UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SITEMA INFORMÁTICO QUE PERMITE EL MANEJO CONTABLE DE UNA AGENCIA AUTOMOTRÍZ

T E S I S

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

GLORIA RAMÍREZ DE ....

DIRECTOR: .Ing. Lucila Patricia Arellano Mendoza

MEXICO, D.F. **2004**

Índice

Sistema informático que permite el manejo contable de una agencia automotriz

Capítulo I

Antecedentes de la empresa

I.1. LA EMPRESA

En 1959, Nissan Motor Co. llega a México como distribuidora de autos de marca Datsun. El 11 de septiembre de 1967, se constituye Nissan Mexicana S.A. de C.V. En 1966, se produce el primer automóvil mexicano: Datsun Sedan Bluebird.

En 1972, inicia la exportación de unidades a Latinoamérica. En 1975, Nissan Mexicana es pionera al establecer el primer laboratorio de pruebas de gases contaminantes de vehículos en México. En 1981, cambia la imagen de Datsun a Nissan en todo el mundo.

En 1995, Nissan Mexicana es nombrada base de comercialización regional en virtud de que el 60% de las exportaciones Nissan a Latinoamérica provienen de México. A partir de 1999, Nissan Mexicana produce desde su planta en Aguascalientes del modelo Sentra para todo el continente Americano.

En 2000, Nissan Mexicana llega a la producción de tres millones (3,000,000) de vehículos. Se inicia la producción del Scénic, primer vehículo dentro de la Alianza Nissan y Renault.

En 2001, Nissan Mexicana alcanza la cifra de UN MILLÓN de autos Tsuru vendidos en su historia.

La filosofía de Nissan es “Nuestro primer compromiso es la satisfacción de nuestros clientes. Al esforzarnos con diligencia en aumentar nuestra base de clientes, contribuimos al progreso y al enriquecimiento de la sociedad”

El círculo rojo representa al sol naciente de Japón y la sinceridad. La barra azul representa el cielo, y el concepto de la marca significa: "La sinceridad trae el éxito". (Ver Ilustración I.1)

Ilustración I.1

I.1.1 La distribuidora de autos

NISHI, S.A. DE C. V. forma parte de Nissan Mexicana como distribuidora de autos, que de igual manera dentro de sus actividades cuenta con la primera prestación de servicio y venta de refacciones a clientes NISSAN.

Dados los antecedentes de la compañía Nissan, Nishi no puede estancarse en el ámbito de desarrollo tecnológico, de manera que el mejoramiento de su sistema mejorará todos los procesos internos de la empresa que actualmente se llevan a cabo manualmente.

I.2. ÁREA CONTABLE

El área contable procesa la información recibida de las operaciones efectuadas por los vendedores o prestadores del servicio dentro de la empresa. Esta información almacenada debe ser correctamente dirigida para su proceso, es decir contar con su respectivo debe y haber, es decir que a cualquier cargo le corresponde su respectivo abono a todo tipo de cuenta.

Cuando la información como las facturas de venta de un auto, o por la prestación de un servicio enviada por el módulo de ventas y servicios llega a manos del área contable, se genera una póliza que respalda dicha información, haciendo que las cuentas contables dentro de este módulo tengan movimientos en su debe y haber, de manera que el total de los “debe” menos el total de los “haber” resulte cero.

Así, se ve que es necesario contar con un sistema que automatice la forma en la que se generan y/o cancelan facturas, ya sea en la venta o en la prestación de los servicios. De igual manera en la toma de decisiones administrativas es necesario tener a la mano reportes que contengan la información íntegra, concisa y consistente.

Sin embargo todas estas necesidades no se pueden satisfacer a partir del sistema actual, pues la base de datos contiene información repetida, existen tareas que no efectúa por si sólo el sistema, y se deben llevar a cabo por el mismo contador y corregir de esta forma los posible errores que contengan los reportes finales, proceso que puede efectuarse automáticamente con un sistema más completo y dirigido al área correspondiente.

I. 3. REQUERIMIENTOS

Dentro de los requerimientos que la empresa solicita está el poder llevar a cabo reportes automáticamente, es decir, el sistema deberá ser capaz de interactuar con otros sistemas para obtener información de forma automática con la mínima intervención del usuario.

Para ello el sistema debe considerar ciertos factores que a continuación se enumeran para que al término del mismo, se obtenga el sistema que nos permita mejorar el anterior al no permitir duplicidades en la información, incongruencia con los datos, evitar procedimientos manuales los cuales solían provocar errores durante su captura.

Los factores a incluir serán:

1. Un catálogo de cuentas.- En el que se visualice la información correspondiente a cada una de las cuentas generadas originalmente por el contador. Así como permitir (dependiendo del tipo de usuario) dar de alta, editar o eliminar cualquiera de ellas.

   Esta opción, debe permitir el visualizar cualquier periodo anterior al activo, así como la posibilidad de cambiar de ejercicio.

   Dado que no todos los usuarios tienen los mismos permisos para realizar todo tipo de actividades, éstas estarán restringidas en diferentes pantallas, es decir dependerá del periodo y ejercicio activo, así como de la información a actualizar por los anteriores periodos.

2. Un catálogo de pólizas.- Dentro de esta opción se deberá encontrar un submenú que incluya:

   - Mantenimiento de pólizas.- se permiten hacer altas, bajas y cambios sólo para el periodo activo, sólo consultas a periodos anteriores o cerrados, acceso a periodos con restricción por password

   - Contabilización diaria.- Visualización de una lista general de las pólizas en stand-by, permite visualizar y contabilizar las cuentas existentes en los diversos departamentos

   - Recalcular pólizas.- Toma la información inconsistente del periodo activo, permite hacer el recálculo de todas las pólizas, por departamento, una a una o por fecha

3. Generación de reportes.- En este módulo se encuentra una lista de reportes que ayudarán en la toma de decisiones a nivel gerencial. Tales como;

   - Balanza de comprobación.- es una lista de cuentas con el número de cuenta, descripción, saldo inicial, debe, haber, saldo final; y sus respectivos totales
   - Libro de pólizas.- Despliega las pólizas creadas en el periodo seleccionado, incluyendo el detalle de las cuentas afectadas, así como los totales parciales, es decir de cada póliza, así como el de todas las pólizas.
   - Estados financieros:
     1. Balance general que es un comparativo general como su nombre lo indica de los pasivos contra los activos de la empresa.
     2. Estado de situación financiera que representa en forma generalizada por Patrimonio = Recursos – Obligaciones.
   - Estado de resultados.- Donde se refleja la utilidad o pérdida de la empresa a partir de la fórmula Ingresos – Egresos = Resultado (utilidad o pérdida)

Capítulo II

Bases teóricas

II.1 CONTABILIDAD

La contabilidad es una técnica que se ocupa de registrar, clasificar y resumir las operaciones mercantiles de un negocio con el fin de interpretar sus resultados, para que los gerentes a través de ella puedan orientarse sobre el curso que siguen sus negocios mediante datos contables; permitiendo así conocer la estabilidad, la solvencia y la capacidad financiera de la empresa.

II.1.1 Historia

La contabilidad se remonta desde tiempos muy antiguos, cuando el hombre se ve obligado a llevar registros y controles de sus propiedades porque su memoria no bastaba para guardar la información requerida. Se ha demostrado a través de diversos historiadores que en épocas como la egipcia o romana, se empleaban técnicas contables que se derivan del intercambio comercial.

La Revolución Industrial provocó la necesidad de adoptar las técnicas contables para poder reflejar la creciente mecanización de los procesos, las operaciones típicas de la fábrica y la producción masiva de bienes y servicios. Con la aparición, a mediados del siglo XIX, de corporaciones industriales, propiedades de accionistas anónimos, el papel de la contabilidad adquirió aún mayor importancia.

Actualmente la contabilidad, es una herramienta empresarial que permite el registro y control sistemático de todas las operaciones que se realizan en la empresa, por ende no existe una definición concreta de la contabilidad aunque todas estas definiciones tienen algo en común.

La teneduría de libros, parte esencial de cualquier sistema, ha ido informatizándose a partir de la segunda mitad del siglo XX, por lo que, cada vez mas, corresponde a los ordenadores o computadoras la realización de estas tareas. El uso generalizado de los equipos informáticos permitió sacar mayor provecho de la contabilidad utilizándose a menudo el término “procesamiento de datos”, actualmente el concepto de teneduría ha decaído en desuso.

La contabilidad hacia el siglo XXI se ve influenciada por tres variables:

- Tecnología.
- Complejidad y globalización de los negocios.
- Formación y educación.

II.1.2 Sistema de información contable

Un sistema de información contable comprende los métodos, procedimientos y recursos utilizados por una entidad para llevar un control de las actividades financieras y resumirlas en forma útil para la toma de decisiones.

La información contable se puede clasificar en dos grandes categorías: la contabilidad financiera o la contabilidad externa y la contabilidad de costos o contabilidad interna.

1. La contabilidad financiera o externa muestra la información que se facilita al público en general, y que no participa en la administración de la empresa, como son los accionistas, los acreedores, los clientes, los proveedores, los analistas financieros, entre otros, aunque esta información también es de mucho interés para los administradores y directivos de la empresa. Esta contabilidad permite obtener información sobre la posición financiera de la empresa, su grado de liquidez y sobre la rentabilidad de la empresa y tiene como objeto facilitar al público información sobre la situación económico-financiera de la empresa.

2. La contabilidad de costos o interna estudia las relaciones costos – beneficios – volumen de producción, el grado de eficiencia y productividad, y permite la planificación y el control de la producción, la toma de decisiones sobre precios, los presupuestos y la política del capital. Esta información no suele difundirse al público y se dedica a facilitar información a los distintos departamentos, directores y planificadores.

Para el sistema contable de cualquier empresa, se deben ejecutar tres pasos básicos

1. Registro de la actividad financiera: en un sistema contable (que será propósito de este proyecto proporcionar el adecuado para las necesidades de la empresa) se debe llevar un registro sistemático de la actividad comercial diaria en términos económicos. En una empresa se llevan a cabo todo tipo de transacciones que se pueden expresar en términos monetarios y que se deben registrar en los libros de contabilidad. Una transacción se refiere a una acción ya terminada más que a una posible acción a futuro.

2. Clasificación de la información: un registro completo de todas las actividades comerciales implica comúnmente un gran volumen de datos, demasiado grande y diverso para que pueda ser útil para las personas encargadas de tomar decisiones. Por tanto, la información de debe clasificar en grupos o categorías. Se deben agrupar aquellas transacciones a través de las cuales se recibe o paga dinero.

3. Resumen de la información: para que la información contable utilizada por quienes toman decisiones pueda ser analizada, ésta debe ser resumida. Por ejemplo, una relación completa de las transacciones de venta de una empresa automotriz seria demasiado larga para que cualquier persona se dedicara a leerla. Los empleados responsables de comprar mercancías necesitan la información de las ventas resumidas por producto. Los gerentes de almacén necesitaran la información de ventas resumida por departamento, mientras que la alta gerencia necesitará la información de ventas resumida por almacén.

Estos tres pasos que se han descrito: registro, clasificación y resumen constituyen los medios que se utilizan para crear la información contable. Sin embargo, el proceso

contable incluye algo mas que la creación de información, también involucra la comunicación de esta información a quienes estén interesados y la interpretación de la información contable para ayudar en la toma de decisiones comerciales.

Un sistema contable debe proporcionar información a los gerentes y también a varios usuarios externos que tienen interés en las actividades financieras de la empresa.

II.1.2.1 Utilización de la Información Contable

La contabilidad va mas allá del proceso de creación de registros e informes. El objetivo final de la contabilidad es la utilización de esta información, su análisis e interpretación. Los contadores se preocupan de comprender el significado de las cantidades que obtienen. Buscan la relación que existe entre los eventos comerciales y los resultados financieros; estudian el efecto de diferentes alternativas, por ejemplo la compra o el arriendo de un nuevo edificio; y buscan las tendencias significativas que sugieren lo que puede ocurrir en el futuro.

Si los gerentes, inversionistas, acreedores o empleados gubernamentales van a darle un uso eficaz a la información contable, también deben tener un conocimiento acerca de cómo obtuvieron esas cifras y lo que significan. Una parte importante de esta comprensión es el reconocimiento claro de las limitaciones de los informes de contabilidad. Un gerente comercial u otra persona que esté en posición de tomar decisiones y que carezca de conocimientos de contabilidad, probablemente no apreciara hasta qué punto la información contable se basa en estimativos mas que en mediciones precisas y exactas.

II.1.2.2 Características de un sistema de información contable efectivo.

Un sistema de información bien diseñado ofrece control, compatibilidad, flexibilidad y una relación aceptable de costo / beneficio.

Control : un buen sistema de contabilidad le da a la administración control sobre las operaciones de la empresa. Los controles internos son los métodos y procedimientos que usa un negocio para autorizar las operaciones, proteger sus activos y asegurar la exactitud de sus registros contables.

Compatibilidad: un sistema de información cumple con la pauta de compatibilidad cuando opera sin problemas con la estructura, el personal, y las características especiales de un negocio en particular.

Flexibilidad: para el fácil acceso a ella y de fácil manipulación según sean las necesidades de la empresa

Y por último debe ofrecer los beneficios que la empresa demande a un costo razonable.

II.1.2.3 Objetivos de la información contable.

La información contable debe servir fundamentalmente para:

a) Conocer y demostrar los recursos controlados por un ente económico, las obligaciones que tenga de transferir recursos a otros entes, los cambios que hubieren experimentado tales recursos y el resultado obtenido en el periodo.
b) Predecir flujos de efectivo.
c) Apoyar a los administradores en la planeación, organización y dirección de los negocios.
d) Tomar decisiones en materia de inversiones y crédito.
e) Evaluar la gestión de los administradores del ente económico.
f) Ejercer control sobre las operaciones del ente económico.
g) Fundamentar la determinación de cargas tributarias, precios y tarifas.
h) Ayudar a la conformación de la información estadística nacional.
i) Contribuir a la evaluación del beneficio o impacto social que la actividad económica representa para la comunidad.

II.1.2.4 Cualidades de la información contable

Para poder satisfacer adecuadamente sus objetivos, la información contable debe ser comprensible, útil y en ciertos casos se requiere que además la información sea comparable.

- La información es comprensible cuando es clara y fácil de comprender.
- La información es útil cuando es pertinente y confiable.
- La información es pertinente cuando posee el valor de realimentación, valor de predicción y es oportuna.
- La información es confiable cuando es neutral, verificable y en la medida en la cual represente fielmente los hechos económicos.

La tecnología que a través del impacto genera el aumento en la velocidad con la cual se generan las transacciones financieras, a través del fenómeno Internet. La segunda variable de complejidad y globalización de los negocios, requiere que la contabilidad establezca nuevos métodos para el tratamiento y presentación de la información financiera. La última variable relacionada con la formación y educación requiere que los futuros gerentes dominen el lenguaje de los negocios.

Aunque la contabilidad ha logrado su progreso más notable en el campo de los negocios, la función contable es vital en todas las unidades de nuestra sociedad. Las grandes compañías por acciones son responsables ante sus accionistas, ante las agencias gubernamentales y ante el público. El gobierno, los estados, las ciudades y los centros educativos, deben utilizar la contabilidad como base para controlar sus recursos y medir sus logros. La contabilidad es igualmente esencial para la operación exitosa de un negocio, una universidad, una comunidad, un programa social o una ciudad.

Retomando el concepto de informe contable, éste resume la información que debe ser registrada en el sistema contable. En el registro de la actividad económica, los contadores se basan en 'transacciones' completas, es decir en:

- Eventos que originan cambio inmediato en los recursos financieros u obligaciones de los negocios
- Eventos que pueden ser medidos objetivamente en términos monetarios

La fuerza inicial de este enfoque sobre las transacciones radica en la confiabilidad de la información que está registrada. Las decisiones que toma la gerencia están basadas principalmente en la información que desarrolla el sistema de contabilidad. Por lo tanto la gerencia necesita asegurarse de que la información contable que recibe es segura y contable.

Toda esa información registrada se encuentra almacenada como un conjunto de datos que deben ser administrados de forma específica. Por tanto la mejor opción para manejarlos es a través de una base de datos que pueda asegurar al usuario características tales como:

- Acceso a la información de forma restringida
- Integridad de la información
- Mínima redundancia, etc.

II.2 BASES DE DATOS

Una base de datos puede resumirse en una serie de datos (también llamada simplemente información) relacionados entre sí, con base en este concepto, se puede definir un sistema de bases de datos como un sistema para archivar en una computadora, es decir un sistema computarizado cuyo propósito general es mantener información y hacer que esté disponible cuando se solicite.

Una base de datos es un modelo de un modelo previo, es decir es el modelo de otro previamente hecho por el usuario, el cual es la idea que éste tiene del mundo real, por ejemplo si la empresa es de venta de automóviles y refacciones, la base de datos no representa directamente la venta de ellos, si no quien la haya diseñado la va a modelar según la imagen que tenga de la empresa, por lo que los datos serán simplemente un nombre, descripción de un auto, precios etc.

Cada base de datos varia según el nivel de detalle que tenga, algunos pueden ser muy simples, como nombres y cuentas por pagar, o tan detalladas como sus direcciones, teléfonos, trabajos, etc. El nivel que tenga esta base dependerá de sus propias necesidades, es decir de la información necesaria para llevar a cabo las funciones que la empresa demande.

El modelo de la base debe ser dinámico por los cambios en los negocios, el movimiento de la gente que va y viene, la introducción de nuevos productos, así como la

obsolencia de otros, el dinero que se gasta y el que se gana, etc. por lo que una base de datos que sea estática no podría servir a una empresa real.

II.2.1 Modelo de datos

El modelado de datos es el proceso en el que se crea una representación de los datos desde el punto de vista del usuario. Esta es la tarea más importante en el desarrollo de las aplicaciones de la base de datos, realmente efectivas. La creación del modelo es la base para todos los trabajos subsecuentes en el desarrollo de bases de datos así como para sus aplicaciones.

Podemos esquematizar la arquitectura de un sistema manejador de bases de datos como sigue

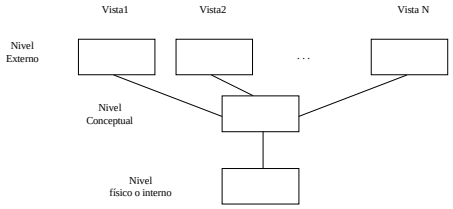


Ilustración II.1

Nivel interno Es la representación a nivel inferior de una BD, es la forma como se almacenan realmente los datos.

Nivel conceptual Se describen cuales son los datos reales y que la relación existe entre ellos. Este nivel contiene toda la BD en términos de unas estructuras sencillas.

Nivel externo o de visión Es el más cercano al usuario es decir, es el que se ocupa de la forma como los usuarios individuales perciben la información.

Se cuentan con diferentes formas de modelar los datos, este modelo se efectuará a partir de los datos reales, es decir dentro del nivel conceptual, y se pueden utilizar alguno de los siguientes tipos:

- Modelo de datos en red.- Representa los datos en estructuras de retículo de tipos de registros conectados por interrelaciones uno-uno o uno-muchos.
- Modelo de datos jerárquico.- Modelo de dato en el cual todas las interrelaciones son estructuradas como árboles.
- Modelo de datos relacional.- Un modelo de datos donde los datos se representan en forma de tabla.
- Modelo orientado a objetos.- un modelo que captura los significados de las entidades del mundo real y sus interrelaciones.

Para nuestro caso, sólo será necesario estudiar los modelos relacionales, y el orientado a objetos, ambos basados en el modelo conceptual entidad-relación

II.2.1.1 Modelo relacional

Este modelo consiste en un conjunto de tablas, (con un nombre único) donde cada una de las columnas es una relación de datos, estas relaciones para nuestro modelo son un conjunto de datos con ciertas características en común, como lo serían todos los nombres de los empleados de una empresa, o todas las fechas en que fueron contratados, o las edades de todos ellos.

Otra característica importante de este modelo es que para agrupar todos los datos en la base de datos es necesario tener más de una tabla, es decir varios conjuntos de relaciones, también llamados esquemas, por ejemplo para la empresa que tiene almacenados a sus empleados y a los clientes de cada uno de ellos sería una pérdida de espacio tener almacenada toda la información de un empleado por cada cliente que atienda, de esta manera se tuvo la necesidad de manejar más tablas o esquemas.

Las relaciones que forman un esquema tienen propiedades específicas como se muestra en la Ilustración II.2.

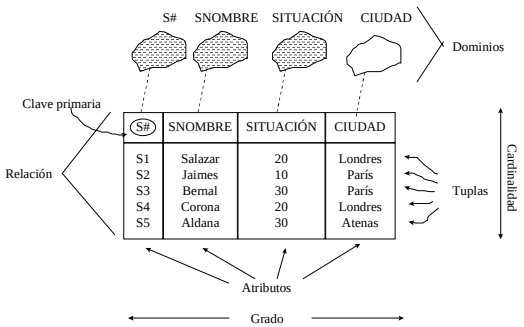


Ilustración II.2

y cumpliendo con estas condiciones:

- No existen tuplas repetidas
- Las tuplas no están ordenadas
- Los atributos no están ordenados
- Todos los valores de los atributos son atómicos

Para tener acceso a la información de las tablas ya mencionadas es necesario contar con ciertos lenguajes previamente establecidos, llamados “lenguajes de consulta”; dichos lenguajes son por lo general de alto nivel.

Uno de los lenguajes de consulta clasificado como de procedimiento (se genera la respuesta a la consulta después de escribir la expresión por medio de una secuencia de operaciones) es el álgebra relacional que contiene 5 operaciones básicas (Ver tabla II.1).

OPERACIÓN	DESCRIPCIÓN	SÍMBOLO
SELECCIONAR	Extrae las tuplas específicas de una relación dada, presenta sólo las TUPLAS que satisfagan la condición especificada.	σ
PRODUCTO	A partir de 2 relaciones especificadas, construye una relación que contiene todas las posibles combinaciones de tuplas uno de cada uno de las 2 relaciones	X
UNIÓN	Construye una relación formada por todas las tuplas que aparecen en cualquier a de las 2 relaciones especificadas (mismo tipo)	$\cup$
PROYECTAR	Extrae los atributos especificados de una relación dada	π
DIFERENCIA	Construye una relación formada por todas las tuplas de la primera relación que no aparezcan en la segunda de las 2 relaciones especificadas	–

Tabla II.1

Por otro lado el cálculo relacional es un lenguaje sin procedimientos; donde no se especifica la forma en que se va a obtener la información, simplemente se da una descripción formal de la información deseada. Toda expresión en cálculo relacional siempre tendrá la forma

$$\{t \ / \ p(t)\}$$

Que se interpreta como el conjunto de tuplas t tales que la proposición P se cumpla.

La solución de toda consulta en cálculo relacional es una relación que se define por una lista resultado y por una sentencia de cualificación. La lista resultado define los atributos de la relación solución. La sentencia de cualificación es una condición usada para determinar qué valores de la base de datos actual van a la relación solución.

Existen otros lenguajes para la consulta de información comerciales que explicaremos en otro capítulo tales como SQL.

II.2.1.2. Modelo orientado a objetos

La programación orientada a objetos surgió durante los últimos 10 años, este nuevo estilo ha demostrado tantos avances significativos sobre la programación tradicional que casi todos los nuevos desarrollos en software son orientados a objetos, este tipo de programación a diferencia de la tradicional, ve los programas como conjuntos de estructuras de datos que tienen tanto los datos de los elementos como las instrucciones del programa.

Una diferencia entre la programación tradicional y la orientada a objetos, es que en la tradicional está organizada primero la lógica y después los datos, mientras que la orientada a objetos es de manera contraria.

Para entender más profundamente este modelado es necesario conocer ciertos términos:

- Un objeto es una estructura encapsulada, que tiene tanto atributos como métodos. El término encapsulado significa que está completo por sí mismo; los programas externos a un objeto no saben nada de su estructura y no le es necesario saberlo. El aspecto que tenga externamente un objeto se refiere exclusivamente a su interfaz.

- Una interfaz consta de los atributos y métodos que son visibles al mundo exterior.

- Los atributos de un objeto están organizados en una estructura particular, y son a su vez muy parecidos a los atributos de los objetos semánticos.

- Los métodos de un objeto, son secuencias de instrucciones que el objeto ejecuta. En un programa se encuentran muchos objetos que tienen métodos en común. Para reducir la redundancia en la programación, los objetos son subclasificados desde la clase más general. Es decir los objetos de la clase más general tienen inherentes todos los atributos y métodos de los objetos siguientes en la clasificación.

- Una clase objeto comprende la estructura lógica de un objeto, el objeto en sí, un nombre, atributos y métodos.

- Una biblioteca clase objeto es un grupo de clases objeto. Y las instancias de objetos son llamadas simplemente objetos.

Los objetos son creados llamando un objeto constructor, programas que obtienen memoria y crean las estructuras necesarias para inicializar un objeto. Los objetos destructores son programas que inhiben objetos y liberan memoria. Los programas pueden ser transitorios o permanentes. Un objeto transitorio existe solamente en la memoria volátil durante la ejecución de un programa. Cuando termina el programa, el objeto se pierde. Un objeto persistente sobrevive la ejecución de un programa y puede ser releído desde memoria.

II.2.2 Modelo entidad-relación

Este es un modelo conceptual que para poder entender su funcionamiento, es necesario definir los siguientes conceptos:

Entidad.- es lo que puede ser identificado en el ambiente de trabajo del usuario. Es algo importante a construir para los usuarios del sistema.

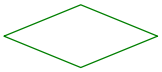
EJEMPLOS	SÍMBOLO
Empleado Cliente Vendedor Producto	

Atributos.- Las entidades tienen atributos, (también llamados propiedades) los cuales describen las características de la entidad.

EJEMPLOS	SÍMBOLO
Nombre del empleado Fechas de entrega Código de tarea	

La entidad empleado tiene dentro de sus valores al empleado1 y/o al empleado2, cada uno de estos ejemplos lo llamaremos instancia.

Relaciones.- La forma en que se asocian una a otra las entidades es por medio de las relaciones, las cuales pueden tener atributos por sí solas.

EJEMPLOS	SÍMBOLO
No. de venta (vendedor-cliente) Facturación (producto-cliente)	

Identificadores: Cada instancia tiene identificadores, los cuales son atributos que llaman o identifican cada una de las instancias. Por ejemplo las instancias de la entidad empleado pueden ser identificados por su numero de seguro social, numero de empleado o nombre del empleado. Estas instancias no podrían ser identificadas por el salario o tarea que llevan a cabo, ya que normalmente estos atributos no son usados para el papel de nombrar algo específico. El identificador puede ser formado por uno o más atributos de la entidad, el cual puede o no ser único. Estos identificadores también son llamados llaves.

- Llave.- característica principal que diferencia una entidad de otra
- Superllave.- conjunto de atributos que identifica como único a una identidad
- Candidato.- superllave mínima

- Llave primaria.- llave candidato seleccionada por el programador

Limitantes de mapeo.- cardinalidad de mapeo, las entidades pueden generarse de 4 formas distintas, es decir las combinaciones que se crean a partir de unir 1 o muchos elementos de cada entidad con los de otra (Ver tabla II.1).

SÍMBOLOS	
M:M	M:1
1:M	1:1

Tabla II.2

II.3 ANÁLISIS DE DISEÑO DE SISTEMAS

Todo análisis debe incluir un trabajo arduo dentro de un área administrativa para poder ejecutar de forma más simple el diseño de dicho sistema, este trabajo debe incluir los siguientes puntos.

II.3.1. Actividades administrativas

- Propuesta.- la propuesta describe los objetivos del proyecto y de cómo debe llevarse a cabo. Generalmente incluye un costo y un programa estimado.

- Planeación del proyecto y calendarización.- se refiere a la identificación de actividades, y/o los obstáculos producidos o generados durante el proyecto. Por lo tanto un proyecto debe ser redactado para guiar el desarrollo del proyecto hacia las metas requeridas.

- Costos.- La estimación de costos es una actividad asociada que se refiere a la estimación de recursos requeridos para completar el plan.

- Monitoreo.- Es una actividad continua dentro del proyecto. El administrador debe conservar el trayecto del progreso del proyecto, y compararlo el actual con el ya planeado así como con los costos.

- Selección y evaluación de personal.- Idealmente el mejor personal, es el más hábil, con la experiencia apropiada, sin embargo en la mayoría de los casos no se puede pagar el tipo de personal ideal.

- Presentación y reportes escritos.- El administrador del proyecto es generalmente el responsable de hacer un reporte acerca del proyecto tanto al cliente, como a la organización de contrato. Estos reportes deben ser concisos, coherentes, y concretos.

II.3.2. Planeación del proyecto

El administrador del proyecto debe anticipar los problemas que puedan surgir y preparar posibles soluciones para los mismos.

Una planificación redactada al inicio del proyecto debe ser usada como un administrador del proyecto. Además del plan de proyecto, los administradores deben desarrollar otro tipo de planes tales como:

- Plan de calidad.- que describe los procesos de calidad y los estándares que serán utilizados.

- Plan de validación.- Describe los accesos, recursos y calendario usado por el sistema de validación.

- Plan de administración de configuración.- Describe los procedimientos de administración y las estructuras usadas.

- Plan de mantenimiento.- Pronostica los requerimientos del mantenimiento del sistema, costos del mismo y esfuerzo requerido.

- Plan de desarrollo del personal.- Describe cómo la destreza y experiencia del equipo será utilizado.

II.3.2.1. Plan del proyecto

Es un documento sencillo que incluye los diferentes tipos de planes. Los detalles de este plan varía dependiendo de la empresa y el tipo de proyecto, sin embargo, la mayoría de las planeaciones incluyen las siguientes secciones:

- Introducción.- Describe brevemente los objetivos del proyecto y establece las restricciones sin afectar la administración del proyecto.

- Organización.- Contiene la forma en que se organiza el equipo del proyecto, el personal implicado y los roles que van a desempeñar.

- Análisis de riesgos.- Describe los posibles riesgos del proyecto, la probabilidad de que estos surjan y las estrategias de reducción de los mismos.

- Análisis de requerimientos.- Incluye el hardware y software de soporte requerido para llevar a cabo el desarrollo, si tiene que comprarse, estimaciones de los costos, calendario de entrega, etc.

- Calendarización.- Explica las independencias entre las actividades, el tiempo estimado requerido para alcanzar cada meta, y la forma en que se repartirán las actividades entre el personal
- Monitoreo y mecanismos de reporte.- Se describen los reportes que deben ser generados, y los mecanismos de monitoreo usados.

II.3.3. Calendarización

Se estima el tiempo y recursos requeridos para completar las actividades y organizarlas de manera coherente.

Este paso tiende a complicarse por el hecho de que diferentes proyectos utilizan diferentes métodos y lenguajes de implantación. Si la técnica del proyecto es avanzada, las estimaciones iniciales deben ser lo más optimistas posibles, aún cuando se traten de considerar todas las eventualidades podemos encontrar frecuentemente en diversas producciones retrasos frecuentes por presentarse problemas no anticipados, los programas por lo tanto deben ser actualizados continuamente conforme la información se encuentre disponible.

Esta programación o calendarización incluye la separación del proyecto total en diversas actividades considerando el tiempo requerido para realizarlas. Algunas veces las actividades pueden efectuarse en paralelo. La programación debe coordinar estas actividades y organizarla de manera que el trabajo efectuado sea lo más optimo posible.

Se deben evitar las situaciones en las que el proyecto sea retrasado por completo. Las actividades principales deben durar por lo menos una semana (aunque este tiempo depende de la magnitud del proyecto.) Cuando se encuentran actividades que duran de 8 a 10 semanas estas deben subdividirse y así hacer una mejor programación con las subactividades.

En la planeación del programa no se debe asumir que todas las etapas estén libres de problema, como sería la enfermedad de algún miembro del equipo, la descompostura de alguna herramienta, o simplemente el retraso de un programa.

Además del tiempo en el calendario, se deben estimar los recursos necesarios para completar cada una de las tareas. El principal recurso es el humano, otros pueden ser el espacio en disco requerido, el tiempo necesario para utilizar hardware especializado, tales como simuladores, tarjetas, etc.

Una regla importante sería estimar el tiempo, como si nada fuera a salir mal, e ir incrementando el tiempo conforme se van cubriendo problemas anticipadamente. Existen administradores que aumentan 30% extra para problemas anticipados y otro 20% para otros problemas que puedan surgir.

II.3.3.1. Gráfica de barras y red de actividades

Esto se refiere a notaciones gráficas que son usadas para ilustrar el programa del proyecto.

La gráfica de barras muestra quien es responsable de qué actividad y cuando es el comienzo y fin de dicha actividad. La red de actividades muestra las dependencias entre las diferentes actividades del proyecto.

Estas gráficas pueden ser generadas automáticamente desde una base de datos. Una vez que se conocen el tiempo estimado por tarea y sus dependencias se puede llevar a cabo la red de actividades, que muestra la secuencia de tales actividades, qué actividad se efectúa en paralelo con qué otra, y cuales deben efectuarse en secuencia.

II.3.4. Estudio de viabilidad

Una tarea importante dentro de la planeación del proyecto es la de anticipar riesgos que podrían afectar a la planeación del mismo, así como la calidad de los programas que se estén desarrollando.

El resultado del análisis de riesgos debe ser documentado junto con un análisis de las consecuencias cuando se haya presentado uno de ellos.

Se le llama manejo de riesgos al proceso de identificar los riesgos que puedan ocurrir en el proyecto así como documentarlos para poder minimizar el efecto que ejerzan sobre el proyecto.

Un riesgo es la probabilidad de que alguna circunstancia adversa suceda, y pueda ser una amenaza para la empresa, el sistema o un proyecto menor. Los riesgos pueden clasificarse como de:

Proyecto.- Afectan los recursos o la planeación
Producto.- Afectan la calidad o el rendimiento del software en desarrollo.
Negocio.- Afectan el desarrollo de la empresa u organización.

Aunque claro está que pueden existir riesgos que pertenezcan a las tres clasificaciones.

Existen diversos tipos de riesgos que dependiendo del proyecto lo van a afectar de una forma específica. Tales como:

Abandono del proyecto por parte del personal
Cambio de personal administrativo
Falta de disponibilidad del equipo (Hardware)
Cambios en los requerimientos
Subestimación en el tamaño del proyecto
Cambio en la tecnología

Competencia de otros productos

El proceso que en general se sigue para el manejo o administración de riesgos es el siguiente:

Identificación de riesgos
Análisis de riesgos
Planeación de riesgos
Monitoreo de riesgos

La identificación consiste en descubrir los posibles riesgos. A pesar de que en esta etapa no se debe efectuar ningún tipo de valoración, en la práctica usualmente no se consideran los riesgos que provoquen mínimas consecuencias, o que tengan poca probabilidad de ocurrencia.

Los tipos de riesgos pueden ser:

Tecnológicos
Humanos
De organización
De herramientas
De requerimientos
De estimación

El análisis se efectúa en cada riesgo individualmente para que se encuentren la probabilidad y la seriedad del mismo, lo cual va a depender al 100% del administrador del proyecto, por lo cual no se pueden medir numéricamente, aunque algunos casos se pueden redondear de la siguiente manera:

Poca probabilidad → < 10%
Baja → 10-25%
Moderada → 25-50%
Alta → 50-75%
Muy alta → >75%

Ahora los efectos se pueden clasificar como:

Catastrófico
Serio
Tolerable
Insignificante

Los resultados obtenidos deben ser tabulados para que se puedan identificar de mayor a menor seriedad.

Entre mayor sea la información recopilada, mejores resultados se obtendrán en el análisis y como dicha información puede cambiar durante el proyecto, será necesario actualizarla para mejores resultados en el análisis.

Ya que los riesgos tienen su categoría definida y han sido analizados exhaustivamente debe existir un juicio acerca de cuáles serán los riesgos que realmente valdrá la pena tomar en cuenta durante el proceso. Este juicio dependerá de la combinación entre la probabilidad de ocurrencia y la seriedad de sus efectos.

La planeación es el punto en el que se elaborarán las estrategias necesarias para contrarrestar todo tipo de riesgos. Obviamente como en los puntos anteriores es difícil establecer reglas específicas para diversos tipos de riesgos, aunque las estrategias pueden considerarse dentro de las siguientes categorías:

De prevención.- Al seguir esta estrategia, la probabilidad de ocurrencia se reduce.
De minimización.- Con ésta se reducirá el impacto que produzca el riesgo.
Plan de contingencia.- En dado caso que lo peor llegara a suceder, estar preparado para contrarrestarlo.

El monitoreo se refiere a la constante valorización de los riesgos para decidir si cambiará la probabilidad de ocurrencia o los efectos que causen, claro que los riesgos no pueden ser monitoreados directamente, sino por medio de otros factores. El monitoreo de riesgos debe ser un proceso continuo y en cada revisión sobre el progreso del proyecto, los riesgos clave deber ser considerados por separado y discutirlos con su debida importancia.

II.4. Lenguaje de modelado Unificado (UML)

Es una especificación de notación orientada a objetos. Divide cada proyecto en un número de diagramas que representan diferentes vistas del proyecto. Estos diagramas en conjunto representan la arquitectura del proyecto. UML también intenta solucionar el problema de propiedad de código que se da con los desarrolladores, al implantar un lenguaje de modelado común para todos los desarrollos, se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.

II.4.1. Diagramas

La explicación estará basada en los diagramas, en lugar de basarse en vistas o anotaciones, ya que estos son la base de UML. Cada diagrama utiliza la anotación necesaria y la suma de los diagramas crean las diferentes vistas existentes en UML, las cuales son:

- Vista casos de uso: Se forma con los diagramas de casos de uso, colaboración, estados y actividades.
- Vista de diseño: Se forma con los diagramas de clases, objetos, colaboración, estados y actividades.

- Vista de procesos: Se forma con los diagramas de la vista de diseño. Recalcando las clases y objetos referentes a procesos.
- Vista de implantación: Se forma con los diagramas de componentes, colaboración, estados y actividades.
- Vista de despliegue: Se forma con los diagramas de despliegue, interacción, estados y actividades.

Se tienen dos tipos diferentes de diagramas:

- Los que dan una vista estática del sistema:

    - Diagrama de clases: muestra las clases, interfaces, colaboraciones y sus relaciones. Son los más comunes y dan una vista estática del proyecto.
    - Diagrama de objetos: es una diagrama de instancias de las clases mostradas en el de clases. Muestra las instancias y como se relacionan entre sí. Se da una visión de casos reales.
    - Diagrama de componentes: muestra la organización de los componentes del sistema. A cada componente le corresponde una o varias clases, interfaces o colaboraciones.
    - Diagrama de despliegue: Muestra los nodos y sus relaciones. Un nodo es un conjunto de componentes. Se utiliza para reducir la complejidad de los diagramas de clases y componentes de un gran sistema. Sirve como resumen e índice.
    - Diagrama de casos de uso: Muestran los casos de uso, actores y sus relaciones. Muestra quien puede hacer que y las relaciones que existen entre las acciones (casos de uso). Son muy importantes para modelar y organizar el comportamiento del sistema.

- Y los que dan una visión dinámica:

    - Diagrama de secuencia, diagrama de colaboración: Muestra los diferentes objetos y las relaciones que pueden tener entre ellos, los mensajes que se envían entre ellos. Son dos diagramas diferentes, que se puede pasar de uno a otro sin pérdida de información, pero que nos dan puntos de vista diferentes del sistema. En resumen, cualquiera de los dos es un diagrama de interacción.
    - Diagrama de estados: muestra los estados, eventos, transiciones y actividades de los diferentes objetos. Son útiles en sistemas que reaccionen a eventos.
    - Diagrama de actividades: es un caso especial del diagrama de estados. Muestra el flujo entre los objetos. Se utilizan para modelar el funcionamiento del sistema y el flujo de control entre objetos.

Como se puede observar el número de diagramas es muy alto, en la mayoría de los casos excesivos, y UML permite definir solo los necesarios, ya que no todos son necesarios en todos los proyectos.

II.4.1.1. Diagramas recomendados

Los diagramas a representar dependerán del sistema a desarrollar, para lo cual se efectúan las siguientes recomendaciones dependiendo del sistema y se deberán adaptar a las características de cada desarrollo.

- Aplicación monopuesto: diagrama de casos de uso, diagrama de clases, diagrama de interacción
- Aplicación monopuesto, con entrada de eventos: Añadir Diagrama de estados
- Aplicación cliente servidor: Añadir Diagrama de despliegue y diagrama de componentes, dependiendo de la complejidad
- Aplicación compleja distribuida: Todos

Así se tiene que para una aplicación sencilla se deben realizar entre tres y seis tipos de diagramas, y para una aplicación compleja unos nueve tipos. El tiempo dedicado a la realización de los diagramas es proporcional al tamaño del producto a realizar. Para la mayoría de los casos se tendrá suficiente con tres o cuatro diagramas. Se debe tomar en cuenta que UML esta pensado para el modelado tanto de pequeños sistemas como de sistemas complejos, y que los sistemas complejos pueden estar compuestos por millones de líneas de código y ser realizados por equipos de centenares de programadores.

II.4.2. Diagrama de casos de uso

Un caso de uso especifica un requerimiento funcional, se emplean para visualizar el comportamiento del sistema, una parte de él o de una sola clase. De forma que se pueda conocer como responde esa parte del sistema. El diagrama de uso es muy útil para definir como debería ser el comportamiento de una parte del sistema, ya que sólo especifica como deben comportarse y no como están implantadas las partes que define. Por ello es un buen sistema para documentar partes del código que deban ser reutilizables por otros desarrolladores. El diagrama también puede ser utilizado para que los expertos del dominio se comuniquen con los informáticos o en todo caso sin llegar a niveles de complejidad.

En el diagrama se encuentran diferentes símbolos que pueden mantener diversas relaciones entre ellas:

- Caso de uso: representado por una elipse, cada caso de uso contiene un nombre, que indica su funcionalidad. Un caso de uso debe tener las siguientes características:

    a) Ser siempre inicializado por un actor, es decir el actor debe directa o indirectamente ordenar al sistema desarrollar el caso de uso. Ocasionalmente, el actor no estará conciente de la inicialización del caso de uso.
    b) Un caso de uso debe entregar algún tipo de valor tangible al usuario. El valor no tiene que ser siempre un valor de salida, pero debe ser discernible.
    c) Debe ser una descripción completa.

Los casos de uso pueden tener relaciones con otros caso de uso tales como:

  - Include: Representado por una flecha
  - Extends: Una relación de un caso de uso hacia un caso de uso B indica que el caso de uso B ejecuta la funcionalidad de caso de uso A.
  - Generalization: Es una típica relación de herencia.

- Actores: Se representan por un muñeco. Sus relaciones son:
  - Communicates: Comunica un actor con un caso de uso, o con otro actor
- Parte o límite del sistema (System boundary): Representado por un cuadro, identifica las diferentes partes del sistema y contiene los casos de uso que la forman.

Se puede emplear el diagrama de dos formas diferentes, para modelar el contexto de un sistema y para modelar los requisitos del sistema.

- Modelado del contexto

Se debe modelar la relación del sistema con los elementos externos, ya que son estos elementos los que forman el contexto del sistema.

Los pasos a seguir son:

1. Identificar los actores que interactúan con el sistema
2. Organizar a los actores
3. Especificar sus vías de comunicación con el sistema

- Modelado de requisitos

La función principal, o la mas conocida del diagrama de casos de uso es documentar los requisitos del sistema, o una parte de él. Los requisitos establecen un contrato entre el sistema y su exterior, definen lo que se espera que realice el sistema, sin definir su funcionamiento interno. Es el paso siguiente al modelado del contexto, no indica relaciones entre autores, sólo indica cuales deben ser las funcionalidades (requisitos) del sistema. Se incorporan los casos de uso necesarios que no son visibles desde los usuarios del sistema.

Para modelar los requisitos es recomendable:

1. Establecer su contexto, para lo que también podemos usar un diagrama de casos de uso.
2. Identificar las necesidades de los elementos del contexto (actores).
3. Nombrar esas necesidades, y darles forma de caso de uso.
4. Identificar qué casos de uso pueden ser especializaciones de otros, o buscar especializaciones comunes para los casos de uso ya encontrados.

II.4.3. Diagrama de clases

Forma parte de la vista estática del sistema. En el diagrama de clases, será donde se definirán las características de cada una de las clases, interfaces, colaboraciones y relaciones de dependencia y generalización.

- La Clase.- Representada por un rectángulo que dispone de tres apartados, el primero para indicar el nombre, el segundo para los atributos y el tercero para los métodos. Cada clase debe tener un nombre único, que las diferencie de las otras. Un atributo representa alguna propiedad de la clase que se encuentra en todas las instancias de la clase. Los atributos pueden representarse sólo mostrando su nombre y su tipo, e incluso su valor por defecto. Un método u operación es la implantación de un servicio de la clase, que muestra un comportamiento común a todos los objetos. En resumen es una función que le indica a las instancias de la clase que hagan algo.

- Relaciones entre clases.- Existen tres relaciones diferentes entre clases, dependencias, generalización y asociación. En las relaciones se habla de una clase destino y de una clase origen. Representadas así por una flecha. Las relaciones se pueden modificar con estereotipos o con restricciones.

  ■ Dependencias

  Es una relación de uso, es decir una clase usa a otra, que la necesita para su cometido. Se representa con una flecha discontinua va desde la clase utilizadora a la clase utilizada. Con la dependencia mostramos que un cambio en la clase utilizada puede afectar al funcionamiento de la clase utilizadora, pero no al contrario. Aunque las dependencias se pueden crear tal cual, es decir sin ningún estereotipo UML permite dar más significado a las dependencias, es decir concretar más, mediante el uso de estereotipos.

- Estereotipos de relación Clase-objeto

  - Bind: La clase utilizada es una plantilla, y necesita de parámetros para ser utilizada, con Bind se indica que la clase se instancia con los parámetros pasándole datos reales para sus parámetros.
  - Derive: Se utiliza al indicar relaciones entre dos atributos, indica que el valor de un atributo depende directamente del valor de otro.
  - Friend: Especifica una visibilidad especial sobre la clase relacionada. Es decir podrá ver las interioridades de la clase destino.
  - InstanceOF: Indica que el objeto origen es una instancia del destino
  - Instantiate: Indica que el origen crea instancias del destino
  - Powertype: indica que el destino es un contenedor de objetos del origen, o de sus hijos
  - Refine: Se utiliza para indicar que una clase es la misma que otra, pero más refinada, es decir dos vistas de la misma clase, la destino con mayor detalle.

- Generalización

Es la herencia, donde tenemos una o varias clases padre o superclase o madre, y una clase hija o subclase. UML también nos permite modificar la relación de Generalización con un estereotipo y dos restricciones.

- Estereotipo de generalización
  - Implementation: El hijo hereda la implantación del padre, sin publicar ni soportar sus interfaces

- Restricciones de generalización
  - Complete: La generalización ya no permite más hijos
  - Incomplete: Podemos incorporar más hijos a la generalización
  - Disjoint: solo puede tener un tipo de tiempo de ejecución, una instancia del padre sólo podrá ser de un tipo de hijo
  - Overlapping: puede cambiar de tipo durante su vida, una instancia del padre puede ir cambiando de tipo entre los de sus hijos

- Asociación

Especifica que los objetos de una clase están relacionados con los elementos de otra clase. Se representa mediante una línea continua, que une las dos clases. Se puede indicar el nombre, multiplicidad en los extremos, su rol y agregación

II.4.4. Diagrama de objetos

Forma parte de la vista estática del sistema. En este diagrama se modelan las instancias de las clases del diagrama de clases. Muestra a los objetos y sus relaciones, pero en un momento concreto del sistema. Estos diagramas contienen objetos y enlaces. En los diagramas de objetos también se pueden incorporar clases, para mostrar la clase de la que es un objeto representado. En este diagrama se muestra un estado del diagrama de eventos. Para realizar el diagrama de objetos primero se debe decidir que situación queremos representar del sistema.

II.4.5. Diagrama de componentes

Se utilizan para modelar la vista estática de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema.

En él se situarán bibliotecas, tablas, archivos, ejecutables y documentos que formen parte del sistema. Uno de los usos principales es que puede servir para ver qué componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema.

Todo objeto UML puede ser modificado mediante estereotipos, los estándar, que define UML son:

- Executable
- Library
- Table
- File
- Document

II.4.6. Diagramas de despliegue

En éste se indica la situación física de los componentes lógicos desarrollados. Es decir se sitúa el software en el hardware que lo contiene. Cada Hardware se representa como un nodo. Un nodo se representa como un cubo, un nodo es un elemento donde se ejecutan los componentes y representan el despliegue físico de estos componentes. Como los componentes pueden residir en más de un nodo se puede situar el componente de forma independiente, sin que pertenezca a ningún nodo, y relacionarlo con los nodos en los que se sitúa.

II.4.7. Diagrama de secuencia

El diagrama de secuencia forma parte del modelado dinámico del sistema. Se modelan las llamadas entre clases desde un punto concreto del sistema. Es útil para observar la vida de los objetos en sistema, identificar llamadas a realizar o posibles errores del modelado estático, que imposibiliten el flujo de información o de llamadas entre los componentes del sistema. En el diagrama de secuencia se muestra el orden de las llamadas en el sistema. Se utiliza un diagrama para cada llamada a representar

Es imposible representar en un solo diagrama de secuencia todas las secuencias posibles del sistema, por ello se escoge un punto de partida. El diagrama se forma con los objetos que forman parte de la secuencia, estos se sitúan en la parte superior de la pantalla, normalmente en la izquierda se sitúa al que inicia la acción. De estos objetos sale una línea que indica su vida en el sistema. Esta línea simple se convierte en una línea gruesa cuando representa que el objeto tiene el foco del sistema, es decir cuando él está activo.

II.5. Sistemas Manejadores de Bases de Datos Relacionales

Los sistemas manejadores de bases de datos son los programas que nos sirven de interfaz entre los datos de bajo nivel almacenados en la BD y los programas de aplicación:

- Interacción con el manejador de archivos
- Implantación de la integridad
- Puesta en práctica de la seguridad
- Respaldo y recuperación
- Control de concurrencia

La mayoría de las aplicaciones Cliente-Servidor funcionan bajo una arquitectura de dos capas en lenguajes de cuarta generación. Estas aplicaciones son bifurcadas en las siguientes capas: El llamado front-end (la interfaz del usuario, llamadas a SQL, aplicación de escritorio, etcétera) y el llamado Back-end (servidor de Bases de datos SQL, Sistema operativo multitareas, etc.).

El proceso front-end se desarrolla en algún lenguaje de 4ª generación (4GL). Se llama front-end dado que es la capa en donde el usuario interactúa con su PC. El proceso back-end es el servidor de bases de datos. Se llama así dado que típicamente reside en un servidor central en un entorno controlado.

Uno de los problemas es la dificultad de manipular los cambios en el front-end. Es decir, ¿Qué pasa si desea alterar las consultas SQL dado que una columna ha sido añadida a la tabla de Base de datos? ¿Y qué cuando los más antiguos distribuidores ahora serán marcados como inactivos (no eliminados) de la base de datos?

En estos casos, varios (a veces decenas, tal vez cientos o miles) de estaciones de trabajo clientes necesitarán ser actualizadas con una nueva versión del front-end simultáneamente al cambio en la base de datos. Este no es un cambio sencillo, sobre todo si las aplicaciones cliente están geográficamente dispersas.

La seguridad puede ser establecida por el back-end del servidor de bases de datos o por la aplicación que sirve de front-end. Cada una tiene sus limitaciones, el primero consiste en dar privilegios a los objetos de la base de datos y a los usuarios. Sin embargo, las empresas no requieren sólo asegurar cuales datos pueden ser actualizados o accedidos, sino cómo. En cuanto al segundo punto, que es el más usado, aunque el usuario puede acceder a la base de datos con su identificación, tiene dos problemas:

1. Dado que ninguno de los objetos en la base de datos es segura, cualquier usuario pudiese tener acceso total a la misma con alguna otra herramienta de front-end (Como Excel, Access, etcétera.).
2. La implantación de la seguridad deberá ser desarrollada, probada y mantenida en absolutamente toda la red (no importa dónde se encuentren las estaciones cliente).

II.5.1. Front-End

Este tema se refiere a la visualización que tiene el usuario con el sistema.

A continuación se presenta un estudio de las opciones viables según los requerimientos del sistema para desarrollar lo que se conoce como Front-End, es decir el ambiente en el que el usuario final va a trabajar e interactuar con la maquinaria de bases de datos que lo respaldan.

Sybase Power Builder

Esta es una herramienta orientada a objetos que soporta todas las características gráficas de Windows, ofrece una conectividad rápida y sencilla con las bases de datos remotas y para el manejo de datos en el lenguaje SQL casi en su totalidad.

Power Builder 6.0 forma parte de la familia de Sybase Powersoft. Estas son algunas de sus características:

Soporte a desarrollo de componentes para aplicaciones multi-hilo distribuidas.

- MTS ("Soporte a Monitores de transacciones") de Microsoft y Jaguar CTS de Powersoft
- Capacidad de generar clientes ultradelgados para el Web, mediante HTML dinámico
- Depurador instantáneo con innovadora interfaz de usuario, agrega nuevas opciones que permite poner a punto la operación y el desempeño de las aplicaciones
- Extiende el soporte sobre plataformas UNIX incluyendo ahora IBM AIX y HP-UX que junto con las existentes (Win95f, WinNT, MacOS y Sun Solaris) la reafirman como una tecnología abierta.
- Nueva versión habilitada con códigos únicos para la creación de aplicaciones para múltiples idiomas.
- Soporta los diferentes movimientos del ratón, para mejorar la navegación dentro de las ventanas.
- Implanta barras de herramientas en el estilo de Microsoft Office 97.

También está orientado al desarrollo de aplicaciones cliente-servidor, por lo que cuenta también con múltiples interfaces para bases de datos como SQL Server, Oracle, Informix, Sybase, etc.

Tiene asistentes para formas web para el cliente, consulta de bases de datos y conjunto de resultados. Y generación de HTML en tiempo de ejecución al igual que un motor de JavaScript para mover datos hacia el web.

Delphi 5.0

Delphi es una herramienta de desarrollo que combina los beneficios de un RAD Rapid Application Development (ambiente de diseño visual) con un poderoso compilador nativo y el acceso escalable a bases de datos.

- Lenguaje estructurado orientado a objetos.- Delphi utiliza un lenguaje estructurado orientado a objetos: Object Pascal. El lenguaje provee la facilidad de programación de un lenguaje de alto nivel 4GL, el alto desempeño y el poderío de un lenguaje 3GL. Delphi soporta conceptos avanzados de programación tales como encapsulamiento, herencia, polimorfismo y manejo de excepciones.

- Interrelación con diferentes bases de datos.- Uno de los aspectos más destacados lo constituyen los componentes que Borland ha incluido en Delphi para el desarrollo de aplicaciones completas de bases de datos. No se está limitado a un formato de datos determinado, sino que se tiene acceso a cincuenta formatos de datos diferentes a través de controladores suministrados por terceros (IDAPI Integrated Database Application Programming Interface (Interfaz programada de aplicación de la base de datos integrada), y ODBC). Entre estos se encuentran todos los estándares importantes de bases de datos en el área del PC como Xbase. Parados, Access, etc.

- Generador de interfaces visuales.- Permite crear rápidamente aplicaciones de forma visual seleccionando los componentes de una aplicación, lo que facilita a los desarrolladores la creación de la interfaz con el usuario.

- Debugger (depurador) gráfico. Posee un poderoso debugger gráfico que permite localizar y corregir errores en el código. El programador puede poner puntos de ruptura, examinar y cambiar variables, recorres el código paso a paso y entender exactamente el comportamiento del programa.

- Escalabilidad.- También es posible acceder de forma muy cómoda a servidores de bases de datos de otros sistemas (por ejemplo UNIX) por medio del SQL que constituye un estándar de lenguaje de uso general para consultar y modificar datos administrados por los servidores especiales de bases de datos como Oracle Sybase, Informix.

Visual Basic 6.0

Visual Basic es una de las herramientas de desarrollo de Microsoft caracterizada porque es un lenguaje interactivo que permite su aprendizaje rápido.

- Sistema orientado para crear soluciones en ambiente Windows
- Permite el desarrollo en ambiente cliente-servidor
- Permite el uso de otras aplicaciones de Office para usarse como un componente en aplicaciones propias
- Compila aplicaciones en formato DLL para reutilización de componentes.
- Tiene la posibilidad de incorporar controles preconstruidos por terceros
- Cuenta con ayuda en línea robusta.
- Incluye motor de datos vía DAO Data Access Object (Objeto de acceso de datos)
- Tiene gran capacidad y velocidad en su depurador

Este lenguaje ha crecido conforme han ido apareciendo las nuevas versiones. Se orienta sobre todo al desarrollo empresarial y de redes. Además incluye diversas utilerías y herramientas para la programación.

Se pueden generar DDE Dynamic Data Exchange (intercambio de datos dinámico) para aplicaciones basadas en ventanas; y permite también el uso de objetos OLE.

II.5.2. Back-End

El back-end está formado por el sistema manejador de la base de datos, es decir la herramienta con la que se va a manejar toda la información para obtenerla de manera conveniente y eficientemente

Ahora bien, algunos de los manejadores que se analizaron son:

ORACLE

- Administración de grandes volúmenes de datos.
- Contiene mecanismos que controlan el acceso a los datos.

Oracle proporciona ciertos procedimientos de realización de copias de seguridad y recuperación de los datos, las instrucciones SQL estándar se admiten en las bases de datos Oracle

Configuración: Cliente/servidor.- Los usuarios acceden a la base de datos desde su computadora personal (cliente) a través de una red, y la base de datos se encuentra en una computadora diferente (servidor)

La opción de consulta en paralelo (Parallel Query) permite sacar partido del procesamiento de consultas en computadoras dotadas de más de una unidad central de proceso (procesador). En las máquinas con un solo procesador, así como en aquellas con múltiples procesadores sin la opción de consulta en paralelo, sólo un proceso puede acceder a la base de datos y mostrar los datos que responden al criterio de selección definido.

Procedimientos almacenados.- Se pueden almacenar programa (o segmentos de código) en la base de datos Oracle, para realizar funciones de importancia para nuestro sistema.

Disparadores de base de datos (triggers).- son segmentos de código almacenados en la base de datos, y que se disparan como respuesta a sucesos que tienen lugar en las aplicaciones.

El servidor Oracle, al que normalmente se denomina motor de la base de datos, funciona con un amplio conjunto de herramientas de desarrollo, herramientas de consulta para usuario final, aplicaciones comerciales y herramientas de gestión de la información de ámbito corporativo.

Visual FoxPro

Es un entorno de desarrollo para escribir aplicaciones de bases de datos. Proviene de la generación xBASE de lenguajes de programación, que incluye dBASE II y III, Clipper, FoxBASE y FoxPro, entre otros.

La característica principal de las tablas xBASE es que contienen información que describe el archivo en los primeros caracteres del archivos mismo. Cuando un programa de FoxPro abre una tabla, lee el encabezado para averiguar qué hay en el archivo. Una vez que una tabla ha sido definida, simplemente se abre y se usa.

Otra característica es que en estos lenguajes, son interpretados, es decir la ventana de comandos está siempre disponible, o puede ser llamada con una sola tecla, de tal manera que se pueden teclear comandos y ver qué hacen.

Las ventanas de seguimiento y depuración permiten ejecutar el código línea por línea y observar los valores vigentes de las variables o los campos de las tablas; adicionalmente, se puede hacer una pausa en el programa, no sólo en una línea específica, sino también cuando se cumpla cierta condición.

También usa ahora verdaderas clases con herencia, encapsulamiento y polimorfismo. Cada clase tiene propiedades, eventos y métodos. Los desarrolladores pueden tener cualquiera de estas clases base y desarrollar nuevas clases. Se pueden guardar estas clases en librerías de clases y usarlas en esas aplicaciones, reduciendo la necesidad de volver a desarrollar y probar código.

SQL Server

SQL Server se implantó como un servicio MS SQL en Windows NT. Este servicio es el motor del núcleo de las bases de datos y, de todos los componentes de paquete SQL server, es el único capaz de modificar datos. SQL Server protege los datos de valor definiendo las acciones que pueden efectuar los clientes (a través de permisos). Entre éstas se incluyen las siguientes:

- Utilizar las reglas comerciales definidas a través de disparadores y procesos de almacenamiento.
- Evitar que dos usuarios accedan a la vez al mismo dato.
- Obligar que los datos que se encuentran almacenados en ubicaciones distintas conserven cierta consistencia.

Agente SQL Server

El Agente SQL Server (también conocido como SQL Executive) es el encargado de suministrar servicios de programación a SQL Server. Puede ejecutar trabajos, es decir, dar una serie de pasos cuando se le indique. Un paso puede ser un sencillo comando de sistema operativo o un script con un lenguaje de programación como Visual Basic.

Este agente puede trabajar con los envíos de avisos a los operadores. Por ejemplo, si ocurre un error serio mientras se intenta la recepción de cierta información procedente de una base de datos, el Agente SQL Server enviará automáticamente mensajes de correo electrónico al buscapersonas del operador para avisarle de la existencia de cualquier error.

SQL Server es un buen ejemplo de un sistema “autoportante”: controla las tablas de datos según las reglas que ellos mismos establecen a través de las tablas de datos. Cuando instala el software de SQL Server, automáticamente se crean cuatro bases de datos conocidas como bases de datos del sistema. Estas bases de datos son las siguientes:

MASTER.- Como puede sugerir el nombre, la base de datos master es la encargada de guardas la información que se utiliza con la mayoría de las operaciones básicas de SQL Server. En esta base de datos se encuentra la información sobre las cuentas de los usuarios y la configuración del sistema. Asimismo, tiene información sobre dónde localizar las bases de datos que crean los clientes. Dada la importancia de esta base conviene hacer periódicamente copias de seguridad.

MODEL.- Es la única de las cuatro que se pueden modificar con ciertos sucesos. Siempre que se crea una base de datos, SQL Server inicia una copia de model. Si quiere que algún elemento aparezca en todas las bases de datos de su servidor, tendrá que añadirlo a la base de datos model para que cada vez que se haga una copia de ella, se incluya dicho elemento.

TEMPDB.- En ella se almacenan todas las tablas temporales que se generan durante la ejecución de los procesos. Esta base se crea automáticamente cada vez que se inicia SQL Server y no es necesario hacer ninguna modificación en ella.

MSDB.- Se utiliza para que Agente SQL Server guarde la información que necesita para procesar trabajos y alertas.

Una base de datos SQL Server es un contenedor en el que se guardan varios tipos de objetos SQL Server:

- Tablas
- Vistas
- Procedimientos de almacenamiento
- Opciones predeterminadas
- Reglas
- Disparadores
- Índices Tipos de datos
- Limitaciones
- Diagramas

Una de las formas que tiene SQL Server para proteger los datos es a través de registros de transacciones, bases de datos especiales que se utilizan para registrar la actividad de un cliente.

Cuando se crea una base de datos, SQL Server crea automáticamente un registro de transacciones para ella. Este registro es un archivo especial en el que se guarda toda la actividad relacionada con dicha base de datos. Siempre que un usuario agregue, elimine o modifique la información de la base de datos, el registro de transacciones tomará nota de la

acción. Por defecto, estos registros tienen el mismo nombre que la base de datos, pero utilizan otra extensión .

El registro de transacciones proporciona cierto nivel de seguridad a los datos de SQL Server. En caso de fallo, es posible aplicar las transacciones que se encuentran en esta archivo en otra copia de la base de datos.

Administrando

La administración de las bases de datos es un trabajo que han de efectuar tanto los administradores como los desarrolladores de éstas. Según se van diseñando aplicaciones que utilizan SQL Server, tendrá que crear y mantener bases de datos. También tendrá que deshacerse de aquellas que se hayan quedado obsoletas.

SQL Server proporciona un entorno de administración que es realmente flexible. En general, a través de las herramientas de SQL Server hay varias formas de administrar las bases de datos.

- Ejecutando los asistentes del Administrador Corporativo de SQL Server
- Ejecutando declaraciones SQL con la herramienta Analizador de consultas
- A través de los menús, íconos y cuadros de diálogo que se encuentran dentro del Administrador corporativo SQL.
- Ejecutando los asistentes desde una aplicación que utilice objetos SQL-NS

Uno de los trabajos más importantes que tiene que hacer un administrador de bases de datos es gestionar los registros que generan las bases de datos. Aunque la cantidad de información de la base permanezca más o menos constante, si se modifica cualquier operación que modifique datos añade una fila al registro que no se borrará nunca.

Capítulo III

Análisis del sistema

III.1. Planteamiento del problema

El problema en general radica en que la empresa necesita que sus libros contables puedan manejarse de una manera más eficiente así como sencilla y con una cantidad mínima o de ser posible nula de errores manuales.

El primer paso es poder cubrir toda la información que manejan los libros contables, esta información, aunque ya está clasificada, no siempre tiene una organización óptima por la que se puede caer en repeticiones en los datos, así como la pérdida de alguno de ellos y de consultas con datos erróneos.

Para evitar todos esos obstáculos para el manejo de la información, es necesario el uso de una base de datos eficazmente manejada, es decir utilizar un sistema manejador de bases de datos con el que se pueda obtener información de forma conveniente.

Dentro de los sistemas manejadores de bases de datos DBMS (por sus siglas en inglés) podemos encontrar:

Oracle, Visual Fox Pro, SQL Server

Otro de los obstáculos a vencer es que la empresa necesita que su contador general pueda hacer modificaciones a la información almacenada, que otros usuarios dentro de la misma empresa puedan tener acceso a la información en forma de reportes, dado que estos son fáciles de interpretar, y por último que desde otro sistema u otra parte del sistema completo puedan tener acceso a la información para poder extraerla y manejarla a su conveniencia de forma externa a este sistema.

Para llevar a cabo la tarea de que usuarios externos al diseño de la base de datos y al administrador del sistema pueda tener acceso a la base es necesario contar con un medio fácil de manejar.

En otras palabras estamos en la etapa en la que tenemos la información muy bien almacenada y lista para ser utilizada, pero para el usuario la forma en la que tengamos la información almacenada, así como la manera en que tengamos acceso a ella es transparente, y esto es, porque el usuario final necesita tener una forma de conectarse a la base de datos sin tener los conocimientos necesarios para ello. Con el propósito de que el usuario pueda tener acceso a la información fácil y rápidamente es que necesitamos de una interfaz por medio de la cual haya una interacción con él.

Esta interfaz está compuesta por una serie de pantallas por medio de las cuales el usuario con una pequeña introducción al sistema, y con conocimientos básicos de computación será capaz de interactuar con la base de datos.

Para la elaboración de dicha interfaz pueden utilizarse las siguientes herramientas:

Sybase Power Builder, Delphi y Visual Basic

III.2. Análisis de la información

Para poder efectuar un análisis completo y correcto de la información con la que contamos, utilizaremos UML como lenguaje modelador.

Obteniendo de esta manera el siguiente análisis:

III.2.1. Casos de uso

Basándose en los conceptos ya vistos en el capítulo anterior se pueden describir los componentes de los casos de uso como sigue:

.

III.2.1.1 Actores del sistema

- Contador general
- Sistema externo que maneja ventas y servicios.
- Administrador del sistema.
- Otros usuarios.

III.2.1.2 Acciones de los actores

- Sistema externo de ventas y servicios:
  Ingresar pólizas correspondientes a los trabajos y/o ventas realizadas

- Contador general:
  Crear cuentas contables
  Modificar el contenido de las cuentas (características)
  Eliminar cuentas contables
  Revisar y modificar las pólizas ingresadas, así como manipular los saldos de las cuentas afectadas por las mismas

- Otros usuarios
  Consultar cuentas y pólizas
  Verificar datos y saldos de cuentas y pólizas
  Imprimir reportes sobre reportes

III.2.1.3 Descripción de los casos de uso

- Identificar usuario (efectuado por todos los actores al entrar al sistema)
  Dependiendo del tipo de usuario, recibirá los privilegios para manejar el sistema, es decir le permitirá efectuar diferentes actividades dependiendo del nivel de seguridad en el que se encuentre el usuario.

- Ingresar pólizas (únicamente realizado por el sistema externo)
  Se toman los datos desde un archivo de texto totalmente ajeno al sistema, con la información de la póliza a incorporar.

- Crear cuentas contables (contador general)

Desde la pantalla de cuentas contables ingresa a la opción de altas, introduciendo los datos requeridos.

- Modificar el contenido de las cuentas (contador general)

Escoge la opción de editar, donde el sistema permite ciertos cambios, dependiendo de las características propias de la cuenta.

- Eliminar cuentas contables (contador general)

Desde la opción de bajas se verificará con la naturaleza de la cuenta, así como de sus dependencias para el permiso de eliminarla.

- Revisar y modificar las pólizas ingresadas (contador general)

A través del submenú contabilización diaria y del recálculo de pólizas, el actor decide el siguiente paso de las pólizas seleccionadas.

- Consultas de cuentas, pólizas y reportes (todos)

Dichas consultas se realizan desde diferentes puntos del sistema, que consiste simplemente en visualizar los reportes requeridos.

Diagrama jerárquico funcional

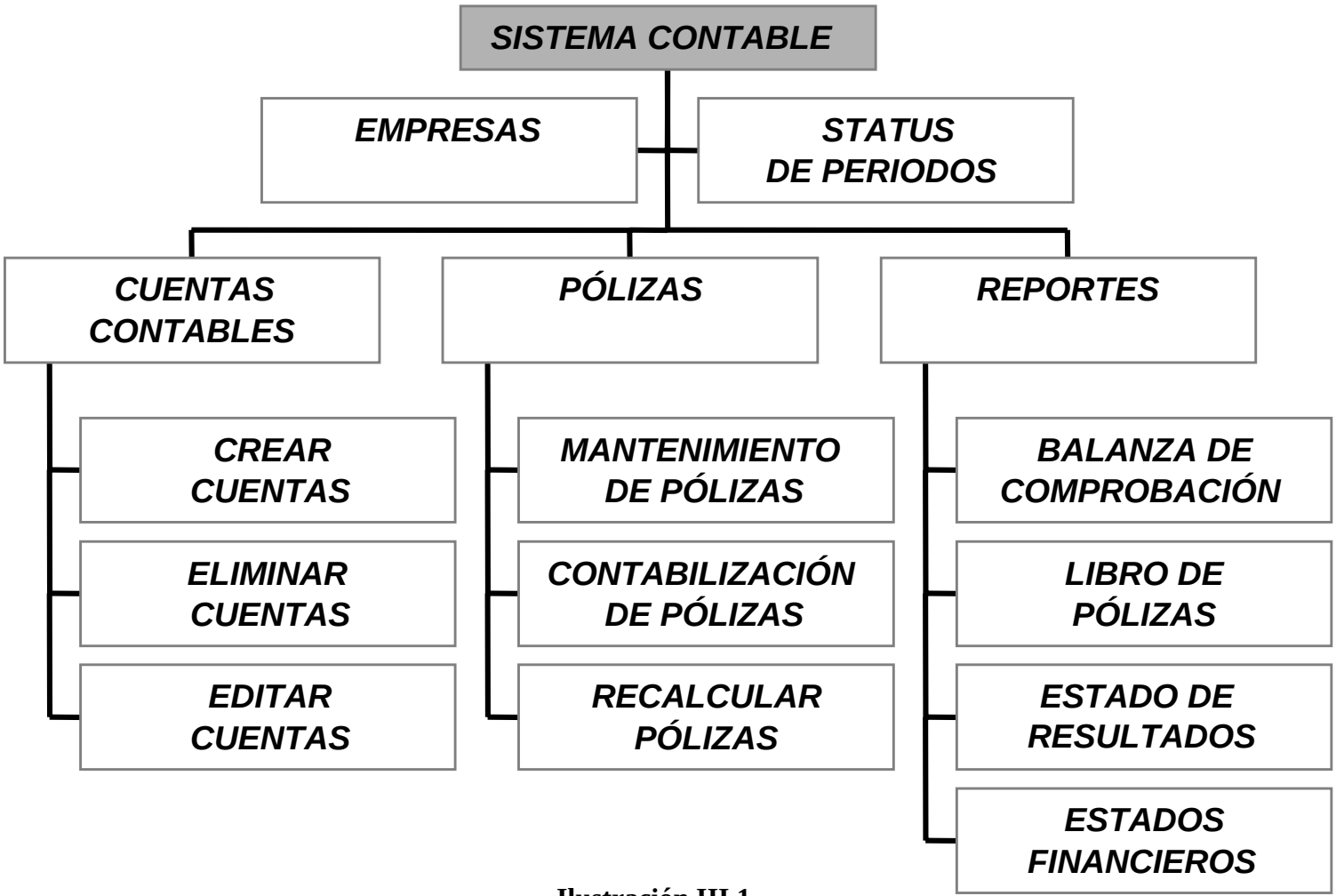


Ilustración III.1

Diagrama de casos de uso

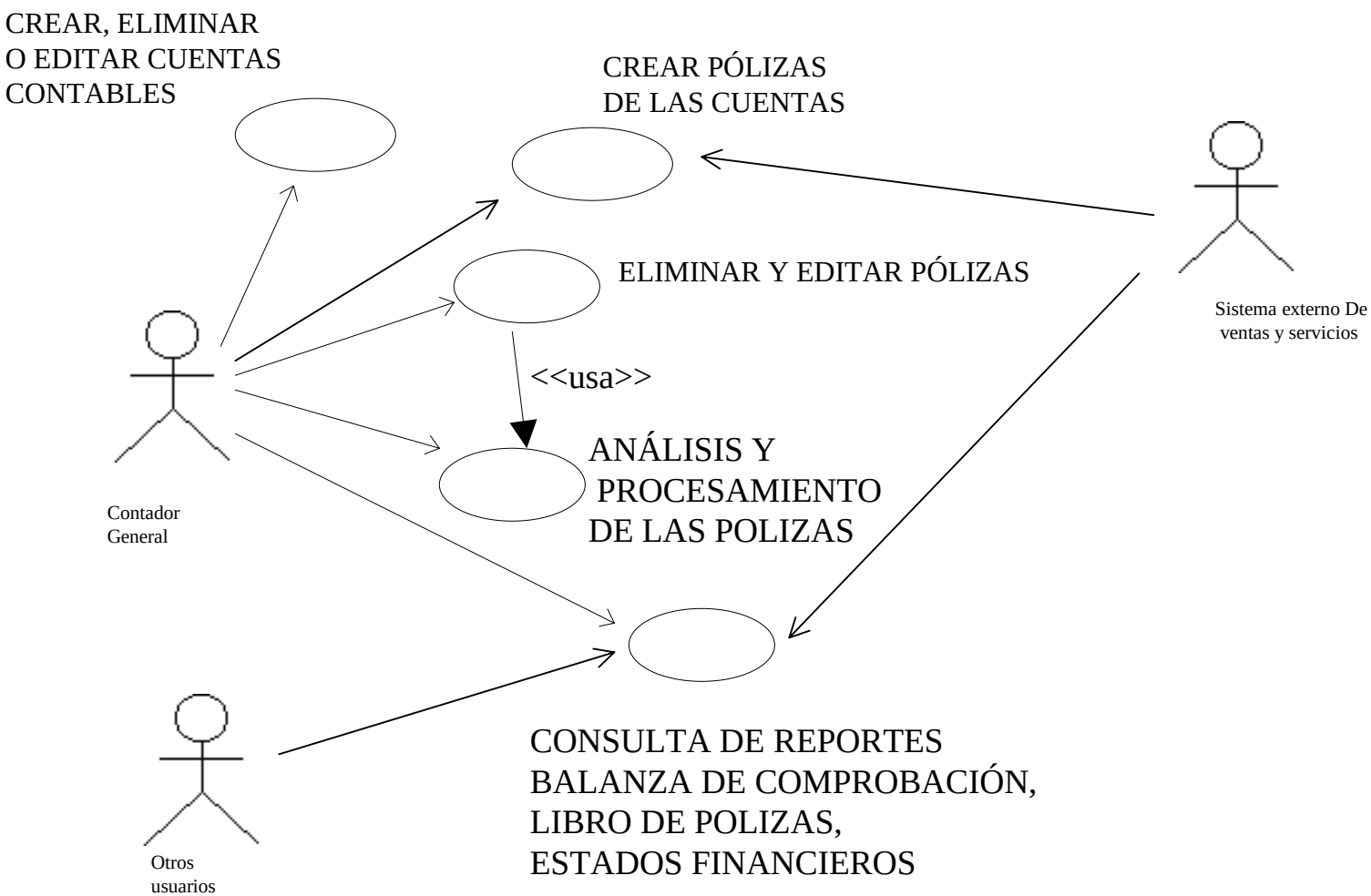


Ilustración III.2

Diagrama de secuencia

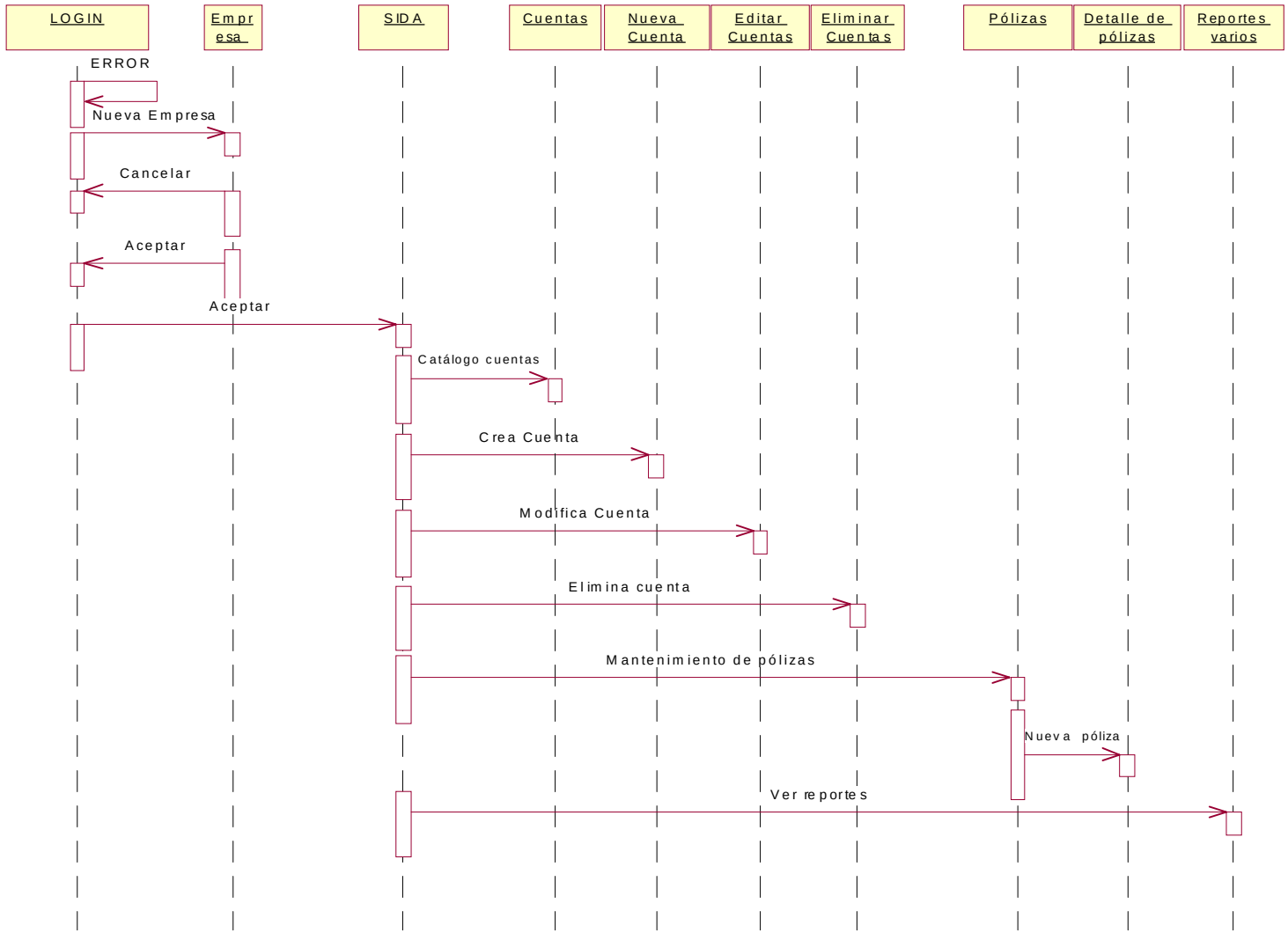


Ilustración III.3

Diagrama de clases

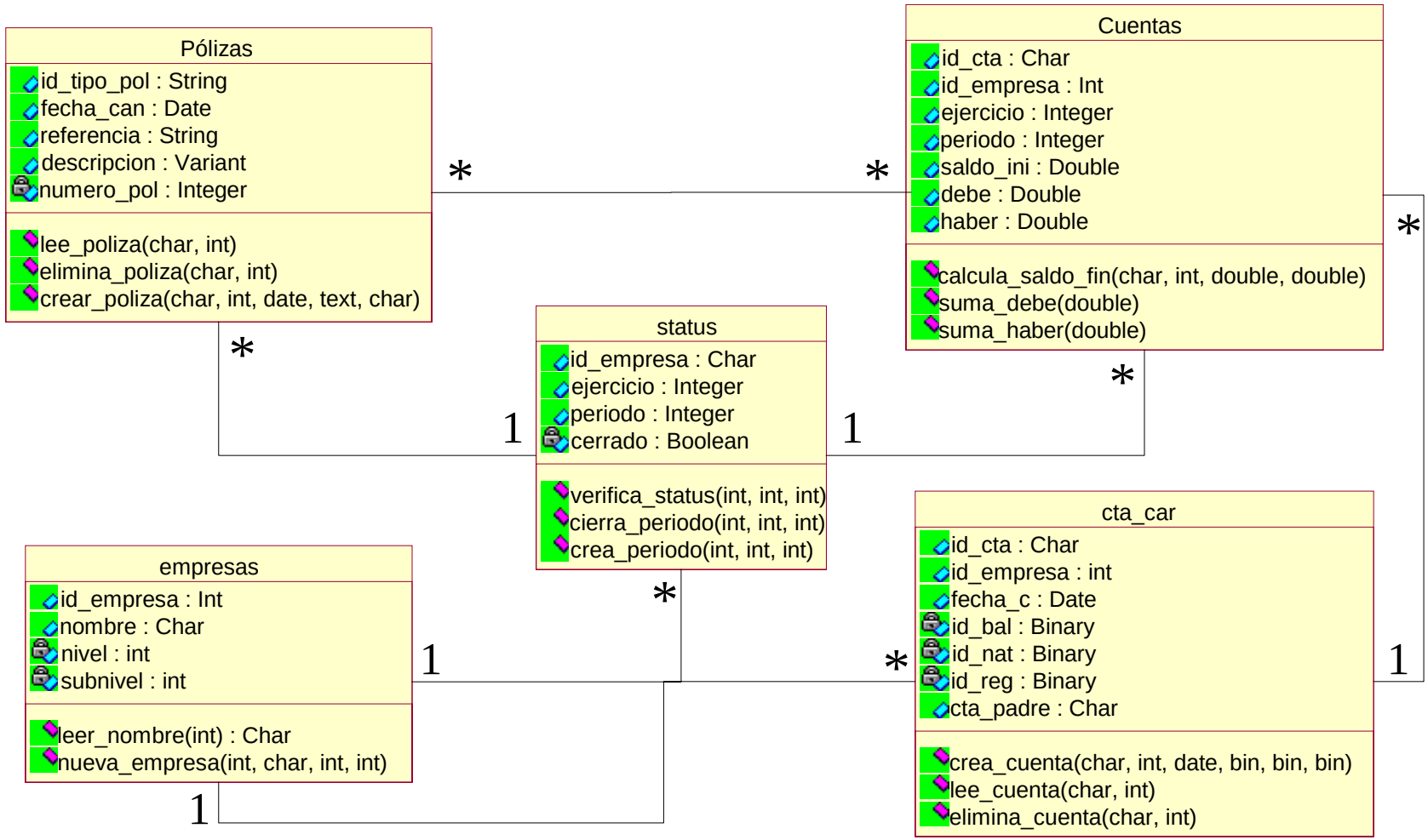


Ilustración III.4

Diagrama entidad-relación

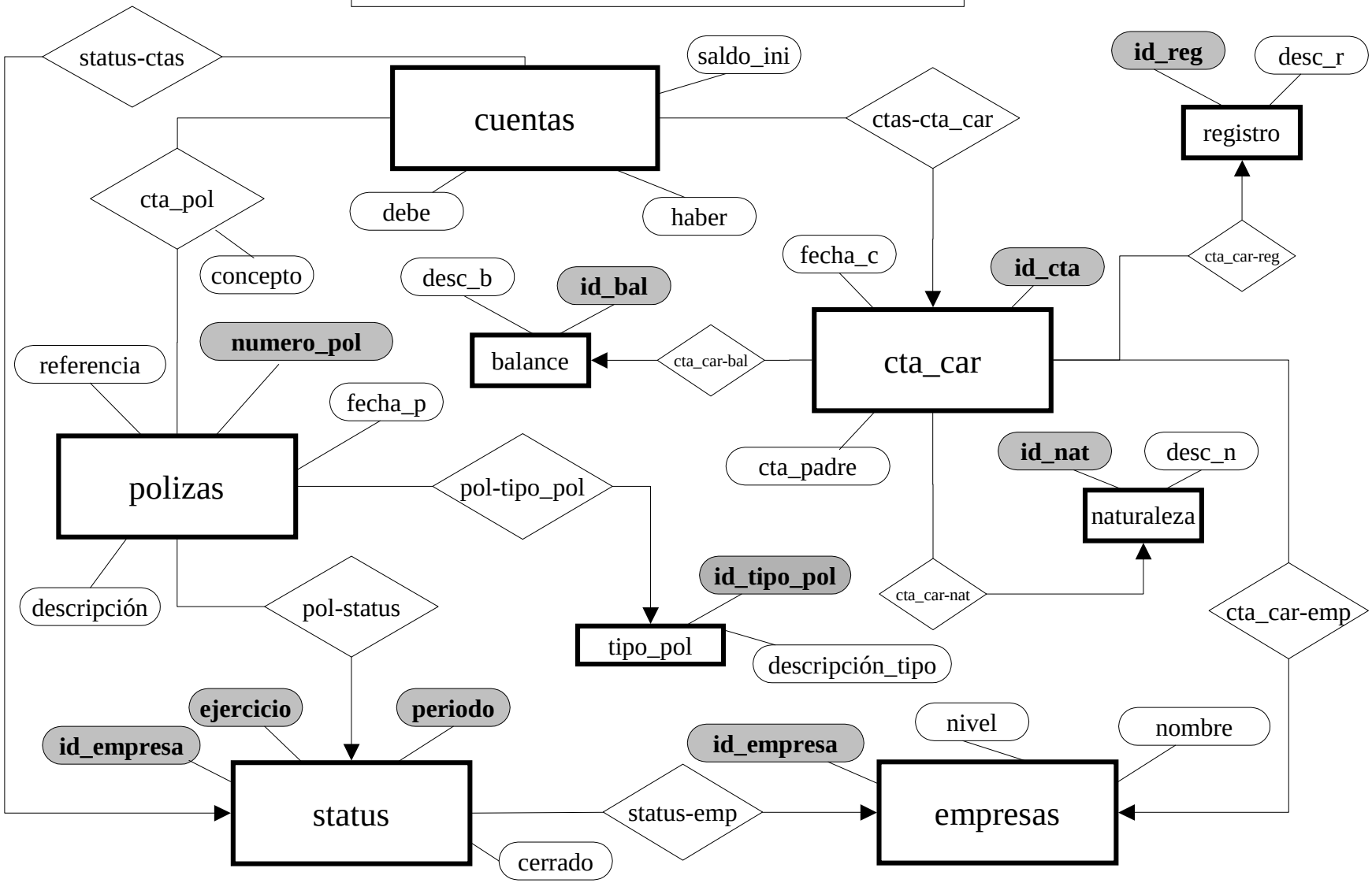


Ilustración III.5

III.3. Propuesta de solución

Una propuesta factible debe construirse a través de decisiones importantes sobre varias opciones, estas permiten escoger la herramienta correcta para desarrollar el front-end y el back-end adecuado.

Cabe señalar que la ponderación utilizada en las tablas III.1 y III.2 deben leerse como el número uno con menor afinidad con la característica en cuestión y el número tres con el de mayor.

La comparación de las herramientas a comparar para el desarrollo de front-end se encuentran en la Tabla III.1

CARACTERÍSTICAS	SYBASE POWER BUILDER	DELPHI	VISUAL BASIC
Conectividad a bases de datos remotas	**3**	**3**	**3**
Contiene una ayuda en línea robusta	**2**	**2**	**3**
Posibilidad de desarrollo para formas web	**3**	**2**	**3**
Contiene depurador gráfico	**3**	**3**	**3**
Popularidad	**1**	**2**	**3**
Facilidad en el aprendizaje del lenguaje	**1**	**2**	**3**

Tabla III.1

Y con respecto a las herramientas para poder desarrollar el back-end se podrán ver en la Tabla III.2

CARACTERÍSTICAS	ORACLE	VISUAL FOX PRO	SQL SERVER
Admite instrucciones SQL	**3**	**3**	**3**
Configuración cliente/servidor	**3**	**2**	**3**
Fácil manejo de objetos (tablas, diagramas, etc)	**1**	**2**	**3**
Popularidad	**1**	**1**	**3**
Fácil adquisición	**1**	**2**	**3**
Facilidad de aprendizaje y manejabilidad	**1**	**2**	**3**

Tabla III.2

Del análisis anterior se puede observar que Visual Basic y SQL Server obtuvieron la puntuación mayor, ahora bien dado que la empresa no cuenta con grandes recursos para la rápida adquisición de nuevas herramientas, que el ambiente con el que se le dará mantenimiento al sistema es Windows y que la empresa ya cuenta con las herramientas anteriores, se pueden tomas éstas como la selección más viable.

Por otro lado la herramienta para el front-end que quedó en segundo lugar es Delphi 5.0, sin embargo la facilidad de aprendizaje de la misma no es tan práctica como en Visual Basic.

Y la otra herramienta para el back-end que se podría tomar en cuenta es superada por SQL-Server en la configuración cliente-servidor y manejo de tablas y/o diagramas que son actividades esenciales para el desarrollo del sistema.

Por lo tanto las herramientas seleccionadas son Visual Basic y SQL-Server.

Capítulo IV

Diseño y desarrollo del sistema

IV.1. Diseño de la base de datos

El diseño de una base de datos es básicamente el convertir el Diagrama Entidad-Relación en una estructura de datos relacional. Las etapas que debe contener esta conversión son el diseño del esquema, su normalización y la obtención del diccionario de datos.

IV.1.1. Diseño del esquema

De forma conceptual se tiene que el esquema es la estructura lógica general de la base de datos, que se escribe utilizando un lenguaje de definición de datos, el DDL conceptual. Dicho esquema está formado por varias relaciones que cumplen ciertas reglas ya mencionadas en el capítulo II. Al definir el esquema, se obtiene una vista conceptual en la que se tendrá el contenido total de la base de datos.

Los esquemas de la base de datos se encuentran definidos de la siguiente manera:

Esquema_empresas (id_empresa, nombre, nivel, subnivel)

Esquema_status (id_empresa, ejercicio, periodo, cerrado)

Esquema_cuentas (id_cta, id_empresa, ejercicio, periodo, saldo ini, debe, haber)

Esquema_polizas (id_tipo_pol, numero_pol, fecha_can, referencia, descripcion, id_empresa, periodo, ejercicio)

Esquema_cta_car (id_cta, id_empresa, fecha:_c, id_balance, id_registro, id_naturaleza, cta_padre, desc_cta)

Esquema_cta_pol (id_cta, id_empresa, ejercicio, periodo, id_tipo_pol, numero_pol, concepto, debe, haber)

Esquema_tipo_pol (id_tipo_pol, descripcion_tipo)

Esquema_balance (id_balance, descripcion_b)

Esquema_registro (id_registro, descripcion_r)

Esquema_naturaleza (id_naturaleza, descripcion_n)

IV.1.2. Normalización

Durante el diseño de una base de datos, se puede caer en ciertos errores como:

- Repetición de la información.
- Incapacidad de representar la información.
- Pérdida de información.

Para evitar caer en estos errores es necesario llevar a cabo una normalización de las relaciones obtenidas a partir de los esquemas anteriores. Para entender la normalización es necesario mencionar como se define una dependencia funcional:

Dada una relación R, el atributo Y(R) depende funcionalmente del atributo X(R) sí y solo sí un solo valor de Y(R) está asociado a cada valor X(R) en cualquier momento dado. Los atributos X y Y pueden ser compuestos, lo que se denota:

$$R.x \rightarrow R.y$$

PRIMERA FORMA NORMAL

Una relación está en primera forma normal sí y solo sí todos los dominios simples subyacentes contienen solo valores atómicos.

SEGUNDA FORMA NORMAL

Una relación está en segunda forma normal sí y solo sí está en primera forma normal y todos los atributos clave dependen por completo de la llave primaria.

TERCERA FORMA NORMAL:

Una relación r está en tercera forma normal sí y solo sí está en segunda forma normal y todos los atributos no clave dependen en forma no transitiva de la llave primaria.

FORMA NORMAL DE BOYCE COOD

Caso especial de la tercera forma normal, entra en uso cuando hay varias llaves candidato, esas llaves candidato son compuestas o las llaves candidato se traslapan

Dependencia multivaluada: Para una relación R con los atributos A, B y C la dependencia multivaluada R (A) $\rightarrow\rightarrow$ B se cumple en R sí y solo sí el conjunto de valores en B corresponde a un par dado en R (valor de A, valor de C) y depende solo del valor de A y es independiente del valor de C.

Una relación está en cuarta forman normal sí y solo sí está en forman normal Boyce-Cood y no contiene dependencia funcionales multivaluadas.

IV.1.3. Diccionario de datos

El contenido del diccionario puede considerarse como “datos acerca de los datos”, es decir, definiciones de otros objetos en el sistema, y no sólo “datos en bruto”. En particular, en el diccionario de datos se almacenarán físicamente todos los diversos esquemas y correspondencias (externos, conceptuales, etc).

Un diccionario completo incluirá también referencias cruzadas para indicar, por ejemplo, cuáles programas usan cuáles partes de la base de datos, cuáles usuarios requieren cuáles informes, qué terminales están conectadas al sistema., etc.

RELACIONES

RELACIÓN	LLAVE PRIMARIA	DESCRIPCIÓN
empresas	id_empresa	incluye todas las empresas que pueden tener acceso al sistema
status	id_empresa, ejercicio, periodo	incluye la situación actual en la que se encuentra tal o cual periodo por empresa, es decir si está abierto o cerrado
cuentas	id_cta, id_empresa, ejercicio, periodo	contiene el debe y el haber de las cuentas de todas las empresas, por ejercicio y periodo
polizas	id_empresa, ejercicio, periodo, id_tipo_pol, numero_pol	contiene las características que diferencian a las pólizas como la fecha en la que se canceló, así como la descripción y el periodo, ejercicio y empresa al que pertenecen
cta_car	id_cta, id_empresa	contiene las características de todas las cuentas de las empresas
cta_pol	id_cta, id_empresa, ejercicio, periodo, id_tipo_pol, numero_pol	es la relación que surge entre las cuentas que son afectadas por tal o cual póliza
tipo_pol	id_tipo_pol	en esta relación se encuentran las identificaciones de los tipo de pólizas que se pueden generar por departamento
balance	id_balance	contiene una identificación para el tipo de cuenta ya sea de resultados o de balance
registro	id_registro	tiene una identificación para la clase de cuenta ya sea de acumulación o de registro
naturaleza	id_naturaleza	contiene la identificación para la naturaleza de la cuenta ya sea acreedora o deudora

ATRIBUTOS

NOMBRE	SE ENCUENTRA EN LAS RELACIONES	PERMITE VALORES NULOS	TIPO DE DATO	DESCRIPCIÓN
id_empresa	empresas, status (ll_f), cuentas (ll_f), cta_car (ll_f), cta_pol (ll_f), polizas (ll_f)	no	entero	es el identificador de las empresas compuesto por iniciales de la empresa y un número consecutivo referente a su orden en el que se dió de alta en el sistema
nombre	empresas	no	caracteres (20)	es el nombre completo de la empresa
nivel	empresas	no	entero	es el número de niveles que la empresa maneja en sus cuentas
subnivel	empresas	no	entero	es el número de dígitos por nivel permitido
ejercicio	status, cuentas (ll_f), cta_pol (ll_f), polizas (ll_f)	no	entero	se refiere al año
periodo	status, cuentas (ll_f), cta_pol (ll_f), polizas (ll_f)	no	entero	se refiere al número del mes
cerrado	status	no	bit	con '0' indica que el periodo en ese ejercicio está cerrado con '1' indica que el periodo en ese ejercicio está abierto
saldo_ini	cuentas	si	float	se refiere al saldo con el que inicia la cuenta en un periodo y ejercicio determinado
debe	cuentas	si	float	se refiere al total de ‘debe’ en el periodo para la cuenta
haber	cuentas	si	float	se refiere al total de ‘haber’ en el periodo para la cuenta

numero_pol	polizas, cta_pol (ll_f)	no	entero	este número se crea al generar la póliza en su respectivo departamento
fecha_can	polizas	si	fecha	se refiere a la fecha en la que fue cancelada dicha póliza
referencia	polizas	no	caracteres(10)	clave creada en el momento de ser generada
descripcion	polizas	no	varchar	breve explicación sobre la finalidad que tiene cada póliza
id_cta	cta_car, cuentas (ll_f), cta_pol (ll_f)	no	caracteres (29)	identificador para cada una de las cuentas
fecha	cta_car	no	fecha	fecha en la que se creó la cuenta
cta_padre	cta_car	si	caracteres (29)	es llave foránea de si misma, pues los datos que se encuentran en este atributo se obtienen del atributo id_cta
concepto	cta_pol	no	varchar	descripción en la que se especifica la modificación sobre una cuenta con base en una póliza
id_tipo_pol	tipo_pol, polizas (ll_f), cta_pol (ll_f)	no	caracteres (3)	clave o iniciales con las que se identifica cada tipo de póliza existente
descripcion_tipo	tipo_pol	no	varchar	nombre completo de cada tipo de pólizas
id_balance	balance, cta_car (ll_f)	no	binario	forma binaria de identificar el tipo de cuenta tipo: binario
descripcion_b	balance	no	caracteres (10)	forma binaria de identificar la naturaleza de la cuenta
id_registro	registro, cta_car (ll_f)	no	binario	forma binaria de identificar la clase de la cuenta
descripcion_r	registro	no	caracteres (10)	nos indica si se refiere a de acumulación o de registro

id_naturaleza	naturaleza, cta_car (ll_f)	no	binario	forma binaria de identificar la naturaleza de la cuenta
descripcion_n	naturaleza	no	caracteres (10)	nos indica si se refiere a acreedora o deudora

IV.2. Diseño y construcción del Back-End

La construcción del Back-End consiste en una estructura de sistema modular de fácil manipulación en la captura, búsqueda y actualización de la información. Considerando que el elemento más importante de cualquier sistema de información es necesario crear la propia base de datos, y a partir de este paso construir las tablas de la misma base, asignar los tipos de datos, así como las propiedades de cada una de las entidades, determinar las llaves primarias, foráneas y las relaciones entre las tablas.

Se creó una base de datos en SQL Server de nombre SIDA que significa: Sistema Integral De Agencias dentro del servidor seleccionado.

Las tablas por las que está formada la base de datos son las siguientes:

Tabla **empresas**

Columna	id_empresa	nombre	nivel	subnivel
Valores nulos	no	no	no	no
Tipo de dato	entero	caracteres (10)	entero	entero
Tipo de llave	primaria	--	--	--

Tabla **status**

Columna	id_empresa	ejercicio	periodo	cerrado
Valores nulos	no	no	no	no
Tipo de dato	entero	entero	entero	bit
Tipo de llave	primaria / foránea	primaria	primaria	--

Tabla **cuentas**

Columna	id_cta	id_empresa	ejercicio	periodo	saldo_ini	debe	haber
Valores nulos	no	no	no	no	si	si	si
Tipo de dato	caracteres (29)	entero	entero	entero	float	float	float
Tipo de llave	primaria / foránea	primaria / foránea	primaria	primaria	--	--	--

Tabla **polizas**

Columna	id_tipo_pol	numero_pol	id_empresa	periodo	ejercicio	fecha_can	descripcion	referencia
Valores nulos	no	no	no	no	no	si	no	no
Tipo de dato	caracteres (3)	entero	entero	entero	entero	fecha	varchar	caracteres (10)
Tipo de llave	primaria / foránea	primaria	primaria / foránea	primaria / foránea	primaria / foránea	--	--	--

Tabla **cta_car**

Columna	id_cta	id_empresa	fecha	id_balance	id_registro	id_naturaleza	cta_padre
Valores nulos	no	no	no	no	no	no	si
Tipo de dato	caracters (29)	entero	fecha	binario	binario	binario	caracteres (29)
Tipo de llave	primaria	primaria / foránea	--	--	--	--	--

Tabla **cta_pol**

Columna	id_cta	id_empresa	ejercicio	periodo	id_tipo_pol	numero_pol	concepto	debe	haber
Valores nulos	no	no	no	no	no	no	no	si	si
Tipo de dato	caracteres (29)	entero	entero	entero	caracteres (3)	entero	varchar	float	float
Tipo de llave	primaria / foránea	primaria / foránea	primaria / foránea	primaria / foránea	primaria / foránea	primaria / foránea	--	--	--

Tabla **tipo_pol**

Columna	id_tipo_pol	descripcion_tipo
Valores nulos	no	no
Tipo de dato	caracteres (3)	varchar
Tipo de llave	primaria	--

Tabla **balance**

Columna	id_balance	descripcion_b
Valores nulos	no	no
Tipo de dato	binario	caracteres (10)
Tipo de llave	primaria	--

Tabla **registro**

Columna	id_registro	descripcion_r
Valores nulos	no	no
Tipo de dato	binario	caracteres (10)
Tipo de llave	primaria	--

Tabla **naturaleza**

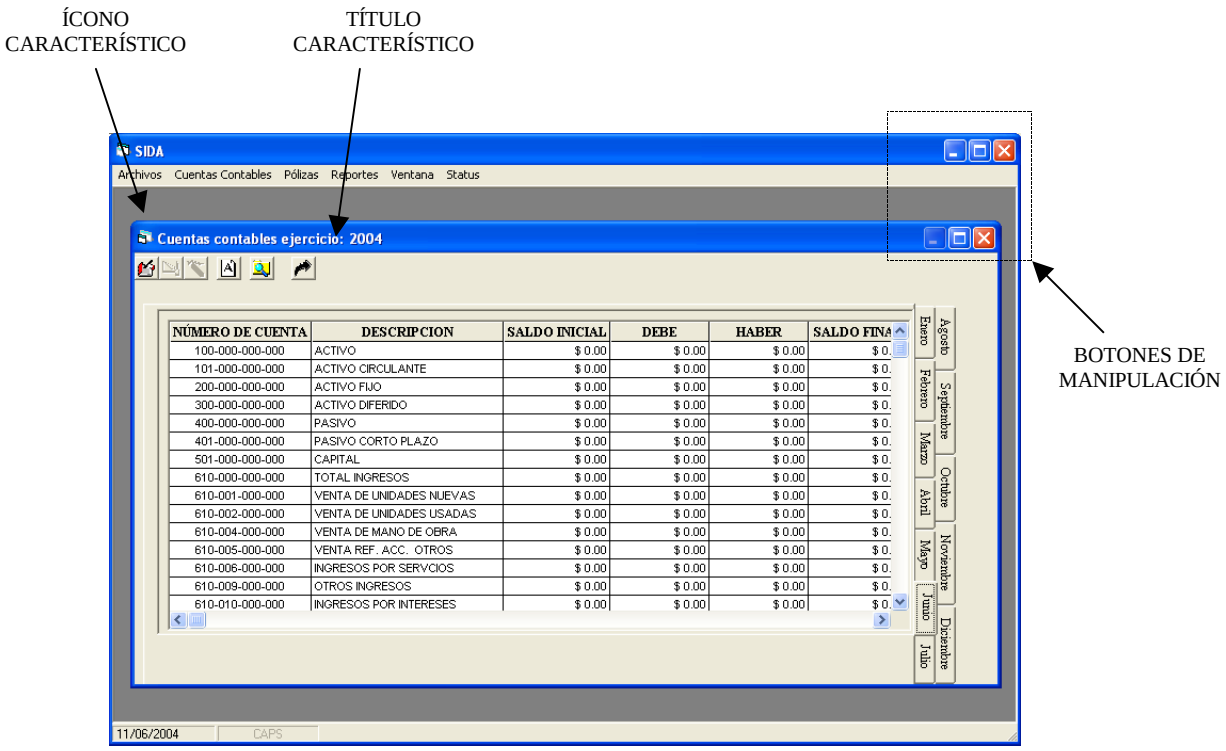
Columna	id_naturaleza	descripcion_n
Valores nulos	no	no
Tipo de dato	binario	caracteres (10)
Tipo de llave	primaria	--

IV.3. Diseño y construcción del Front-End

La elaboración y construcción final de la aplicación, es lo que será el front-end o vista final, que permitirá la manipulación de la información del sistema así como la generación de reportes de forma práctica y sencilla.

Para permitir una mayor funcionalidad del sistema y disminuyendo la posibilidad de error al operarlo, se ha diseñado un front-end integrado básicamente por botones de acción o comandos, de fácil operación mediante la utilización del mouse o del teclado. Esto tiene la apariencia que nos presenta una aplicación Windows, es decir lo más amigable hacia el usuario posible. Para el diseño del Front-End se tomaron ciertas consideraciones (también observadas en la figura IV.1) previas, como:

- Los colores de todas las pantallas o formularios (como se conocen en el ambiente de Visual Basic) se basarán en los preestablecidos por el usuario dentro del ambiente Windows.
- Dado que el sistema se desarrolló bajo el esquema MDI (Múltiple Document Interface) la navegación entre las pantallas se puede hacer por medio del menú general o por el menú de íconos en cada una de las pantallas contenidas en el.
- El tamaño de estas pantallas estará determinado por el espacio necesario para desplegar la información necesaria, es decir el tamaño requerido será el menor para poder visualizar dicha información.
- Todas las pantallas deberán contener:
    - Título de la pantalla
    - Cuando sea posible, botones de cerrar, restaurar y/o minimizar
- Estos formularios siempre estarán localizados en el centro del formulario principal MDI
- Los botones contenidos en estos formularios se definieron de un tamaño estándar.
- Lo que respecta a la navegación entre los formularios se cuenta con el ya conocido menú Ventana en el formulario principal el cual permite también un fácil arreglo de los formularios a utilizar.



IV.4. Pruebas e integración

La confiabilidad de un sistema es parte fundamental para su calidad. Para poderse llamar confiable es necesario que al utilizarse no se produzcan fallas peligrosas o costosas. Sin embargo se podría esperar que cumplan estas características en su totalidad pero es prácticamente imposible desarrollar software que se pueda demostrar que esté libre de errores, por lo que se debe usar métodos y técnicas que incluyan la detección de errores, su corrección y tolerancia.

La prueba es el proceso de ejecutar un programa con la intención de hallar errores, es decir, hacer que el programa falle. Así, una prueba exitosa es la que encuentra un error.

La prueba se debe llevar a todo lo largo del desarrollo del sistema, ya que cumple con el propósito de identificar aquellos problemas desconocidos, mas no demostrar la perfección del sistema.

IV.4.1. Prueba de código

La estrategia de pruebas de código examina la lógica del programa. Para seguir este método de prueba, el analista desarrolla casos de prueba que verifiquen la ejecución de cada instrucción en el programa o módulo; es decir, se prueba cada ruta del programa. Una ruta es una combinación específica de condiciones manejadas por el programa.

Un ejemplo sencillo es seguir la lógica que del siguiente segmento de código que sólo incluye el evento “DropDown” de una cuadro de lista:

CÓDIGO	INDICACIONES
Private Sub cmbempre_DropDown()	Cabecera de la función
Dim empre As String Dim rsempre As New ADODB.Recordset	Declaración de variables

If cmbempre.ListCount = 0 Then
cmbempre.AddItem "(Nueva Empresa)"
End If
	Si la lista está vacía comenzar por la opción de nueva empresa

If cmbempre.ListCount = 1 Then rsempre.Source = "select nombre from empresas" rsempre.Open , cn	Leer la tabla de empresas
If Not (rsempre.BOF And rsempre.EOF) Then rsempre.MoveFirst Do Until rsempre.EOF	Si existen valores en la tabla empresas y hasta que llegue al final...
cmbempre.AddItem rsempre.Fields("nombre") rsempre.MoveNext	Agregar los datos en la lista
Loop End If End If	Terminar los ciclos
End Sub	Fin de la función

Tabla IV.1

El ejercicio anterior se llevó a cabo en todas y cada una de las partes que conforman el código completo.

IV.4.2. Pruebas parciales

En este tipo de pruebas, se verifican los programas que conforman a un sistema. Se centran primero en los módulos independientes entre sí, para localizar los errores. Esto permite detectar errores en el código y lógica, contenidos dentro de ese único módulo. Aquellos errores que resultan de la integración entre módulos se evitan inicialmente.

El siguiente segmento de código parece muy sencillo, sin embargo cada una de las funciones escritas en mayúsculas implican un código complejo en sí.

```
...
  sel = LEEESTADO(periodosel, ejerciciosel)
  ant = LEEESTADO(periodoant, ejercicioant)

  If sel = 0 Then 'cerrado
    MsgBox "El periodo ya fue cerrado", vbOKOnly + vbCritical, "ERROR"
    Exit Sub
  End If
  If sel = 1 Then ' abierto
    MsgBox "El periodo ya está abierto", vbOKOnly + vbCritical, "ERROR"
    Exit Sub
  End If
  If sel = 2 Then ' no existe
    If ant = 2 Then
```

```
    MsgBox "Este periodo no se puede abrir", vbOKOnly + vbCritical, "ERROR"
    Exit Sub
   End If
   If ant = 1 Then
    Call ABRIRPERIODO(periodosel, ejerciciosel)
    MsgBox "El periodo ha sido abierto", vbOKOnly
    'llenar el siguiente periodo con las cuentas que están en cuentas
    Call ABRIRCUENTAS(periodosel, ejerciciosel)

   End If
  End If
...
```

Ahora, los errores posibles serán fácilmente detectables pues se tienen pequeños módulos que integran un todo.

Los casos de prueba necesarios para las pruebas parciales deben verificar cada condición u opción.

Si el módulo recibe una entrada o genera una salida, también se necesitan casos de prueba para examinar el rango de valores esperado, incluyendo los datos válidos e inválidos. Como en el siguiente ejemplo se observa que el tipo de datos que entran a la función son de tipo entero, y no tendrá datos de salida

```
Public Sub CERRARPERIODO(periodo As Integer, ejercicio As Integer)
  Dim query(4) As String

  query(0) = "update status set cerrado = 0 "
  query(1) = "where id_empresa = " & frmLogin.empresasel
  query(2) = " and ejercicio = " & ejercicio
  query(3) = " and periodo = " & periodo
  frmLogin.cn.Execute query(0) & query(1) & query(2) & query(3)

End Sub
```

Por lo que en el momento en que se llama a esta función, los datos de entrada serán únicamente de tipo entero, sin esperar respuesta alguna.

Si el módulo se diseña para llevar a cabo iteraciones con procesos específicos contenidos dentro de un ciclo, es recomendable ejecutar cada condición frontera: cero iteraciones, una iteración en el ciclo y el máximo de iteraciones en el ciclo.

Esto se puede ver en el siguiente ejemplo donde para cero iteraciones el código verifica que ha llegado al final del archivo sin haber entrado al ciclo, puede ejecutar una o más iteraciones sin problema alguno dado que la condición de final de archivo (EOF) se cumplirá una y sólo una vez llegado el archivo a su fin, pues siempre se encuentra moviendo su indicador al siguiente registro, lo que nos garantiza llegar al final en un determinado momento.

```
Do Until rs1.EOF
  cmbEjercicio.AddItem rs1.Fields("ejercicio")
  rs1.MoveNext
Loop
```

IV.4.3. Pruebas del sistema

Las pruebas de sistema no prueban el software en sí, sino la integración de cada módulo en el sistema. También buscan las discrepancias entre el sistema y su objetivo original, especificaciones y documentación del sistema. La preocupación principal es la compatibilidad de los módulo individuales.

Se trata de hallar áreas en donde los módulos hayan sido diseñados con especificaciones distintas para la longitud, tipo de datos y nombres de los elementos de los datos.

Un pequeño ejemplo para estas pruebas es el diseño para la tabla de relación cuentas con pólizas (Ver tabla IV.2) que contiene campos de tipo carácter, entero, variable, y flotante, de esta manera se puede apreciar que al ejecutar inserciones, operaciones o simplemente consultas los datos cumplen con las espectativas, así como con las exigencias y necesidades del sistema.

Tabla **cta_pol**

Columna	id_cta	id_empresa	ejercicio	periodo	id_tipo_pol	numero_pol	concepto	debe	haber
Valores nulos	no	no	no	no	no	no	no	si	si
Tipo de dato	caracteres (29)	entero	entero	entero	caracteres (3)	entero	varchar	float	float
Tipo de llave	primaria / foránea	primaria / foránea	primaria / foránea	primaria / foránea	primaria / foránea	primaria / foránea	--	--	--

Tabla IV.2

Las pruebas de sistema deben verificar que los tamaños de los archivos son adecuados y que los índices se han construido en forma adecuada. Se deben probar a nivel de sistema los procedimientos de ordenamiento y reindexación, que se supone están presentes en los módulos de nivel interior, para ver que realmente existen y que logran los resultados que esperan los módulos.

IV.4.4. Pruebas de especificación

Para llevar a cabo las pruebas de especificación se examinan las especificaciones que señalan lo que el programa debe hacer y cómo lo debe llevar a cabo bajo diferentes condiciones. Después se desarrollan casos de prueba para cada condición o combinación de condiciones y se mandan para su procesamiento. Por medio del estudio de los resultados, se puede determinar si el programa funciona de acuerdo con los requerimientos específicos.

Por ejemplo el siguiente código representa la respuesta a la necesidad de no permitir modificar ningún tipo de cuenta o póliza de periodos que ya han sido cerrados.

```
If Not rs.Fields("cerrado") Then
  resp = MsgBox("Este periodo ya fue cerrado" & Chr$(13) & "¿Desea verlo como sólo lectura? ", vbOKCancel)
  If resp = vbOK Then
    Toolbar1.Visible = False
    MDIsida.mnuAltasCta.Visible = False
    MDIsida.mnuEditarCta.Visible = False
    MDIsida.mnuEliminarCta.Visible = False
  Else
    Set TabStrip1.SelectedItem = TabStrip1.Tabs(CInt(Mid(frmAltas.PRIMERPERIODO(), 5, 2)))
    Exit Sub
  End If
End If
```

En este código se puede observar que al entrar a un periodo previamente cerrado, las opciones con las que se puede llevar a cabo un alta, edición o eliminación permanecerán invisibles al usuario, y este es una de las principales necesidades de la empresa.

IV.4.5. Prueba de Caja Blanca

Pruebas basadas en el conocimiento sobre la lógica y estructura interna. Usualmente dirigidas a la lógica.

IV.4.6. Prueba de Caja Negra

Pruebas funcionales basadas en los requerimientos sin conocimiento sobre como fue construido el sistema y usualmente dirigidas a los datos.

IV.4.7. Prueba alfa

A esta prueba se le conoce también como la de verificación, y su objetivo es el de detectar errores. Se lleva a cabo ejecutando un programa en un ambiente simulado. Por lo general, se solicita la intervención de un usuario para que ejecute la parte a verificar, mientras el analista observa y anota los errores en los que se incurre durante su ejecución, así como las reacciones del usuario ante el programa.

IV.4.8. Prueba beta

Esta prueba es conocida como la prueba de validación, y se refiere al proceso del uso del software en un ambiente no simulado para encontrar errores. La retroalimentación de la fase de validación generalmente produce cambios en el software para resolver los errores y fallas que se descubren. Se elige un conjunto de instalaciones usuarias que ponen a trabajar el sistema en un ambiente real. Estas instalaciones usan el sistema en las actividades cotidianas, también procesan transacciones en directo y producen salidas

normales del sistema. El sistema está a prueba en toda la extensión de la palabra, excepto que los usuarios están advertidos de que esta usando un sistema que puede fallar. Sin embargo las transacciones que están procesando y las personas que usan el sistema son reales.

También se ejecutaron varias pruebas más generales (Ver tabla IV.3) al sistema, además de los ejemplos mencionados en la explicación de cada prueba.

Tipo de pruebas	Ejecuciones
DE CÓDIGO	AL COMPILARSE Y PODER CREAR EL ARCHIVO EJECUTABLE LO EFECTUÓ DE MANERA SATISFACTORIA.
PARCIALES	PARA CADA MÓDULO (EN ESTE CASO SUBMENÚ) SE EFECTUARON PRUEBAS DE CÓDIGO, ASÍ COMO PRUEBAS EN LA ENTRADA Y SALIDA DE DATOS OBTENIÉNDOSE BUENOS RESULTADOS.
DEL SISTEMA Y DE ESPECIFICACIÓN	SE COMPROBARON QUE LA ENTRADA DE DATOS ASÍ COMO LOS RESULTADOS DE LOS MISMOS CUMPLIERAN CON LOS REQUISITOS ORIGINALES PASANDO ASÍ ESTA PRUEBA SATISFACTORIAMENTE
DE CAJA NEGRA	TODOS LOS MÓDULOS QUE CONFORMAN EL SISTEMA FUERON PROBADOS DESDE SU EXTERIOR, ES DECIR COMPROBANDO ÚNICAMENTE QUE DADAS CIERTAS ENTRADAS SE OBTENGAN LAS SALIDAS REQUERIDAS
DE CAJA BLANCA	CADA UNO DE LOS MÓDULOS FUERON PROBADOS INTERNAMENTE DE MODO QUE EFECTUARAN TODAS SUS OPERACIONES ADECUADAMENTE Y NO SÓLO ARROJARA LA SALIDA CORRECTA, OBTENIÉNDOSE TAMBIÉN RESULTADOS SATISFACTORIOS.

Tabla IV.3

IV.5. Generación de reportes

Uno de los principales requerimientos de la empresa fue el de poder obtener reportes de manera automática, dentro de los reportes requeridos podemos encontrar los siguientes:

- Catálogo de Cuentas

Este catálogo contiene en su cabecera y/o pie de página:

- Título del reporte
- El nombre de la empresa
- La fecha de elaboración
- El periodo que abarca el reporte
- Número de página

Y en específico:

- El número de la cuenta
- Descripción de la cuenta
- Y saldo final

Todos estos datos actualizados a la fecha de elaboración en el periodo solicitado.

- Balanza de comprobación

Para la cabecera y/o pie de página tenemos:

- Título del reporte
- Periodo que abarca
- Fecha de elaboración
- Número de página

En específico contiene:

- Número de cuenta
- Descripción de la cuenta
- Saldo inicial
- Debe
- Haber
- Saldo final
- Totales de saldos deudores y acreedores
    - para saldo inicial, debe, haber y saldo final

- Libro de pólizas o Diario General

Para la cabecera y/o pie de página contiene:

- Título del reporte
- Número de folio
- Fecha de elaboración
- Número de página

En específico cuenta con:

- Respecto a las pólizas en general
    - Tipo
    - Número
    - Concepto
    - Referencia
- Respecto al detalle de la póliza
    - Número de cuenta
    - Descripción de la cuenta
    - Concepto del movimiento
    - Debe
    - Haber
- Y totales de pólizas en el debe y haber

- Estados financieros (Balance general)

En el encabezado o pie de página se encuentan:

- Nombre de la empresa
- Título del reporte
- Fecha
- Número de página

En específico se tiene:

- Cuatro columnas que contiene
    - Concepto
    - Fecha de comparación 1
    - Fecha de comparación 2
    - Diferencia
- Esas columnas contienen los saldos finales de las cuentas englobadas en:
    - Activos
        - Circulante
        - Fijo
        - Diferido
    - Pasivo
        - Corto plazo

        - Largo plazo
    - Capital
- Todos estos datos con sus respectivos totales al final de cada columna

- Estado de resultados (de pérdidas y ganancias)

En la cabecera y/o pie de página se encuentran:

- Nombre de la empresa
- Título del reporte
- Fecha
- Número de página

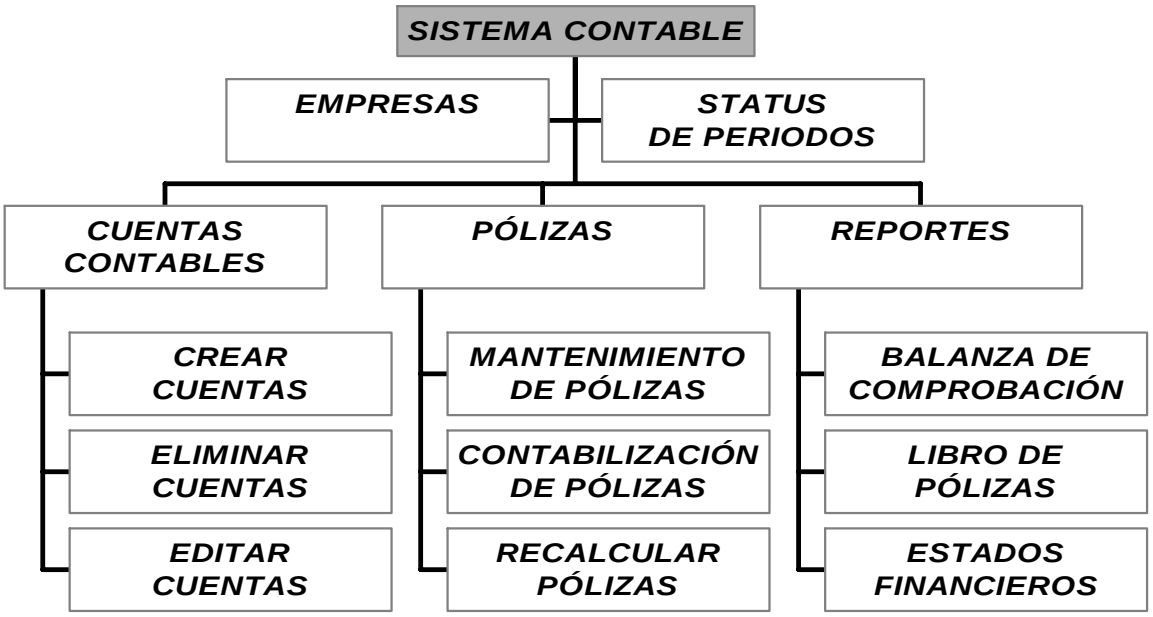
En específico se tiene:

- Relación de ingresos
- Relación de egresos
- Diferencia entre ambos
- Desplegando así la utilidad (pérdida) neta
- Y todos los datos anteriores para:
    - El mes actual
    - El acumulado hasta la fecha
    - El promedio

Manual Técnico

El sistema fue desarrollado bajo el siguiente esquema:

- Interfaz de usuario: Visual Basic 6.0
- Administrador de Bases de Datos: SQL Server 7.0
- Sistema Operativo: Windows 2000 Family Server (NT)
- Este sistema se rige bajo el siguiente diagrama funcional:



La forma de acceso a estos módulos se lleva a cabo por medio del siguiente menú.

Cuentas Contables	Pólizas	Reportes	Status
Crear	*Mantenimiento*	*Balanza de comprobación*	
Eliminar	*Contabilización*	*Libro de pólizas*	
Editar	*Recalcular*	*Estados Financieros*	

Los tipo de usuarios que tienen acceso a este sistema son los siguientes:

Contador General

Sistema Externo de ventas y servicios

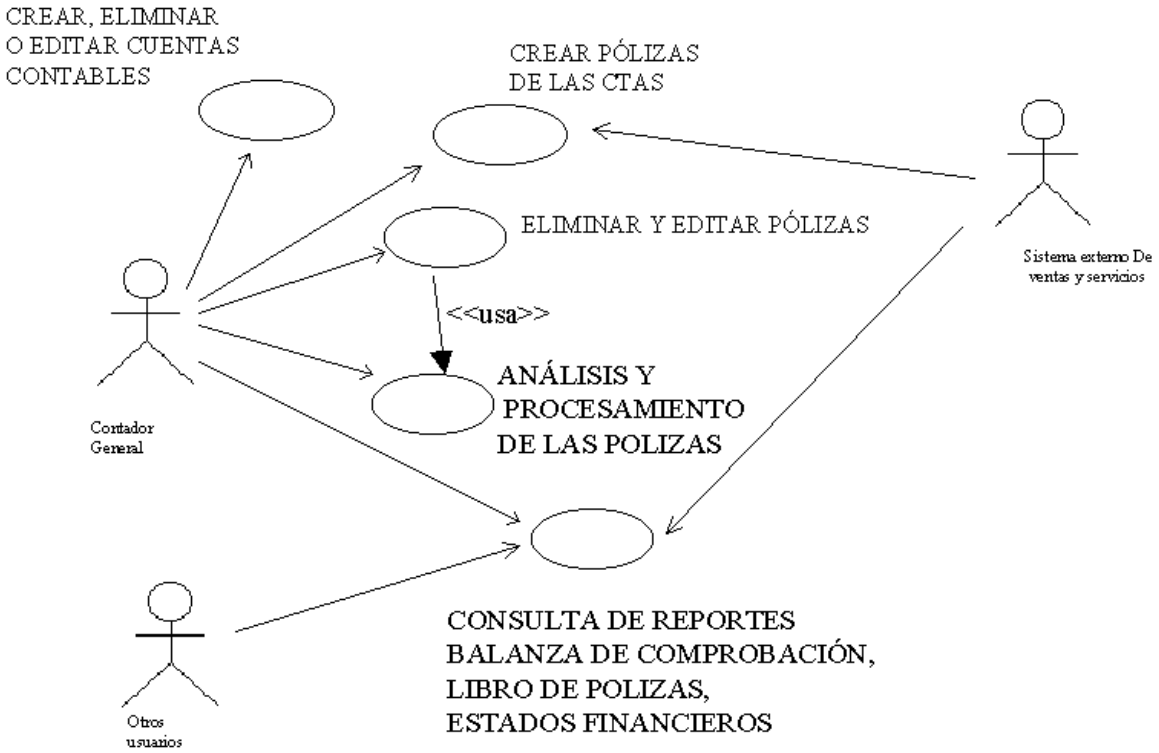
Otros usuarios

Para el contador general el sistema es libro abierto, es decir que puede crear eliminar y editar tanto cuentas como pólizas así como cualquier otro tipo de consulta.

El sistema de ventas y servicios, puede tener acceso para generar pólizas que correspondan a sus movimientos, así como para realizar consultas en general.

Y cualquier otro usuario que tenga acceso al sistema, podrá hacer las consultas de todos los reportes generados.

Lo anterior se puede ver de manera gráfica como sigue:



La interfaz de usuario fue desarrollada en un entorno MDI (Multiple Document Interface) que se refiere a que todas las ventanas forman parte de una más grande, dado que el sistema no requiere ejecutar más de un evento a la vez así como las transacciones con el administrador de la Base de Datos.

Las ligas entre la interfaz y la base de datos se llevan a cabo de forma remota, para que los usuarios finales puedan tener acceso al sistema desde cualquier equipo externo por lo que se utilizó ODBC (Open Data Base Connectivity) siendo éste el medio por el cual se pueden comunicar el usuario a través de la interfaz con la base de datos.

La base de datos contiene los siguientes esquemas:

Esquema_empresas *(id_empresa, nombre, nivel)*

Esquema_status *(id_empresa, ejercicio, periodo, cerrado)*

Esquema_cuentas *(id_cta, id_empresa, ejercicio, periodo, saldo ini, debe, haber)*

Esquema_polizas *(id_tipo_pol, numero_pol, fecha_can, referencia, descripcion. periodo, ejercicio, id_empresa)*

Esquema_cta_car *(id_cta, id_empresa, fecha_c, id_balance, id_registro, id_naturaleza, cta_padre)*

Esquema_cta_pol *(id_cta, id_empresa, ejercicio, periodo, id_tipo_pol, numero_pol, debe, haber)*

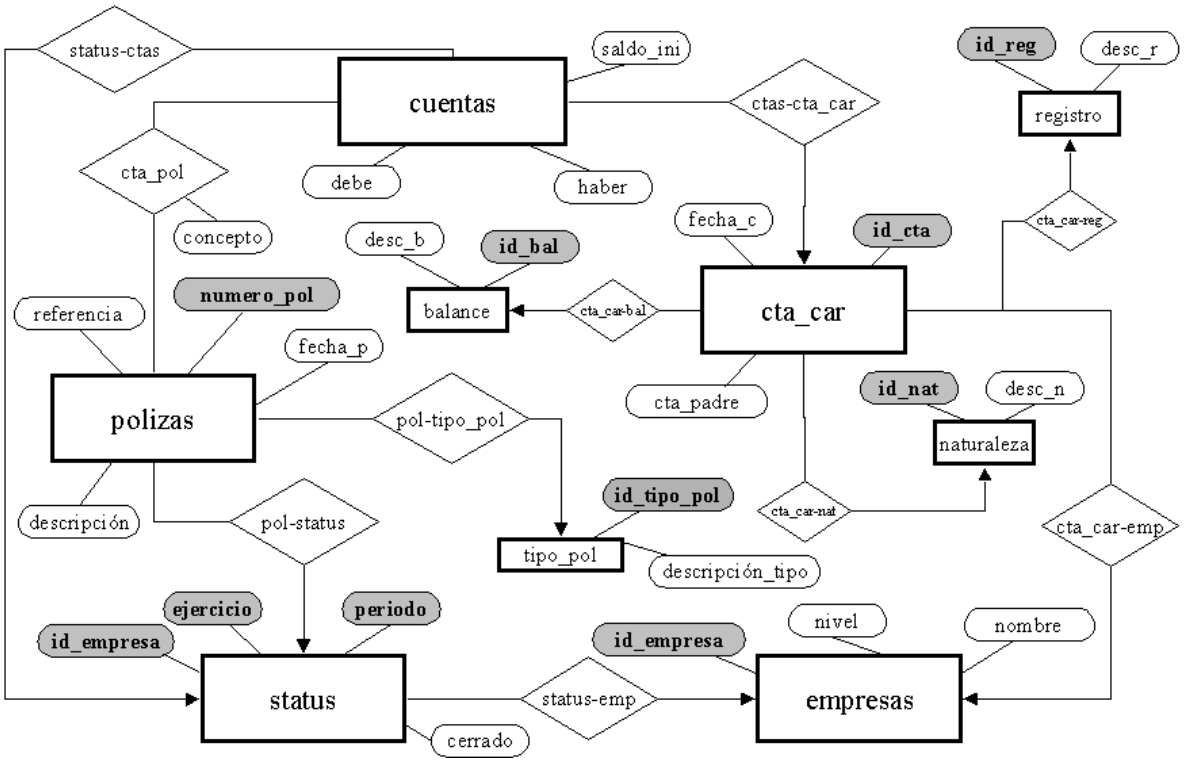
Esquema_tipo_pol *(id_tipo_pol, descripcion_tipo)*

Esquema_balance *(id_balance, descripcion_b)*

Esquema_registro *(id_registro, descripcion_r)*

Esquema_naturaleza *(id_naturaleza, descripcion_n)*

Estos esquemas se crearon a partir del siguiente diagrama entidad relación:



En el diagrama anterior se puede observar que las entidades son:

cuentas

cta_car

status

polizas

empresas

tipo_pol

balance

registro

naturaleza

Con la información anterior podemos encontrar que en la base de datos están definidas las siguientes tablas:

Nota: id se refiere a todos los identificadores que forman parte de una llave primaria

empresas

Columna	*id_empresa*	*nombre*	*nivel*
Valores nulos	no	no	no
Tipo de dato	entero	caracteres (10)	entero

Llp:{id_empresa}
El nombre de la empresa no debe contener más de 10 caracteres ya que en este espacio no se encuentra el nombre real de la empresa.
El nivel se refiere al nivel máximo al que pueden llegar las cuentas de esta empresa.

status

Columna	*id_empresa*	*ejercicio*	*periodo*	*cerrado*
Valores nulos	no	no	no	no
Tipo de dato	entero	entero	entero	bit

Llp:{id_empresa, ejercicio, periodo}
El ejercicio es el año, y periodo el mes, para el cual, cerrado nos ofrece la condición en la que se encuentra la empresa para tal mes y año, por lo que cerrado nos muestra si está cerrado (0) o abierto(1).

cuentas

Columna	*id_cta*	*id_empresa*	*ejercicio*	*periodo*	*saldo_ini*	*debe*	*haber*
Valores nulos	No	no	no	no	si	si	si
Tipo de dato	Caracteres (29)	entero	entero	entero	float	float	float

Llp:{id_cta, id_empresa, ejercicio, periodo}
En saldo_ini se encuentra el saldo con el que inicia la cuenta para el perioro y el ejercicio actual.
Llf:{id_empresa, ejercicio, periodo}

polizas

Columna	*id_tipo_pol*	*numero_pol*	*fecha_can*	*referencia*	*descripcion*	*id_empresa*	*ejercicio*	*periodo*
Valores nulos	no	no	no	no	no	no	no	no
Tipo de dato	caracteres (3)	entero	fecha	texto	caracters (55)	entero	entero	entero

Llp:{id_tipo_pol, numero_pol, ejercicio, periodo, id_empresa}
Llf:{id_tipo_pol}, {id_empresa, ejercicio, periodo}

cta_car

Columna	*id_cta*	*id_empresa*	*fecha_c*	*id_balance*	*id_registro*	*id_naturaleza*	*cta_padre*
Valores nulos	no	no	no	no	no	no	si
Tipo de dato	caracters (29)	entero	fecha	binario	binario	binario	caracteres (29)

Llp:{id_cta, id:empresa}
La cuenta padre debe existir por si misma como cuenta, y puede que no tenga cuenta padre, pues la cuenta a la que se esté refiriendo puede ser una cuenta padre única.
Llf:{id_cta, id_empresa}, {id_balance}, {id_registro}, {id_naturaleza}

cta_pol

Columna	*id_cta*	*id_empresa*	*ejercicio*	*periodo*	*id_tipo_pol*	*numero_pol*	*concepto*
Valores nulos	no	no	no	no	no	no	no
Tipo de dato	caracteres (29)	entero	entero	entero	caracteres (3)	entero	caracteres (30)

Llp:{id_cta, id_empresa, ejercicio, perido, id_tipo_pol, numero_pol}
El concepto se genera al tener una relación entre la póliza y la cuenta.
Llf:{id_cta, id_empresa, ejercicio, periodo}, {id_tipo_pol}

tipo_pol

Columna	*id_tipo_pol*	*descripcion_tipo*
Valores nulos	no	no
Tipo de dato	caracteres	caracteres

Llp:{id_tipo_pol, descripcion_tipo}

balance

Columna	*id_balance*	*descripcion_b*
Valores nulos	no	no
Tipo de dato	binario	caracteres (10)

Llp:{id_balance}

registro

Columna	*id_registro*	*descripcion_r*
Valores nulos	no	no
Tipo de dato	binario	caracteres (10)

Llp:{id_registro}

naturaleza

Columna	*id_naturaleza*	*descripcion_n*
Valores nulos	no	no
Tipo de dato	binario	caracteres (10)

Llp:{id_naturaleza}

Lo que respecta a los archivos, el proyecto en visual basic se llama SIDA.prj

El nombre de los archivos para las formas en visual basic así como los controles más utilizados en el proyecto siguen la siguiente convención:

Forma	frm*nombre*
Botones	boton*nombre*
Etiquetas	lbl*nombre*
Opciones	opt*nombre*
Cuadros de texto	txt*nombre*
Cuadros combinados	cmb*nombre*

Manual de Usuario

Instalación

Antes de poder llevar a cabo la instalación debe asegurarse de contar con los requerimientos mínimos de hardware y software para poder ejecutarlo.

Hardware

- Procesador 486 DX2/66MHz o mayor
- Al menos 16 Megabytes de RAM
- Por lo menos 5 Megabytes libres en disco duro
- Drive para CD-ROM

Software

- Microsoft Windows 98 o superior
- En caso de ser administrador, necesitaría Windows NT SERVER, así como SQL SERVER 7.0

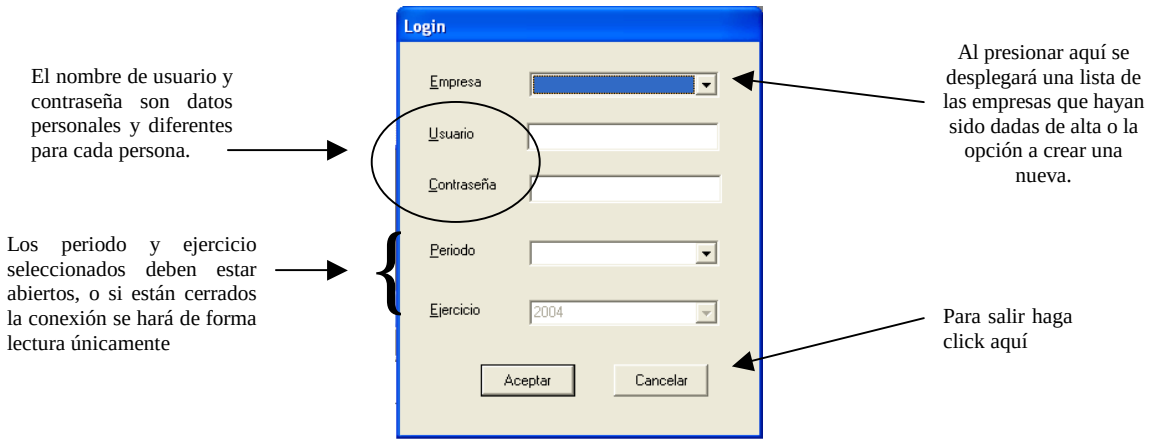
Ahora ya está listo para la instalación.

1. Coloque el CD en el drive de CD-ROM de su computadora.
2. Seleccione ejecutar del menú inicio en la barra de tareas.
3. En el cuadro de diálogo teclee la letra en la que se encuentra el CD-ROM, seguido de \setup.
4. Presione enter,
5. Siga las instrucciones de instalación.

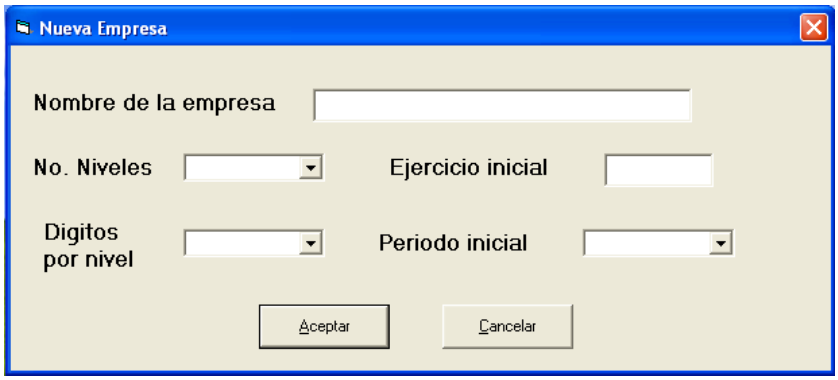
Referencia general

Para poder empezar a trabajar con este software necesita dar doble click sobre el icono que se generó en su escritorio llamado SIDA.

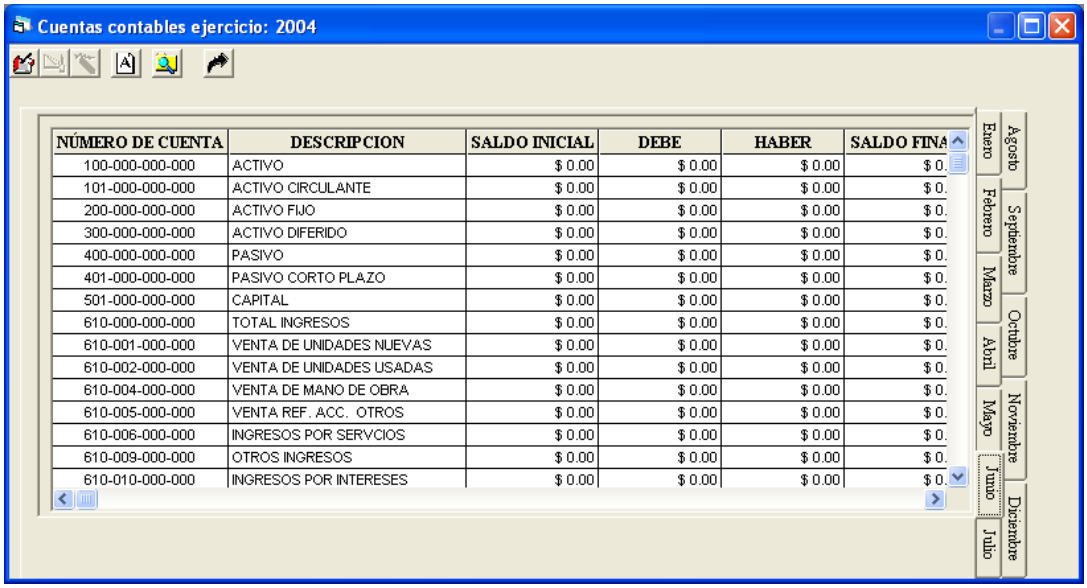
Posterior mente aparecerá un cuadro de diálogo como el que sigue:



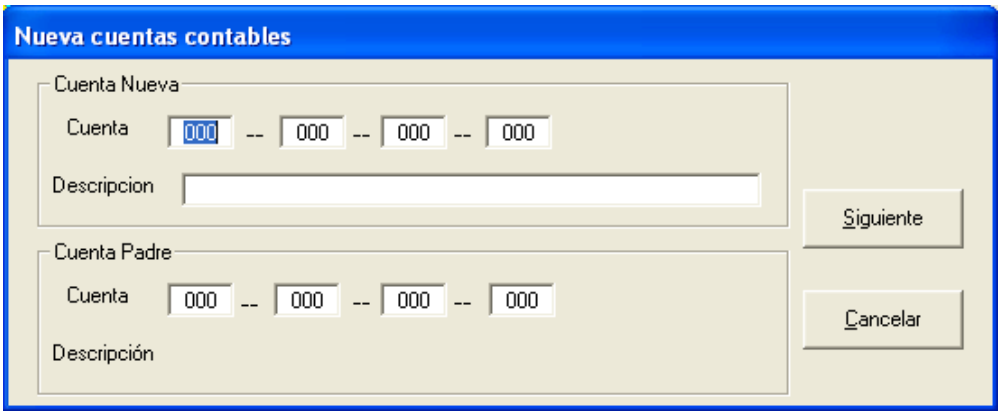
Para crear una empresa nueva se tiene la siguiente pantalla:



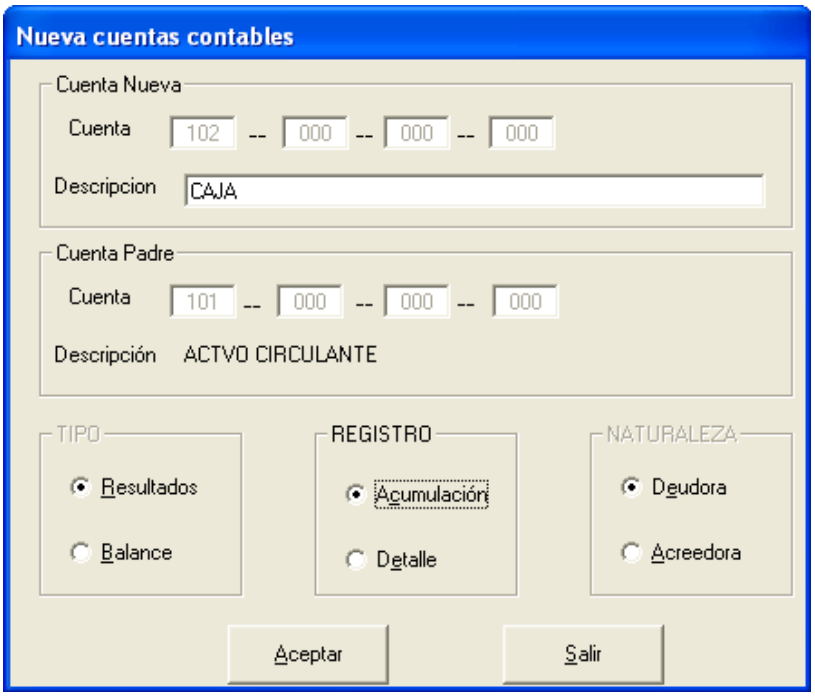
La pantalla siguiente es la correspondiente a las cuentas contables del presente ejercicio:



Para crear una cuenta nueva se hace click en “CREAR CUENTA NUEVA”

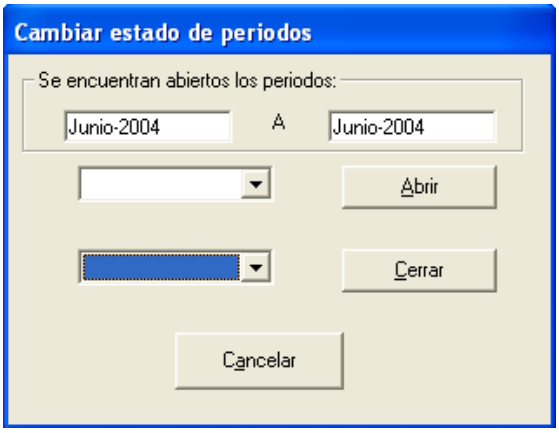


y al presionar siguiente se tendrá:

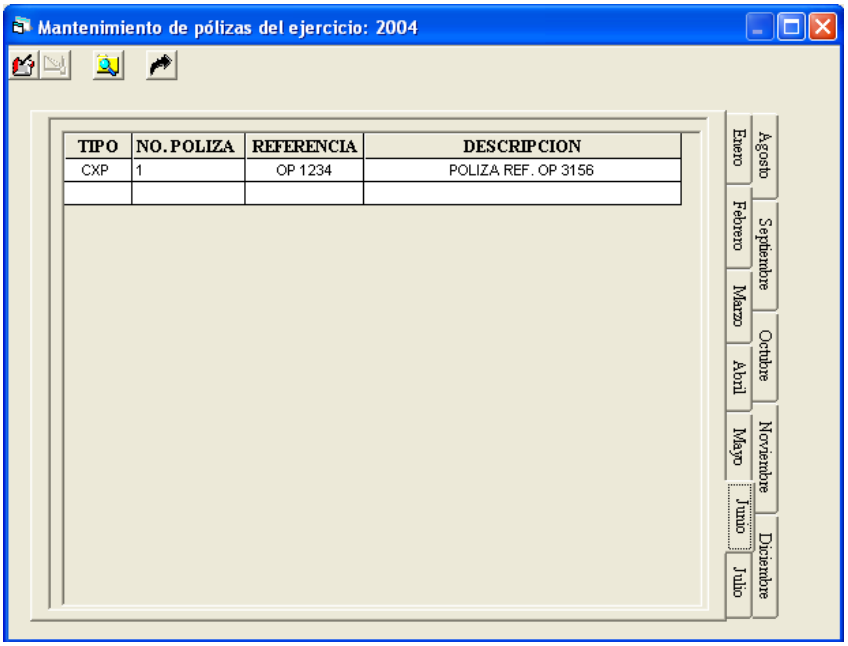


Para eliminar y editar primero se presenta una pantalla como en el catálogo y al elegir una de ellas para editarla:

Al elegir el menú para cambiar el status aparecerá:

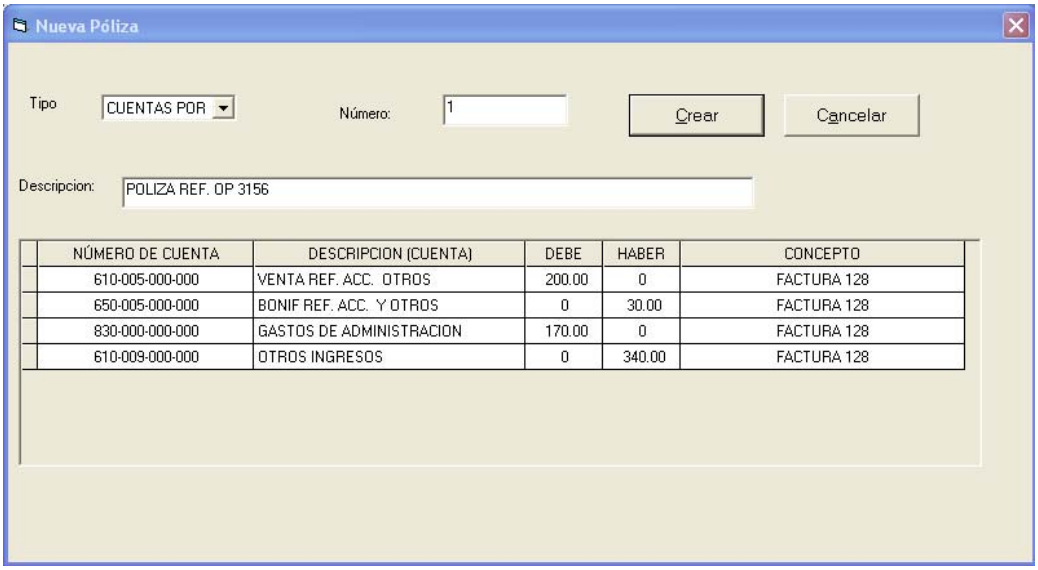


Lo que corresponde al menú de pólizas, en “mantenimiento de pólizas” aparece la siguiente pantalla:

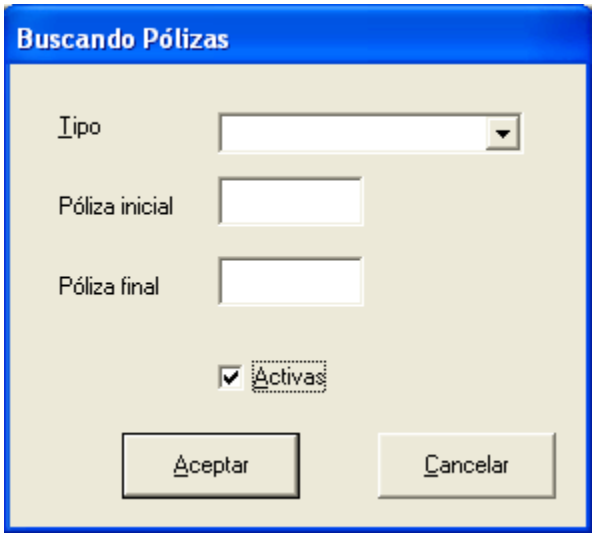


En esta pantalla se observa que puede escoger entre los diferentes periodos para ver las pólizas,

Para poder crear pólizas nuevas se tiene la próxima pantalla:



Ahora, si se desea buscar una póliza en especifico o un grupo de ellas, debe hacer click en “Buscar” y aparecerá la siguiente pantalla:



La búsqueda se puede efectuar indicando cualquiera de las características o todas si así se desea, por ejemplo si necesita buscar una póliza de la que se conoce su número, simplemente escribir ese número en póliza inicial y póliza final, o ver únicamente las canceladas, no debe estar seleccionado el cuadro de “Activas”.

Cabe mencionar que todas estas búsquedas se limitan a la empresa solicitada en un principio, así como el ejercicio que se seleccionó y el periodo.