon	Razones por las que se crea el Programa.
Tipo_Programa	Tipo de Programa.
Importancia	Importancia del Programa en la estrategia del desarrollo institucional.
Apoyo	Apoyo al Programa de otras áreas de la Institución.
Nivel	Nivel del Programa
Cve_Solicitud	Clave de la Solicitud
Fecha_Aprobacion	Fecha de Aprobación del Programa
Cve_Programa	Clave del Programa
Especialidad	Especialidad u opciones terminales del Programa
Nombre_Programa	Nombre del Programa
Fecha_Inicio	Fecha de Inicio del Programa
Sede	Lugar donde se imparte el Programa : Dependencia=0, Institución=1 y Otro lugar=2

Entity Table Column Name	Entity Table Column Datatype
Metas	VARCHAR(1000)
Antecedentes	VARCHAR(1000)
Razon	VARCHAR(1000)
Tipo_Programa	CHAR(18)
Importancia	VARCHAR(1000)
Apoyo	VARCHAR(1000)
Nivel	CHAR(18)
Cve_Solicitud	CHAR(2)
Fecha_Aprobacion	DATE
Cve_Programa	CHAR(5)
Especialidad	VARCHAR(1000)
Nombre_Programa	VARCHAR(100)
Fecha_Inicio	DATE
Sede	SMALLINT

Entity Name : **Tesis**
Entity Definition : Describe los datos fundamentales de la Tesis del Alumno

Entity Attribute Name	Entity Attribute Definition
Cve_Alumno	Clave del Alumno
Titulo	Título de la tesis
FTermino	Fecha estimada de terminación de la tesis
Avance	Porcentaje aproximado del avance
Tutor	Nombre del tutor designado
Asesor	Nombre del asesor designado (en caso de no ser el tutor)
Comite	Comité tutoral designado

Entity Table Column Name	Entity Table Column Datatype
Cve_Alumno	CHAR(5)
Titulo	VARCHAR(100)
FTermino	DATE
Avance	VARCHAR(10)
Tutor	VARCHAR(50)
Asesor	VARCHAR(50)
Comite	VARCHAR(100)

Entity Name : **TesisProf**
Entity Definition : Información sobre las Tesis de los profesores

Entity Attribute Name	Entity Attribute Definition
Cve_Institucion	Clave de la institución
Licenciatura	Título de la tesis obtenida
Maestria	Tipo de tesis obtenida: 0=Licenciatura, 1=Maestría, 2=Doctorado y 3=Otro
Cve_Profesor	Clave del Profesor
Cve_TesisProf	Clave de la Tesis

Entity Table Column Name	Entity Table Column Datatype
Cve_Institucion	CHAR(6)
Licenciatura	CHAR(18)
Tipo	CHAR(18)
Cve_Profesor	CHAR(5)
Cve_TesisProf	CHAR(18)

Hay que tener en cuenta que los diccionarios de datos son importantes por los siguientes puntos.

1. Nos permiten documentar las características del sistema.
2. Facilita el análisis de los detalles para evaluar las características y determinar donde deben realizarse los cambios.
3. Permite localizar errores y omisiones en el sistema, previamente a la fase de pruebas.

Una vez hecho lo anterior, procedemos a generar la base de datos. Erwin también nos permite por medio de una serie de pasos, generar toda la estructura de la BD haciendo uso de los comandos SQL.

Ya que hemos generado el modelo de la base de datos, entonces procedemos a crear el "cascarón" que albergará y protegerá a la propia base de datos. Para esto va a ser necesario auxiliarnos de una herramienta que viene incluida en el manejador de base de datos de Interbase, llama "Interbase Windows ISQL".

3.3 Programación del Sistema

Una vez que hemos generado los diccionarios de datos, ya nos encontramos en condiciones de empezar a programar el sistema para la explotación de la información.

Coma ya habíamos mencionado, la programación la vamos a realizar en Delphi versión 3.0. Lo primero que debemos hacer es programar el servidor de nuestra aplicación y una vez terminado, programar el cliente.

Debido a que no pretendemos escribir un texto sobre algoritmos, técnicas y lenguajes de programación, no entraremos a detalle en los puntos de programación.

Solo haremos mención a que anexamos en un disco flexible, los programas fuentes del Cliente, como del Servidor.

Cuando terminamos la etapa de la programación, generamos mediante un Instalador, los discos de instalación, primero del Servidor y posteriormente del cliente. Esto para facilitar la instalación en cada uno de los equipos donde se realizan las pruebas y finalmente donde el sistema vaya a trabajar.

Pruebas

Una vez concluido el desarrollo del sistema es necesario proceder a la instalación, pruebas y evaluación, con el fin de implementarlo. Esto no significa que el sistema ya se encuentre terminado en su totalidad, sino que se puede operar, evaluar y en cualquier instante se puede dar mantenimiento a todo el Sistema.

Esta etapa es un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y la codificación.

Conforme se va programando cada uno de los módulos que componen el Sistema, también se van realizando pruebas a la par. Cada módulo se termina cuando:

1. Este cumple con los objetivos para las que fue diseñado
2. No presenta errores ni warnings.

Por lo tanto a estas alturas ya se han realizado toda una serie de pruebas unitarias.

Normalmente se desarrolla un Plan de Prueba, que determina el tiempo y el esfuerzo necesarios para asegurar la calidad del sistema entregado. En muchos sistemas las pruebas adquieren una nueva importancia, porque el entorno del sistema varía de una instalación a otra, por lo cual deben de probarse la mayor cantidad de facetas posibles.

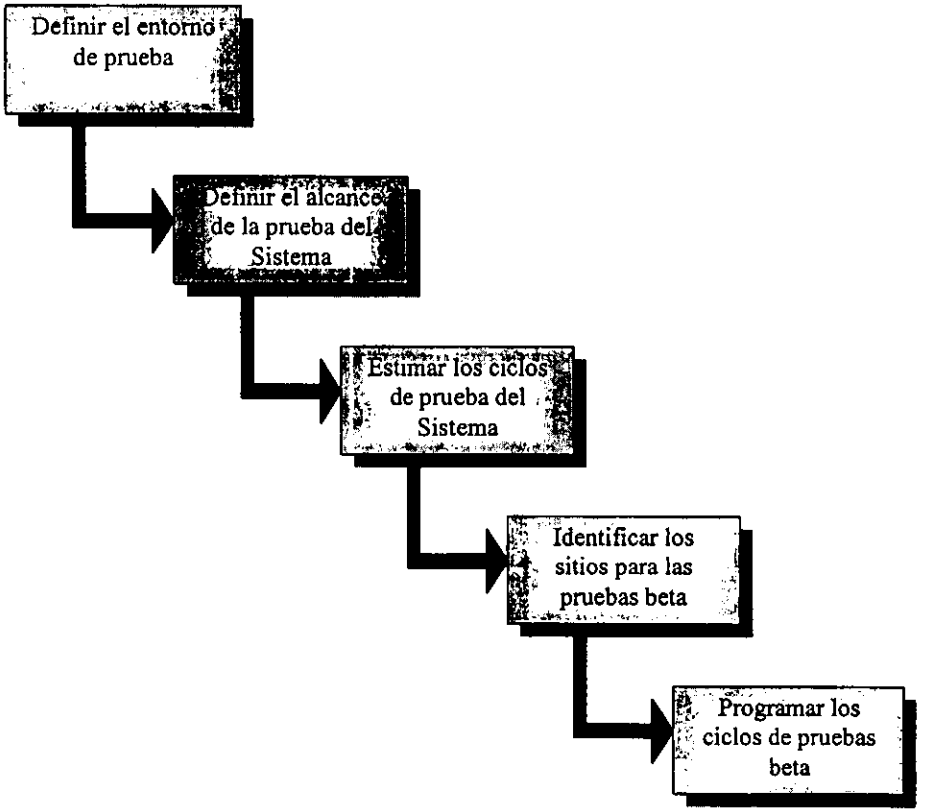
Además de los ciclos usuales de pruebas unitarias, de integración, de sistema y de aceptación, los sistemas cliente/servidor también planearán pruebas beta.

Esta tarea determina cómo dependen las pruebas una de la otra y estiman su duración. Entonces podrán establecerse los costos, los requerimientos de recursos y el impacto sobre los tiempos estimados.

Los planes de las pruebas beta, deberán ser utilizados para probar la operación del programa sobre distintas configuraciones y no como un sustituto de pruebas unitarias, de integración o de aceptación. Los proyectos pueden resultar normal que se empantanen fácilmente en las pruebas beta.

El propósito de las pruebas beta es verificar el sistema en un amplio espectro de entornos. Hay que determinar el rango de configuraciones de clientes, incluyendo opciones de hardware, de software y de comunicaciones. Además de las características genéricas como memoria y espacio en disco.

Presentamos a continuación un método para desarrollar un Plan de prueba:



4.1 Entorno de Prueba

En este punto se deben identificar las características del entorno necesario para una prueba adecuada de sistemas y de aceptación. Idealmente los entornos de prueba de sistemas y de aceptación deberán duplicar al entorno de producción. En el caso de los sistemas cliente/servidor, esto por lo regular no es posible dado que el

entorno tiene tantas variables potenciales posibles. Es necesario especificar los componentes técnicos de los servidores, redes y anfitriones, así como los clientes típicos para las pruebas de sistemas. Se debe tratar de limitar las computadoras de los clientes a las configuraciones más comunes.

Para este punto, hemos hecho uso de los recursos disponibles en la Secretaría de Contraloría y Desarrollo Administrativo (SECODAM), debido a la facilidad de acceso que tenemos al equipo por laborar en dicha institución y a la envergadura del mismo.

El Servidor se ha instalado en una computadora con las siguientes características:

- Acer con microprocesador Pentium III a 300 MHz, MMX Technology.
- 64 MB de memoria RAM.
- 8.5 GB de capacidad en disco duro.
- Tarjeta de red Fast Ethernet, Intel 82558.
- Unidad de CD-ROM E-IDE 40X/AKU.

Y los Clientes en equipos con las siguientes características:

- Olivetti con microprocesador Pentium II a 200 MHz.
- 32 MB de memoria RAM.
- 1.5 GB de capacidad en disco duro.
- Tarjeta de red 3COM Fast EtherLink XL

La red con que se cuenta es Ethernet con topología Bus - Estrella :

- Velocidad en el Back-Bone de 100 Mbits.
- El resto de trabajo es de 10 Mbits.
- Aproximadamente 1200 usuarios conectados y
- 40 servidores conectados.

4.2 Alcance de las pruebas del sistema

Es necesario realizar cada una de las pruebas en las áreas siguientes:

Pruebas de entorno

Se debe asegurar que las piezas necesarias se encuentren en su lugar y que estas funcionen correctamente para la prueba.

Esto lo hicimos por medio de una sencilla prueba, donde se enciende cada una de las computadoras que se utilizan, entrando como usuarios permitidos para el acceso a la red. Entramos a una aplicación para ver los grupos de trabajo, donde se encuentran dichos equipos y empezamos a enviar paquetes de información, los cuales llegaron sin problema alguno de manera íntegra, por lo cual esta prueba ha sido totalmente adecuada.

Pruebas de falla

Deben reconocerse las fallas, la recuperación de pruebas y los procedimientos de recuperación.

En este paso no tenemos problema con la red, debido a que los equipos cuentan con las herramientas apropiadas que se la pasan "escuchando" a la red, y

en caso de existir un fallo, inmediatamente se activa una aplicación de recuperación del entorno de trabajo adecuado.

La base de datos se encuentra protegida en dos aspectos. Primero, los clientes en cada operación cuentan con un código para la protección en caso de errores o problemas con la conexión con la base de datos.

Segundo, el servidor que es el que tiene la base de datos, a su vez cuenta también con un código de protección en cada acceso y además la protección propia del manejador de la base de datos, por lo cual la información se encuentra bien protegida.

Pruebas de volumen

Se verifica el rendimiento bajo cargas pico. Esto se realiza con todos los clientes conectados al Servidor y trabajando al mismo tiempo, realizando operaciones a la base de datos por medio del sistema.

Conectamos hasta 12 clientes al mismo tiempo, trabajando en las operaciones que el sistema contempla, como son de consulta, búsqueda, guardado y modificación de datos.

No se tuvo ningún tipo de error, lo que sí se pudo observar fue un retardo (no gravoso) en la respuesta del servidor hacia los clientes.

Aseguramiento

Para verificar que el sistema puede manejar correctamente cualquier entrada y que no se puede violar ni la privacidad ni la seguridad

Esta seguridad se encuentra tanto en el servidor como en el cliente. En el servidor, solo una persona –que es el Administrador- tiene acceso a la información, él cuenta con una clave de usuario única para realizar todo tipo de manejos.

En la parte del cliente se tiene una clave de acceso universal, el cual permite realizar las operaciones normales, excepto la de borrado, que es de uso exclusivo del Administrador.

4.3 Ciclos de prueba del sistema

Se agrupan los requerimientos de pruebas en sesiones manejables, o sea en ciclos de prueba. Se debe tratar de utilizar datos creados por un ciclo de prueba como datos de entrada para otro.

Hay que definir las condiciones que se han de revisar dentro de cada ciclo de prueba, así como el resultado. Se deben seguir los caminos normales que traza el sistema.

Para esto hemos realizado las pruebas de tal forma, que nos suponemos en el papel de un usuario consciente y experto, donde sabemos los datos que estamos capturando y conocemos la información final que se debe obtener.

4.4 Sitios de prueba beta

Estos necesitarán personal y equipo. El probador beta ideal es un usuario de computadora capacitado, que sabe cómo buscar e identificar problemas. Sin embargo este probador también deberá tener la perspectiva de un novato de computación, que se confunde por cosas que usuarios más avanzados toman como natural, incluyendo al programador mismo.

Hemos solicitado, para hacer esto, la ayuda de compañeros de trabajo, ya familiarizados con los sistemas y equipo de cómputo para realizar las pruebas beta. Se nos hicieron algunas observaciones las cuales consideramos e hicimos los ajustes que se nos requirieron. También pedimos la ayuda a compañeros no acostumbrados ni conocedores de sistemas, a los cuales se les indicaron los pasos a seguir.

En ambos casos, salvo las observaciones comentadas, las pruebas fueron totalmente exitosas y sin fallas alguna.

4.5 Ciclos de prueba beta

Contra lo que se cree, las pruebas beta no tienen como objetivo definir las especificaciones del sistema o hacer una depuración al código. Su propósito es el de descubrir problemas de configuración que no hayan sido verificados en pruebas unitarias, de integración, o de sistemas. Cuando un sistema llega a la prueba beta, la mayor parte de los errores ya han sido detectados.

Nuestras prueba beta, consistieron en la generación de los discos de instalación, tanto en la parte del cliente como en la parte del servidor. Se le pidió a cada una de las personas que nos ayudó, que lo instalaran y después nos comentaran cada uno de los puntos observados como raros o erróneos, los cuales afortunadamente no se presentaron.

Conclusiones

Para el caso del presente trabajo, fueron diversos los obstáculos que atravesamos, uno de ellos, fue el de no poder contar con el equipo ni las instalaciones que se tienen en la División de Estudios de Posgrado de la Facultad de Filosofía y Letras, sobre todo en la red, ya que es parte necesaria para el funcionamiento del sistema, debido al paro que se presentó en la Universidad.

Se tuvo que hacer frente a esta situación, utilizando otra red con características ya antes mencionadas.

Cuando se desarrolla un sistema, se requieren de ciertos requisitos para poder cumplir con la finalidad con que se diseña, estos van desde su problemática y necesidad de solución, hasta la puesta en marcha del mismo.

La problemática principal que se presentó, sucedió durante la etapa del análisis, ya que el contacto para el intercambio de información e ideas con gente de Filosofía y Letras, se vio muy acotado. Esto se debe a dos razones principales, la primera, tuvo que ver con que no se proporcionó la información suficiente, y la poca que se tuvo, no fue de la calidad suficiente y la segunda por el motivo del Paro Universitario.

La metodología utilizada considera las metodologías de análisis de requisitos y el análisis estructurado, así como la experiencia de los diseñadores. El análisis de requisitos se utilizó en la investigación preliminar, lo cual ayudó a entender de manera más clara el planteamiento del problema. El análisis estructurado permite, mediante el empleo de diagramas, evitar caer en redundancias y tolera la realización de un desarrollo más legible y pulcro que al momento de dar mantenimiento, resulte sencillo y fácil de entender.

El uso de estas metodologías permitió cumplir con los objetivos planteados en un inicio, donde se pretendió desarrollar un sistema no redundante y que obedeciera con los requisitos necesarios para lograr un sistema útil y funcional.

Para facilitar la compresión del Sistema de Información para Programas de Posgrado de la Facultad de Filosofía y Letras (SIPPFFYL) , se cuenta con los diagramas de flujo de datos, el diccionario de datos y el diagrama entidad–relación. La utilización de una base de datos relacional evita el uso indiscriminado de recursos.

La programación, es un punto clave, donde se construye la interfaz con el usuario, que tiene como objetivo ingresar, modificar y explotar la información. El lenguaje utilizado, es de quinta generación y está orientado a objetos, el cual permite realizar desarrollos más puros y eficientes.

Es importante mencionar que en el presente trabajo no se satura al lector con manuales de usuario, técnicos, ni de operación y mantenimiento, ya que parte de los objetivos es mostrar la forma de realizar desde el análisis hasta el desarrollo de un caso real, el cual no va enfocado a los usuarios finales, razón que explica el por que los manuales no se presentan.

SIPPFFYL no concluye aquí, este puede ser reutilizado en otros organismos con la misma problemática, adecuándolo a las situaciones propias, lo cual es permisible ampliamente, debido a que el sistema es susceptible a mejoras y a un crecimiento en el futuro.

Finalmente, y a pesar de los obstáculos que se generaron a raíz del paro estudiantil, el desarrollo del presente trabajo concluyó, cumpliendo en su totalidad los objetivos propuestos.

Glosario

En este Glosario se enlistan los términos utilizados en el presente trabajo.

ActiveX	Lenguaje desarrollado por Microsoft para la elaboración de aplicaciones exportables a la red y capaces de operar sobre cualquier plataforma a través, de navegadores. Permite dar dinamismo a las páginas web.
Administrador del Sistema	En una estación de trabajo multiusuario, se denomina así al usuario especial encargado del mantenimiento y la gestión global del sistema.
Agent	Es en el modelo cliente-servidor, la parte del sistema que realiza la preparación e intercambio de información por cuenta de una aplicación del cliente o del servidor.
Alfanumérico	Campo de datos o conjunto de caracteres que incluye letras, números y otros caracteres especiales (símbolos de puntuación, espacios, símbolos matemáticos...).
Algoritmo	conjunto de instrucciones que configuran el procedimiento paso a paso para resolver un determinado problema en una cantidad finita de tiempo. A partir de los algoritmos se forman los programas.
Alias	Nombre usualmente corto y fácil de recordar que se utiliza en lugar de otro nombre usualmente largo y difícil de recordar.

Término	
ANSI	(American National Standard Institute). Instituto de Estándares Nacionales Americanos. ANSI es la principal organización que auspicia el desarrollo de estándares tecnológicos en los Estados Unidos. ANSI trabaja con grupos de la industria y es el miembro de Estados Unidos de la Organización Internacional para la Estandarización (ISO) y la Comisión Electrotécnica Internacional (IEC). Algunos estándares computacionales que tienen tiempo en ANSI son ASCII y SCSI.
API	Interfaz de Programas de Aplicación Una API es un método específico prescrito por un sistema operativo de computadora o por otro programa de aplicación por medio del cual un programador que escribe un programa de aplicación puede hacer solicitudes del sistema operativo o de otra aplicación.
Aplicación	Programa de software utilizado para efectuar una tarea particular, como un tratamiento de textos o un paquete gráfico. Este termino se utiliza indistintamente junto con el de "programa".
Aplicación cliente	Es el programa cuyos documentos pueden aceptar objetos vinculados o incrustados
Archivo	Es el conjunto de datos de un documento o aplicación a los cuales se ha asignado un nombre para su almacenamiento en el disco.

Término	Significado
ASCII	(American Standard Code for Information Interchange). Código Estándar Americano para el Intercambio de Información. ASCII es el formato más común para archivos de texto en computadoras y sobre Internet. En un archivo ASCII, cada caracter alfabético, numérico o especial, es representado con un número binario de 7 bits. 128 posibles caracteres son definidos. Sistemas operativos basados en DOS y UNIX, exceptuando a Windows NT, usan ASCII para sus archivos de texto. Windows NT usa un nuevo código llamado Unicode. ASCII fue desarrollado por ANSI.
Authentication	Verificación de la identidad de una persona o de un proceso para acceder a un recurso o poder realizar determinada actividad. También se aplica a la verificación de identidad de origen de un mensaje.
Base de datos	Conjunto de información almacenada en cualquier formato. Generalmente el término se aplica a textos o información gráfica almacenada en una computadora y accesible de forma sistemática. La información suele estar dividida en registros, y éstos, a su vez, en campos.
Biblioteca	Es un conjunto de subrutinas y programas que pueden utilizarse para realizar funciones que no están implementadas en el compilador.
BLOB	Objeto Binario de Gran Tamaño. Es un campo que permite almacenar grandes cadenas de bits, imágenes, música o video.

Término	
Browser	Navegador. Aplicación para visualizar documentos WWW y navegar por el espacio Internet. En su forma más básica son aplicaciones hipertexto que facilitan la navegación por los servidores de información Internet
Bug	Término aplicado a los errores descubiertos al ejecutar un programa informático. Fue usado por primera vez en el año 1945 por Grace Murray Hooper.
Business ObjectBroker	Componente de MIDAS que proporciona equilibrio de cargas y seguridad frente a los fallos a cualquier servidor, sin codificación o esfuerzo adicional.
Byte	Conjunto significativo de ocho bits que representan un carácter.
CGI	Common Gateway Interface. Programas usados para hacer llamadas a rutinas o controlar otros programas o bases de datos desde una página Web. También pueden generar directamente código HTML.
Cliente	Un sistema o proceso que solicita a otro sistema o proceso que le preste un servicio. Una estación de trabajo que solicita el contenido de un archivo a un servidor, es un cliente de este servidor.
Códigos de control	Códigos que especifican comandos o instrucciones de formato, tales como avances de línea o retornos de carro, en un archivo de texto. Generalmente aparecen precedidos por un símbolo de intercalación.

Compilador	Programa que traduce un listado de instrucciones en un lenguaje a código-máquina. Normalmente los compiladores se diseñan para traducir lenguajes de alto nivel como Delphi.
Computadora	Dispositivo electrónico programable que es capaz de recibir datos, en un formato especificado, someterlos a un proceso y emitir, una información o señales de control, como resultado de ese proceso.
Constante	Posición de memoria, referenciada por un nombre de constante o identificador, donde se almacena un valor que no puede cambiarse.
ConstraintBroker	Componente de MIDAS que permite mantener las reglas de integridad en una posición central y minimiza el tráfico en la red.
Correo Electrónico	Sistema de transmisión de información a través de un canal de comunicaciones.
Delphi	Es un lenguaje de programación visual de alto nivel, que se encuentra orientado a objetos.
Diseño Conceptual	Diagrama en donde aparecen todos los elementos que intervienen en la problemática a resolver.
DNS	Sistema de Nombres de Dominio. Base de datos distribuida que gestiona la conversión de direcciones de Internet expresadas en lenguaje natural a una dirección IP. P.e. 172.31.1.178.

Dominio	Sistema de denominación de Hosts en Internet. Los dominios van separados por un punto y jerárquicamente están organizados de derecha a izquierda. P.e. cosmos.com.mx
Ejecutable	El código ejecutable es una serie de instrucciones que se ejecutan en la computadora. Los programas ejecutables normalmente tienen extensión o atributo específico según el sistema operativo.
Encapsulado	Es el conjunto de datos y métodos en un solo lugar.
Erwin	Herramienta para modelado y diseño de bases de datos Relacionales. La primera versión apareció en 1990.
Ethernet	Sistema de red de área local de alta velocidad. Se ha convertido en un estándar de red corporativa.
Extranet	Interconexión entre dos o más organizaciones a través de sistemas basados en la tecnología Internet.
Firewall	Sistema que se coloca entre una red local e Internet. La regla básica es asegurar que todas las comunicaciones entre dicha red e Internet se realicen conforme a las políticas de seguridad de la organización que lo instala. Además, estos sistemas suelen incorporar elementos de privacidad y autentificación.
FTP	Protocolo de Transferencia de Archivos. Protocolo que permite a un usuario de un sistema acceder a, y transferir desde, otro sistema de una red.

Término	Significado
Fuente	Conjunto de letras, números, signos de puntuación y símbolos a los que se ha dado un determinado tamaño y adoptan un diseño común.
Gateway	Programa o dispositivo de comunicaciones que transfiere datos entre redes que tienen funciones similares pero implantaciones diferentes.
Gestor de Base de Datos	También conocido como Administrador de Base de Datos. Es la parte de la Base de Datos que nos permite realizar operaciones y manejos sobre esta misma.
Host	Computadora que, mediante la utilización de los protocolos TCP/IP, permite a los usuarios comunicarse con otros sistemas anfitriones de una red. Los usuarios se comunican utilizando programas de aplicación, tales como el correo electrónico, Telnet, WWW.
Hostname	Nombre dado a una computadora.
HTML	Lenguaje de Marcado de Hipertexto. Lenguaje en el que se escriben las páginas a las que se accede a través de navegadores WWW. Admite componentes hipertextuales y multimedia.
HTTP	Protocolo de Transferencia de Hipertexto. Protocolo usado para la transferencia de documentos WWW.
InterBase	Es un manejador de bases de datos relacional que provee un rápido procesamiento de transacciones y datos en ambientes multiusuarios. Corre en Windows 95 y 98, Windows NT, Novell Netware y Unix.
Icono	Representación gráfica de un objeto.

Término	
Interface	Conexión entre dos componentes de hardware, entre dos aplicaciones o entre un usuario y una aplicación.
Internet	Red de telecomunicaciones nacida en 1969 en los EE.UU. a la cual están conectadas millones de personas, organismos y empresas en todo el mundo, y cuyo rápido desarrollo está teniendo importantes efectos sociales, económicos y culturales, convirtiéndose de esta manera en uno de los medios más influyentes de la llamada "Sociedad de la Información" y en la "Autopista de la Información" por excelencia.
Internet Address	Dirección Internet. Dirección IP que identifica de forma inequívoca un punto de conexión en Internet.
Internet Protocol	Protocolo Internet. Conjunto de reglas que regulan la transmisión de paquetes de datos a través de Internet. La versión actual es IPv4 mientras que en el proyecto Internet2 se intenta implementar la versión 6 (IPv6), que permitiría mejores prestaciones dentro del concepto QoS (*Quality of Service*).
Intranet	Red propia de una organización, diseñada y desarrollada siguiendo los protocolos propios de Internet, en particular el protocolo TCP/IP.

Término	Significado
IP	Protocolo de Internet, que provee las funciones básicas de direccionamiento en Internet y en cualquier red TCP/IP. Dirección de 32 bits definida por el Protocolo Internet en STD 5, RFC 791. Se representa usualmente mediante notación decimal separada por puntos. Un ejemplo de dirección IP es 172.31.7.12.
ISP	Proveedor de Servicios Internet. Organización, habitualmente con fines de lucro, que además de dar acceso a Internet, les ofrece una serie de servicios (por ejemplo, hospedaje de páginas web, consultoría de diseño e implantación de webs e Intranets, etc., etc.).
Iteración	En los lenguajes de programación, estructura organizativa que representa la repetición de las instrucciones del programa de forma autónoma por contraposición a la recursividad, que supone la dependencia de unos determinados módulos respecto de sí mismos para poderse ejecutar, o a la simple estructura secuencial, que supone la ejecución de las instrucciones de principio a fin del código.
Java	Lenguaje de programación desarrollado por Sun para la elaboración de pequeñas aplicaciones exportables a la red y capaces de operar sobre cualquier plataforma a través, normalmente, de navegadores WWW.

LAN	Red de Área Local. Red de datos para dar servicio a un área geográfica máxima de unos pocos kilómetros cuadrados, por lo cual pueden optimizarse los protocolos de señal de la red para llegar a velocidades de transmisión de hasta 100 Mbps.
Lenguaje de Programación	Conjunto de instrucciones que permiten al programador pensar claramente sobre la complejidad del problema a resolver, de manera que pueda ordenarlas convenientemente para la creación de un programa ejecutable por la computadora. Se dividen en lenguajes de alto y bajo nivel según se acerquen más o menos a las formas de comunicación humana, respectivamente.
Login	Procedimiento de entrada a un sistema multiusuario. También, nombre que identifica al usuario en su máquina y que se introduce en dicho procedimiento.
Loop	En un lenguaje de programación, es un conjunto de instrucciones que se repiten siguiendo una estructura iterativa, mediante el empleo de estructuras condicionales al principio o al final del mismo o con la asignación de un número fijo de veces en las que deberá ejecutarse el loop.
Memoria	Lugar donde la computadora almacena datos y programas.

Término	Significado
Menú	Lista de los comandos disponibles en la ventana de una aplicación. Se designan por un nombre que aparece en la barra situada en la parte superior de la ventana.
Metadatos	Es la información estructural de la base de datos, algunos de estos son: dominios, generadores, tablas, triggers.
Métodos	Son las formas en que se controlan los objetos.
MIDAS	Multi-tier Distributed Application Services Suite. Proporciona a los desarrolladores una serie de componentes avanzados, servidores y tecnologías esenciales para desarrollo de aplicaciones multinivel. Soporta los sistemas operativos más populares y los estándares del mercado.
Modelo Cliente-Servidor	Forma común de describir el paradigma de muchos protocolos de red.
Multihilos	Es la aplicación de base de datos particionada en unidades lógicas, las cuales corren en conjunto o en equipos separados y permiten compartir la información desde una red local hasta Internet.
Multinivel	Es una arquitectura que anula las limitaciones impuestas por las estructuras de uno o dos niveles sin comprometer ninguna parte del proceso de desarrollo.
Multiproceso	Sistema que no divide las tareas concurrentes por programas como lo hace la multitarea, sino que permite que un programa pueda tener varias partes que se ejecuten simultáneamente.

Término	
Multitarea	Capacidad de algunas computadoras y sistemas operativos para ejecutar varias aplicaciones al mismo tiempo, dividiendo las operaciones del procesador en tareas concurrentes.
Multiusuario	Aplicación que puede ser utilizada por varios usuarios al mismo tiempo.
Objeto	Son los datos y los métodos mediante los cuales se controla a los propios datos.
OLE	Object LinKing and Embeded. Objetos Enlazados e Incrustados,
OLEEnterprise	Incrustación y transportación de datos a través de aplicaciones con las características necesarias.
PASCAL	Lenguaje de programación que nace en 1970 como una versión reducida del ALGOL original de principios de los 60. Se normalizó en 1983, año desde el que se expandió de la mano de Borland, llegando a ser el principal lenguaje de enseñanza de la informática hasta el apogeo del C, debido a ser el que mejor refleja la estructura y el pseudocódigo.
Password	Contraseña ó palabra clave que restringe el uso de una computadora o archivo.
Programa	Serie de instrucciones especialmente codificadas que realizan una tarea específica a ejecutar por el ordenador.

Término	Significado
Programa de Procesamiento por Lotes	Archivo de texto que contiene comandos del sistema operativo, de forma que al ser ejecutado se realiza la tarea instruida por cada uno de los comandos del archivo, tal como si se hubiesen escrito directamente a continuación del símbolo de sistema.
Programa Emergente	Residente cargado en la memoria, que no es visible hasta presionar una determinada combinación de teclas o hasta que tenga lugar un determinado hecho, tal como la recepción de un mensaje.
Programa Fuente	Conjunto de instrucciones de alto nivel.
Programa Objeto	Conjunto de instrucciones en lenguaje máquina que la computadora puede interpretar y ejecutar.
Programador	Persona que escribe programas para que la computadora realice una tarea específica.
Protocolo	Descripción formal de formatos de mensaje y de reglas que dos computadoras deben seguir para intercambiar dichos mensajes.
Proxy	Servidor especial encargado, entre otras cosas, de centralizar el tráfico entre Internet y una red privada, de forma que evita que cada una de las máquinas de la red interior tenga que disponer necesariamente de una conexión directa a la red.
Red	Grupo de computadoras interconectadas por cables u otros medios y que utilizan un software específico que les permite compartir dispositivos, e intercambiar información.

Término	
Reglas de Negocio	Son las reglas que permiten que núnca sea posible llevar a cabo acciones no válidas en una aplicación.
Remote DataBroker	Componente de MIDAS que crea aplicaciones de bases de datos distribuidas.
Respaldo	Copia de software hecha con el fin de poder reponerlo en caso de daño físico o lógico.
Servidor	En una red, es una computadora compartida por múltiples usuarios que posee las operaciones centrales y es dueña de la información.
Source	Lenguaje que se emplea para escribir un programa informático.
SQL	Structured Query Language. Es el lenguaje estándar para almacenar y manipular información en bases de datos relacionales.
TCP	Protocolo de Control de Transmisión. Norma orientada a la conexión, que en general se parece al protocolo de transporte del modelo OSI, pero es completamente diferente a este en cuanto a sus formatos y detalles.
TCP/ IP	Transmission Control Protocol/Internet Protocol. Protocolo de control de transmisión/ protocolo Internet. Los protocolos definen las reglas de comunicación. TCP/IP se diseñó específicamente para la interconexión de diferentes tipos de equipos de computadoras.
Trigger	Permite automatizar ciertas tareas como inserción y borrado, directamente en la Base de Datos.

Término	Significado
UNIX	Sistema operativo multitarea, multiusuario. Gran parte de las características de otros sistemas están basadas en este sistema muy extendido para grandes servidores.
Variable	En archivos de proceso por lotes, caracter que se sustituye por un nombre cuando se ejecuta el archivo de proceso por lotes.
Variable de Entorno	Variable que almacena información relativa al entorno de trabajo del usuario y es previamente definida por este, tal como la unidad, la ruta o el nombre de archivo asociados a un nombre simbólico que pueda ser utilizado por el sistema.
Windows	Entorno Gráfico de trabajo desarrollado por la empresa Microsoft para la arquitectura PC desde 1987 como respuesta al entorno nativo de las computadoras de Apple.
Windows - NT	Sistema operativo de 32 bits desarrollado por Microsoft para convertirse en el sistema operativo del futuro. Soporta multitarea real, puede correr tanto sobre procesadores RISC como CISC, y es capaz de ejecutar tanto programas DOS como Windows, POSIX y OS/2, todo ello en multitarea. Incluye seguridad de datos y un conjunto de caracteres de 16 bits, que le permite mostrar caracteres no románicos. Diseñado para funcionar sobre diferentes sistemas y procesadores.

Término	Significado
Windows 95	Nuevo Sistema Operativo de Microsoft presentado el 24 de agosto de 1995 tras la campaña publicitaria más cara de la historia informática y precedido de dos años de fase beta.
WWW	World Wide Web. Sistema de información distribuido, basado en hipertexto, potenciado en los 90's por Tim Berners Lee, investigador en el CERN, Suiza. La información puede ser de cualquier formato (texto, gráfico, audio, imagen fija o en movimiento) y es fácilmente accesible a los usuarios mediante los programas navegadores.
XML	Lenguaje Extensible de Marcado. Sistema desarrollado para promover el uso del lenguaje SGML en la red. XML no es un lenguaje sino un metalenguaje, es decir, sirve para crear lenguajes. No es una extensión ni un componente de HTML.

Referencias

Bibliografía

Análisis Moderno Estructurado,
Edward Yourdon,
Prentice Hall,
México D. F., 1990.

Análisis y Diseño de Sistemas
Keneth E. Kendall y Julie E. Kendall,
Prentice Hall Hispanoamericana,
México D.F., 1991.

Análisis y Diseño Orientado a Objetos,
James Martin y James J. Odell,
Prentice Hall,
México D.F., 1992.

Aprendiendo Delphi 2 en 21 días,
Dan Osier, Steve Grobman y Steve Batson,
Prentice Hall,
México D.F. 1996.

Cliente / Servidor, Guía de Supervivencia,
Robert Orfali, Dan Harkey y Jeri Edwards,
Mc Graw Hill,
México D. F., 1998.

Erwin Methods Guide,
Platinum Technology,
Berkeley California, Estados Unidos, 1998.

Getting Started
InterBase Server for Windows 95 & Windows NT
Borland International Inc.
Scott Valley California, Estados Unidos, 1998.

Introducción a los sistemas de bases de datos,
Date C.J.,
Addison Wesley Iberoamericana,
México, D.F.,1986.

Direcciones Electrónicas

Club Delphi
http://www.clubdelphi.com/

Delphi Super Page
http://www.delphizine.com/

Glosario básico Inglés-Español para usuarios de Internet
http://www.ati.es/novatica/glosario/glosario_internet.txt

La Página Orientada a Objetos
http://www.ctv.es/USERS/pagullo/

Revista Mundo Delphi
http://mundodelphi.com/