UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

CONSTRUCCION DE UN CENTRO DE INFORMACION PARA UNA INSTITUCION BANCARIA SOBRE ARQUITECTURA CLIENTE – SERVIDOR.

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A N:

MARIA ALICIA JAIME LOA
ENRIQUE LEGLISSE CISNEROS
JAIME RICARDO MARTINEZ SOTO
JUAN CARLOS SOTO CASTILLO

ASESOR: M.I. LAURO SANTIAGO CRUZ

MEXICO, D.F. MARZO DE 1999

PAGINACION

DISCONTINUA.

Agradecimientos

Este trabajo está dedicado principalmente a la **Universidad Nacional Autónoma de México**, la institución que nos brindó la oportunidad de realizar una carrera profesional, así como a todo el profesorado de la Facultad de Ingeniería quienes dedicaron su tiempo y esfuerzo para darnos las bases necesarias para desarrollar una formación propia.

Se hace un digno reconocimiento al **M. en I. Lauro Santiago Cruz** quien dirigió el presente trabajo, brindándonos las facilidades y el apoyo que se requirieron para la elaboración de esta tesis.

Dedicatorias

Dedico este trabajo sobretodo a **Dios** por haber estado siempre a mi lado en los momentos en que más lo necesite.

A mis padres que amo y admiro, **Sr. Eusebio Soto Arias y Sra. Manuela Castillo de Soto**, quienes en todo momento me brindaron su apoyo y jamás se apartaron de mi lado, pero lo más importante, a quienes debo mi vida y mis logros personales.

A mis Hermanos **Roberto, Olga, Bety, Memo, Alfredo y Ara**, con quienes compartí mi niñez y gran parte de mi vida de estudiante.

A mi amada esposa **Berenice**, quien me inspira a superarme día tras día con su amor y apoyo.

A mis sobrinos **Fabis, Paola, Beto, Gina, Gaby, Carem, Chuy y Aideé**, sigan adelante, la vida es difícil y deben prepararse bien.

A mis compañeros y amigos, **Ricardo, Alison y Enrique**, por la oportunidad de realizar este trabajo con ellos.

A **Gerardo Saldivar**, quien incondicionalmente me brindó su apoyo en todo momento.

Dedicatorias

A **Dios**, por todo y tanto que me dá.

A mis padres, **Enrique y Angeles**, por su amor, por el apoyo incondicional y la fuerza e inspiración para llevar a cabo este y muchas cosas más, los amo.

A mis hermanos, **Angeles, Daniel y Teresa** por tanto que me dan y tanto que he aprendido de ellos. Los quiero mucho.

A mis tíos **Ana, Rosa y Salvador**, por brindarme apoyo y regalarme su experiencia, por tanto que me han dado y tanto que he aprendido de ellos, son únicos.

A **Angélica, Jesús, Gerardo y Nadia**, por darme la oportunidad de ser padre y amigo, siempre los amaré.

A **Claudia, Omar, Gabriela, Gloria y Alberto**, por ser mis hermanos y cómplices, mi ayuda y mi soporte y por tanto y tanto que me han compartido. Siempre cuentan conmigo.

A **Alison, Oswaldo, Graciela y Laura**, por ser apoyo y consuelo en los momentos personales y profesionales más importantes, la enseñanza que me dieron es mucha.

A **Mónica y Alejandro**, mis compadres, por compartir conmigo lo más valioso que pudieran tener, su hijo.

A **Alison, Ricardo y Juan Carlos**, por la paciencia, por tanto que pasamos juntos y sobre todo por lo que podemos hacer juntos, somos un gran equipo.

A **Herman, Carmen y Raymundo** por ser grandes amigos, me enseñaron a conquistar otras metas y a darme diferentes alegrías y satisfacciones, gracias por su apoyo.

A **Miguel**, por ser la persona que me ha enseñado más en más corto tiempo, eres único, gracias por aparecer en este grupo y en este momento.

A **SOFTTEK** por ser la escuela y a **PCA**, por darme la oportunidad de despuntar y de crecer como jamás lo imaginé, son muy importantes y son la vía para desarrollar todo lo que aprendí.

Nuevamente a **Claudia y a Alison** por ser la amistad más pura que he conocido, son lo más increíble que me ha pasado jamás.

A ti, que estás leyendo este trabajo, espero podamos ayudarte a solucionar tus problemas o al menos interesarte en esto que hicimos con tanto esfuerzo, espero te sea útil.

Enrique Leglisse

Dedicatorias

A **Dios,** por todas las cosas que nos ha dado y enseñado.

A mis papás **Jaime Martínez Herrera y María de los Angeles Soto**, que me han apoyado en todo momento de mi vida y con el ejemplo me han enseñado a disfrutar cada instante, gracias siempre por su alegría de vivir y darnos a mi, a mis hermanas y a todos lo mejor de ustedes, su amor.

A mis hermanas **Nora y Erika** que me han dado todo su cariño y paciencia, solo espero ser tan buen hermano como ellas lo han sido.

A mis tíos **Raquel, Cándido y Concepción, María, Baltazar, Eduwiges, Rolando, Marbella**, y a sus familias, por apoyarme y enseñarme desde niño, que todo lo que desees se puede lograr.

A **Alison**, por ser esa persona a la que tanto amo y admiro, que me enseña a ser mejor cada día con su paciencia, compresión y cariño.

A **Ricardo y Salvador, Adrián B.** por su amistad tan grande que me han dado, su forma única de ver la vida y compartir la suya conmigo.

A **Juan Carlos, Gerardo y Enrique** por ser como son y darme lo mejor de cada uno, los admiro son únicos.

Jaime Ricardo

INTRODUCCION

"Competencia" es la palabra; la sociedad actual cuenta con una infinidad de posibilidades de elección para solventar sus necesidades, de servicios y/o productos; lo cual coloca a los proveedores en una situación de competencia muy difícil, ya que si un cliente no se siente cómodo con el servicio que se le ofrece, tiene más opciones para encontrar entre ellas la que más le agrade.

Como se dice en los negocios: "Para conseguir un cliente se puede necesitar de mucho tiempo; pero para perderlo sólo basta un segundo."

Este panorama se vuelve más interesante, cuando tomamos en cuenta que gracias a los avances tecnológicos, las distancias y fronteras parecen no existir. La sociedad conoce más sobre estándares de calidad, ya no en su país sino a nivel internacional; lo cual los convierte en clientes mucho más exigentes.

Todo esto sucede a gran velocidad y abarca a todos los sectores, en especial al de nuestro tema de estudio; el financiero, representado por los bancos. Uno de los indicadores más importantes sobre el nivel de desarrollo de cualquier país es su sistema financiero, es por esto que debe verse a los bancos, más que como empresas, como instituciones que juegan un papel relevante en la estabilidad económica nacional.

La apertura comercial, nos lleva a una competencia que exige mejoras inmediatas a los Sistemas de Información de los bancos, para poder ofrecer a la sociedad que crece constantemente una gama de servicios que se actualizan constantemente, haciéndolos más eficientes que antes.

El objetivo de este trabajo es diseñar e implementar un Centro de Información para BANPAIS, en una arquitectura Cliente - Servidor, con una *interface* gráfica de fácil acceso para el usuario. Esto como parte del proyecto de actualización tecnológica del banco.

BANPAIS realiza la mayoría de sus operaciones en *Mainframe*, incluso las de presentación de informes a la dirección, la que requiere de esta información en forma dinámica, confiable y en una presentación que les dé la facilidad de procesarla posteriormente en distintas formas y obtener estimaciones estadísticas con escenarios de alta probabilidad.

El diseño de esta aplicación contempla, el mantener a los niveles directivos y ejecutivos bien informados sobre el estado actual en que se encuentra el banco, esto con la finalidad de que se tomen decisiones y se realicen acciones a tiempo, para mantener y aumentar el nivel de calidad en el servicio.

La descripción del presente trabajo está organizado de la siguiente manera:

Capítulo 1: Antecedentes. Comprende los aspectos teóricos básicos del manejo de información, como son: modelos de desarrollo y tipos de bases de datos, en los cuales se apoya el resto del documento.

Capítulo 2: Generalidades. Contiene definiciones y conceptos de lo que es el Sistema Financiero Mexicano así como Banca Múltiple y sus servicios.

Capítulo 3: Descripción del Proyecto: En este apartado se delimita el problema, se describen las posibles soluciones y se muestra el análisis que determina la solución definitiva.

Capítulo 4: Planeación, Diseño y Desarrollo del Sistema: Aquí se describen las técnicas a utilizar en el diseño, la estructura lógica, procesos, diseño de tablas y estándares a cumplir y respetar en el desarrollo del sistema.

Capítulo 5: Pruebas, Liberación y Mantenimiento: En este apartado se describen las pruebas y validaciones a las que se somete al sistema, previas a su liberación; la generación de manuales de usuario, los tipos de mantenimiento que se le aplica al sistema, finalmente la presentación de las cargas de información y la puesta en marcha a escala nacional. mantenimientos preventivos.

Capítulo 6: Resultados y Conclusiones: Esta sección propiamente nos muestra los resultados y las conclusiones a los objetivos planteados, entre los cuales se presenta al sistema como base para la migración de la información a Banorte, a la aplicación como incremento del valor agregado a los sistemas de la institución, así como las experiencias personales que nos deja el desarrollo de este trabajo.

Bibliografía: Aquí se encuentran las referencias bibliográficas, hemerográficas, así como las direcciones de las páginas en Internet, que se consultaron durante el desarrollo del trabajo.

Apéndices: Finalmente en esta sección, se encuentran: Las metodologías de programación utilizadas, el código de los programas, el glosario de términos y el análisis costo beneficio.

INDICE

4. PLANEACION, DISEÑO Y DESARROLLO DEL SISTEMA

5. PRUEBAS, LIBERACION Y MANTENIMIENTO

6. RESULTADOS Y CONCLUSIONES

BIBLIOGRAFIA

APENDICES

CAPITULO 1

ANTECEDENTES

En este capítulo se describirán brevemente los conceptos de Ingeniería de *Software*, así como los elementos necesarios para elaborar una aplicación que resuelva un problema real, cubriendo las necesidades del usuario y tomando en cuenta los recursos con los que cuenta.

De igual forma se dará una introducción a los conceptos de Programación Estructurada, Características de los sistemas, Bases de Datos y desarrollo de aplicaciones en arquitectura Cliente / Servidor.

1.1 Historia

¿Qué es la Ingeniería de *Software*?

La Ingeniería de *Software* comparte, junto con otras ramas de la Ingeniería, el enfoque pragmático para el desarrollo y mantenimiento de aplicaciones tecnológicas; existen, sin embargo, diferencias significativas entre esta ingeniería y las otras, siendo la fuente principal de dichas diferencias la falta de leyes físicas para la programación, la intangibilidad del producto y lo oculto de las interfaces entre los diversos módulos de programación.

La ingeniería de *Software* se define como la disciplina tecnológica preocupada de la producción sistemática y mantenimiento de los productos de *Software* que son desarrollados y modificados en tiempo y dentro de un presupuesto definido.

Esta ingeniería tiene como objetivo optimizar el aprovechamiento de los recursos destinados a dichos productos de *Software* que se desarrollen en un medio real.

Entendemos como medio real el hecho de que actualmente los usuarios requieren que el Software este a su disposición, es decir que lo tengan a la mano y puedan manipularlo, y no que el software les ponga limitaciones a pesar de las ya conocidas (velocidad de proceso, capacidad de almacenamiento, etc.).

La madurez de la capacidad de la Ingeniería de Software en una organización puede ser medida en términos del grado en el cual, los resultados del proceso por medio del cual el software es desarrollado pueden ser precedidos.

- Tiempos de Entrega. Predecir la cantidad de tiempo requerido para desarrollar un producto de software.
- Recursos Humanos. Predecir los recursos (número de personas, cantidad de espacio en disco, etc.) requeridos para desarrollar un producto de software.
- Presupuesto. Predecir el costo de desarrollar un producto de software.
- Prioridades.

Nacimiento y Evolución de la Ingeniería del Software

A lo largo de la historia de la humanidad, la necesidad de obtener recursos que permitan realizar las tareas cotidianas de manera más eficiente fue cada vez más grande, estos acontecimientos dieron lugar al nacimiento de máquinas que permitían realizar cálculos matemáticos en menor tiempo y en mayor cantidad que un ser humano.

Estas máquinas también evolucionaron y requirieron de instrucciones de operación más elaboradas y complejas, surgiendo así las generaciones de computadoras y lenguajes de programación que permitieron resolver problemas más complejos. La tabla 1.1 muestra la evolución de lo que hoy conocemos como Ingeniería de *Software*.

Año	Acontecimiento
1950	Lenguajes de máquina y ensamblador (Primera y Segunda Generación)
1958	Nacimiento de COBOL y FORTRAN (Lenguajes de Tercera Generación)
1960	Creación de ALGOL 60 (Primer Lenguaje "Estructurado")
1964	Lanzamiento del PL/Y
1968	Literatura sobre Programación Estructurada
1970	Cursos sobre Programación Estructurada en los U.S.A.
1974	Literatura sobre Ingeniería de *Software* (Análisis Estructurado, Diseño Estructurado, etc.)
1978	Cursos sobre Programación Estructurada en México
1980	Programación Orientada a Objetos (*SmallTalk*), Introducción de primeras Herramientas CASE
1985	Aparición de Lenguajes de "Cuarta Generación" (Dbase, CSP, SISINF)
1990 [1]	Comercialización de Herramientas CASE Integradas (ICASE)

Tabla 1.1. Historia de la Ingeniería de Software.

Características de los Sistemas

Ya que los programas no tienen propiedades físicas, no están sujetos a las leyes de gravedad; no existen leyes o ecuaciones que permitan guiar el desarrollo de los programas, se limita el número de directrices y lineamientos fundamentales disponibles para encausar el diseño y la instrumentación de un proyecto de programación.

La variedad de enfoques en el desarrollo de programas está limitada solamente por la creatividad y el ingenio del desarrollador, para minimizar la probabilidad de error en la selección del enfoque adecuado, es importante considerar las características de los sistemas (tabla 1.2).

[1] Actualmente existe una acalorada discusión sobre lo que se consideran "Lenguajes de Quinta Generación"

El Usuario desea en un sistema	El Sistema debe ser Comprensible	El Sistema debe ser Modificable	El Sistema debe ser Confiable
Minimizar costo y tiempo de desarrollo	Documentación completa y clara	Un manejo manual del sistema flexible	Minimizar el número de fallas
Que cumpla con sus Requerimientos	Programación legible y ordenada	Programas con información flexibles	Detectar errores en su diseño
Que sea fácil de utilizar	Lógica sencilla y estructurada	Archivos de diseño flexible	Fácil de rastrear
Que no requiera de mantenimiento excesivo	Archivos con información ordenada en forma lógica	Programas que manejan información flexibles	Identificar la información válida e inválida

Tabla 1.2. Características de los sistemas.

1.2 Modelos de Desarrollo y Bases de Datos

Podemos hablar de tres modelos de desarrollo para describir un sistema, estos son: Modelo de Objeto, Dinámico y Funcional.

Modelo de Objeto

El modelo de objeto describe la estructura de objetos en un sistema, sus identidades, sus relaciones con otros objetos, sus atributos y sus operaciones. El modelo de objetos contiene diagramas de objetos. Un diagrama de objetos es un gráfico cuyos nodos son objetos clases y cuyos arcos son relaciones entre clases. Las clases se acomodan en jerarquías compartiendo estructuras y comportamientos comunes y están asociados con otras clases. Las clases definen los atributos acarreados por cada instancia objeto y las operaciones que cada objeto desempeña o lleva a cabo.

El objetivo de construir el modelo de objeto es capturar esos conceptos del mundo real que son importantes para una aplicación. Al modelar un problema de ingeniería, el modelo de objeto debe contener términos familiares para los ingenieros; al modelar una *interface* de usuario, deben contemplarse términos de dominio de aplicación. El modelo de diseño describe cómo resolver el problema.

Modelo Dinámico (Eventos)

El modelo dinámico describe esos aspectos del sistema concernientes con el tiempo y la secuencia de operaciones –Eventos que marcan cambios, secuencias de eventos, estados que definen el contexto de eventos, así como la organización de eventos y estados -. El modelo dinámico captura control, el aspecto de un sistema que describe las secuencias de operaciones que ocurren, sin referirse a lo que las operaciones hacen, en qué operan, o cómo están implementadas.

El modelo dinámico se representa gráficamente con diagramas de estado. Cada diagrama de estado muestra las secuencias del evento y estado permitidos en un sistema para una clase de objetos. Los diagramas de estado se refieren a otros modelos. Las acciones en el diagrama de estados corresponden a funciones del modelo funcional; los eventos en un diagrama de estados se convierten en operaciones en el modelo de objeto.

Modelo Funcional (Estructural)

El modelo funcional describe esos aspectos del sistema relacionados con las transformaciones de valores –funciones, mapeos, constantes y dependencias funcionales -. El modelo funcional captura lo que hace un sistema sin referirse en cómo o cuándo lo hace.

El modelo funcional se representa con diagramas de flujo de datos. Los diagramas de flujo de datos muestran dependencias entre valores y el cálculo de valores de salida de valores de entrada y funciones, sin referirse al momento o a la respuesta de las funciones que se ejecutan.

Diferencias entre metodologías Orientadas a Objetos y Funcional

El desarrollo orientado a objetos invierte la metodología previa orientada a funciones, como con las metodologías de Jourdan y De Marco. En estas metodologías, el énfasis principal se coloca en especificar y descomponer la funcionalidad del sistema. Esto podría parecer el modo más directo de implementar un objetivo deseado, pero el sistema resultante puede ser frágil. Si los requerimientos cambian, un sistema basado en la funcionalidad requiere reestructuración masiva.

Por el contrario, el desarrollo orientado a objetos se enfoca primero en identificar objetos del dominio de la aplicación, después ajustar procedimientos alrededor de éste. Aunque esto se ve más indirecto, el software orientado a objetos es mejor cuando los requerimientos evolucionan, ya que esta basado en el marco de trabajo del dominio de la aplicación y no en requerimientos funcionales de un solo problema.

Programación Estructurada

Objetivos de la Programación Estructurada

Ya que la ingeniería de *Software* se preocupa por el desarrollo y mantenimiento de la tecnología moderna, es necesario utilizar técnicas de resolución de problemas comunes a todas las ramas de la ingeniería; estas técnicas sientan las bases de la planeación y administración de proyectos, análisis de sistemas, diseño metódico, elaboración cuidadosa, validación profusa y mantenimiento continuo del producto. Es decir:

- Abatir costos de mantenimiento.
- Reducir tiempos de entrega.
- Generar sistemas confiables.
- Mejorar la administración del sistema.
- Eficacia en la Programación.
- Reducir la transferencia de control dentro de la programación (go to's).

Para efectuar esto, se requiere aplicar una notación adecuada, así como de herramientas y técnicas en cada área. La Programación Estructurada permite realizar un desarrollo adecuado que logre que la estructura del programa modele las características naturales de un problema determinado.

Técnicas de la Programación Estructurada

Debido a la falta de propiedades tangibles de un producto de *Software*, deben tomarse medidas especiales para determinar su situación durante el desarrollo, de tal manera que resulta fácil que una persona optimista diga que el producto se encuentra en un 95% completo, pero difícil para un ingeniero de programación o para su jefe asentar el progreso real del producto y definir las zonas del problema utilizando sólo su intuición.

Para garantizar que un producto de *Software* sea desarrollado bajo una metodología que permita cumplir con el objetivo de la programación estructurada deben tomarse en cuenta las siguientes características de desarrollo:

- Estructuras de Control.
- Modularidad.
- Diseño *Top-Down.*
- Diseño *Botton-Up.*
- Estructuras de Datos.

Estructuras de Control

Son aquellas que permiten llevar un orden lógico durante la programación, tales como: Estructuras SEQUENCE, IF - THEN - ELSE, CASE, WHILE - DO, REPEAT UNTIL, LOOP, EXIT – IF..

Modularidad

La Modularidad consiste en dividir el programa en varios grupos de subprogramas que reciben el nombre de módulos, los cuales pueden ser llamados a ejecución desde cualquier parte del programa. Esta división debe contemplar los efectos de las posibles modificaciones que se realizarán al programa desde el cual fue llamado el módulo.

Un módulo debe tener un tamaño que sea fácilmente comprensible, no debe ser llamado desde otro módulo que dependa de él y deben estar bien identificados el Nombre del Módulo, ¿Cuál es la función que realizará? y ¿Cuáles son las *interfaces* externas con otros módulos? (¿Qué espera recibir de entrada y Qué regresa como salida?).

Diseño *Top - Down*

El Diseño *Top - Down* es una metodología para diseñar programas usando las ideas de modularidad y de estructuras de control, consiste en atacar el diseño de lo general a lo particular.

Esta metodología es útil en cualquier tipo de problema, en particular en aquellos problemas que no conocemos o que son muy extensos. Los pasos a seguir en este diseño son:

- Comenzar el diseño del programa con el módulo principal.
- Definir que módulos se requieren de acuerdo a la lógica del problema (no del programa).
- La importancia de los módulos radica en **qué** es lo que harán, no **cómo** lo harán.
- Enfocar la atención en un módulo a la vez sin distraerse por los detalles de los demás módulos del programa.

Diseño *Bottom - Up*

Al igual que el Diseño *Top - Down*, este diseño se basa en las ideas de modularidad y estructuras de control, sólo que esta metodología ataca el diseño de lo particular hacia lo general. Esta metodología puede ser útil en problemas ya bien conocidos y particularmente útil si se usa parcialmente, o sea, parte del diseño es *Top - Down* y parte es *Bottom - Up*.

Los pasos a seguir en este diseño son:

- Comenzar el diseño del programa identificando las funciones elementales.
- Atacar una por una las funciones identificadas.
- Definir un módulo para elaborar la función identificada.
- Identificar otras funciones que requieran la utilización de los módulos ya diseñados.
- Atacar una por una las funciones ya identificadas hasta desarrollar un módulo que tenga como función el objetivo del programa.

Estructuras de Datos

Las Estructuras de Datos así como las funciones, representan una parte importante del programa, son muy diversas y pueden definirse como agrupaciones de datos que siguen una organización y un objetivo. Algunas de las más comunes en problemas típicos para aplicaciones administrativas son: Identidades, Acumuladores y Tablas.

Identidades

Las Identidades se definen como estructuras de datos que agrupan todos los campos necesarios para identificar un registro en sus distintos niveles. Las Identidades se usan para identificar uno o varios registros de un archivo y verificar la clasificación de los mismos.

Las ventajas de la Estructura de Datos Identidad son:

- Facilidad de manejo.
- Facilidad de modificación (añadir o eliminar un nivel de corte).

Acumuladores

Son una Estructura de Datos que agrupan todos los campos requeridos para acumular valores en un nivel de corte. La ventaja de esta estructura está en ver estos valores como a una unidad en lugar de verlos como distintos campos.

Tablas

Es una Estructura de Datos que agrupa una repetición de campos, y las variables requeridas para controlar dicha repetición. Las ventajas de esta estructura son:

- Ver todo el conjunto como una unidad.
- Facilidad de manejo.
- Definición y simulación de operaciones sobre la tabla (buscar elementos y modificarlos).
- Fácil de maniobrar (aumentar o disminuir la capacidad de la tabla).

1.3 Bases de Datos

Dentro de la tecnología de la información, las técnicas de bases de datos han ido aumentando su importancia y sus aplicaciones en los últimos años. La razón para ello ha estado en la demanda, siempre creciente, de utilización de grandes masas de datos, frecuentemente integrados, y con requisitos de alta productividad en su manejo, tanto en el desarrollo de aplicaciones por programadores, como por parte de usuarios

finales. Las bases de datos han respondido a esta necesidad, permitiendo estructurar éstos con técnicas más formalizadas, de uso más sencillo y con mayor capacidad para reflejar su significado.

Los principios teóricos de las bases de datos relacionales se establecieron y consolidaron en la década de los 70´s, durante los 80´s se diversificaron y hoy día en los 90 están pasando del ámbito de la investigación al de las realizaciones prácticas, expandiéndose rápidamente sus aplicaciones.

Al mismo tiempo, continúa la investigación teórica, pues los conceptos relacionales forman una base sólida para la exploración de nuevas áreas, como por ejemplo el enriquecimiento semántico del modelo de datos, las bases de datos distribuidas, nuevos tipos de datos (gráficos, voz), uso de técnicas de inteligencia artificial para formar bases de datos de conocimientos y otros.

Información

Dado que tanto el concepto de información como el de datos se utilizan en muy diferentes contextos y con múltiples matices, se acotará el significado de estas palabras al ámbito que nos interesa.

La información puede adoptar formas muy diversas. Por ejemplo:

- *Signos Escritos*: Libros, periódicos, facturas, letras de cambio, apuntes de contabilidad, cartas, jeroglíficos, dibujos, partituras musicales, inscripciones en piedra, en papiro, etc.
- *Sonidos*: Lenguaje hablado, música, los silbidos que son utilizados por comunidades indígenas, etc.
- *Señales de cualquier tipo*: Las señales de tráfico, los lenguajes con banderas entre los barcos, las señales de humo entre los indios del lejano Oeste, etc.

En todos estos ejemplos hay algo en común: un mensaje. Es decir, unas ideas vertidas por un ser humano sobre un soporte material para que otro ser humano las reciba. Las ideas representadas por el mensaje son la información de éste. Para nosotros, por tanto, el concepto de información va unido al proceso de la comunicación humana.

Nuestro ámbito de estudio es más restringido. Solo nos interesa la información que pueda representarse en mensajes procesables por máquinas. Las computadoras son máquinas que procesan cadenas de signos o caracteres, a gran velocidad y en grandes volúmenes.

Por lo tanto, nos restringiremos a la información aportada por mensajes formado por cadenas de signos.

Datos

Analicemos que tipos de mensajes son los que nos interesan. Consideraremos mensajes construidos con signos, combinando éstos en unidades de complejidad creciente ateniéndose a unas reglas que vienen definidas por el lenguaje en que se escribe el mensaje.

Para construir estos mensajes partimos de un conjunto finito que llamaremos alfabeto. A los elementos de este conjunto los llamaremos signos. Concatenando signos del alfabeto se forman unas cadenas, de longitud finita. No todas las combinaciones posibles de signos forman cadenas válidas en un lenguaje determinado. A las que sí lo son las llamaremos palabras. El conjunto de todas las palabras forman el vocabulario del lenguaje.

Muy frecuentemente las aplicaciones de las computadoras procesan mensajes construidos con un lenguaje muy simple, en el que normalmente cualquier combinación obtenida concatenando palabras del vocabulario constituye un mensaje formalmente válido. Cada palabra de un mensaje tiene un significado predefinido en función de la posición que ocupa dentro de él. Las palabras que forman el vocabulario se llaman *datos*.

Usando una terminología más común en ingeniería, cada mensaje es un registro y cada registro contiene varias palabras cuyo significado está ligado a su posición dentro del registro. El "hueco" o posición dentro de cada registro que ocupa un dato se llama campo. Los registros que tienen campos con el mismo significado predefinido se suelen agrupar en conjuntos llamados archivos.

Los mensajes que nos interesan son un subtipo de estos registros, formado por aquellos que tienen un número fijo de campos. En este caso los conjuntos de registros de igual significado, y por consiguiente con el mismo número de campos, se llaman tablas o relaciones. Los campos se suelen llamar atributos.

Este tipo de registros, sus conjuntos o tablas, sus componentes, atributos y datos, y la información presentada por ellos, constituyen finalmente la materia que vamos a tratar en las páginas siguientes.

Esquema Conceptual de Datos

Sea una empresa o entidad de cualquier tipo, que desea almacenar datos que reflejan información sobre sus actividades; hay que empezar acotando que parte del mundo exterior nos interesa representar en los datos. Estará formada por todos los objetos y acontecimientos cuyo conocimiento nos permita una mejor gestión de nuestras actividades.

Para ello, debemos aprender, comprender y conceptualizar este mundo, transformándolo en un conjunto de ideas y definiciones, que formen una imagen fiel del comportamiento del mundo real. A esta imagen del mundo exterior la llamaremos modelo conceptual.

Para construir un buen modelo, debemos utilizar una buena dosis de procesos mentales de abstracción, análisis y síntesis; además de la colaboración del personal directivo de la empresa.

Una vez definido el modelo conceptual, lo transformaremos en una descripción de datos, atributos y tablas, incluyendo las posibles interrelaciones entre estos elementos y su significado. A esta descripción la llamaremos esquema conceptual de datos.

A la operación de transformar el modelo conceptual en un esquema conceptual, la llamaremos diseño lógico de datos (por ello, al esquema conceptual también lo llamaremos a veces esquema de diseño).

Ya definido el esquema conceptual, hay que traducirlo a estructuras almacenables en soportes físicos controlados por la computadora, normalmente discos magnéticos. Esta transformación se le suele llamar diseño físico de datos.

Un buen diseño lógico debe producir un esquema conceptual que sea una imagen fiel y completa del modelo conceptual, incluyendo algunas interrelaciones y condiciones semánticas, es decir, aquellas que son consecuencia del significado de los datos.

A continuación definiremos los objetos con los que se construye el esquema conceptual: atributos, tablas, dominios, etc.

Dominios

Un dominio Di, es el conjunto de palabras, es decir, de ristras de caracteres tomados de un alfabeto dado. En general nos interesará definir también un conjunto de operadores sobre estas palabras, para poder manipularlas. Por ejemplo operaciones aritméticas y de comparación.

Un dominio puede ser un conjunto finito o infinito:
Ejemplos:

- Alfabeto:(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, /, -, ').
- Dominio D1: (0, 10, 11, 12, 13, 14, 15).
- Dominio D2: Conjunto de los números naturales.
- Dominio D3: (0, 1, 2, ½, 1/3, 2/3).
- Dominio D4: Conjunto de los números racionales.
- Dominio D5: (0, 1, 2, -1, -2).
- Dominio D6: (0, 3, 14, -3, 14).

D2 y D4 son infinitos.

El producto cartesiano Di y Dj, es otro conjunto, cuyos elementos son todas la parejas de valores que se pueden formar tomando un elemento de Di y otro de Dj. Lo representaremos como (Di x Dj).

Relaciones

Supongamos un conjunto D, de dominios:
D = (D1, D2, D3, ..., Dn).

Definamos entre ellos un criterio de orden que nos permita colocarlos a todos en una secuencia de menor a mayor (o de precedente a siguiente):
Secuencia: D1 < D2 < D3 < ... < Dn .

Formemos ahora un producto cartesiano con algunos o todos los dominios de D:
D1 x D2 x ... x Dk.

Los factores del producto anterior pueden estar repetidos. Por tanto: puede ser k = n, k < n o k > n. Los factores los colocamos en el orden definido por la secuencia citada anteriormente. Ejemplo: D1 x D2 x D3; D1 x D1 x D4; D1 x D2 x D2 x D3; etc.

Una relación, R, es un subconjunto finito de un producto cartesiano formado con las reglas expuestas. Si representamos con $\subseteq$ la propiedad de que un conjunto esté contenido en otro tendríamos:
$R \subseteq (D1 \times D2 \times D3 \times \ldots \times Dk)$. Cada elemento de una relación se llama tupla.

Se llama grado de una tupla al número de componentes que tiene. El grado de una relación es el de sus tuplas. Como R es un conjunto, todas sus tuplas deben ser diferentes. Es decir, en una relación no puede haber tuplas repetidas. Además, no hay un criterio de orden definido entre ellas.

Atributos

A cada dominio que participa en una relación se le da un nombre que lo identifica en forma única para esa relación. Entonces una tupla puede representarse o bien como una lista ordenada de valores:
t= < t1, t2, t3, ..., tk>, donde:
t1 = valor del primer dominio,
t2 = valor del segundo dominio,
t3 = valor del tercer dominio, etc.

O bien como una lista desordenada de valores identificados por sus nombres de atributos: t = < ATR3=t3, ATR2=t2, ATR1=t1, ..., ATRk = tk>.

Claves

Ya se ha dicho que como una relación R es un conjunto, todas sus tuplas han de ser diferentes, es decir, no se repiten sus valores.

A todo conjunto de atributos con esta propiedad se le llama superclave. Sea una superclave, tal que formada por los atributos A, B, C y D de una relación R. Si no es posible quitarle ninguno de sus atributos, sin que se pierda su cualidad de superclave, se dice que es una clave de R. En definitiva, una clave de R es un conjunto de atributos de R que no puede tomar valores repetidos, y tal que cualquier subconjunto suyo sí puede tomar valores repetidos, es decir, no es una superclave.

Toda relación tiene al menos una clave. Pero puede tener más de una. Esto depende del significado de la relación. Entre los valores que toma una clave de una relación y las tuplas de ésta existe una correspondencia biunívoca, de modo que las claves pueden utilizarse como identificaciones de las tuplas de la relación. El único medio de dirigirnos a una tupla determinada, distinguiéndola de todas las demás, es mediante los valores de los atributos de algunas de sus claves.

Tablas

Si colocamos todas las tuplas de una relación una debajo de otra, alineando los componentes correspondientes de cada atributo en una columna, y colocamos en la cabecera de cada columna el nombre de su atributo, obtendremos una representación de la relación a la que designaremos con el nombre de TABLA. Cada tupla forma una fila de la tabla.

Extensión e Intensión de una Relación

Al transcurrir el tiempo, una relación puede sufrir modificaciones para reflejar los cambios de mundo real que representa. Al añadir, borrar o modificar alguna tupla de una relación, ésta se transforma en otra distinta, pero con las mismas características.

Aquellas propiedades de una relación que no varían a lo largo del tiempo se conocen con el nombre de INTENSION. Dicho de otro modo, la intensión viene definida por todas las propiedades comunes a toda la familia de relaciones obtenibles al actualizar una dada. Estas características invariantes al actualizar una relación son las siguientes:

- El nombre de la relación.
- Los dominios.
- El producto cartesiano de dominios sobre el que se construye la relación.
- Los nombres de los atributos.

La intensión también la conoceremos como ESQUEMA de la relación. Llamaremos extensión de una relación a las tuplas que contiene en un momento dado, que pueden variar al actualizarla. El esquema se representa poniendo el nombre de la relación, seguido por los nombres de los atributos entre paréntesis y separados por comas. Así por ejemplo, R(A, B, C) es el esquema de una relación R con tres atributos A, B, C.

Base de Datos Relacional

Llamaremos Base de Datos Relacional, a un conjunto finito de esquemas de relaciones. Cada tupla de una relación indica que sus componentes están ligados entre sí por unas determinadas propiedades, que constituyen el significado de la relación. La información que una tupla determinada aporta a un usuario viene definida en parte por el significado de la relación y en parte por lo valores de sus atributos.

Asignar significado a un esquema de relación no es imprescindible desde un punto de vista mecanista del modelo relacional, pues aún cuando las tuplas que se incluyeran en una relación se hubieran elegido al azar, se seguiría teniendo una relación formalmente válida. Existen intentos de extensión del modelo relacional básico para que el sistema capture y gestione algo más del significado de las relaciones.

1.4 Arquitectura de la Información

En el pasado "los buenos tiempos" de los *mainframe*s, el modelo de cómputo era de *Host* y terminales tontas. Un grupo de terminales de vídeo con teclados conectados, con un poco o ningún tipo de "inteligencia" incorporada, estaban conectadas a una gran computadora central.

La computadora central - el *mainframe* - colocada en un gran salón especialmente configurado, rodeada por aire acondicionado y devotos sirvientes en batas blancas de laboratorio. En respuesta a las atenciones de las "batas blancas", la computadora central hacía todo - esto actualmente se conoce como ambiente de cómputo.

El *mainframe* procesaba todos los datos, administraba todas las lecturas y escrituras a disco, indicaba las rutas de impresión primero a los *buffers* y después a las impresoras, y manipulaba todas las imágenes de las terminales conectadas para entrada y despliegue de datos. Los usuarios hacían una petición de información o de impresión desde la terminal tonta, la computadora central hacía todo, lo hacía bien y lo hacía rápido.

El *mainframe*, sin embargo, tenía algunos pormenores, requería de un pequeño pelotón de programadores de computadoras para desarrollar las instrucciones que fueran necesarias para que el trabajo se hiciera. También requería de altos niveles de experiencia para comprender su sistema de archivos, esto restringía a personas ordinarias de intentos para explorar la información que tenía en su interior.

La gente que se dedicaba a atender las necesidades del *mainframe* así como la que se encargaba de programar las instrucciones eran muy hábiles y bien entrenados. Y como suele suceder en situaciones como ésta, desarrollaban su propia comunidad, la cual se centraba alrededor de las necesidades del *mainframe*, y posteriormente con el tiempo fallaban en obtener respuestas adecuadas para las necesidades del negocio que era propietario del *mainframe.*

Modelo Computadora de Escritorio

Como un negocio crece y cambia, éste debe tener más información, a una velocidad siempre creciente. La necesidad por un ambiente de cómputo que respondiera más rápido, llevó al desarrollo de una pequeña unidad autosuficiente en cuanto a procesamiento se refiere (computadoras personales de escritorio) que podían ser colocadas en el escritorio de las personas.

Este *mainframe* en miniatura podía hacer de todo; entrada de datos, manipulación de los mismos y reportes, desde la comodidad de la propia oficina, y sin tener que lidiar con la comunidad de gente que atendía al *mainframe*. Personal administrativo ordinario comenzaba a comprender que ellos podían tomar el control de los datos y su presentación, llegando más allá de lo que el *mainframe* podía hacer por ellos. En definitiva, ellos podían investigar la información.

Para obtener información de un ambiente de cómputo altamente estructurado, se debe conocer la forma de realizar preguntas "correctas". Y en la planeación estratégica para negocios, normalmente la pregunta no está bien definida; ya que se requiere de la capacidad de visualizar datos, combinar y recombinar elementos de datos en un orden casi ilógico, todo en un esfuerzo para obtener nueva información. La metodología *mainframe* no estaba preparada para soportar este tipo de actividad.

Igualmente el proceso de escritorio tiene sus problemas. Cada computadora de escritorio se convierte en una "isla de información". Todos los usuarios de computadoras tienen sus muy personales formas de almacenar y presentar su información, algunos utilizan programas de hojas electrónicas para guardar su información, otros utilizan procesadores de texto y documentos; y otros más utilizan sistemas de manejo de bases de datos.

Sin tener un grupo que controle un estándar para los programas, se llega a tener una diversidad de paquetes de información y aunque algunos de estos datos actualmente tengan mucho valor, por causa de restricciones técnicas, el compartir información entre computadoras de escritorio y el *mainframe* es virtualmente imposible.

Modelo Red de Área Local (LAN)

Entrando la era de las redes de área local. La LAN utiliza cableado físico y capas de *Software* para construir conexiones entre todas estas "islas de información" en los escritorios individuales. La idea fue el conectar a todas las computadoras personales y permitir a sus usuarios compartir archivos y periféricos como impresoras y escáners.

La primera LAN introdujo su propio tipo de modelo de cómputo - servidor de archivos. Todas las computadoras en la LAN eran conocidas como nodos, donde existen dos tipos de diseño, por jerarquías o el llamado de punto a punto. En el diseño de punto a punto, un nodo podía solicitar un servicio a otro nodo, como "Envíame una copia de ese archivo que está en tu disco", o "Imprímeme este trabajo en la impresora que tienes conectada".

La LAN tipo punto a punto, daba la capacidad a cada nodo de actuar como un servidor (un proveedor de archivos o servicios de impresión para el resto de la red).

La LAN jerárquica fue configurada un poco diferente, con un solo nodo de servidor que provee diversos servicios a los otros nodos, aquí es de donde proviene el nombre que inherentemente marca la diferencia, los nodos son llamados clientes.

Cada nodo cliente puede preguntar al servidor por una copia de un archivo almacenado en el disco duro del servidor, solicitar al servidor la impresión de un documento, solicitar al servidor un envío de fax, etc.

El servidor provee todos los servicios del sistema, como administración de archivos y transferencia de documentos, pero los nodos cliente hacen todo el procesamiento de la información y presentación en pantalla. Este es el punto donde esta división del trabajo comenzó a ser llamada "cliente – servidor ".

Modelo Cliente – Servidor

La idea que existe detrás del proceso cliente/servidor es dividir la carga de trabajo y distribuirla a todos los componentes dentro del ambiente de cómputo. Existen diversas formas para configurar un ambiente cliente/servidor, pero básicamente, éste gira alrededor del concepto de balance de carga. El ambiente óptimo para cliente/servidor debe ser diferente para cada compañía o cada departamento de una compañía, dependiendo de las necesidades y requerimientos del negocio.

El ideal es tener uno o varios nodos servidores, administrando los archivos, controlando transferencia de documentos, procesando algunos datos (cuando sea apropiado) y tener nodos clientes para hacer la otra parte del proceso de la información, presentaciones y dibujo en pantalla.

Un ambiente cliente-servidor está típicamente formado por los siguientes tres componentes:

- ***El Cliente***: El cliente o los clientes son las computadoras de escritorio, que hacen solicitudes de tareas al servidor. Normalmente, la computadora cliente es responsable de interpretar las instrucciones capturadas por medio de teclados, ratón u otro periférico de entrada; procesar estas instrucciones, y (si se necesita procesamiento adicional que ella misma no pueda realizar) enviarlas al elemento apropiado dentro de la red. El cliente es responsable por el despliegue en pantalla y controlar la personalización del ambiente para el usuario. Maneja la presentación de la información e instrucciones que son enviadas de cualquiera de los servidores en la red.

- ***El Servidor***: El servidor es la computadora que actúa como un proveedor de servicios para el resto de las computadoras en una red. Típicamente es responsable de las tareas a nivel sistema; dirigir solicitudes de impresión a la impresora apropiada, dirigir consultas de información a las bases de datos correctas, regresar archivos y aplicaciones a los nodos clientes después de una solicitud, y actuar como un concentrador de comunicaciones para el fax módem.

- ***La Red***: (y cualquier elemento intermedio) que conecta al cliente y el servidor – sin una red el esquema cliente/servidor no existe. La red física – los cables y conectores – que por sí mismos son únicamente el comienzo. Apilados en el cableado físico, esta capa sobre capa de *Software* que comprende el sistema operativo de red y los protocolos de comunicación. Adicionalmente, algunos ambientes de cliente/servidor que requieren de una respuesta en extremo rápida y/o necesitan manejar altos niveles de actividad – una base de datos OLTP (*online transaction processing*), por ejemplo puede existir un elemento intermedio que administre el flujo de instrucciones y búsquedas que fluyen hacia el servidor.

La computación cliente/servidor comenzó como un sistema de información departamental y ahora se mueve hacia las empresas; cruzando a través de los límites departamentales y divisionales, dando servicio a la corporación entera.

Beneficios del Modelo Cliente – Servidor

Hoy día el clima en los negocios es muy competitivo, cualquier compañía desde los gigantes industriales a las corporaciones de una sola persona tienen que operar más eficientemente que nunca. Para algunos esto significa reducciones; para otros, eficiencia significa reingeniería de procesos. La tecnología cliente/servidor ha hecho posible la reingeniería en muchas corporaciones. Actualmente es muy difícil separar la causa del efecto.

El introducir conceptos cliente/servidor requiere un replanteamiento del proceso de negocio. La reingeniería efectiva, donde la autoridad y responsabilidad es colocada en las manos de empleados, requiere que los empleados tengan la información necesaria, la cual es mejor accesada por sistemas de información cliente/servidor.

Una vez que hemos abarcado los antecedentes principales a considerar en nuestro trabajo, en el siguiente capítulo comentaremos las funciones que los Bancos realizan dentro de la economía de un país.

CAPITULO 2

GENERALIDADES

En este capítulo trataremos de dar una perspectiva de lo que es la Banca Comercial Mexicana, abordando un poco la estructura del Sistema Financiero Mexicano, así como su historia y los servicios que ésta ofrece al público, para dar un panorama general de la importancia de la Banca Comercial dentro de la economía nacional y su importancia como institución.

2.1 Conceptos Generales sobre Mercados Financieros

A efecto de tener una mejor comprensión del papel que desempeñan los organismos financieros conocidos como Bancos, es preciso referirnos a las funciones que los mismos realizan dentro de la economía de un país. Los Bancos en conjunto representan un papel fundamental en el desempeño de las políticas monetarias de una nación, además de que constituyen una fuente importante de generación de empleo estable.

Individualmente los Bancos constituyen un esquema de empresas susceptibles de operar exitosamente o de fracasar en sus actividades, con lo cual benefician o lesionan, respectivamente los intereses de los inversionistas.

En forma específica los Bancos en el país son elementos partícipes de lo que en esencia podríamos denominar el Sistema Financiero Mexicano.

El Estado en su función primordial de órgano rector de la economía del país, establece la reglamentación y las condiciones necesarias para permitir la operación de todas aquellas transacciones de carácter público o privado que tengan una connotación monetaria.

En realidad, cuando se habla de un sistema financiero se está haciendo referencia a la representación práctica del sistema económico de un país, es decir, de las características, regulación y ámbitos físicos en donde se desarrollan actividades de tipo monetario.

De hecho, tal sistema viene a ser conformado por un núcleo de mercados financieros que en sí, están orientados al desempeño de las siguientes funciones:

1. Proporcionar los medios y mecanismos necesarios para que se lleven a cabo de una manera ordenada y eficiente transferencias de fondos entre entidades que generan excedentes de fondos y aquellas que carecen y requieren de los mismos.

2. Mantenerse dentro de los márgenes de disponibilidad y eficiencia operativa, bajo cualquier circunstancia, aun en aquellos casos de desaceleración o recesión económica.

3. Contener características de adaptabilidad en materia de recursos y desarrollo, en funciones a las variaciones existentes en la actividad económica del país.

4. Proporcionar las condiciones necesarias que permitan a la Hacienda Pública mantener la regulación y el control de los factores monetarios y de crédito, evitando con ello la presencia de elementos distorsionantes en cuanto al equilibrio de los diferentes sectores que conforman el sistema económico nacional.

Los propósitos y consecuentes funciones de los mercados financieros pueden ser cubiertos desde dos diferentes ámbitos o dimensiones:

- En forma directa. Que se da cuando el prestador de los bienes monetarios asume la totalidad del riesgo derivado de la transacción de préstamo que a su vez genera una obligación para quien recibe en forma temporal.

- En forma indirecta. Que se produce cuando una tercera entidad, denominada Institución Financiera, juega el papel de intermediaria entre ambos contratantes.

A los mercados directos se les conoce generalmente bajo el rubro de mercados de dinero, y los mismos, se caracterizan en forma básica por la duración de las transacciones respectivas, que no excede a un periodo de un año.

Por su parte los Mercados Indirectos, de los cuales forman parte los Bancos, son mejor conocidos como Mercados de Capitales y en los mismos la duración de las operaciones con valores es superior a un año.

El Sistema Financiero Mexicano

En esencia la presencia de mercados financieros implica necesariamente la presencia de un conjunto de elementos interrelacionados que permita y promueva su funcionamiento. A dicho conjunto puede darse el título de sistema financiero, mismo que por lo general suele verse de moderada a altamente regulado por el sector gubernamental de un país como resultado de su decisivo impacto en el comportamiento de la economía de una nación.

El sistema financiero mexicano está formado por una serie de entidades de carácter público y privado a cuya cabeza se encuentra la Secretaría de Hacienda y Crédito Público, máximo organismo regulador y rector del sistema de quien jerárquicamente depende en forma directa el Banco de México, el cual es un organismo público descentralizado del Gobierno Federal con personalidad y patrimonio propios que en su papel de Banco Central del país detenta las siguientes funciones:

- Regular la emisión y circulación de la moneda, crédito y cambios.
- Operar como banco de reserva de las instituciones de crédito y a la vez regular el servicio de cámara de compensación en lo referente a transacciones entre tales instituciones.
- Servir como tesorero del gobierno federal y actuar como su agente financiero en operaciones de crédito interno y externo.
- Fungir como asesor del gobierno federal en materia económica y financiera.
- Participar en el Fondo Monetario Internacional y demás organismos existentes en materia de cooperación financiera internacional o que agrupen a bancos centrales.

Las funciones anteriores las desempeña mediante tres órganos primarios administrativos, como lo son su Director General, una Comisión de Crédito y Cambios y la Junta de Gobierno.

A su vez, de la Secretaría de Hacienda y Crédito Público, dependen en forma directa dos organismos desconcentrados de supervisión y vigilancia denominados:

- Comisión Nacional Bancaria y de Valores.
- Comisión Nacional de Seguros y Fianzas.

Mismos que por su parte tienen la facultad de emitir las disposiciones necesarias para el ejercicio de sus funciones, así como de aplicar las sanciones que en términos de las leyes correspondientes determinen sus respectivas juntas de gobierno.

De la cúspide del Sistema Financiero Mexicano se derivan tres subsistemas que son los siguientes:

- Bancario.
- Bursátil.
- Seguros y Fianzas.

Subsistema Bancario

Se subdivide por su parte en los componentes siguientes:

- Comisión Nacional Bancaria y de Valores.
- Banco de México.
- Instituciones de Crédito.
- Organizaciones Auxiliares de Crédito.
- Patronato del Ahorro Nacional.
- Fideicomisos Públicos del Gobierno Federal.

A su vez, las instituciones de crédito se agrupan en dos ramas principales:

- Banca Múltiple.
- Banca de Desarrollo.

La Banca Múltiple o Banca de Servicios Múltiples, equivalente a lo que se conoce en otros países como Banca Comercial, tiene como función primordial la captación de recursos, provenientes del público dentro del mercado nacional para fines de su colocación, en base al otorgamiento de créditos a personas físicas (personas con derechos y obligaciones por sí mismas) y morales (agrupaciones formadas por varias

personas físicas), mediante operaciones pasivas o contingentes; quedando el intermediario financiero obligado a restituir los recursos captados, más, en su caso, los intereses correspondientes al plazo y la tasa pactadas.

Las organizaciones auxiliares del crédito corresponden en nuestro país a las siguientes áreas de actividad:

- Arrendadoras Financieras.
- Almacenes Generales de Depósito.
- Uniones de Crédito.
- Empresas de Factoraje Financiero.
- Sociedades de Ahorro y Préstamo.
- Sociedades Financieras de Objeto Limitado.
- Las demás que otras leyes consideren como tales, como lo son las Casas de Cambio.

La Banca de Desarrollo gira en torno al financiamiento de inversión en infraestructura civil, tal como puentes, carreteras, presas, etc.

Subsistema Bursátil

El Subsistema Bursátil se enfoca principalmente en los siguientes campos de acción:

- Mercado de Dinero.
- Fondos de Inversión.

El Objetivo del Mercado de Dinero consiste en crear los mecanismos necesarios para realizar las operaciones pactadas en Moneda Nacional, UDIS y Moneda Extranjera entre Instituciones Bancarias, abarcando los siguientes instrumentos tanto para personas físicas como morales:

- Pagaré liquidable al vencimiento en Moneda Nacional.
- Pagaré con rendimiento liquidable al vencimiento en UDIS.
- Cuenta de inversiones.
- Preestablecidos.

Por su parte, los Fondos o Sociedades de Inversión se conforman por una agrupación de personas interesadas en invertir su capital en los diferentes productos que ofrece el fondo de inversión, ya sea a renta fija o variable.

Las Sociedades de Inversión pueden participar en la Bolsa, Uniones de Crédito y Productos Propios de la Sociedad en cuestión.

Seguros y Fianzas

El campo de acción de este subsistema se basa en crear programas de protección de cualquier índole tanto para personas físicas como morales, es decir, las instituciones que prestan sus servicios como aseguradoras o afianzadoras, estudian las necesidades de los clientes prospecto para ofrecerles un producto adecuado a las mismas.

Las compañías aseguradoras "venden" seguridad para sus clientes, o dicho de otra forma, "compran" el riesgo al que sus clientes se encuentran expuestos dentro de sus acciones cotidianas, siempre y cuando, las cláusulas del contrato firmado por el asegurado así lo establezcan. La compañía aseguradora deberá indemnizar al asegurado por la cantidad establecida al momento de realizar el contrato, por otra parte, el asegurado deberá cubrir una cantidad (prima) en un periodo determinado dentro de las cláusulas del seguro para cubrir el costo del mismo.

Los diversos riesgos que pueden afectar negativamente la economía de una persona física o moral se encuentran ubicados en cinco áreas:

1. **Riesgos de las propiedades físicas**: Son todos aquellos eventos que afectan bienes que forman parte del patrimonio de asegurado. Encontramos aquí el seguro de incendio, huracán, explosión, y los de transporte marítimo y de automóviles.

2. **Riesgos que nacen de las leyes**: Son aquellas eventualidades en las que la acción u omisión de una persona causa un daño y por lo tanto la ley exige, de acuerdo al tipo de daño, una reparación que puede estar representada por montos económicos o bien alguna sanción, por ejemplo el seguro de Responsabilidad Civil, del tipo: general, familiar, de accidentes personales de los viajeros y el profesional.

3. **Riegos nacidos de actos criminales**: Son todas aquellas afectaciones económicas provenientes de la comisión de actos delictivos por parte de terceros, Por ejemplo: robo, fraude, etc.

4. **Riesgos consecuenciales o intangibles**. Están representados por la falta de percepciones económicas generada por la improductividad de un bien material que se ha dañado en alguna forma y queda por lo tanto temporalmente impedido para

su explotación, por ejemplo perdida de utilidades, perdida de rentas etc. En este tipo de riesgo encontramos el seguro de obras civiles en construcción, montaje o rotura de maquinaria , equipo de contratistas, calderas y aparatos sujetos a presión, equipo electrónico, cristales, de anuncios luminosos, etc.

5. **Riesgos personales**: Son todos aquellos eventos que afectan la existencia, integridad física, salud o vigor vital en una persona, por ejemplo muerte, incapacidad, accidentes, etc.

2.2 Banca de Servicios Múltiples

La Banca de Servicios Múltiples que existe en la actualidad en el país tiene sus primeros antecedentes en el siglo pasado, cuando, y mediante la aportación de capitales preponderantemente extranjeros, las instituciones operaron bajo condiciones tanto favorables como adversas durante las primeras tres décadas del presente siglo.

Aunque como consecuencia de la nacionalización de la Industria Petrolera se observó como todos los bancos extranjeros, menos uno, abandonaron el país, dejando en manos de inversionistas nacionales la propiedad de las instituciones de crédito.En aquel entonces los bancos operaban en forma especializada, pudiéndose distinguir 5 tipos de ellos:

- Bancos de Depósito.
- Bancos de Capitalización.
- Bancos Hipotecarios.
- Bancos de Ahorro y Préstamos para la Vivienda.
- Sociedades Financieras.

Con posterioridad, en 1970 se modificó la Ley Bancaria vigente en ese entonces para dar cabida al referido concepto de Banca Múltiple, la que vino a agrupar en una sola entidad las funciones y facultades de esos cinco diferentes Bancos.

Hacia fines del año de 1982 el Gobierno Federal optó por estatizar la Banca Privada, refiriéndose a los Bancos como Instituciones Nacionales de Crédito. La Banca permaneció en el poder del estado hasta el año de 1991, año en que se inició la llamada desincorporación de las Instituciones Bancarias, las cuales, durante el periodo en el que formaron parte del gobierno, vieron reducido su número de más de 40 empresas iniciales a sólo 18, como producto de una serie de fusiones que se realizaron en ese entonces.

Como consecuencia de la crisis económica de fines del año de 1994, se produjo una disminución dramática en el número de Instituciones de Crédito en el país, que a mediados del referido año alcanzaba un número de 35 Bancos, dado que muchos de ellos en mala situación financiera fueron adquiridos por Bancos de mayores dimensiones o con mejor posición de resultados y capitalización, así como por instituciones extranjeras, que a partir de 1995 tuvieron autorización para detentar hasta el 100% del capital social.

A continuación se presenta una lista de los principales Bancos de carácter comercial existentes en el país. Tanto nacionales como extranjeros:

- Banamex
- Bancomer
- Serfín
- Bital
- Inverlat
- BBV
- Santander
- Banorte
- Banpaís
- Bancreser
- Promex
- Inbursa
- Interacciones
- Chase Manhattan Bank México
- Bank of America México
- Banco de Boston
- City Bank

Resulta importante señalar que en la actualidad la casi totalidad de la Banca privada del país opera como participante de grupos financieros en donde, además de los bancos, se tiene el control sobre entidades financieras tales como Casas de Bolsa y de Cambio, Aseguradoras, Afianzadoras, Empresas de Factoraje, Almacenadoras y Arrendadoras.

Actualmente existen instituciones de Banca Múltiple, de capital nacional, extranjero y mixto, las cuales se encuentran de nuevo en manos de particulares en lo que se refiere a su administración y control de decisiones. En razón del continuo proceso de transformación que han sufrido las Instituciones Bancarias de éste y otros países,

resulta difícil identificar en el presente patrones comunes de funcionamiento entre los mismos.

Sin embargo, en términos generales se pueden distinguir las siguientes áreas básicas de operación:

- La captación de dinero mediante atractivos intereses de cuentas de ahorro a corto, mediano y largo plazo representa un porcentaje importante para generar reservas monetarias que se emplearán a su vez en el otorgamiento de créditos o en reinversiones del capital.
- Se efectúan préstamos, tanto a empresas como a un individuo y adicionalmente, se invierte en valores los excedentes.
- Los Bancos, al igual que cualquier otra empresa, son operados en relación a un propósito de lucro, es decir, obtención de utilidad por sus servicios.
- Estas instituciones deben de una manera permanente buscar y obtener un equilibrio entre el deseo de obtener niveles razonables de rentabilidad, derivados de sus funciones de crédito e inversión, y un rendimiento esencial de operar dentro de niveles altos de seguridad y liquidez.

Una manera muy simple de poder entender como opera un Banco de tipo Comercial o Múltiple, consiste en observar su estado de situación financiera. Por ejemplo, vamos a suponer las siguientes cifras expresadas con un nivel máximo de agregación:

Estado de Situación Financiera al 31 de Diciembre de 1996.
Cifras en Millones de pesos

Efectivo	12.80	Depósitos	218.30
Inversiones en Valores	11.14	Otros Pasivos	38.50
Préstamos	174.00	Capital Contable	
Activos Fijos Netos	65.60	Capital Social	9.00
		Superávit	3.40
Total Activo	269.20	Total Pasivo	269.20

En el que el lado izquierdo muestra una partida sobresaliente la cuenta de Préstamos, que son precisamente el importe que le deben a la institución las personas morales y físicas a las cuales les ha concedido créditos que aún se encuentran pendientes de liquidar, es decir, que la función de prestar dinero es una de las funciones más importantes de una institución bancaria.

Por otra parte, el lado derecho de este estado financiero nos reporta como partida sobresaliente el renglón de los depósitos, que corresponde a las entregas efectuadas de los cuentahabientes del banco para fines de su custodia y/o inversión, es decir, corresponde a la captación por Ahorro, la cual constituye otra función fundamental de este tipo de instituciones.

En los siguientes apartados se realizará una explicación general sobre:

- Cuentas de Cheques.
- Cuentas de Ahorro.
- Cuentas de Inversiones.
- Tesorería y Riesgos (Banca Especializada)

Los productos anteriores se consideran como los más importantes sobre los cuales gira una Banca de Servicios Múltiples.

Cuentas de Cheques

Una Cuenta de cheques se define como un contrato mediante el cual la institución bancaria acepta depósitos en efectivo o en títulos de crédito (pagarés, letras de cambio, etc.) a favor del cliente; quien podrá hacer disposiciones a través formatos de cheques que la Institución bancaria otorgará al depositante.

Se pueden distinguir los siguientes productos de Cuentas de Cheques:

- Cuentas de cheques en Moneda Nacional.
- Cuentas de cheques en Dólares Americanos.
- Cuenta Maestra.
- Cuenta Productiva.

Cuentas de Cheques en Moneda Nacional

Es un instrumento susceptible de ser utilizado tanto por personas físicas como morales, el cual se constituye mediante depósitos de dinero, que a su vez se hacen accesibles por medio de la expedición de cheques.

En este instrumento intervienen tres sujetos: *El Librador* que es el propietario de la cuenta, *El Librado* que es el Banco en el que se encuentran depositados los fondos, y *El Beneficiario* que es el sujeto a nombre de quien se expiden los cheques.

Las cuentas de cheques tienen como ventajas principales para los clientes las siguientes:

- Seguridad al conservar los fondos en efectivo que están protegidos de robo o extravío.
- Comodidad al utilizar la chequera para poder hacer pagos y evitar traer consigo sumas importantes de dinero.
- Permite con facilidad llevar un registro de depósitos y retiros.
- Se recibe un estado de cuenta mensual, que permite llevar un control del manejo de los fondos.
- La expedición de cheques nominativos es un comprobante fiel de pago de adeudos.

Cuentas de Cheques en Dólares Americanos

Este Instrumento Funciona igual que la Cuenta de Cheques en moneda Nacional, con la característica de que sus principales clientes son personas morales domiciliadas en cualquier parte de la república y personas físicas que radican en poblaciones en una franja de 20 kilómetros a la línea divisoria internacional, así como en los Estados de Baja California y Baja California Sur.

Las persona morales podrán depositar en cualquier parte del país, en tanto que las personas físicas solo podrán hacerlo en las sucursales ubicadas dentro de la franja fronteriza mencionada. Los mismos deberán ser en dólares americanos, en documento o billete.

Cuenta Maestra

Este instrumento consiste en dos cuentas, una cuenta eje, la cual requiere un monto mínimo para poder funcionar, y una cuenta de inversión, sobre la cual se generarán intereses capitalizables en la cuenta eje.

Los depósitos generan rendimientos y están a la vista, es decir, disponibles en cualquier momento, para este fin se dispone de una chequera especial. Tales

rendimientos se calculan sobre saldos promedios diarios y se capitalizan al día primero del mes siguiente acreditándose su importe en la Cuenta Maestra.
Se pueden girar todos los cheques que se deseen, aunque sólo los primeros (5 a 10) son gratuitos, y los subsecuentes originan el pago de una comisión.

Los impuestos causados por los intereses pagados son directamente retenidos por el Banco y liquidados posteriormente por el mismo.

Cuenta Productiva

Tiene las mismas características y ventajas de una Cuenta de Cheques, a excepción de que paga intereses, que se calculan sobre el porcentaje de rendimiento de los CETES, y en función a un valor que se incrementa en la medida que aumenta el importe de la inversión.

Los intereses se capitalizan mensualmente, y a diferencia de una cuenta de cheques se suele cobrar por cada cheque expedido.

Cuentas de Ahorro

Es un instrumento que permite obtener un interés neto anual capitalizable mensualmente, que se ejerce mediante una libreta, cuyos saldos se actualizan en forma semestral.

El dinero depositado se encuentra a la vista y para depositar o retirar se debe acudir a la sucursal, a efecto de anotar en la referida libreta las anotaciones correspondientes.

Generalmente asociado con un seguro de vida sin costo alguno o examen médico aunque la edad del depositante no debe exceder de 55 años para este beneficio.

Se considera patrimonio y por lo tanto no es embargable, salvo cuando se trate de pensiones alimenticias o de créditos otorgados por el Banco.

Se documenta mediante un recibo de valores en administración, a efecto de evitar trámites legales en el caso de pérdida o extravío.

Cuentas de Inversiones

Fondos Bancarios. Instrumento exclusivo de personas físicas, en donde el capital se encuentra invertido en un fideicomiso, que se maneja a través de dos saldos.

- Saldo Invertido, que es el total de la inversión del cliente en el fondo.
- Saldo Disponible, que es la parte del saldo invertido que ya cumplió dos días hábiles en el fondo.

Los depósitos en estos fondos suelen generar intereses desde el mismo día en que se efectúen, mediante capitalización diaria del rendimiento. Los recursos se encuentran disponibles dos días después de haberse efectuado.

Para el manejo de estos fondos se requiere que el cliente tenga una cuenta de cheques tradicional, productiva o maestra, a efecto de poder operar en ésta los movimientos del Fondo.

Se genera también un estado de cuenta mensual por las operaciones efectuadas en el fondo.

Servicios Fiduciarios

Se refieren al establecimiento de fideicomisos, los cuales son contratos en donde se destinan ciertos bienes o derechos a un fin lícito y determinado, por parte de cualquier persona física o moral.

En un contrato de fideicomiso intervienen los siguientes sujetos: El Banco en su calidad de Fiduciario, que es quien recibe de una persona física o moral llamada Fideicomitente, ciertos bienes o derechos para que realice determinados fines, en beneficio del propio fideicomitente o de terceras personas que éste mismo señale, a quienes se les denominará fideicomisarios.

- Fideicomisos de Seguro

Las personas que cuentan con Pólizas de seguro de vida pueden designar a través de un fideicomiso, a una institución fiduciaria como beneficiaria, para que a su fallecimiento ésta proceda a efectuar el cobro de la suma asegurada y utilizarla para cumplir con las finalidades establecidas en el fideicomiso, entregando los beneficios correspondientes a los fideicomisarios elegidos en vida por el fideicomitente.

- Fideicomiso de Inversión

Mediante este fideicomiso el fideicomitente le hace entrega al fiduciario de una determinada cantidad de dinero, a fin de que éste lo administre, invierta o reinvierta en títulos o valores que ofrezcan la mayor rentabilidad y seguridad, en beneficio del fideicomitente generalmente, o bien, de otras personas designadas por él.

- Fideicomiso Inmobiliario

En este fideicomiso se transmite en forma temporal a la institución fiduciaria la propiedad de un determinado inmueble para que éste lo conserve hasta que el fideicomitente lo indique, o para que en su caso lo transmita a los fideicomisarios.

- Fideicomiso de Garantía

Mediante este contrato se garantiza el cumplimiento de una obligación y puede constituirse sobre bienes inmuebles o sobre valores y derechos.

En caso de incumplimiento se procede por parte de la fiduciaria a la venta de los bienes objeto del fideicomiso, a efecto de que con su producto se cubra el adeudo correspondiente.

- Fideicomiso para Inmigrantes Rentistas

Por medio de este contrato se permite que extranjeros que deseen radicar legalmente en el país puedan hacerlo de conformidad con las leyes migratorias, dado que garantiza su solvencia económica por medio de los productos generados por los bienes objetos del fideicomiso, generalmente fondos entregados a la institución para su inversión.

- Fideicomiso para Uso y Aprovechamiento de Inmuebles

Cualquier persona física o moral puede a través de este fideicomiso, utilizar o aprovechar un inmueble sin que para ello le sea necesario adquirir la propiedad del mismo.

- Fideicomiso de Fondo de Ahorro

Con este fideicomiso una empresa, sus trabajadores, o ambos, pueden crear un fondo mensual de ahorro, aportando periódicamente cantidades con las cuales se

constituirá el fideicomiso de referencia, a efecto de que la institución fiduciaria proceda a administrar ese fondo invirtiéndolo en valores de alto rendimiento.

- Fideicomiso para la Adquisición de Inmuebles dentro de Zonas Prohibidas (Para el establecimiento de Maquiladoras)

La Constitución Política Mexicana prohibe a los extranjeros adquirir en propiedad el dominio directo de bienes inmuebles en ciertas zonas prohibidas, como lo es la franja fronteriza

Mediante este tipo de fideicomisos el extranjero tiene oportunidad de asegurar y disfrutar una propiedad sin que se constituyan derechos reales sobre los mismos.

Mesa de Dinero

El Cliente (Mandante) celebra un contrato de mandato en el cual el Banco (Mandatario) invierte en nombre de aquel sus recursos en instrumentos del mercado de dinero, tales como: Certificados de la Tesorería (CETES), Pagarés y Aceptaciones Bancarias: a los plazos acordados en el contrato.

Si al vencimiento del contrato de mandato el cliente no gira nuevas instrucciones, el Banco acreditará el monto de la inversión y sus rendimientos en una cuenta de cheques, también denominada cuenta eje. Respecto de este instrumento se emite un estado de cuenta mensual, con el desglose de todas las operaciones pactadas durante el período.

2.3 Banca Especializada

La división de Banca Especializada es responsable de las operaciones de la tesorería, de banca patrimonial (personas físicas con grandes inversiones), así como operaciones de casa de bolsa, de cambios y de servicios fiduciarios. La unidad de comercio exterior adscrita a la división de banca especializada es responsable del diseño, estructuración, operación, normatividad y asesoría en productos de crédito y programas relacionados con la exportación e importación.

Tesorería es una unidad de negocio creada para atender a los usuarios de tasas (intereses determinados por el mercado financiero) de una Institución. El objetivo

principal es lograr consolidar a la entidad bancaria como una unidad de negocios corporativa, rentable y más competitiva en el mercado financiero a través de un servicio especializado y oportuno, con el objeto de lograr una mejor competitividad de las unidades de negocio del banco mediante: captación (determinación de tasas pasivas) y colocación (fijación de tasas activas).

Riesgos

Riesgos es el área encargada de diseñar y coordinar el desarrollo e implantación de los proyectos de información de riesgo y sistemas de decisión de la Dirección de Crédito y Gestión de Riesgos, reforzando las funciones de las áreas participantes en el proceso de crédito; para identificar, valuar y administrar los riesgos de crédito, mercado y operación de la institución; apoyar la toma de decisiones de estrategias de colocación y recuperación; y particularmente en la administración, seguimiento y monitoreo del riesgo de los activos.

Evaluación del Riesgo

Es necesario realizar un análisis formal para identificar los riesgos específicos asociados con los activos críticos de información. El resultado más importante del proceso de evaluación de riesgo es la información que se utiliza para desarrollar e implantar las políticas de seguridad.

Cada unidad de negocio de la institución debe identificar a un ejecutivo responsable para cada aplicación crítica o conjunto de información. Ese mismo ejecutivo debe conducir un ejercicio de análisis de riesgo cada año como mínimo para asegurar tanto la integridad como la disponibilidad y confidencialidad de la información.
Los siguientes riesgos y áreas expuestas deben ser considerados al conducir el análisis de riesgo.

- Impacto sobre el banco si los clientes son atendidos inapropiadamente, o si la información acerca de ellos es inadecuadamente vista, accesada, exhibida, o modificada.
- Impacto sobre la empresa si los empleados violan las licencias, privacidad, o requerimientos legales.

- Impacto sobre la empresa si la gente no autorizada obtiene acceso a información acerca de empleados, estrategias comerciales, información de clientes, políticas y otra información confidencial.
- Impacto sobre el banco si los procedimientos de seguridad reducen la productividad, tiempo de respuesta del sistema o disponibilidad del mismo.
- Impacto sobre el banco si las unidades de negocio y/o sucursales consideran que no tienen el control adecuado sobre la protección de información de los clientes a quienes brindan el servicio.

2.4 BANPAIS como Banca de Servicios Múltiples

Historia de BANPAIS

Desde su fundación BANPAIS ha desempeñado un papel importante en el desenvolvimiento de las actividades económicas del país. A lo largo de más de cien años ha dado muestras de un dinamismo y una gran capacidad de adaptación que le han permitido llegar a ser una institución financiera de primer orden, con recursos, experiencia y personal altamente capacitado que le permite cubrir el amplio campo de los servicios financieros.

BANPAIS, S.A. inicia sus operaciones en 1892, en Monterrey, N.L., con el nombre de Banco de Nuevo León, S.A., institución que dio impulso al mundo de las finanzas, la agricultura, la minería, la industria y el comercio. En este sentido es el primer banco en fundarse con capital económico de la región. En el año de 1937, los señores Salinas y Rocha fundaron la empresa denominada Financiera del Norte, S.A.

En 1977 esa empresa financiera fusiona al Banco del País, S.A. (antes Banco de Nuevo León, S.A.), y a Financiera BANPAIS de Occidente, S.A., Financiera del País, S.A., Financiera del Bravo, S.A., Hipotecaria BANPAIS, S.A. y Almacenadora BANPAIS, S.A.

En el año de 1978 este grupo de empresas se transforma en BANPAIS y surge como una de las primeras Instituciones de Banca Múltiple en México.

En 1979 se fortalece aún más con la incorporación de Banco Comercial Peninsular, S.A. y en 1985 con la fusión de Banco Latino, S.A.

Al igual que el resto de la Banca Mexicana, BANPAIS fue estatizado en 1982, con el decreto de nacionalización de la banca, convirtiéndose a partir de 1983 en BANPAIS, Sociedad Nacional de Crédito.

En 1991, como resultado de la desincorporación bancaria, BANPAIS se constituyó como una institución financiera privada, transformándose de BANPAIS, S.N.C. a BANPAIS, S.A., y cuenta con el apoyo de instituciones filiales que le permiten ofrecer a todos sus clientes paquetes completos de servicios financieros.

Cobertura Comercial de BANPAIS

BANPAIS, S.A. es un banco con cobertura multirregional, cuenta con 160 sucursales bancarias, distribuidas en 58 ciudades de 27 entidades federativas, con presencia en los mercados más importantes de la República Mexicana.

Su estructura comercial se encuentra organizada en cinco direcciones territoriales:

- Territorio Norte.
- Territorio Centro.
- Territorio Pacífico.
- Territorio Sur.
- Territorio México.

Sus oficinas internacionales se encuentran en Nueva York y las Islas Caimán. La figura 2.1. muestra la gráfica de distribución territorial a nivel nacional de BANPAIS.

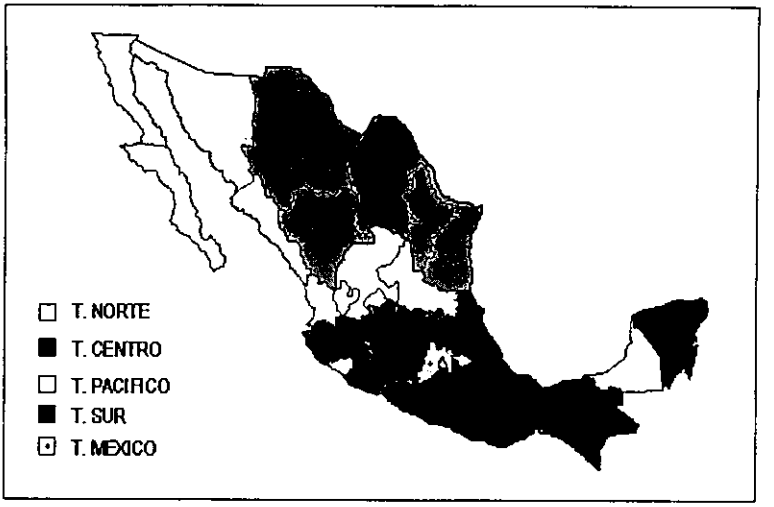


Figura.2.1.Cobertura Nacional de BANPAIS.

Servicios de BANPAIS

BANPAIS representa una buena opción para realizar operaciones internacionales, cuenta con personal especializado y una red de corresponsales a nivel internacional.

También proporciona financiamientos para importar materias primas, refacciones, productos elaborados, maquinaria y equipo y prestación de servicios.

Si el cliente requiere financiamiento para solventar pedidos de exportación, ya sea como exportador directo o como exportador indirecto, BANPAIS ofrece financiamiento acorde a las necesidades.

Si el cliente requiere ofrecer crédito a sus compradores del exterior, BANPAIS paga al cliente de contado sus ventas a plazos mediante una comisión de descuento y la institución bancaria espera el pago de sus importadores.

A nivel Internacional, BANPAIS cuenta con los siguientes servicios:

- Cheques de viajero.
- Compra venta de divisas.
- Transferencia de fondos Giros bancarios.

- Ordenes de pago.
- Cobranzas.
- Remesas en camino.
- Cartas de crédito internacionales.

Financiamientos a la Importación y Exportación

Mediante el desarrollo de Ingeniería Financiera (estudio del flujo de capital, capacidad de pago, capacidad de endeudamiento, conducta del cliente y experiencia del cliente con otros bancos), como parte de los servicios de BANPAIS como banca comercial (banca de servicios múltiples), BANPAIS ofrece a las empresas financiamientos para:

- Adquisición de Complejos Industriales.
- Importación de Maquinaria y Equipo, obteniendo recursos de Organismos Internacionales como la Unión de Bancos Suizos, *Eximbank*, *Exporting Development Co.* etc.
- Exportaciones.
- Productos y materias primas.
- Productos agrícolas, ganaderos y pesqueros.
- Bienes de capital.
- Tecnología.
- Servicios profesionales.
- Préstamos.
- Servicios especializados para maquiladoras.

La figura 2.2 (página siguiente) representa los tipos de financiamientos que BANPAIS ofrece a sus clientes.

Inversiones

Para realizar operaciones de inversión, BANPAIS cuenta con: depósitos a la vista (inversiones con liquidez inmediata) y depósitos a plazo (inversiones con liquidez al finalizar el plazo).

Estas operaciones pueden realizarse tanto en moneda nacional como en dólares.

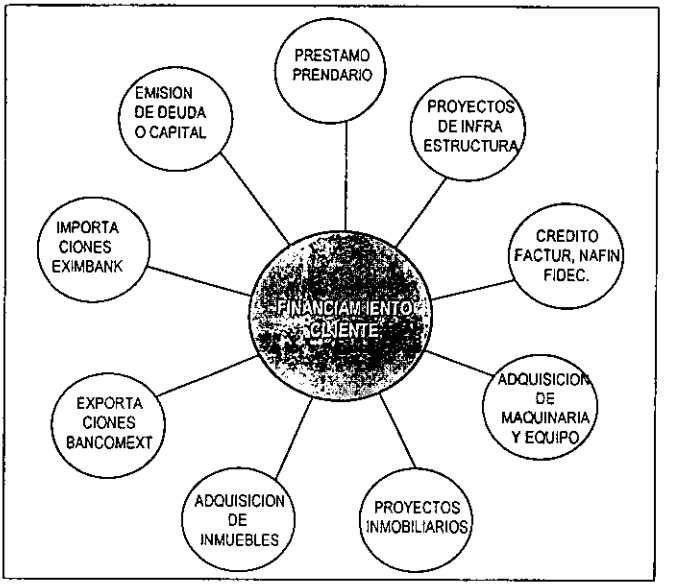


Figura.2.2.Tipos de Financiamiento.

A través de funcionarios especializados, el cliente puede recibir la atención y asesoria financiera, para accesar a financiamientos destinados a apoyar el desarrollo de su empresa, con recursos de:

- Banco Nacional de Comercio Exterior (Bancomext).
- Nacional Financiera, S. N. C. (Nafin).
- Fondo para el Desarrollo Comercial (Fidec).
- Fondo Nacional de Fomento al Turismo (Fonatur).
- Fideicomiso de Fomento Minero (F. F. M.).

Orientación a Empresas

Se orienta a atender dentro de la gran empresa a aquellas que tengan necesidades crediticias superiores a los $15 millones o ventas anuales mayores de $ 60 millones.

Asimismo, atiende las necesidades financieras de las dependencias públicas del gobierno federal, empresas de participación estatal, organismos descentralizados y desconcentrados, gobiernos estatales y municipales.

Servicios

La figura 2.3 refleja los servicios que BANPAIS ofrece a su clientela, estos servicios pueden resumirse como se explica a continuación.

- *Fideicomisos y Avalúos.* El fideicomiso es un contrato en el cual un fideicomitente (cliente) destina ciertos bienes a un fiduciario (Institución Bancaria) de tal manera que la inversión de dichos bienes sea otorgada a un fideicomisario (beneficiario).

- *Cheques e Inversiones.* Consiste en una amplia variedad de productos que le permiten al cliente realizar operaciones de inversión con intereses que le resulten atractivos y un manejo de chequera para hacer uso de su capital sin necesidad de retirar directamente de una terminal electrónica o del banco.

- *Operaciones Cambiarias.* Se refiere a la compra - venta de monedas extranjeras, principalmente monedas duras (libra esterlina, marco alemán, franco francés, etc.) y también de monedas blandas (lempira uruguaya, peso cubano, etc.).

- *Mercado de Dinero.* Ya sea en moneda nacional o extranjera, el cliente puede realizar operaciones de depósitos plazo que le generen rendimientos (intereses + capital) mas atractivos que otro tipo de inversión.

- *Tarjeta de Crédito.* A nivel nacional o Internacional, el cliente puede disponer de una tarjeta de crédito si cumple con los requisitos necesarios, estando en la posibilidad de hacer uso de esta tarjeta en un gran número de establecimientos afiliados.

- *Recolección y Traslado de Valores.* Permite que el cliente inscrito en este servicio se despreocupe por la inseguridad de realizar él mismo un recorrido por cualquier vía de comunicación con un cargamento valioso.

- *Terminal Bancaria y Cajeros.* Es un producto que permite al cliente conectarse desde su computadora personal vía módem (no por Internet) al sistema de servicios al cliente de BANPAIS, desde el cual el cliente puede realizar traspasos, depósitos, pago de servicios, consultas de saldos, etc.

- *Pago de Nóminas.* Las empresas inscritas en este servicio pueden depositar el pago de la nómina de sus empleados en una cuenta de cheques, con la cual los empleados pueden disponer de su dinero, por medio de una tarjeta de débito (sólo dispone del dinero que tiene en la cuenta), en cualquier cajero automático que cuente con el sistema de RED cajeros compartidos (Cajeros de varios bancos con un sistema común), también pueden realizar sus compras en los establecimientos afiliados que acepten este tipo de tarjetas.

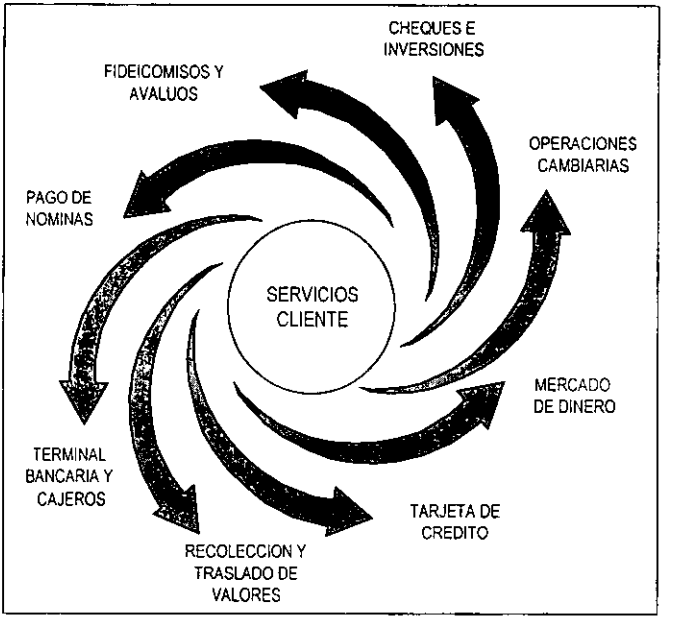


Figura.2.3.Servicios de BANPAIS.

Una vez que contamos con un panorama general de lo que es la Banca Comercial y de los servicios que ésta ofrece, así como de su importancia para la economía nacional, podremos comprender la importancia del buen aprovechamiento de los recursos para beneficio del país. En consecuencia, pasaremos a describir el proyecto como tal y su importancia para la Institución bancaria.

CAPITULO 3

DESCRIPCIÓN DEL PROYECTO

Una vez que se han visto los conceptos básicos del manejo de información, los modelos de desarrollo, tipos de bases de datos y las definiciones más generales de lo que es la Banca de Servicios Múltiples, así como los servicios que ésta ofrece al mercado financiero, pasaremos a definir el proyecto como tal, mostrando su importancia para la institución y la descripción de requerimientos que justificaron su desarrollo e implementación dentro de la misma.

3.1 Exposición del Sistema

Para comprender la importancia del desarrollo de un nuevo sistema, es necesario visualizar el problema que se tenía antes de la puesta en operación del mismo, realizar una propuesta de las diferentes soluciones y explicar las diferencias de operación entre el sistema anterior y el nuevo sistema.

El siguiente análisis tiene la finalidad de justificar la necesidad de contar con un nuevo sistema que satisfaga los requerimientos de la institución que no se están cubriendo con el sistema actual.

Definición del Problema

La oportuna toma de decisiones basándose en la información de los sistemas de Cheques e Inversiones, Ahorros, Fondos Riesgos y Tesorería, es vital para contar con una buena administración que permita alcanzar los niveles de competitividad que el mercado requiere.

Actualmente en BANPAIS se cuenta con la información requerida manejar sistemas de Ahorro, Cuenta de Cheques e Inversiones, Tesorería y Riesgos Financieros; el problema consiste en que esta información está separada en tablas no relacionadas en un ambiente *mainframe*, el cual no presenta ninguna facilidad para los usuarios que requieren de la misma, por lo que se dificulta su extracción y consulta para la elaboración de reportes.

Lo anterior retrasa a los funcionarios en la toma de decisiones y los imposibilita para conocer el estatus de cada uno de los sistemas mencionados anteriormente, así como el rendimiento y la captación de los mismos que ayuda a determinar el nivel de eficacia de los productos que ofrece cada sistema.

Las consultas que llegan a realizarse a través de los operadores no permiten un análisis preciso de la información, ya que los resultados obtenidos no se encuentran actualizados, debido a un desfasamiento de días al realizar la consulta.

No existe la posibilidad de elaborar reportes cruzados, es decir, no hay manera de interactuar con la información de las diferentes bases de datos en el sistema de información dentro del ambiente *mainframe*.

El sistema *mainframe* es de difícil mantenimiento, además carece de información histórica en línea para la consulta y explotación; actualmente se tiene que utilizar un sistema de respaldos, lo cual implica un largo período en tiempos de carga y descarga de la información.

Así mismo no cuenta con *interfaces* para interactuar con otras herramientas de vanguardia que existen en el mercado que ayudan a realizar análisis financieros y estadísticos.

Posibles Soluciones

Como primer propuesta de solución, se pensó en habilitar terminales para cada uno de las persona que requerían del manejo de la información mencionada, sin embargo, el grueso de ellos es constituido por directores, y personal poco familiarizado con el ambiente *mainframe*, lo cual descartó rápidamente esta opción, amén de la difícil capacitación de los usuarios.

Otra propuesta de solución fue el hecho de contratar a personal especializado que tuviera la capacidad de proporcionar la información de manera más ágil y efectiva a quienes realmente la necesitaban, pero el hecho de seguir trabajando con una tecnología que finalmente tiende a desaparecer, en conjunto con la dependencia de la información en cierto personal, hizo que se buscaran más opciones.

Otra de las posibles soluciones fue el comprar un sistema integral que manejara cada una de las cuentas de manera que todas y cada una de ellas fuera ligada por un número único, sin embargo en el mercado no existía un producto que tuviera tal capacidad y se pensó en un desarrollo a la medida como la mejor solución, la cual proporcionaba como ventaja adicional el tener lo que realmente se necesitaba.

El hecho de hacer una inversión mucho menor y una espera de tiempo mucho más corta al llevarla a cabo con el personal que ya estaba involucrado con el manejo del banco, dio lugar a la siguiente propuesta de solución:

La Generación de un Modelo de Datos en arquitectura Cliente-Servidor, que contenga la información de los siguientes grupos de cuentas bancarias:

- Cheques.
- Ahorros.
- Fondos e Inversiones.
- Riesgos y Tesorería.

La descripción anterior de los servicios de la institución bancaria será agrupada, proporcionando en su caso un número único de cliente, que pueda ser tomado como base para el nuevo sistema de consulta que muestre el estatus actual del banco de una forma más precisa y amigable para el funcionario, además de facilitarle la toma de decisiones a través de reportes.

La información del sistema se desarrollará sobre una base de datos relacional, que permita una forma avanzada de consultas por medio de la aplicación Cliente-Servidor.

Esta aplicación de consulta contará con una interfaz gráfica que permita al funcionario accesar la información de manera amigable y así poder explotar su información en forma eficientemente.

Se le ofrece a la institución una herramienta que no requiere de complejas tareas de mantenimiento y comercialmente de bajo costo. Con diversas posibilidades para generar tipos de reportes y amplia gama de *interfaces* para poder comunicar la información con otro tipo de aplicaciones para su análisis.

El nuevo formato de base de datos utilizado permitirá mantener la información histórica en línea sin necesidad de hacer carga y descarga de respaldos.

Descripción del Sistema Actual

El sistema actual es un sistema que realmente, aunque cuenta con la información que se requiere, no está diseñado para la funcionalidad que se pretende utilizar, es decir, el sistema con el que se trabaja en el banco en este momento es únicamente el medio que se utilizó para almacenar la información, la cual está organizada de una manera aislada por lo que dificulta su manejo.

Otra de las características que presenta el sistema actual, es la difícil recuperación de los datos y el hecho de que los reportes que se obtienen de este medio de ninguna manera pueden presentarse a un nivel directivo, que es realmente la utilidad que se requiere de ellos.

El sistema actual no cuenta con una liga directa de la información de las cuentas de un mismo usuario, lo que lo hace además, de lento *performance* y muy poco apropiado para hacer consultas para conocer el estatus de un solo cliente, de manera que esta idea es de difícil aplicación.

Es un hecho que la información histórica del sistema manejado, es de muy difícil manipulación, ya que cada vez que se requiere de su uso, es necesario llevar a cabo cargas sobre tablas temporales o tablas de trabajo diferentes a las que se manejan en el sistema propio, y el hecho de necesitar del área de operación para obtener información histórica lo hace más burocrático e ineficiente.

El manejo de la información, por los puntos expuestos anteriormente, se hace de manera difícil e ineficiente, cuando la institución requiere de precisión y poco tiempo de respuesta para una junta con el comité de dirección en la cual se requiera una ágil toma de decisiones.

Objetivos de la nueva aplicación

Consultar mediante una interfaz gráfica de ambiente Cliente-Servidor amigable para el usuario la información correspondiente a:

1. Estado de las cuentas (activas e inactivas).
2. Monto de las cuentas.
3. Clientes activos y relaciones entre sus cuentas.
4. Captación por zonas.
5. Estado actual de la institución bancaria en conjunto.

Al final del día se pretende que el nuevo sistema sea capaz de:

- Contar con la información más reciente, confiable y oportuna en un tiempo de respuesta óptimo.

- Tener un conocimiento certero y oportuno de cual es el estatus de todas y cada una de las cuentas con las que el banco trabaja.

- Mantener la información de manera organizada para interactuar entre las cuentas de un mismo cliente, por medio de un número único de referencia.

- Aplicar en el diseño una metodología estructurada que garantice:

    - La funcionalidad de los procesos y flujo de información.
    - Disminuir la probabilidad de errores y deficiencias en el diseño.
    - El nivel adecuado en el desempeño del sistema (Consultas ágiles y precisas).
    - La generación de documentación clara y comprensible con la cual, en un futuro, se puedan realizar los mantenimientos que sean necesarios para el sistema.

- Respetar las políticas sobre el desarrollo de sistemas de proyectos en el banco, como:

    - Evitar que los proyectos de programación se excedan del presupuesto.
    - Cumplir con las fechas y compromisos que se hayan establecido.
    - Minimizar el costo de mantenimiento de los sistemas.

Cumpliendo los requerimientos mencionados, se podrá:

- Eliminar la dependencia del personal encargado del *mainframe* para la extracción de información.

- Establecer una base para que el banco pueda obtener un estatus de cada uno de sus clientes, así como del comportamiento en general del banco.

- Proporcionar la información de la manera más fácil y amigable a quienes realmente la necesitan, por lo cual será accesada de manera que se pueda generar sin problema por el usuario final.

- El sistema deberá ser preciso y certero, pues reflejará cual es el estado del banco en el momento de la consulta, con una diferencia máxima de horas a la realidad.

- Proporcionar información estadística, ya que la información histórica se tendrá a la mano de manera eficiente y tan fácil como la información más reciente con la que se cuente.

- Los usuarios podrán obtener reportes que no deberán ser procesados por ninguna otra herramienta para tener un formato y una posibilidad de interacción con otras herramientas más manejables y que darán más diversidad y comprensibilidad y un mejor uso de la información.

- Estará orientado a los directores por medio de los términos que ellos utilizan frecuentemente, lo que les dará una herramienta familiar y evitará la necesidad de depender de terceras personas u otros medio para obtener la información en el momento que se requiera.

Requerimiento de información de la nueva aplicación

El sistema requerirá tomar la información de las tablas en las que actualmente se maneja, y dependerá del uso y el manejo que se le dé para cumplir con los objetivos anteriormente propuestos; es por esto que se encargará de ordenar de manera adecuada la información, lo que implicará un efectivo diseño.

El sistema será madurado por medio de un modelo de prueba al cual se le harán las correcciones necesarias y se irá optimizando dentro del proyecto para que éste pueda ser tomado como base final y tenga las características necesarias para poder cumplir con las finalidades propuestas.

El sistema requerirá de la información propiamente de las tablas de los sistemas de Cheques y Fondos, Ahorros, Inversiones y Tesorería y Riesgos que actualmente se manejan por parte de la institución, amén de ser completada por medio de los conceptos que se apliquen por los usuarios del banco, así como por los criterios que se irán aplicando para procesarla y subirla de manera correcta al nuevo sistema.

Además de la información antes mencionada, se requerirá de múltiples entrevistas con los usuarios y con el personal de operación que ahora maneja la misma para entender completamente le concepto y la idea que deber reflejar la información al ser presentada por y para los usuarios finales.

Debemos, por último, aprovechar los conocimientos que se han adquirido dentro de BANPAIS en cuanto a los conceptos bancarios que se van a utilizar, ya que será responsabilidad nuestra el darle el valor agregado al sistema, así como optimizar su funcionalidad.

En conjunto, requiere de recursos tecnológicos y humanos armonizados para poder cumplir con las finalidades propuestas.

Afectaciones del nuevo sistema al actual

El nuevo sistema deberá ser completamente transparente al actual, por lo que deberán programarse, por ejemplo, los tiempos de carga, que serán el puente y la parte más interactiva de los mismos.

El nuevo sistema no solamente enriquecerá al anterior, sino que será utilizado como el medio para poder conjuntar la información del banco en una sola aplicación, lo que podría conceptualizarse, al ser madurado, como el *DATAWARE HOUSING* de la institución bancaria.

El sistema anterior no podrá ser desplazado prontamente, lo que originará que ambos tengan una convivencia armoniosa para el mejor aprovechamiento de los recursos.

Cabe mencionar que ambos sistemas deberán tener la misma funcionalidad básica, que es el mantener organizada la información, y que el nuevo tendrá más versatilidad, lo que enriquecerá a la institución bancaria como tal.

3.2 Evaluación Técnica

El *Mainframe*

Originalmente se cuenta con un *mainframe Unisys* Serie A 10, dentro del cual se programa en *COBOL* 74 sobre un Sistema Operativo *CANDE*.
Este *Mainframe* se encarga de la parte a analizar sobre: Cuentas de Cheques e Inversiones, Ahorros, Fondos Riesgos y Tesorería.

Descripción del Servidor

No es necesario adquirir un nuevo servidor para la creación del Centro de Información, ya que, anteriormente el Banco había adquirido un servidor *SUN SPARCserver 20*, y actualmente no tiene mucha carga de procesos; identificaremos las características de este servidor y se evaluará la posibilidad de colocar la aplicación a desarrollar y la base de datos dentro del mismo.

Este servidor de *SUN* está diseñado para requerimientos específicos de los grupos de trabajo: como servidor de archivos, servidor de impresión, servidor de aplicaciones, o como el mejor servidor de conectividad para la industria. El *SPARCserver 20* ofrece el poder del multiproceso; más procesadores para manejar múltiples tareas simultáneamente, así como sus consultas separadas a bases de datos, peticiones múltiples de archivos o una aplicación compartida de uso intensivo.

El *SPARCserver 20* modelo 514MP cuenta con: 4 procesadores *SuperSPARC* 50 *MHz* cada uno con *SuperCache*, 64 MB de memoria, 2 discos de 2.1 GB con *interface Fast-SCSI, SunCD* 2 *Plus* de 644 MB interno. Ambiente Operativo *Solaris*, provee el mejor desempeño de red para la industria y completa capacidad de administración del sistema a través de herramientas que mantienen la red en su más alta eficiencia. Elementos de red integrados como conectores de par trenzado y *AUI Ethernet*.

Sybase como Base de Datos

La selección de la Base de Datos no es menos compleja que la del servidor donde va a residir; y aunque el banco tiene una licencia corporativa de *Sybase*, se verificaron las características de compatibilidad y desempeño de *Sybase SQL Server 11*.
Al solicitar información sobre el desempeño del manejador de base de datos sobre un servidor *SUN* a *Sybase*, nos enviaron como parte de la respuesta, artículos relacionados, como:

"En Octubre de 1996, *Sun Microsystems, Inc.* y *Sybase, Inc.* anuncian sus primeros resultados de desempeño de *SQL Server 11*, alcanzando el más alto nivel de respuesta de una base de datos relaciona *UNIX* en sistema *SUN*.

El resultado representa una nueva marca de la plataforma *SUN* en servidores *UNIX* para el *TPC-C* (*Transaction Processing Perfomance Council Benchmark*).

La marca se definió en 4544.60 transacciones por minuto (*tpmC - transactions per minute*) a un precio/desempeño de $396 USD por *tpmC*, demostrando la superioridad en rendimiento alcanzada por las dos compañías". *Sybase SQL server 11* aumenta 3.5 veces el desempeño alcanzado anteriormente con *Sybase SQL Server 10*.

Sybase Inc. es líder mundial en *software* cliente/servidor que enfoca sus esfuerzos en cuatro grandes segmentos de mercado: acceso masivo, proceso de transacciones en línea, *data warehousing* y comercio electrónico."

Perfil de la Computadora Cliente

Se seleccionarán Computadoras Personales para funcionar como clientes del Sistema; en este rubro no se realizará inversión alguna, ya que se cuenta con 10 computadoras, 5 nuevas y 5 de uso intermedio. Las computadoras nuevas tienen la siguientes características:

- Procesador *Pentium* 266 MHz
- Memoria RAM de 32 MB
- Disco Duro de 3.2 GB
- Sistema Operativo *Windows* 95

Y las de uso intermedio, en promedio cuentan con lo siguiente:

- Procesador *Pentium* 166 MHz
- Memoria RAM de 16 MB
- Disco Duro de 2.1 GB
- Sistema Operativo *Windows* 95

Software de Desarrollo

En la primera parte del sistema, para enviar la información del *mainframe* al nuevo servidor, se desarrollarán rutinas llamadas *STORE PROCEDURES* en *CANDE*, el cual se encuentra instalado en el mismo.

Se realizarán sesiones *FTP* de comunicación entre las dos computadoras. Para el desarrollo del *Front-End* del Sistema se seleccionará entre las dos herramientas estándar para desarrollos dentro del Banco, las cuales son: *Visual Basic* 4 y *GQL Windows*; ya que se cuenta con las licencias y permisos de uso.

La configuración de tablas, relaciones y estructuras, se realizarán en *Sybase*, y la comunicación hacia los nodos cliente será por medio de Módulos ODBC hacia *SQL Server*. A lo largo del proyecto y como apoyo para la elaboración de documentos se utilizará el paquete integrado de *Microsoft Office* 95.

Comunicación en Red

Para interactuar con el servidor, los equipos PC utilizarán la red estructurada que ya se encuentra instalada en el Banco, esta red cuenta con cableado *Ethernet* 10 Base T, que cumple con las siguientes características:

- Cable UTP nivel 5.
- Máxima longitud por cable 100m.
- Máxima atenuación de la fuente al destino 100 dB.
- Impedancia del cable, entre 85 y 110 ohms.

Las computadoras cliente cuentan con tarjetas *3Com* 10 Base T; y la red está conectada con concentradores *3Com* de 8 y 32 puertos para conectores RJ45. Cabe hacer mención que estas características son estándar para todo el Banco.

En este momento conocemos claramente la problemática a resolver, requerimientos y funciones a realizar; además de saber las características del equipo de cómputo con el que contamos, pasaremos al siguiente capítulo donde trataremos los puntos importantes del diseño del sistema, así como la implementación y verificación del mismo.

4 PLANEACION, DISEÑO Y DESARROLLO DEL SISTEMA

En el presente capítulo se describen las herramientas de programación, las estructuras de datos y de control a utilizar en el diseño e implementación del sistema, así como la estructura lógica, los procesos, el diseño de tablas y los estándares de codificación a utilizar para conformar el Centro de Información (CDI).

4.1 Diseño de Lógica

Para comprender en forma global la lógica del sistema hablaremos de cuatro etapas o procesos, como se muestra en la figura 4.1: Extracción, Transferencia, Carga y Explotación de la Información; incluyendo en esto, un paso especial que es la Bitácora del Sistema.

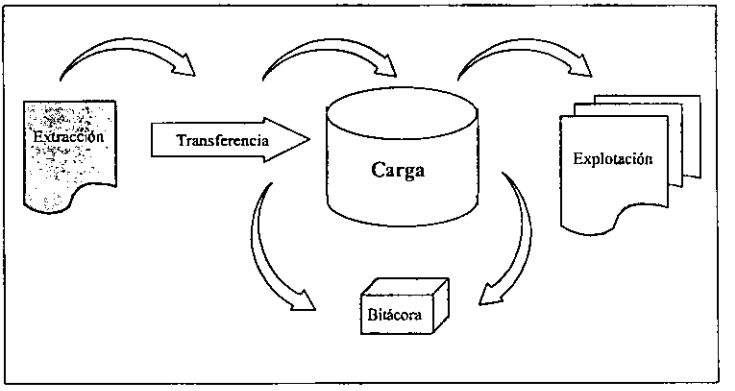


Figura 4.1. Diseño Lógico del sistema.

Es muy importante definir el proceso en base a Módulos, los cuales irán determinando puntos de Control para el manejo de la información, por lo cual generamos diagramas donde cada uno se encargará de conceptualizar una parte muy específica de la aplicación.

La tabla 4.1 muestra los módulos principales dentro de la generación del CDI, así que nos referimos de aquí en adelante a los módulos ahí definidos como las etapas propias del desarrollo. Con el fin de comprender el contenido de la tabla 4.1, mencionaremos a continuación los tipos de cuentas , las zonas que están involucradas en los procesos para la generación del CDI y el tipo de información que se maneja.

Cuenta	Nomenclatura
• Cheques.	CHE
• Fondos.	FON
• Ahorro.	AHO
• Inversión.	INV
• Inversión Mesa de Dinero.	IMD
• Tesorería y Riesgos.	TES

Zonas	Nomenclatura
• Monterrey.	MTY
• Guadalajara.	GUA
• México.	MEX
• Saltillo.	SAL
• Hermosillo.	HER
• Foráneas México.	FOM

Información	Nomenclatura
• Contratos.	CON
• Saldos.	SAL
• Clientes.	CLI
• Direcciones.	DIR
• Bajas.	(IMD e INV)

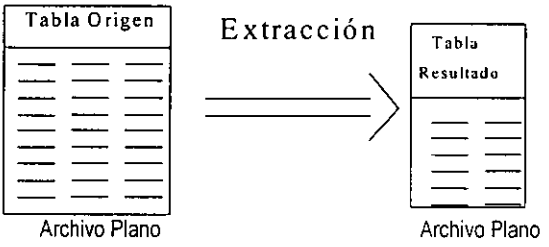
Módulo	PRODUCTO FINAL
EXTRACCION En esta etapa se tiene la información fuente en el ambiente Unisys en tablas maestras independientes. Esta extracción queda en el mismo ambiente Unisys.	Este proceso consiste en tomar una tabla de cualquier sistema (Clientes, Cheques, Ahorro, etc.) tal y como se encuentra diseñada. Estas tablas contienen información histórica de por lo menos cuatro décadas, por lo que es necesario realizar un extracto de la tabla para poder explotar mejor su contenido. Para realizar este procedimiento se requiere de programación en Cobol 74 y CANDE (Command AND Edit), ya que las tablas originales se crearon bajo estas características. UNISYS UNISYS  El resultado de este proceso es la generación de archivos planos dentro del mismo ambiente. Una vez extraída la información, el nombre del Archivo Final queda así: CDI/Zona/Fecha/Cuenta/Información Ejemplo: CDI/MTY/0101/CHE/CON
TRANSFERENCIA Es aquí donde se realiza la transferencia de la Información del ambiente Unisys a UNIX.	El objeto de este proceso es realizar una conversión de un archivo plano a un archivo de texto, el cual permitirá posteriormente ser convertido en una tabla relacional de SYBASE. Es llevado a cabo por medio del FTP (File Transfer Protocol) vía Reflection (emulador que nos permite realizar la conexión para la transferencia de archivos a través de TELNET (Protocolo de emulación). El protocolo TELNET proporciona una capacidad de emulación de terminal que permite al usuario interactuar con cualquier otro tipo de computadora de la red. El protocolo TCP controla la transferencia de los datos, y el IP brinda el mecanismo para encaminarla.

Tabla 4.1. Descripción y Características de las Etapas del Diseño Lógico (Continúa).

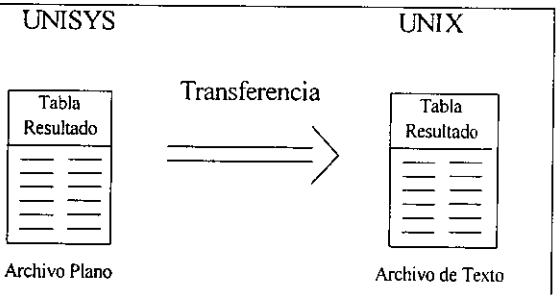
	 Al finalizar esta etapa, se deberá contar con los archivos planos, generados como *archivos.txt* dentro del ambiente UNIX en el directorio /home/clientes/archivos en el servidor Mercurio (Servidor Sun SPARC 20). El nombre del archivo final en ambiente UNIX dentro del directorio especificado es : CTA+INF+ZON+MMDD.TXT CTA = Cuenta, INF = Información ZON = Zona MMDD = Fecha en el formato MesMesDiaDía, utilizando los conceptos definidos en los párrafos anteriores a esta tabla. Ejemplo: AHOCONMTY0101.TXT
CARGA En esta etapa se realiza la carga de la información a la Base de Datos SYBASE, en ambiente UNIX.	Ya que se ha realizado la transferencia de archivos (tablas) para obtener un archivo de texto, lo siguiente es la conversión a un archivo de SYBASE, es decir, la conversión de una tabla individual a una tabla relacional. Para llevar a cabo este proceso se utilizan las herramientas de programación de CShell, instrucciones como bulkcopy y Querys en sesiones de SYBASE. Como resultado de este proceso de carga, se obtendrá la información implementada en la base de datos, la cual es propiamente el CDI, con lo cual queda lista, previa validación al momento de cargarla para su explotación. Los archivos procesados se cambian de directorio y de extensión a .OK.
EXPLOTACION Una vez que se tienen las tablas cargadas, se procede a la generación de resultados a partir de la información.	Finalmente se utilizará Visual Basic para crear la interface final que permitirá el manejo de la base de datos creada a partir de las tablas que se han generado en SYBASE mediante los tres primeros procedimientos. La figura muestra los dos últimos procedimientos.

Tabla 4.1. Descripción y Características de las Etapas del Diseño Lógico (Continúa).

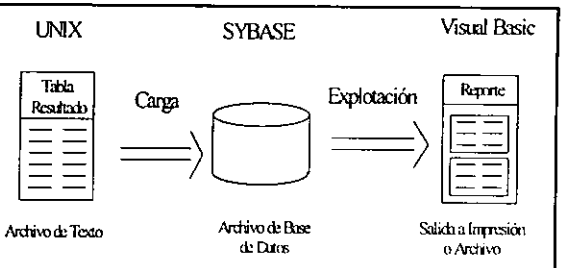
	 Carga y Explotación de Tablas. Se obtienen reportes, tablas y gráficas con opción a pantalla, salida a archivo o impresión.
BITACORA Se realiza una bitácora de cada sistema o cuenta, es decir se realiza un tipo de validación de la información desde su extracción hasta la carga de datos.	Nombre del archivo de la bitácora: BITAHOMTY0101.TXT La cual generaba en el archivos el tipo de información y el número de registros de cada tipo, por ejemplo: Tipo de Inf. Núm. Reg CON 10 CLI 10 SAL 2500 DIR 10 Estos archivos se suben a tablas de bitácora dentro del modelo de igual forma que se hace con los datos.

Tabla 4.1. Descripción y Características de las Etapas del Diseño Lógico.

La periodicidad en ejecución de los procesos es:

- Inicial. No se tiene punto de referencia, se lleva a cabo en cualquier momento que se solicite.
- Diaria. Se refiere a la información dentro del periodo del día que transcurre.
- Mensual. se tiene únicamente para la información de saldos, ya que únicamente se maneja este tipo de información como requerida en esta periodicidad.

En el banco las cuentas en general cuentan con todos los tipos de información (contratos, saldos, clientes, direcciones y bajas), en todas las zonas, pero se tiene las siguientes excepciones:

- La cuenta de Fondos tienen únicamente la información de saldos, es decir, no maneja más que los montos de las cuentas, sin ningún acceso que haga referencia a otro dato, más que un código de operación, que es quien se encarga de ligar el saldo con su respectiva cuenta.

La tabla de Ahorros no cuenta con la zona de Hermosillo (HER), debido a que la captación de esta zona por el producto mencionado es mínima, por lo que está integrada con la zona de Monterrey (MTY).

La explicación de cada proceso se dará con mayor detalle en el apartado 4.3.

Herramientas de Programación

Las herramientas que se emplearán para el desarrollo del proyecto son con las que se cuenta dentro de BANPAIS para el manejo de la información, estas herramientas son:

- Cobol 74 , que servirá para obtener la información de los sistemas contenida en el *Mainframe*.
- CANDE (*Command And Edit*) que será la plataforma sobre la cual se accesará a la información dentro del *Mainframe*, es decir, el sistema operativo dentro del equipo Burroughs.
- Programación en CShell, la cual servirá como medio, en el sistema UNIX, para "subir" la información a la base de datos, por medio de instrucciones propias y adicionales como *bulkcopy*.
- Sybase, que es la base de datos designada por el corporativo para manejar la información y sobre la cual se construirá propiamente el CDI.
- *Visual Basic* que será la herramienta para la construcción del front-end.[1]

Estas herramientas se tienen en el banco, por lo que no existe la necesidad de evaluar alguna otra, o hacerlo con distintos lenguajes de programación, más bien lo que realizaremos en el presente trabajo es explotar los medios y herramientas con los que ya contamos.

La programación dentro de Cobol 74, CANDE y CShell, se realiza de manera estructural o funcional, mientras que en Visual Basic aprovechamos la programación orientada a eventos o dinámica. Sybase, como la base de datos relacional que es, nos brindará las facilidades necesarias para el adecuado funcionamiento del modelo; y *GQLWindows* nos servirá como herramienta de explotación de la información, mientras que *Visual Basic*, servirá como un medio más poderoso para generar los reportes requeridos.

[1] Además de utilizar a *Visual Basic* 4.0 dentro de la aplicación, se generarán los primeros listados con GQLWindows que fungirá como reporteador mientras se da forma a las aplicaciones finales.

Metodologías de Desarrollo

Las metodologías que explotaremos para llevar a cabo el proyecto, dentro de las diferentes etapas del desarrollo, se apegarán a las definidas dentro de la Programación Orientada a Objetos (POO) y a las Técnicas de Ingeniería de Software Aplicadas a la Programación (TISAP), en lo referente a la programación estructurada. Ambas metodologías se verán de manera general dentro del Apéndice "A".

El diseño orientado a objetos comprende una notación para descripciones lógicas (estructuras de clases y objetos) y físicas (arquitectura de modelos y procesos), así como los modelos estáticos y dinámicos del sistema que se está diseñando. Existen tres modelos de diseño (modelo de Objeto, Dinámico y Funcional) los cuales se relacionan de la siguiente forma.

Cada modelo describe un aspecto, pero contiene referencias a otros modelos. El modelo de Objeto describe estructuras de datos donde los modelo Dinámicos y Funcional operan. Las operaciones en el modelo de Objeto corresponden a eventos en el modelo Dinámico y a funciones en el modelo Funcional.

El modelo Dinámico describe la estructura de control de objetos, muestra decisiones que dependen de los valores de los objetos y que causan acciones que cambian valores de objetos e invocan funciones. El modelo Funcional describe funciones invocadas por operaciones en el modelo de Objeto y acciones en el modelo Dinámico.

Las funciones operan en valores de datos especificados por el modelo de Objetos. El modelo Funcional también muestra constantes en los valores objetos.

Dentro de la parte de la Programación de *Shell*, hablaremos de un *Shell Script* como un archivo que contiene una lista de comandos, los cuales son ejecutados por el *Shell* que es el intérprete propio del Sistema Operativo UNIX.

Los *Shell Scripts* no se compilan, debido a que cada línea es interpretada en tiempo de ejecución. Además debemos decir que son compartibles entre diferentes versiones de UNIX y XENIX.

Por último, llevamos a cabo la construcción y ejecución de *Querys* dentro de la base de datos, los cuales serán los encargados de manipular y actualizar la información dentro del proceso de explotación, así como de realizar algunas validaciones en la base de datos, también generando los filtros necesarios para el correcto despliegue de la información al generar los reportes.

Estructuras de Control

Consideraremos dentro de las fuentes de información que llevarán a cabo el abastecimiento de la datos necesarios para la generación del modelo, así que definiremos tanto las tablas involucradas en el sistema como las utilizadas en el banco como parte de otros procesos ajenos al nuestro.

Como fuente principal de información utilizaremos los sistema bancarios existentes que de manera directa se relacionan e interactúan con el *Mainframe*, entre los que podemos mencionar a *Alborada*, que es el sistema de terminales de sucursal, donde propiamente se captura la información de las sucursales a nivel nacional, el *CIE*, que es el sistema de donde pueden realizarse pagos depósitos de varias empresas por medio de una sucursal bancaria (por ejemplo, los pagos de servicios telefónicos) y PROSA, que nos dará información correspondiente a las transacciones realizadas a través de los cajeros automáticos.

Dentro de las principales tablas de *Burroughs* que agrupan la información del banco podemos mencionar los siguientes:

- Cheques.
- Ahorros.
- Fondos.
- Documentos de Plazo (Inversiones Tradicionales).
- Contratos de Cartera.
- Préstamos.
- Documentos de Cartera (Vigentes y vencidos).
- Documentos de Plazo (Inversiones Mercado de Dinero).
- Asignación Inversiones Instrumentos.
- Compras.

En sí, el sistema necesita de varias tablas básicas para su funcionamiento, entre las que destacaremos como las más importantes, las siguientes:

- Contratos.
- Clientes.
- Direcciones.
- Saldos.

Donde "Contratos" cubre las cuentas de: Cheques, Ahorros, Fondos e Inversiones, así como la información propia de Tesorería y Riesgos. Los campos comunes a todas las cuentas son: Número de Contrato, Código de Cliente, Producto, Sub-producto, Sub-sub-producto, Sub-sub-sub-producto, Moneda y Fecha de alta.

Las tablas de Clientes y Direcciones contendrán propiamente datos personales del cliente, lo que nos proporciona la base de datos integrada de la cual podemos generar información tal como reportes por cliente o por zona. De igual forma que la anterior, en estas tablas se almacenará la información de todos los sistemas incluidos dentro del CDI.

En la tabla de Saldos se cubren las cuentas de: Cheques, Ahorros y Fondos. Los campos comunes son: Número de contrato, Tipo de saldo, Fecha y Saldo.

Existen otras tablas adicionales, que de ninguna manera carecerán de importancia, ya que darán la pauta a poder ligar o relacionar los diferentes productos o datos contenidos dentro del sistema.

Estructuras de Datos

Dentro de las estructuras de datos podemos mencionar los arreglos, las variables, las estructuras y uso de memoria.

Los arreglos, entendiéndose éstos como un conjunto ordenado de elementos de datos, se manejan arreglos de una dimensión, los cuales son almacenados y manipulados fácilmente por los programas que se utilizarán, básicamente en *Visual Basic*.

Las variables, teniendo en cuenta el uso de variables locales y globales, siendo importante mayoría las primeras y optimizando el uso de parámetros dentro de los procesos para así tener un uso más adecuado de recursos de hardware, como lo es la memoria.

Las estructuras comprenden un importante concepto en el desarrollo de la aplicación, ya que éstas permiten llevar un orden lógico durante la programación y facilitan el mantenimiento futuro del programa al hacer más legible la codificación del mismo.

La memoria de la computadora es un recurso importante, ya que determina el tamaño y el número de programas que pueden ejecutarse al mismo tiempo, como también la cantidad de datos que pueden ser procesados instantáneamente.

Cuando entran datos en la memoria, el contenido previo de ese espacio de memoria se pierde. Una vez que el dato está en la memoria, puede ser procesado (calculado, comparado y copiado). Entonces los resultados pueden ser sacados desde la memoria a la pantalla, impresora, disco, cinta o canal de comunicaciones.

En nuestro caso, para el uso óptimo de memoria, hacemos énfasis en la apertura y liberación de ésta por los programas, por ejemplo: en COBOL 74 se libera con la instrucción CLOSE, en *Visual Basic* con *Unload form*, por citar ejemplos.

4.2 Estándares de Codificación

Los estándares se van a definir en base al tipo de Sistema Operativo que se utilice en cada una de las partes del Sistema; sabemos del apartado anterior que básicamente son 4 secciones importantes las que lo forman: Extracción, Comunicación ó Transferencia, Carga y Explotación.

Objetivos de los Estándares de Programación

Debido a la extensión del proyecto, participamos cuatro personas en el desarrollo de los módulos que ejecutan cada uno de los procesos antes mencionados, y que forman la estructura lógica del Sistema.

El trabajo en equipo exige establecer una forma de trabajar y ciertas reglas a cumplir para alcanzar un objetivo común, hablando concretamente de los sistemas y programas, estas reglas se refieren a respetar una estructura para escribir el código en los programas, considerando la declaración de variables y constantes, cuerpo del programa y nomenclatura de archivos, tanto de entrada como de salida.

Entre las reglas más importantes utilizadas para el desarrollo de este sistema, se pueden numerar las siguientes:

1. Las variables locales deberán empezar con una "L" en cada procedimiento, mientras que las variables globales deberán empezar con una "G".

2. Cada proceso deberá estar estructurado de tal manera que se procure que el cuerpo completo se pueda visualizar en una sola pantalla, para tener una visión más rápida de lo que hace el proceso.
3. Dentro de la programación en COBOL, las rutinas que se encuentren definidas como líneas entre 100 y el 999 serán parte del programa principal, mientras que las líneas superiores se ocuparán por cada procedimiento, en grupos de 1000.
4. Para la programación en Visual Basic, deberán existir módulos principales, entre los que siempre deberemos encontrar:
   - Inicio.
   - Definición y Operaciones de Base de Datos.
   - Operaciones de Registros de la Base.
   - Proceso.
   - Validaciones.
5. Todos los programas deberán tener una breve descripción de la función que cumplen en el encabezado de los mismos.
6. Cualquier desarrollo deberá especificar cual es su entrada y cual es su salida, y en los diagramas deberá representarse como únicamente una entrada y una salida.

Una vez definidas las reglas a utilizar, aumenta la capacidad de respuesta del equipo de trabajo, disminuye errores de apreciación o de concepto y mantiene clara la idea del objetivo común, que es contar con un Centro de Información, en el tiempo más corto posible.

Estructura General de los Programas

La estructura de los programas varía en función al lenguaje de programación que se utiliza, por lo que se explicarán las diferentes estructuras a utilizar durante el desarrollo del sistema, y el sistema operativo donde son válidas.

En las páginas siguientes se mostrará la estructura de cada una de los módulos que constituyen el CDI.

Etapa del Proyecto: Extracción
Ambiente Operativo: UNISYS/ Burroughs
Lenguaje o Intérprete de Comandos a Utilizar: COBOL74

Estructura: Nos apoyaremos en la sintaxis requerida por COBOL para la escritura de sus programas.
Las primeras líneas en el programa son el encabezado, la *Identification Division* y están formado por:

- Identificación de la sesión a utilizar en el *Mainframe*.
- Identificación del Programa.
- Fecha de Creación del Programa.
- Datos del Autor del Programa.
- Una sección de comentarios en donde se describa de manera general, que es lo que hace el programa y para qué ?

El siguiente grupo de líneas define la *Environment Division* y contiene información acerca de el equipo que se está utilizando y sus medios (*Configuration Section)* y el *Input-Output Section* que determina la liga a los archivos que físicamente serán generados por los desarrollos. Posteriormente encontramos la *Data Division* que tendrá una sección para la definición de la estructura de los archivos utilizados conocida como *File Section*; la *Database Section*, en donde definiremos la tabla maestra de la cual obtendremos la información; el *Working Storage Section*, que será donde definiremos las variable y constante que se utilizan en el programa, así como los parámetros que deberá tener cada uno de ellos para ejecutarse correctamente.

Finalmente vendrá la *Procedure Division*, en donde definiremos el proceso como tal, separando las primeras líneas (definidas como numeración 100 a 199) como el proceso principal y las líneas 1000 en adelante como procedimientos, procurando que cada uno no exceda de los números de 1000 en 1000. Aquí se escribe propiamente lo que es cada una de las instrucciones a realizar durante la ejecución del programa, donde encontraremos los modelos propios de la programación estructurada (ciclos, condicionales, etc.); y las instrucciones para generar los archivos de salida, necesarios para la siguiente etapa del proceso.

En la tesis se presenta el código fuente de uno de los procesos de extracción desarrollados.

Etapa del Proyecto: Extracción/Transferencia
Ambiente Operativo: UNISYS/ Burroughs
Lenguaje o Intérprete de Comandos a Utilizar: UNISYS

Estructura: Aquí vamos a generar un proceso conocido como "*batch*" (este tipo de proceso ejecuta un conjunto de instrucciones de manera secuencial y básicamente este tipo de instrucciones son propias de un ambiente operativo en particular).

A este tipo de procesos, en UNISYS, se les conoce con el nombre de *JOB'S* (trabajos) y su estructura general es la siguiente:

- Encabezado: Contiene el nombre del proceso y los parámetros de entrada.
- Definición del tipo y valor inicial de las variables y constantes.
- Sección del comentario, donde se explica brevemente, que hace el proceso y que deja como resultado de su ejecución.
- Cuerpo de Instrucciones Secuenciales, uso limitado de estructuras de control.
- Definición de la transferencia de archivos.
- Fin del programa.

En esta parte vamos a colocar las llamadas a sesiones de TELNET vía TCP/IP para poder enviar la información de UNISYS a UNIX, van a ser paquetes de información en archivos planos, los cuales son compatibles para realizar la carga de información en el nuevo sistema.

Se presentará posteriormente un ejemplo de un programa utilizado para realizar la transferencia de información.

Etapa del Proyecto: Carga de Datos en SYBASE
Ambiente Operativo: UNIX
Lenguaje o Intérprete de Comandos a Utilizar: *Shell, Querys* SQL

Estructura: Este tipo de programación se utiliza para ejecutar instrucciones que actúan directamente sobre archivos planos y permiten generar conexiones a la base de datos; para esto se generan *STORE PROCEDURES*, que cargan los datos de los archivos planos generados en UNISYS a SYBASE.

- Estándares de los *Store Procedures* (SP)

Estructura de los nombres de los programas:

SP_SISTEMA//FUNCION//VERSION:

- SP (2 posiciones): Indica que el archivo es del tipo *Store_Procedure.*
- UNDERSCORE (una posición): Para separarlo del comentario (los estándares de SYBASE así los manejan).
- SISTEMA (3 posiciones): Pueden ser INV, CHE, AHO, etc.; hay casos en los que el SP puede ser general o privado, por lo tanto tendrán la siguiente nomenclatura:

  GEN: Que indica que la función se puede aplicar para todos los programas.
  Ejemplo: obtener la fecha del sistema.
  PVD: Indica que la función solo puede ser llamada en ese programa.
  Ejemplo: obtener la fecha del sistema en diferentes parámetros.

- FUNCION (3 posiciones): Explicación muy breve del objetivo del programa. Ej.:
  GENCON: Genera Contratos.
  GENSAL: Genera Saldos.
  GENDOC: Genera Documentos, etc.
- VERSION (2 posiciones): Indica la última versión. Ej.: 01, 02, etc.

Ejemplo de la nomenclatura de un *Store Procedure*, que actualiza la Tabla de Contratos de Inversiones:

SP_INVCONACT01.SQL

Nota:

Todos los archivos deberán ser guardados con la extensión SQL.
Todos los *Store Procedures* deberán ir en mayúsculas.

- Estándares de Programación (*Store Procedure*)

Los programas se dividen en 3 etapas:

1. Encabezado
2. Cuerpo del Programa
3. Funciones

1. Encabezado. La estructura del encabezado se muestra en la Tabla 4.2.

Título	Función
Nombre del Programa:	Nombre físico del programa.
Objetivo:	Descripción breve de la función principal del programa.
Fecha:	Fecha de elaboración del programa.
[Fecha modificación]:	En su caso cuando se haya realizado una modificación posterior a la fecha de elaboración.
Entrada:	Parámetros de entrada.
Salida:	(*Return*) valores que regresa la función.

Tabla 4.2. Estructura para el Encabezado de un Store Procedure.

2. Cuerpo del Programa: Es la estructura principal del programa, es decir, el proceso principal. Este a su vez llama a otras funciones (Tercera Etapa).

3. Funciones: Son todos los procedimientos que actuarán en la función principal y/o en otras funciones.

- Declaración de Variables

Para una mayor claridad en los tipos de variables, se hace una pequeña descripción dentro de la declaración de la misma. A continuación se muestra la estructura de una variable: **alcance//tipo de dato//Objetivo de la variable**

- Alcance (1 posición): Existen 3 tipos de alcance y se muestran en la Tabla 4.3.
- Tipo de Dato (1 posición): Indica de que tipo de variable se trata. Existen diferentes tipos de variables y se describen en la tabla 4.4.
- Descripción: Breve explicación de la variable. Generalmente se utilizan de 3 a 6 posiciones.

Alcance	Representación	Descripción
Generales	g	Se utilizan en todas la funciones de los programas. Este tipo de Variables no son recomendables ya que pierden la consistencia de los procedimientos y pueden producir confusión dentro del programa.
Locales	l	Sólo se utilizan en las funciones que hacen referencia a ellas.
Parámetro	p	Parámetros de una función.

Tabla 4.3.Tipos de alcance de las variables en un Store Procedure.

Tipo de Dato	Representación	Descripción
Integer	n	Sólo caracteres numéricos de tipos entero, flotante y doble.
String	s	Carácter alfanumérico.
Boolean	b	Sólo 0 (False) y 1 (True). Sirven para regresar valores de falso o verdadero.
Datetime	d	Tipo Fecha.
Const	c	Representan valores que no cambian en el programa.

Tabla 4.4.Tipos de Variables en un Store Procedure.

Palabras Reservadas: Todas deber ir con mayúsculas. Ej.: *SELECT, FROM, WHERE*

Comentarios: Cada procedimiento debe llevar una breve descripción de lo que hace, la información que es parámetro de entrada, y el resultado que se obtiene como salida, cuando un proceso concluye. Y en algunos casos se escribirán comentarios en las líneas para explicar lo que se está realizando.

Ej.:
//1.1 Obtiene Tasa de Rendimiento
InTasaRendimiento = InInteres * IsFactor

Se presenta más adelante un ejemplo de un desarrollo generado para la carga de información.

Etapa del Proyecto: Explotación de la Información de Sybase
Ambiente Operativo: Windows 95
Lenguaje o Intérprete de Comandos a Utilizar: *Visual Basic 4.0 y GQL Windows.*

Estructura: La programación en este tipo de lenguajes orientado a eventos, y de desarrollo visual, ayudan a ahorrar tiempo en la generación de código, ya que poseen editores de código que revisan la sintaxis y la corrigen mientras el programador escribe.

Para el proyecto se consideran dos formas de explotación de la información, que utilizará el Centro de Información:

La primera es por medio de *GQL Windows*; es una aplicación capaz de generar reportes, a partir de las tablas en SYBASE, de una manera rápida y sencilla casi usuario.

La segunda es desarrollando una aplicación propia en *Visual Basic*, que se conecte a SYBASE y realice las consultas necesarias; además de presentar una variedad de reportes, cambiando tanto su complejidad de consulta, como de su presentación.

Se han propuesto estas dos formas para explotar la información debido a que el Banco posee las licencias del software original, y por lo tanto no es necesario realizar inversión alguna.

Hemos recomendado la utilización de *Visual Basic* para realizar el desarrollo, ya que de esta forma se pueden obtener informes más precisos de la información contenida en la Base de Datos de Sybase. Además de ser una plataforma más estable y que permite tener la posibilidad de crecer el modelo, hasta posiblemente convertirse en el *front-end* de un *Dataware Housing* para el Banco.

Los programas dentro de *Visual* Basic constarán de varios módulos, que junto con las formas definirán al proyecto o a las aplicaciones, entre los módulos que deberán aparecer generalmente encontramos el de Inicio (donde abriremos el ambiente), el de operaciones de Base de Datos (que nos conectará a la misma y contendrá las rutinas relacionadas), el de Validación (que deberá contener las rutinas que validen datos introducidos o procesados) y el módulo de proceso (que contendrá el cuerpo del programa en sí).

Enseguida se muestra el código generado para explotar la información.

Tablas de Proceso

Una parte vital para la realización del proyecto es la definición de las tablas que contendrán la información, tanto en estructura como sus relaciones. En este apartado se mostrarán cual es la estructura propuesta para el manejo de las tablas, tomando como base las consultadas en el *Mainframe* y su nueva estructura en SYBASE.

Unicamente definiremos los bosquejos iniciales de las tablas, las cuales podrán verse finalmente diseñadas en la página 133 de este trabajo en el apartado de Tablas, en donde se muestra tanto el diseño final del modelo como los *layouts* de las tablas generadas.

La tabla de Contratos deberá contener los siguientes aspectos: números de cuenta, productos del banco, monedas, fechas de alta y cancelación de cuentas, sectores, giros, números de relación con otros sistemas del banco como información primordial.

La tabla de Clientes deberá ser fundamental para guardar la información propia de cada una de las personas que tienen una cuenta en el banco, por lo cual se requerirá de información como la cuenta, el tipo de persona, sus datos generales, el sector contable al que pertenece, la actividad que desempeña, su oficina de registro y fechas de ingreso y modificación.

La tabla de Direcciones deberá contener los campos necesarios para ubicar certeramente al cliente, como lo es domicilio, teléfono, dirección de *e-mail* y fechas de alta y actualización.

La tabla de Saldos únicamente contendrá el número de contrato, el cual será la liga para todas las tablas anteriores, así como la fecha y el monto en que se genera el dato.

Con esto definimos inicialmente las tablas principales del sistema, las tablas relacionales contendrán muchas veces catálogos o datos muy propios de cada sistema o cuenta, por lo que no se detallarán como las anteriores en esta sección, sino que se verán dentro del modelo.

Registros Auxiliares

Es importante mencionar que se contará con tablas "de paso", en las cuales se llevarán a cabo las validaciones necesarias para mandar la información a las tablas definitivas, lo cual nos permitirá, al manejar volúmenes más pequeños de información, evitar tener pérdidas de tiempo al validar la información que diariamente deberá ser sumada, así como asegurarnos de la correcta manipulación de la información.

Reportes

Trabajando en conjunto con el usuario, hemos llegado a establecer algunos formatos básicos para los reportes de:

- Cheques.
- Fondos.
- Ahorros.
- Inversión.
- Mesa de Dinero.
- Tesorería.

Los cuales tienen campos de datos en común, que son:

- Nombre de la Tabla Origen.
- Fecha en que se realiza la Consulta.
- Rango de Fechas que abarca la Consulta.
- Area de Consulta: Regional o Nacional.
- Tipo de Consulta: Cliente Particular o Grupo Empresarial.
- Nombre del Ejecutivo que solicita la Consulta.
- Cifras Totales en reportes de montos.
- etc.

Además de información, el usuario propone algunos formatos que incluyen logotipos oficiales del Banco y espacio para firmas de las personas que avalan la información mostrada en ellos.

Una vez consideradas las dos opciones que se tienen para poder explotar la información: GQL Windows o *Visual Basic 4.0*; se realizan pruebas, primero con GQL Windows; y se observa su desempeño.
Esto es por la facilidad y rapidez que ofrece el paquete; lo que nos dará la pauta para iniciar o no una aplicación propia, desarrollada en *Visual Basic*.

Manejo de Archivos

El manejo de archivos dentro del desarrollo del sistema se basa en el hecho de que existen diferentes sistemas bancarios (Cheques, Ahorros, etc.) , diferentes zonas de aplicación y diferentes tipos de información.

En el apartado de Diseño de Lógica se mencionó la nomenclatura para cada sistema, zona y tipo de información; la construcción de los archivos en UNISYS empleados en el sistema se explica a continuación.

Archivo *"CDI / ZONA / FECHA / INF"*.

Donde:

CDI	=	Centro de Información (Cheques, Ahorro, Fondos, etc.).
ZONA	=	Zona geográfica (Monterrey, Guadalajara, etc.).
FECHA	=	Fecha de generación del archivo (formato de mmdd).
INF	=	Tipo de Información del archivo (clientes, saldos, dirección, etc.).

Por otra parte, cuando se realiza la transferencia de archivos de UNISYS a UNIX (archivo de texto) , la construcción del archivo toma las siguientes características:

Archivo *"CDI / INF / ZONA / FECHA.TXT"*.

La tabla 4.5 muestra la relación que existe entre cada centro de información, zonas geográficas y tipo de información.

	CENTRO DE INFORMACION					
	CHEQUES	FONDOS	AHORROS	INVERSIÓN	MESA DE DINERO	TESORERÍA
ZONAS	MTY , GUA MEX , SAL HER , FOM	MTY , GUA MEX , SAL HER , FOM	MTY , GUA MEX , SAL HER , FOM	MTY , GUA MEX , SAL HER , FOM	MTY , GUA MEX , SAL HER , FOM	MTY , GUA MEX , SAL HER , FOM
INFORMA - CION	CON , SAL CLI , DIR	SAL	CON , SAL CLI , DIR	CON , SAL CLI , DIR , BAJ	CON , SAL CLI , DIR , BAJ	CON , SAL CLI , DIR , BAJ
NUMERO DE ARCHIVOS	24	6	24	30	30	30

Tabla 4.5. Manejo de Archivos.

Todos los archivos generados a través del proceso de transferencias se almacenarán en un directorio de respaldo, es decir, la información original no sufrirá ninguna alteración ya que se trabajará con archivos alternos.

Manejo de Campos

El manejo de campos dentro de los archivos del sistema se encuentra en función del tipo de información del que se esté hablando, la tabla 4.6 muestra una relación de los campos que cada sistema requiere para ser procesado, así como la periodicidad (períodos de tiempo) con la que cada uno debe ser generado.

PRODUCTO	CAMPOS	FUENTE	PERIODICIDAD
CHEQUES AHORROS Y FONDOS	Datos Generales	UNISYS	Diario
	Saldos Diarios		Diario
	Saldos Promedios		Mensual
	Saldos Mensuales		Mensual
	Intereses Pagados		Mensual
	Comisiones Cobradas		Mensual
INVERSIONES	Datos Generales	UNISYS	Diario
	Saldos Diarios		Diario
	Saldos Promedios		Mensual
	Saldos Mensuales		Mensual
	Documentos		Diario
MESA DE DINERO	Datos Generales Inv.	UNISYS	Diario
	Emisiones de Docum.		Diario
	Asignaciones		Mensual
	Compras y Valuaciones		Mensual
TESORERIA	Datos Generales	UNISYS	Diario
	Saldos Diarios		Diario
	Saldos Vencidos		Diario
	Documentos		Mensual

Tabla 4.6. Manejo de Campos.

El campo de Datos Generales está integrado por el Nombre del cliente, su Dirección, Teléfono, Número de Cuenta, Cuentas Relacionadas, etc.

La distinción entre campos de tipo numérico, alfanumérico y de fecha, permitirá eliminar la posibilidad de errores por incompatibilidad de valores al realizar operaciones entre campos.

Manejo de Mensajes

El manejo de mensajes dentro del desarrollo del sistema es de suma importancia para el operador del mismo, pues en el caso de ocurrir alguna anomalía detectable por el sistema, éste debe avisar al usuario del tipo de problema que ha ocurrido, de la posible causa que generó el error, y de cómo resolverlo.

Para los operadores de los procesos *Job* en los cuales se realiza la carga de la información, se empleará el formato de mensaje descrito en la figura 4.2:

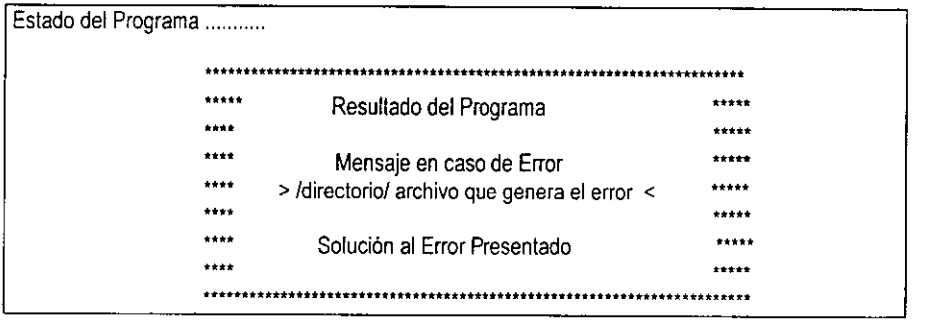


Figura 4.2. Formato de Mensajes.

En las figuras 4.3 y 4.4 se muestra un ejemplo para un mensaje de proceso exitoso y un mensaje de error durante el proceso *Job* en ambiente UNIX.

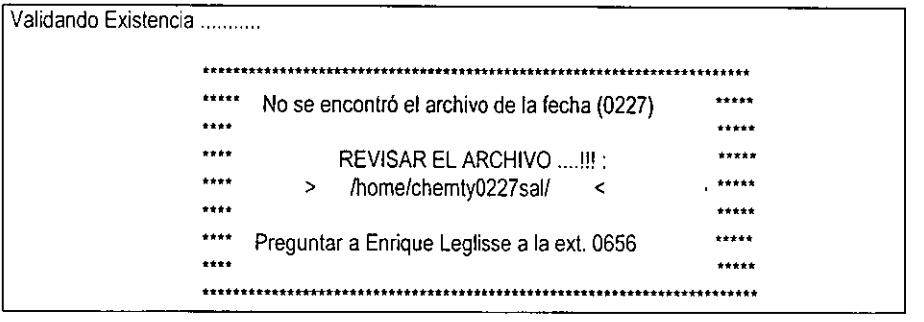


Figura 4.3. Ejemplo de mensajes de error durante el proceso.

```
Validando Existencia .....

**********************************************************************
****     Se encontraron todos los archivos de la fecha (0227)     ****
****                                                              ****
**********************************************************************
```

Figura 4.4. Ejemplo de mensajes proceso exitoso.

Para el manejo de mensajes del usuario final, los formatos son más amigables debido a la programación en *Visual Basic*. La figura 4.5 muestra el formato de mensaje para el usuario en la aplicación.

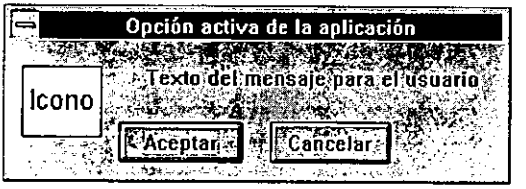


Figura 4.5. Formato de mensajes en Visual Basic.

En este ambiente de programación existen diferentes íconos (dibujos) que representan el *status* del programa tal como se muestra en la figura 4.6.

a) b) c) d)

Figura 4.6. Iconos de mensajes en Visual Basic.

De la figura anterior se derivan los siguientes incisos:

a) Icono de Exclamación.- Representa una advertencia a un proceso que pudiera contener cierto riesgo para el usuario, por ejemplo : eliminar un campo de un reporte de salida.

b) Icono de Información.- Representa una acción en la cual por algún motivo el proceso no fue realizado como debiera, por ejemplo : realizar una consulta por fechas introduciendo como parámetros fechas no válidas.
c) Icono de Interrogación.- Se despliega este icono cuando el usuario omite alguna operación o cuando se requiere confirmar la realización de la misma, por ejemplo: cerrar un reporte sin guardarlo o guardar un reporte con el nombre de uno ya existente.
d) Icono de Alarma.- Representa un aviso de que se realizará una operación crítica para el funcionamiento del sistema, por ejemplo : generar un reporte con intervalos de fechas muy grandes puede causar un desbordamiento de memoria.

La ejecución de cada una de las rutinas que se emplean en el desarrollo del sistema serán indicadas como parte del apéndice "B", en donde definiremos los fuentes y como debe ejecutarse cada rutina individualmente.

Procesos Especiales

Anteriormente se explicó que debido a la gran cantidad de registros almacenados dentro de las tablas originales y para facilitar el proceso de explotación, se requiere realizar un extracto de información dentro del mismo ambiente (UNISYS), de una tabla a origen a una tabla destino.

Posteriormente esta tabla se convierte a formato de texto para generar un archivo de base de datos SYBASE, durante este proceso es necesario tener la certeza de que toda la información se conservará integra, es decir, que no se omitió ningún registro durante este proceso.

Lo anterior se resuelve corriendo un proceso alterno al proceso de transformación, este proceso generará un archivo de bitácora en el cual podrá compararse el número total de registros que se encuentran en el archivo de texto, contra el número total de registros de la tabla original.

Si el número de registros de ambas tablas coincide, quiere decir que la transferencia se llevó a cabo sin ningún problema, por el contrario, si el número de registros es distinto, deberá realizarse un segundo proceso de transferencia a la tabla original y comparar nuevamente el resultado.

Este proceso de generación de archivo de bitácora es denominado como *Proceso Especial*, y se realiza simultáneamente a la transferencia de UNISYS a UNIX.

En los siguientes apartados de este trabajo se presenta el diseño, los diagramas que originó el mismo, la codificación, los estándares utilizados como el código mismo de los desarrollos y características propias del modelo, tales como diagramas de relación, diccionario de datos y una estructura completa del modelo en cuanto a su estructura, tanto los campos, sus tipos y longitudes, como los principales índices utilizados para conformar el modelo.

Cabe mencionar que los datos aquí presentados son una aproximación del modelo final, debido a la confidencialidad de la información de la institución bancaria en la que se llevó a cabo este trabajo.

4.3 Diseño del Proyecto

Bases de Datos

Como se ha descrito anteriormente, el desarrollo de las Bases de Datos del Sistema se realizarán en *Sybase SQL Server 11.*

Sybase es lo que se conoce como Sistema de administración de Bases de Datos Relacionales RDBMS (Relational Database Management System); los DBMS, sin ser todavía RDBMS, están formados por una colección de programas que nos dan la capacidad de almacenar, modificar y extraer información de una base de datos. Existen diferentes tipos de *DBMS*, que abarcan desde pequeños sistemas que funcionan en computadoras personales, hasta los grandes sistemas que se usan en *mainframes*.

Desde el punto de vista técnico, las definiciones de DBMS pueden diferir ampliamente. Los términos como relacional, compartida, plana, y jerárquica, todos se refieren a la manera en que un DBMS organiza su información internamente. Esta organización interna puede afectar en que tan rápida y flexiblemente se puede extraer la información de ella.

Las solicitudes de información a una base de datos son hechas en la forma de una búsqueda (*query*), lo cual es una pregunta estilizada. Por ejemplo, la búsqueda:

SELECT ALL WHERE NAME = "GONZALEZ" *AND AGE*>35.
Está solicitando todos los registros en los cuales el campo de nombre (*NAME*) es GONZALEZ y el campo edad (*AGE*) es mayor de 35.

El conjunto de reglas para construir las búsquedas (*querys*) es conocido como lenguaje de búsquedas (*query language*). Los *DBMS* soportan diferentes lenguajes de búsquedas; entre los cuales podemos mencionar el *SQL (Structured Query Language)*, que es un lenguaje que pretende ser el estándar para la creación de bases de datos robustas que requieran ejecutar búsquedas avanzadas en el diseño de las mismas . Los lenguajes sofisticados para manejar sistemas de bases de datos son llamados de cuarta generación (4GL).

Ahora un *RDBMS* es un tipo de *DBMS* que almacena datos en forma de tablas relacionales.

Las bases de datos relacionales son muy potentes, por la razón de que requieren de muy poca definición de cómo los datos están relacionados y cómo pueden ser extraídos de la base de datos. Como resultado, la misma base de datos puede ser vista de diferentes formas.

Una característica importante de los sistemas relacionales es que es una base de datos única, puede extenderse a diversas tablas. Esto difiere de las bases de datos en archivos planos, en los cuales, cada base es por sí misma una tabla única.

Después de todo, los sistemas a gran escala de bases de datos son *RDBMS*. Pequeños sistemas de bases de datos utilizan otros diseños, que limitan la flexibilidad en la ejecución de búsquedas.

Las características principales de *Sybase SQL Server* 11 son:

Alto desempeño escalable y gran eficiencia con tecnología probada:

- Funciona en una gran variedad de plataformas, desde PC hasta servidores de múltiples procesadores, se puede seleccionar el hardware apropiado para cada trabajo o cambiarlo si se necesita.
- Alcanza grandes niveles de transacción y soporta diversos usuarios a través de su eficiente proceso de multilectura *SQL Server*.
- Procesos de guardado (*Stored Procedures*) y de activación inmediata(*Triggers*), mantienen la integridad de la base de datos.
- Si la integridad de los datos es alterada, un *Trigger* restaura la operación conservando la integridad.
- Los *Stored Procedures* encápsulan la compleja lógica del negocio en unidades compactas de código, que múltiples aplicaciones pueden reutilizar para un mejor manejo de información.

Adaptabilidad de Cargas de trabajo para un máximo aprovechamiento de recursos:

- Se puede configurar cualquier número de memorias caché por servidor, por lo tanto se pueden optimizar completamente los recursos de la base de datos.
- Proceso de transacciones en línea *OLTP (On-Line Transaction Processing)* y aplicaciones de soporte de decisiones, pueden correr en el mismo servidor, aprovechando mejor la inversión en los sistemas de información.
- El Módulo administrador de memoria lógica nos da la capacidad de optimizar recursos de memoria y ajustes en aplicaciones.

Disponibilidad de Información, Incremento de Productividad:

- Discos en espejo, y alta velocidad en restauración y respaldo minimizan el impacto de una falla de hardware en aplicaciones funcionando.
- *SQL Server* soporta completamente restauración y respaldo en línea, haciendo a la información siempre disponible para los usuarios.

Sybase cumple con los estándares:

- *ANSI/ISO SQL*-89 y el nivel de entrada *ANSI/ISO SQL*-92.
- Soporta aplicaciones escritas sobre *ODBC (Open DataBase Connectivity)* y estándares *X/Open XA*.
- Soporta diversos protocolos de red, habilitando virtualmente cualquier máquina cliente para conectarse a *SQL Server* ejecutándose en cualquier plataforma.

Especificaciones Técnicas:

- *SQL Server* 11 está disponible para la mayor parte de las plataformas *UNIX*, así como para *windows NT*.

Requerimientos de Hardware:

- 15 *MB* de *RAM* para *SQL Server.*
- 48 *KB RAM* por cada usuario adicional.
- 6 *MB* de espacio de disco para almacenar el software del sistema.

Estadísticas del producto *Sybase SQL Server:*

- 32, 767 bases de datos por cada *SQL Server.*
- Hasta 4 *terabytes* para el tamaño de la base de datos.

- Se actualizan hasta 8 bases de datos diferidas al mismo tiempo.
- Para una sola transacción se pueden abrir hasta 16 bases de datos.
- Una búsqueda puede cubrir hasta 16 tablas.
- 250 columnas por tabla.
- 251 índices por tabla.
- Los registros por tabla solamente se limitan por el espacio disponible en disco.
- 16 columnas por índice compuesto.
- 30 caracteres por nombre de la base de datos.

Tipos de Datos:

a) Numérico:

- *int*: entre –2,147,483,648 y +2,147,483,647 inclusive.
- *smallint*: entre –32,768 y +32,767 inclusive.
- *tinyint*: entre 0 y 255 inclusive.
- *float*: números de punto flotante en 8 bytes.
- *shortfloat*: números de punto flotante de 4 bytes.
- *money* las columnas de tipo *money*(dinero) almacenan valores exactos entre +/-922,337,203,685,447.5807 dólares (tipo de moneda); con cuatro lugares de precisión decimal, valores numéricos con precisión específica y escala hasta un máximo de 38 dígitos.
- *double precision:* números de punto flotante de 8 bytes.

b) Carácter:

- *char(n):* columnas carácter (letras, números, símbolos) hasta 255 caracteres de longitud.
- *varchar(n):* carácter de longitud variable hasta 255 caracteres.

c) Objetos binarios, *Binary Large Object (BLOB):*

- *text*: carácter de longitud variable, hasta 2 GB de longitud.
- *image*: columnas de longitud variable, hasta 2 GB de longitud.
- *binary*: columnas hasta de 255 bytes de longitud, mantienen datos binarios de longitud fija.
- *varbinary(n)*: columnas de longitud variable, mantienen hasta 255 bytes de datos binarios.

d) Tipo de datos varios:

- *bit*: columnas que mantienen datos 0 y 1.
- *datetime*: fecha y hora del día con una precisión de 1/30 de un milisegundo.
- *shortdatetime*: fecha y hora del día con una precisión de 1 minuto.
- *timestamp*: una columna de este tipo de datos es actualizada automáticamente cuando un registro es alterado.
- *indentidad*: número secuencial colocado por el sistema.

Uso de procedimientos almacenados (*Stored Procedures*).

Es posible tomar un conjunto de procesos por lotes (*batch*), de instrucciones *SQL* y colocarlos en un "procedimiento" almacenado en un servidor de base de datos *Sybase*. Este es llamado proceso almacenado (*Stored Procedure*).

La sintaxis para declarar un *Stored Procedure* es:

create proc {nombre del procedimiento}
as
{instrucción o bloque de instrucciones}

Los *Stored Procedures* están almacenados en el servidor en un formato precompilado. Si diversos usuarios necesitan invocar básicamente la misma búsqueda o el mismo procedimiento, solamente modificando ciertos parámetros, tiene sentido colocar las instrucciones que lo forman dentro de un *Stored Procedure*.

Transact-SQL que es lo que mantiene los *Stored Procedures*, indica que no tienen que ser compilados cada vez que son invocados. Es posible implementar algoritmos que pueden ejecutarse tanto en el cliente como en el servidor como *Stored Procedures*. Aquí pueden haber criterios múltiples que decidirán a donde debe ir el flujo de los programas.

Las tablas temporales "temp" de *Sybase* pueden utilizarse en diversas situaciones. Un *Stored Procedure* puede implementar series de búsquedas (*querys*), cada una basada en una tabla temporal de pasos anteriores. Los datos en las tablas de bases de datos y los datos en las tablas temporales, son todos localizados en los *Stored Procedures*.

Además, un *Stored Procedure* utiliza ciclos del procesador en el servidor, no en la computadora del cliente. Esto puede ser bueno o malo, dependiendo de las especificaciones del sistema cliente servidor que se está diseñando.

Los *Stored Procedures* pueden llamar a otros *Stored Procedures*. Esto permite modularidad en el código. La modularidad es limitada, como siempre, por la naturaleza de las tablas de base de datos que están siendo accesadas.

Es posible dar acceso a los usuarios hacia una tabla usando solamente *Stored Procedures*. De esta forma, pueden ser usados para proveer de vistas lógicas de la información (utilizando la seguridad para reforzar estas consultas).

Opciones de acceso a bases de datos

Para establecer una comunicación entre el sistema de consulta, desde las computadoras de los clientes hacia el servidor de base de datos *UNIX*, se utilizará en el desarrollo de la aplicación los llamados *ODBC (Open DataBase Connectivity)*.

Los *ODBC* son métodos estándar de acceso a bases de datos, desarrollados por *Microsoft Corporation*. La meta de los *ODBC* es hacer posible accesar a los datos desde cualquier aplicación, independientemente del cual es el sistema de administración de base de datos que se esté usando.

Un *ODBC* administra esto insertando una capa intermedia, llamada *driver* de la base de datos, entre la aplicación y el *DBMS*. El propósito de esta capa es traducir las búsquedas (*querys*) de la aplicación en instrucciones que el *DBMS* comprende. Para este trabajo, ambos, la aplicación y el *DBMS* deben de ser compatibles con el *ODBC*, esto es, la aplicación debe de ser capaz de utilizar instrucciones *ODBC* y el *DBMS* debe ser capaz de responder a ellas.

Desde la versión 2.0, el estándar *ODBC* soporta *SQL* estándar. Esto no representa problema alguno para el desarrollo de la aplicación en *Visual Basic* 4, ya que existe un *ODBC* propio de *Visual Basic* para establecer comunicación con *Sybase SQL Server*.

4.4 Diagramas , programación Shell y pantallas del CDI

Diagramas

Los diagramas de procedimientos son una herramienta que además de permitir al programador controlar el flujo de la información, le ayudan además en el mantenimiento del sistema , agregar nuevos módulos al mismo y lo más importante, desarrollar una codificación clara y legible.

A Continuación se mostrarán los diagramas empleados en la elaboración de CDI

El diagrama 4.1 nos muestra la secuencia que se llevo acabo para la construcción del CDI, en tal diagrama observamos los procesos principales (señalados con negritas) que son: Extracción de la Información, Transferencia de los archivos, la Bitácora, la Carga de la Información y finalmente su Explotación. Cabe mencionar que en el transcurso de un proceso a otro se realizaron validaciones de archivos.

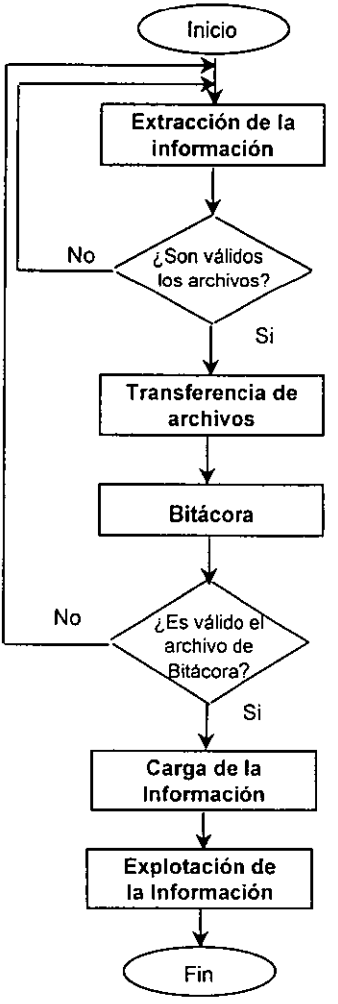


Diagrama 4.1. Procesos principales del CDI.

El diagrama 4.2 nos señala los pasos realizados para hacer la Extracción de la Información del mainframe, la cual se tiene en tablas independientes en el ambiente Unisys, primeramente se abre la sesión de CANDE, después se verifica la conexión a las tablas originales y si es válida tal conexión con la tabla original, se copia la base de datos al nuevo formato, dejando hasta este momento el archivo en el mismo ambiente Unisys. En este proceso utilizamos Cobol 74 como herramienta de programación.

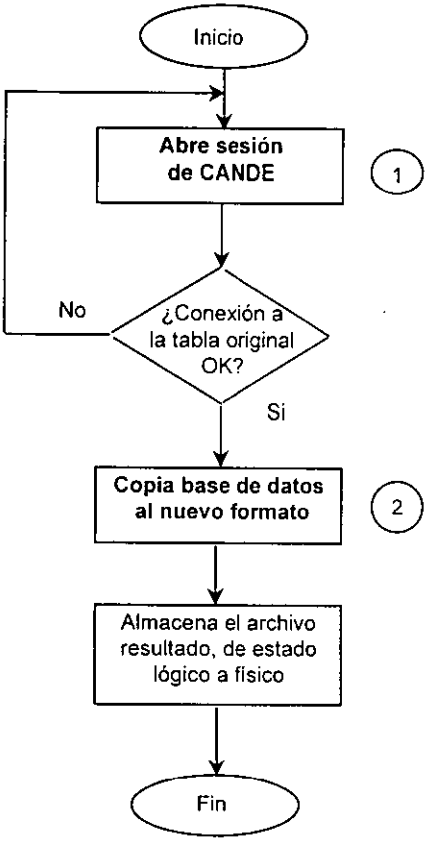


Diagrama 4.2. Proceso de Extracción de la Información.

Para ir al detalle de cómo se abre una sesión de CANDE, veremos el diagrama 4.3, que nos dice primeramente que el usuario inicia la comunicación con el *mainframe* utilizado vía el emulador Reflection, en donde el usuario está debidamente autorizado para tal acceso.

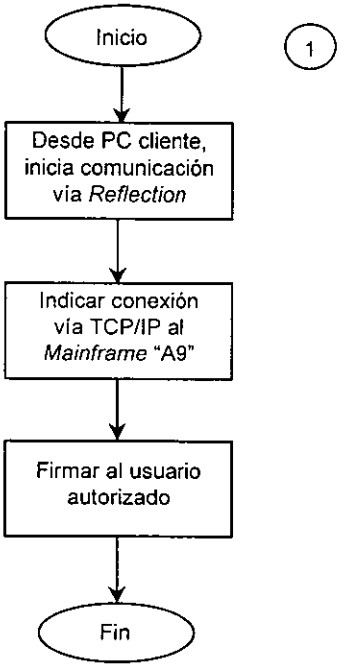


Diagrama 4.3. Subrutina que abre una sesión de CANDE.

El diagrama 4.4 nos señala la subrutina que copia la base da datos al nuevo formato. En donde se pregunta si se trata de una carga inicial, si no lo es, se realiza la lectura de la fecha, finalmente se procesan los registros para ser guardados en el nuevo formato.

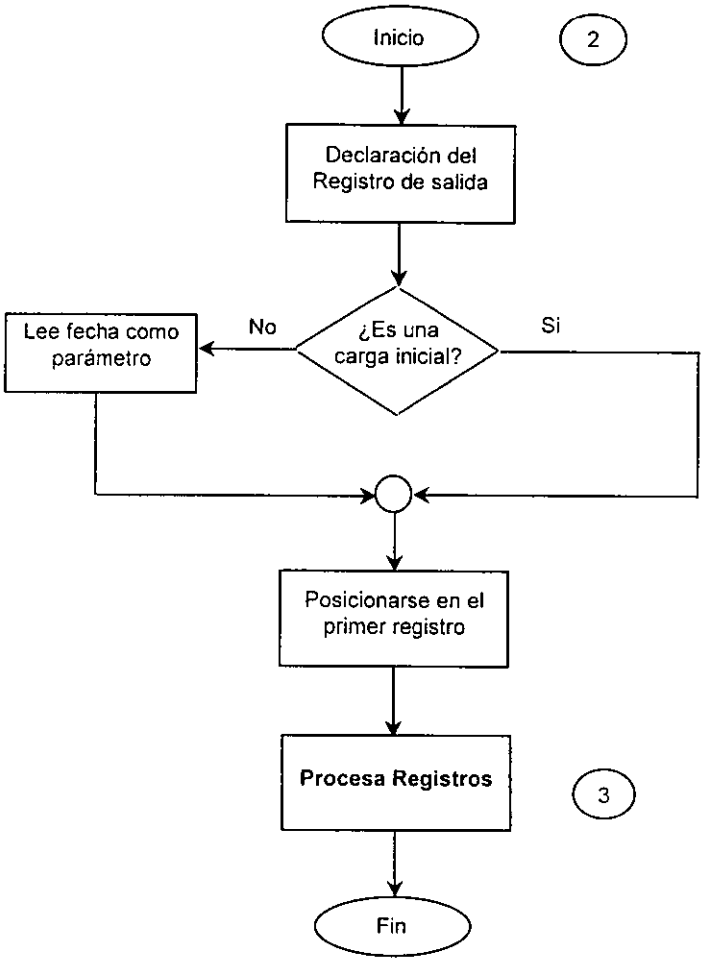


Diagrama 4.4. Subrutina que copia la base de datos al formato nuevo.

Para saber la forma de como se procesan los registros, vemos el diagrama 4.5, en el que observamos la secuencia que debe seguir cada registro leído, los tipos de validaciones, como son: realizar el proceso hasta llegar al ultimo registro, y cumplir con las condiciones de parametrización para poder ser finalmente escrito.

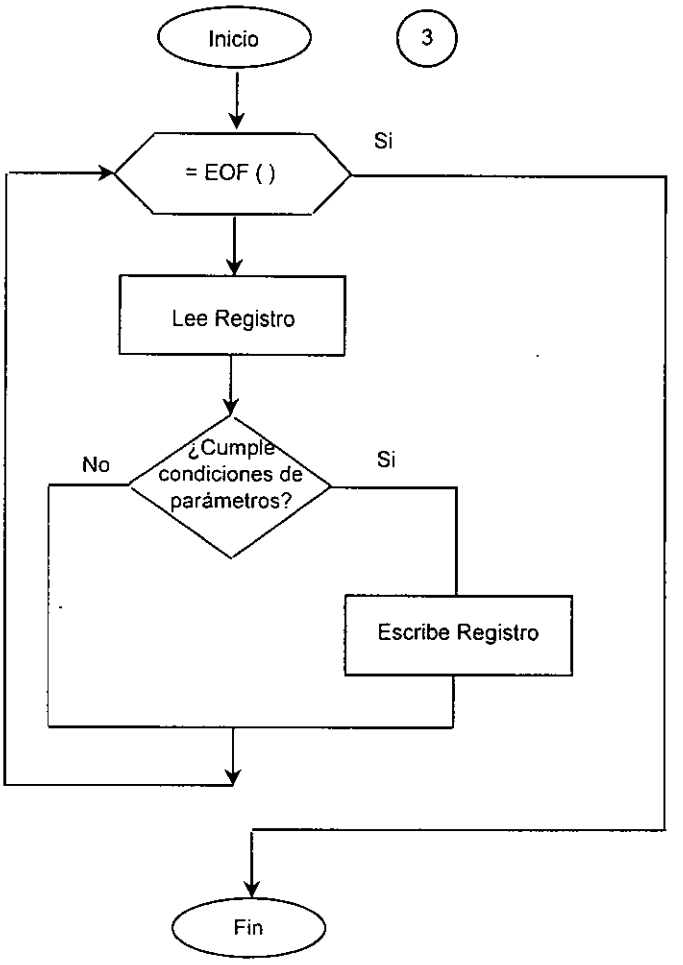


Diagrama 4.5. Subrutina que procesa registros.

Continuando con los procesos principales, vemos en el diagrama 4.6 el de Transferencia de los archivos. En este proceso primeramente se valida la creación de los archivos en Unisys, después se verifica si se encuentran todos los archivos; si no es así termina, de lo contrario continua para iniciar la sesión de TELNET, verificando la conexión y, si todo está en orden, finalmente se transfieren los archivos, preguntando por el número de archivos transferidos.

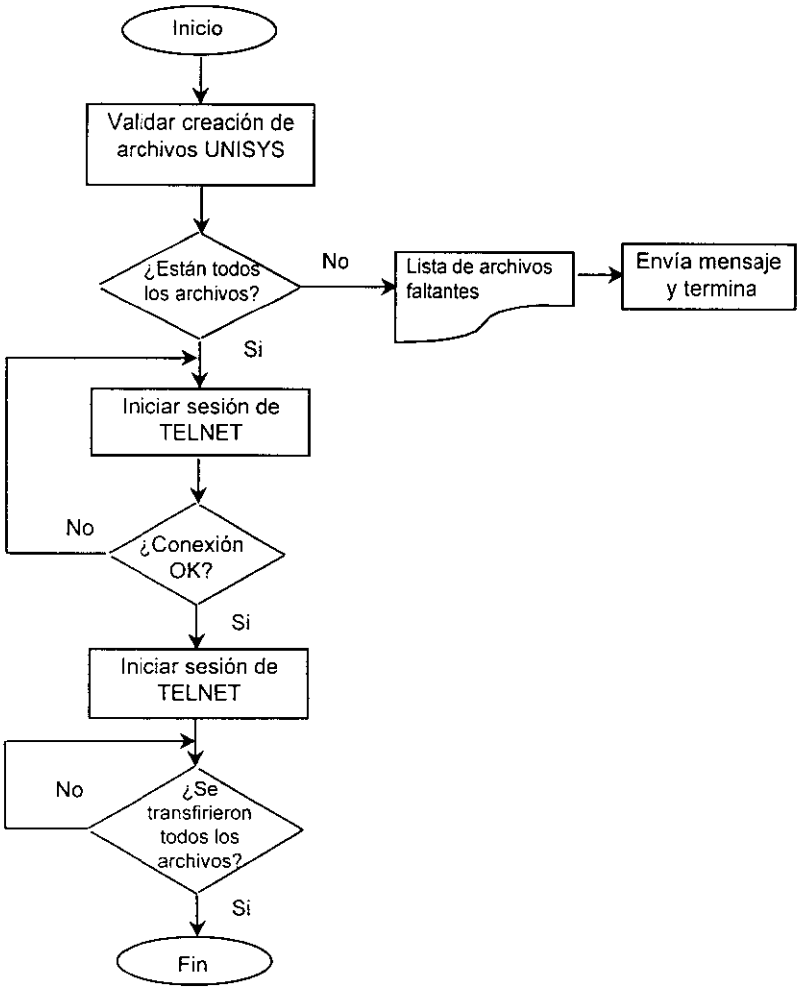


Diagrama 4.6. Proceso de Transferencia de archivos.

En el diagrama 4.7, se nos muestra la secuencia que se sigue para la Bitácora, primeramente se realiza la construcción del archivo de Bitácora en Unisys, después se transfiere y se cargan los archivos de texto en tablas de UNIX, finalmente se realiza el conteo de los registros en UNIX y se muestra el resultado.

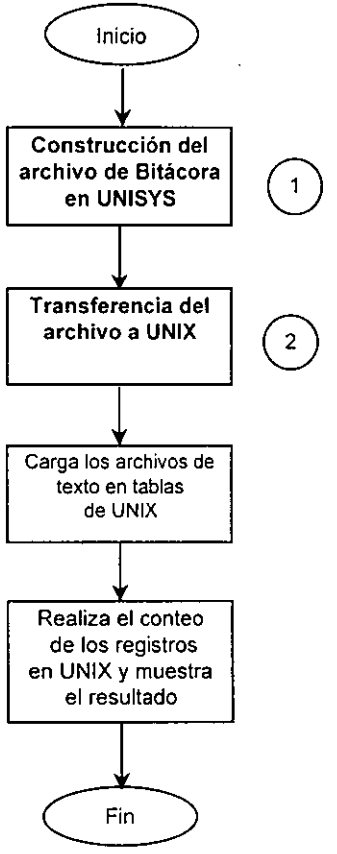


Diagrama 4.7. Proceso de Bitácora.

Para la construcción del archivo de Bitácora en UNIX, vemos que el diagrama 4.8 señala que primero se abren las tabla de Cheques, Ahorros, Fondos, Inversiones e Inversiones Mesa de Dinero que están en Unysis, después se construye el archivo con los datos de: Contrato, Cliente, Saldo, Dirección y Bajas, finalmente se renombra tal archivo, por ejemplo: BITAHOMTY0101.txt

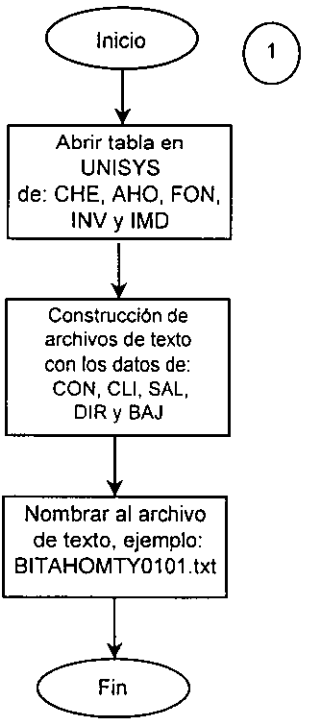


Diagrama 4.8. Subrutina de construcción del archivo de Bitácora en UNIX.

El diagrama 4.9 nos indica la subrutina de la transferencia del archivo a UNIX, en donde, primero se busca el archivo y se abre la conexión vía Reflection, se verifica la conexión, si es incorrecta envía un mensaje y termina, de caso contrario se envía el archivo vía TELNET.

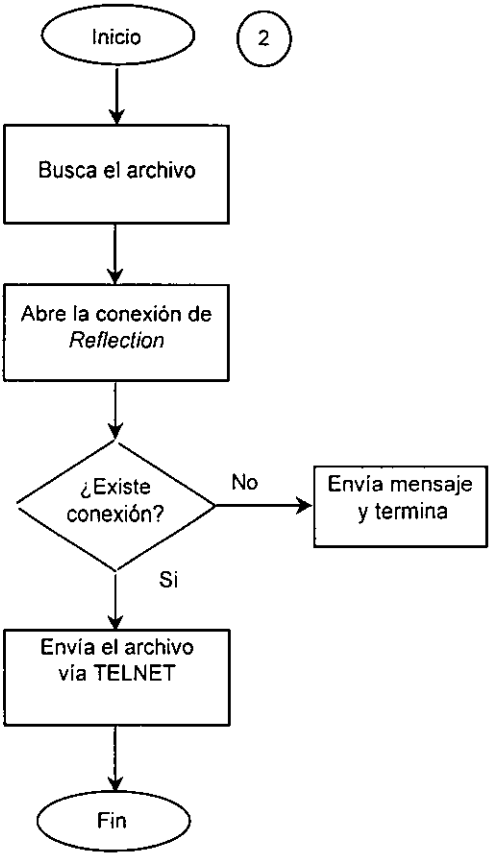


Diagrama 4.9. Subrutina de transferencia del archivo a UNIX.

El siguiente proceso principal es la Carga de la Información. El diagrama 4.10 nos señala el flujo de la carga, donde, primero se va al directorio llamado home/ctes/archs, luego verifica la existencia y las zonas, si existen continúa con el procesamiento de los archivos (lo vemos a detalle en el diagrama 4.11), de caso contrario envía mensaje y termina después de listar los archivos faltantes.

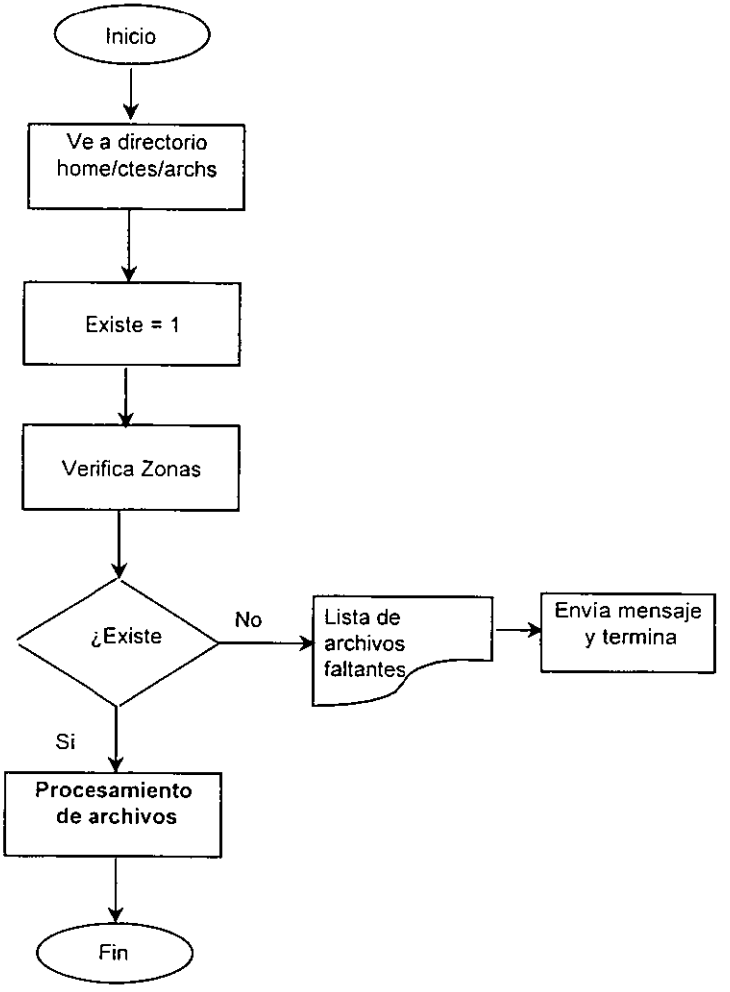


Diagrama 4.10. Proceso de Carga de la Información.

La subrutina del procesamiento de los archivos se presenta en el diagrama 4.11, el cual nos indica las validaciones que se le realizan a los archivos, por ejemplo, se pregunta si son todos los archivos, si es así, termina el procesamiento, en el caso de que falten archivos, pregunta nuevamente la existencia de los archivos, si es no se carga la tabla definitiva con la instrucción "bcp", en caso contrario se hace una carga a una tabla temporal, se validan los datos y se carga la información en una tabla definitiva (proceso que veremos en el diagrama 4.12), finalmente se verifica si son todos los archivos.

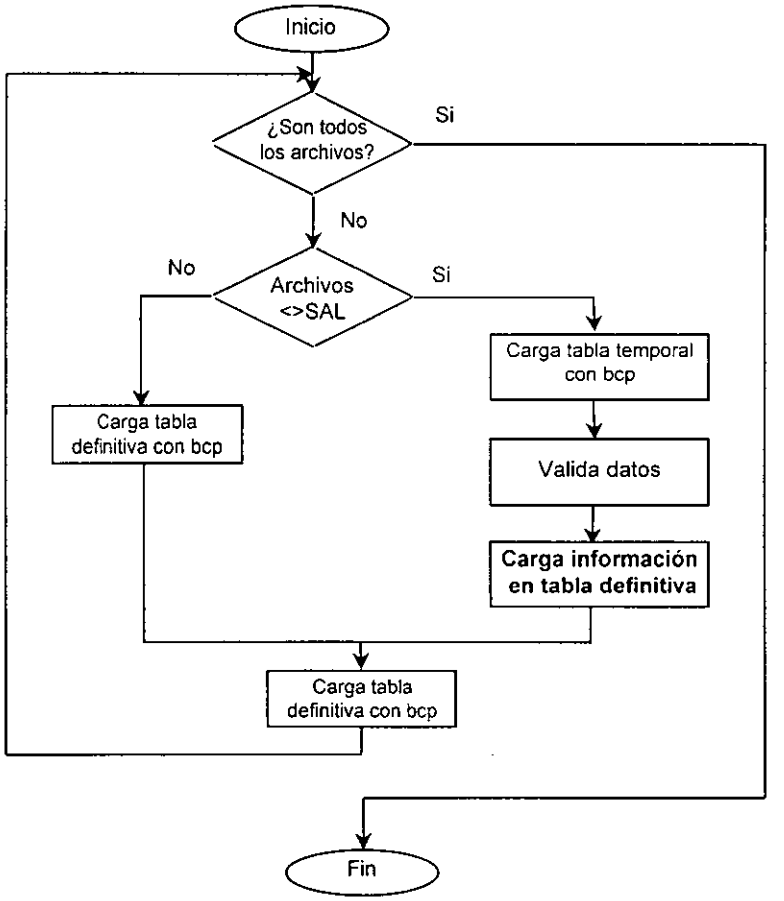


Diagrama 4.11. Subrutina de procesamiento de archivos.

El diagrama 4.12 nos señala la subrutina de Carga de la información en la tabla definitiva. Primeramente se abre la sesión de Sybase, después pasa por los tipo de validación, de los que hablaremos en los diagramas 4.13 y 4.14, una vez que se valida se introduce la información en tablas se Sybase.

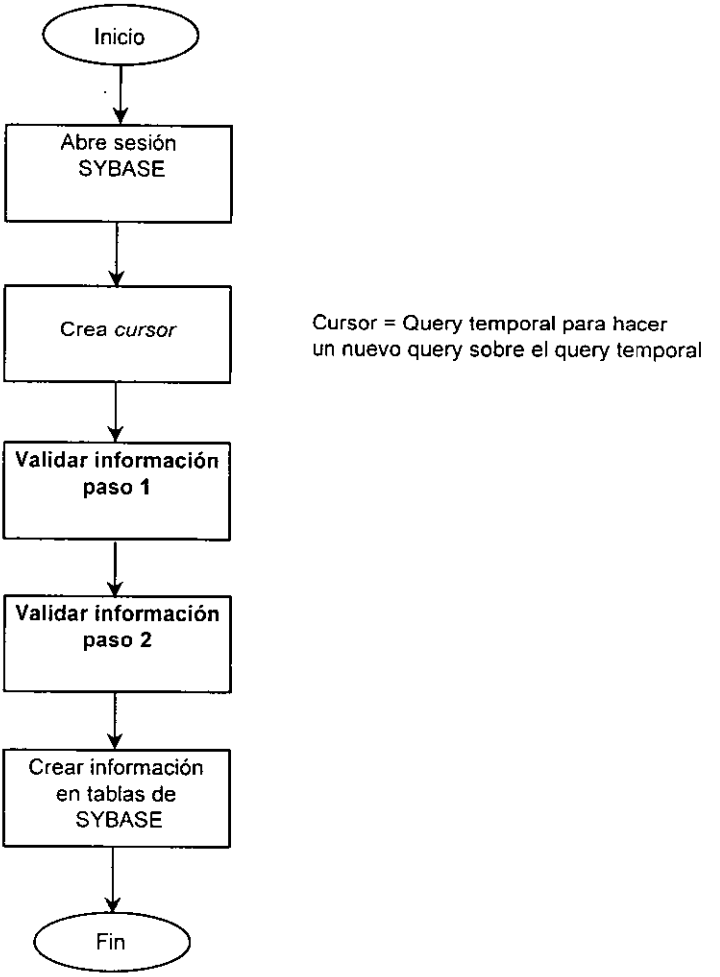


Diagrama 4.12. Subrutina de carga información en tabla definitiva.

Primera validación de la información, ver diagrama 4.13. De manera secuencial se va verificando la existencia de: Constante, Cuenta, Dirección, Saldo y Bajas; si alguna de estas no existe se enciende una bandera y continúa.

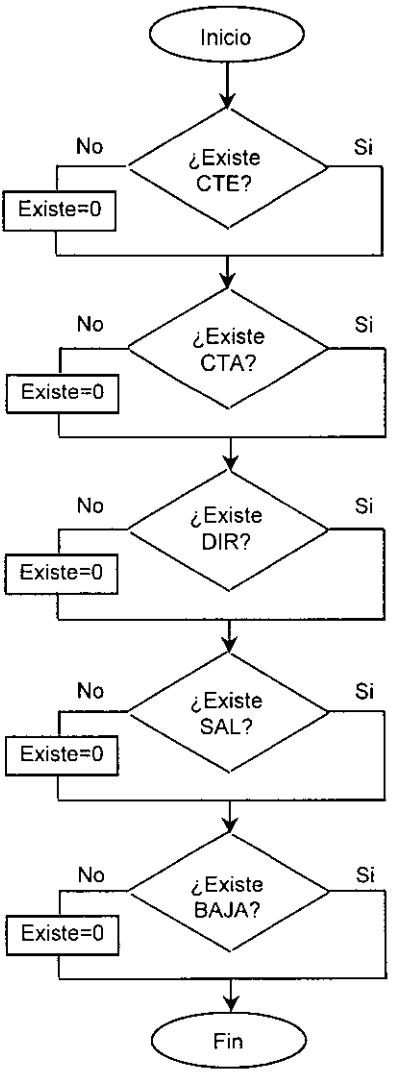


Diagrama 4.13. Paso 1 de la subrutina de validar información.

Segunda validación de la información, ver diagrama 4.14. Primeramente se hace una comparación de los registros en UNIX con los registros de Sybase, estos registros son: Cliente, Dirección, Contrato, Saldos y Bajas; posteriormente se compara el número de registros en UNIX contra los de Sybase, si no son iguales envía un mensaje y termina; en caso contrario genera un reporte de resultados y verifica la información de dicho reporte, si no ésta completa termina, de caso contrario se carga la información.

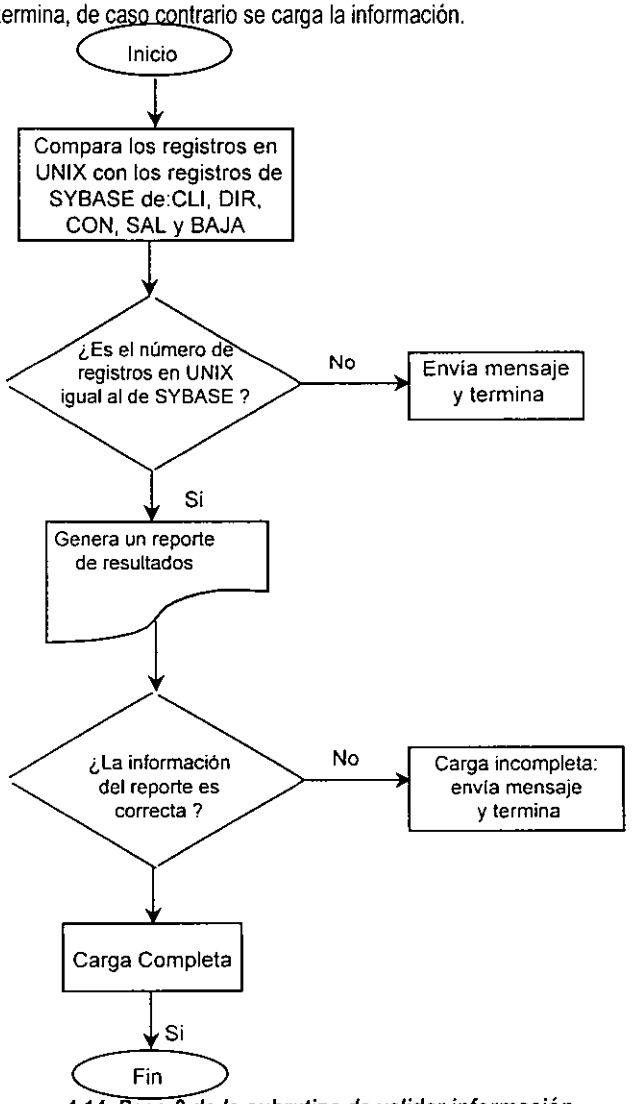


Diagrama 4.14. Paso 2 de la subrutina de validar información.

El proceso principal final es la Explotación de la Información, como lo vemos en el diagrama 4.15. Nos dice que se realiza una lectura de la información almacenada en Sybase con la ayuda de Visual Basic, posteriormente se procesa tal información vía Query´s y finalmente se generan reportes (los vemos a detalle en el diagrama 4.16).

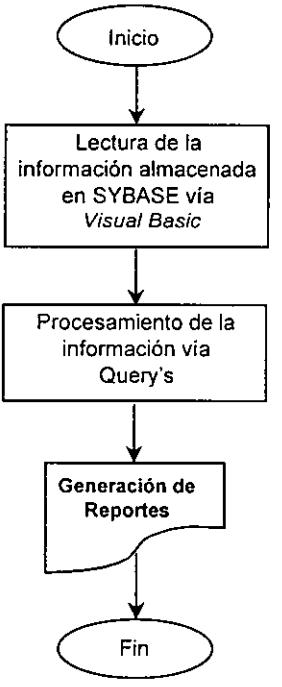


Diagrama 4.15. Proceso de Explotación de la información.

La subrutina de generación de los reportes, la observamos en el diagrama 4.16, en donde, como resultado de la Explotación de la información, ésta puede quedar finalmente en forma de tabla, gráfica o bien en reporte.

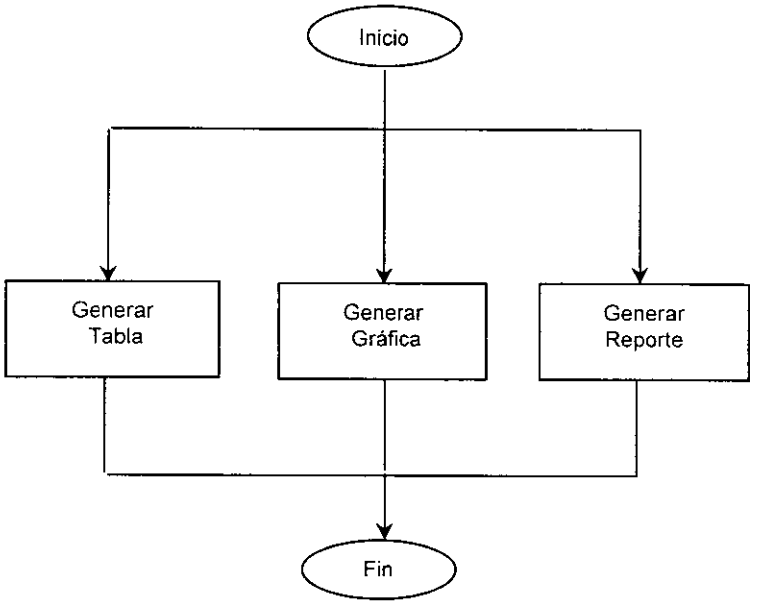


Diagrama 4.16. Subrutina de generación de reportes

Programación en Shell (Cshell y Shell Script)

Parámetros

Para la ejecución de cada uno de los desarrollos de COBOL74, así como los *JOBS* que los utilizan, y los desarrollos generados en *CShell*, muchas veces fue necesario pasar parámetros de ejecución, que ayudarían a la correcta obtención de resultados a partir de los mismos, estos son ejemplos de algunos de esos parámetros dentro de la ejecución de rutinas:

- Para *COBOL* 74 y los *JOBS*

Los archivos deberán generarse bajo dos parámetros que serán dados por el usuario, siendo el primero las 3 letras que representen a la zona y el segundo la fecha con la que el programa discriminará la información requerida por el usuario únicamente, en el formato AAMMDD.

?SO MESS
para desplegar mensajes del programa
ST CDI/CHE/F/CARDIA/020(**"MTY","970415"**)
para ejecutar el programa
Accept: Se va a procesar mensual (S/N) ?
pregunta el programa por el parámetro
?#### AX S
para procesar mensual
?#### AX N
para procesar diario

El *JOB* por su parte, pasará al programa objeto, resultado de la compilación del código de *COBOL*, el parámetro necesario para su ejecución de la siguiente forma:

RUN CDI/CHE/O/CARDIA/020(**FECHA**)
FECHA es resultado de quitarle los dígitos del año al parámetro pasado al JOB por medio de la instrucción DROP, como se ve en el código

- Programación *Cshell*

En Unix

$ isql -Uusuario -Ppassword_usuario **para entrar al ambiente**
1> SP_AHOCARDIA01
2> GO **para lanzar el programa**

Programación en *Shell Script*

Como se mencionó anteriormente, se utilizará programación en *Shell* durante el desarrollo del sistema de Centro de Información, a continuación se explicará que es un *Shell*, que es un *Shell Script* y cuáles son sus características.

La programación en *Shell* se define como un programa creado para sistemas operativos manejado por comandos, especialmente para UNIX. Un archivo generado en *Shell* es un conjunto de instrucciones que se ejecuta línea por línea tal como lo hace un archivo *Batch* (archivo por lotes).

El *Shell Script* es un guión de *Shell*, un archivo de órdenes UNIX el cual contiene una lista de comandos que son ejecutadas (ver tabla 4.7) por el *Shell* mediante el comando *chmod*. Estos archivos no se compilan debido a que cada línea es interpretada en su tiempo de ejecución. A continuación se mostrarán las características de la programación en *Shell Script*.

Modo de Ejecución

a)	Un programa *Shell Script* puede ser ejecutado usando un *subShell*, el cual toma sus instrucciones del archvio *Shell Script*.
b)	El *subShell* ejecuta el *Shell scritp* como una lista de archivos .
c)	Cualquier texto de comando que sea escrito después del signo "#" dentro de un *Shell scritp* será ignorado por el *Shell*.
d)	Un *Shell Script* será abortado cuando se encuentre un comando de salida (*exit*) o la instrucción de final de archvo (EOF).

Tabla 4.7. Modo de Ejecución.

Un archivo en *Shell Script* puede ser ejecutado de dos formas, la ejecución sin permisos del usuario y la ejecución con permisos del usario.

En la primera se utiliza el comando: ***$Sh nombre_archivo***. Con esta instrucción se ejecuta el *Shell Script* aún cuando no se tengan permisos de ejecución.

Para ejecutar un *Shell Script* con permisos del usuario es necesario que en la primera línea se escriba: ***#!/bin/sh*** , posteriormente, se debe cambiar los permisos al modo de ejecución con la línea: ***$ chmod +x nombre_archivo*** y finalmente se ejecuta con la instrucción: ***$ nombre_archivo.***

Para probar las entradas del *Shell Script*, se debe emplear el siguiente comando :

$sh -x nombre_archivo.

Donde la opción ***-x*** despliega cada línea del comando que se va a ejecutar.

En el siguiente ejemplo de *Shell Script* se muestra el código de un proceso de carga de datos.

```
# Valida la Bitácora

#!/bin/sh
lsCdi= 'IMD'
lcMov= 'D'

#Construte las variables de Ambiente
lsLin= ' '              # lee cada línea de archivo de la zona
errOr= 0
lsPath= /home/clientes/archivos    # Path principal

#Valida zona por zona el archivo de Bitácora
cdiInvBit sh $lsCdi $lsMMDD $lcMov cdi$lsCdi'PZ.txt' cdi$isCdi'Zona.txt'

#Construye el archivo de Contratos y Documentos
lsBaj=$lsPath/$lsCdi"BAJ"$lsLin$lsMMDD".TXT"
lsMov=$lsPath/$lsCdi"MOV"$lsLin$lsMMDD".TXT"
lsDoc=$lsPath/$lsCdi"DOC"$lsLin$lsMMDD".TXT"
lsCte=$lsPath/$lsCdi"CTE"$lsLin$lsMMDD".TXT"
lsDir=$lsPath/$lsCdi"DIR"$lsLin$lsMMDD".TXT"
```

Pantallas del CDI

En este apartado se mostrarán las pantallas empleadas en el desarrollo del sistema, en el caso de los procesos de Extracción, Transferencia y Carga, sólo se emplearán pantallas de mensaje para los operadores del servidor (explicados en el apartado de mensajes), los cuales le informarán el resultado del procedimiento, ya que no existe interacción por parte de los operadores para realizar dichos procesos, pues como se explicó anteriormente, los programas *Shell* se encargan de ello.

En el caso del proceso de Explotación, se realizará una *interface* gráfica mediante *Visual Basic* que le permita al usuario realizar consultas, reportes y modificaciones de información de registros (en base de datos Sybase). Este programa tiene como objetivo independizar al usuario final (ejecutivos) de los operadores del servidor para realizar las operaciones descritas anteriormente, con esto, el proceso de toma de decisiones en base a resultados se llevará a cabo en forma práctica e inmediata.

Las siguientes figuras muestran la propuesta de pantallas del proceso de Explotación que utilizarán los funcionarios finales.

1.- Pantalla de bienvenida.

Figura 4.7. Pantalla de bienvenida.

2.- Pantalla principal del CDI.

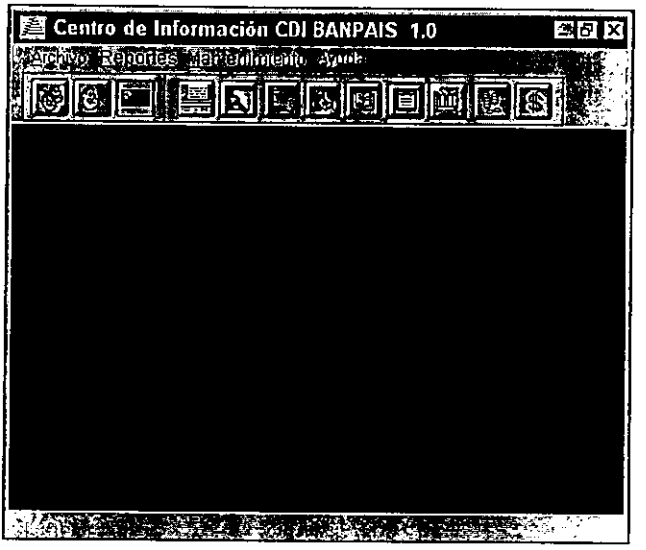


Figura 4.8. Pantalla principal del CDI.

La pantalla Principal cuenta con una barra de herramientas (figura 4.9) que le permiten al usuario ejecutar diversas funciones. A continuación se explicarán cada uno de los comandos de la barra de herramientas dichas funciones .

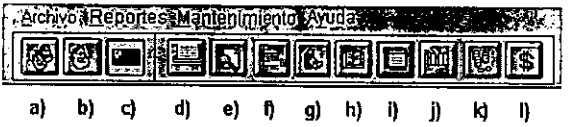


Figura 4.9. Barra del menú principal.

La Barra de herramientas del CDI cuenta con las siguientes opciones:

a) *Consulta por clientes.* Permite buscar información específica de un cliente (fig. 4.10) introduciendo cada uno de los siguientes parámetros:

- Número de Cliente.
- Nombre (s) .
- Apellidos (s).
- R.F.C.

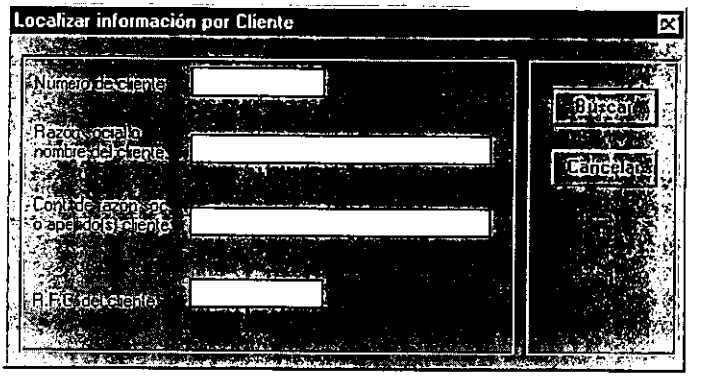


Figura 4.10. Consulta de clientes.

b) *Consulta por grupo (asociación de empresas)* . Puede especificarse el número de identificación del grupo o el nombre del grupo para realizar la consulta tal como se muestra en la figura 4.11.

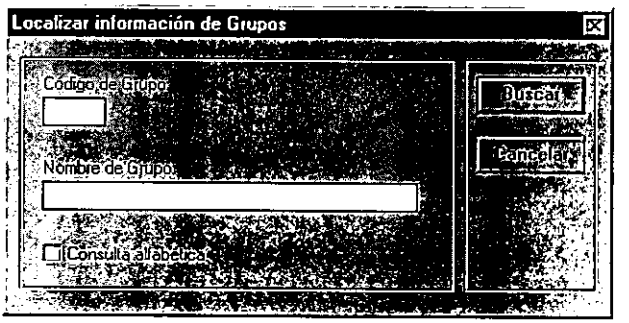


Figura 4.11. Consulta por grupos.

c) *Consulta por número de cuenta.* Permite la búsqueda por un número de cuenta (figura 4.12), se especifica el producto (código de servicio) y la región.

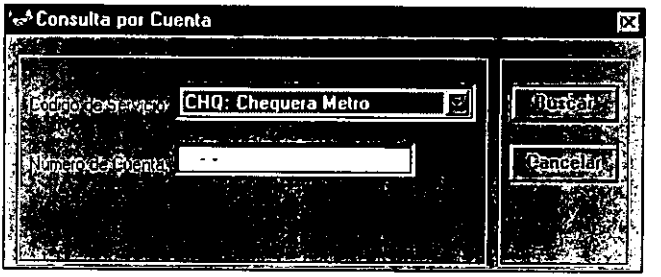


Figura 4.12. Consulta por número de cuenta.

d) *Imprimir pantalla* . Manda una impresión de la pantalla activa.

e) *Resultado de la consulta.* La figura 4.13 muestra el resultado de una consulta, en esta figura se muestran como ejemplo las cuentas asociadas al cliente localizado en la búsqueda.

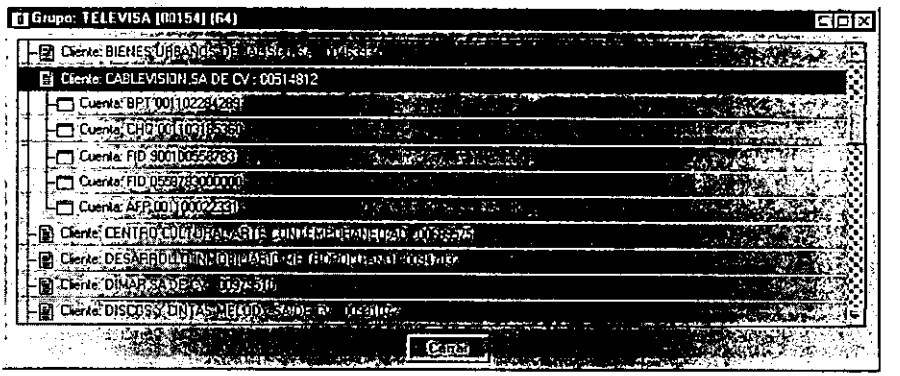


Figura 4.13. Resultado de la consulta.

f) *Despliegue de plaza y sucursal.* Muestra la plaza y la sucursal a la cual pertenece la(s) cuenta(s) en el resultado de la búsqueda (ver figura 4.14).

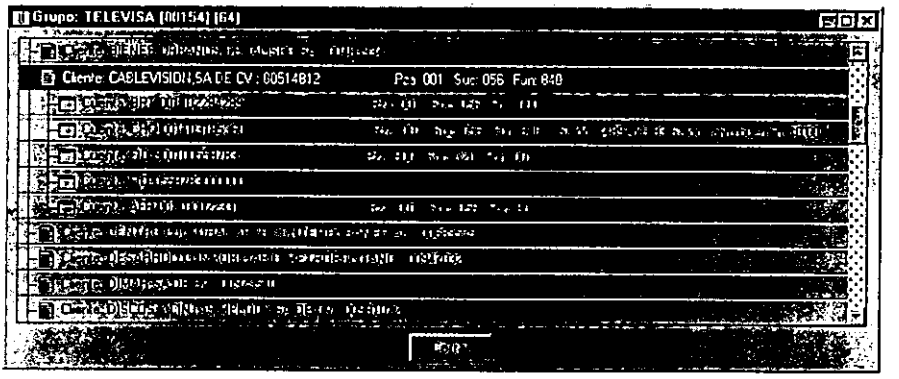


Figura 4.14. Despliegue de plaza y sucursal.

g) *Catálogo de Servicios.* La figura 4.15 muestra un catálogo de los servicios que están disponibles en el Centro de Información, este catálogo es de gran utilidad si no se está familiarizado con la nomenclatura de los servicios de BANPAIS.

Figura 4.15. Catálogo de servicios.

h) Información general del cliente. Se refiere a una pantalla (figura 4.16) con los datos generales del cliente que se esta consultando.

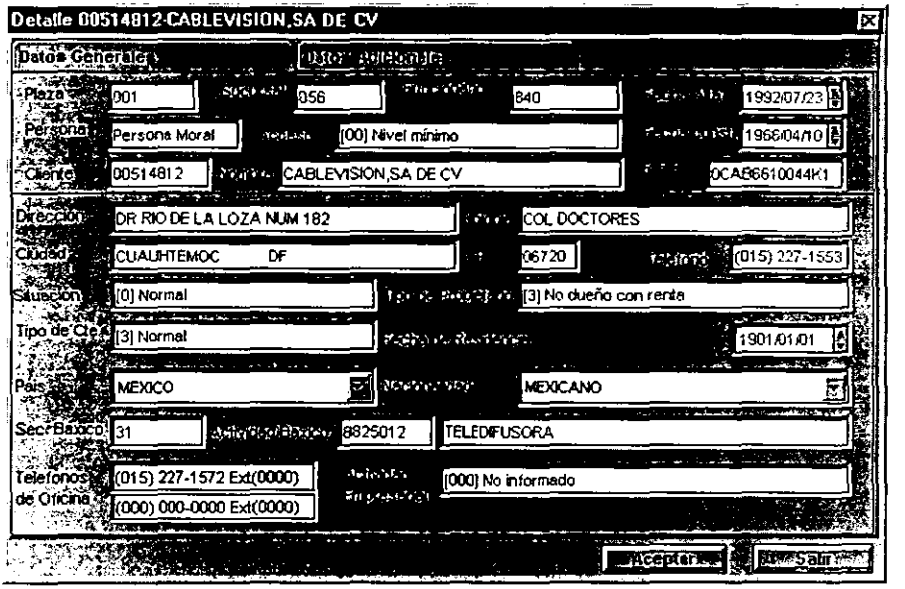


Figura 4.16. Información general del cliente .

La información anterior permite identificar el tipo de actividad que realiza el cliente, así como su situación actual con el banco es decir, si es cliente normal (sin problemas de pago), de solvencia suficiente (cuenta con lo necesario para pagar) moratorio, (no paga a tiempo) etc.

i) Detalles administrativos del grupo. Para mostrar los detalles administrativos de un grupo, se despliega la siguiente pantalla a la que hace referencia la figura 4.17.

Esta información básicamente le es útil al funcionario encargado de elaborar los reportes de los principales accionistas del grupo en cuestión, así como su porcentaje de participación en el mismo. Este reporte se entrega semestralmente a la Comisión Nacional Bancaria

Figura 4.17. Detalles de grupo.

j) Consulta por fechas.- Si se desea realizar una consulta por rangos de fechas, aparecerá la pantalla que se muestra en la figura 4.18, con la solicitud de fecha inicial y fecha final, el rango permitido para realizar esta consulta no debe exceder de 60 días.

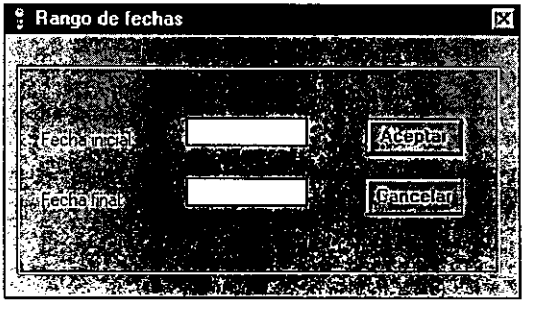


Figura 4.18. Rango de fechas.

k) *Consulta de saldos de cartera.-* Para realizar esta operación es necesario especificar el tipo de consulta que se desea efectuar, es decir, por centro de asignación, por grupo o por funcionario (Figura 4.19).

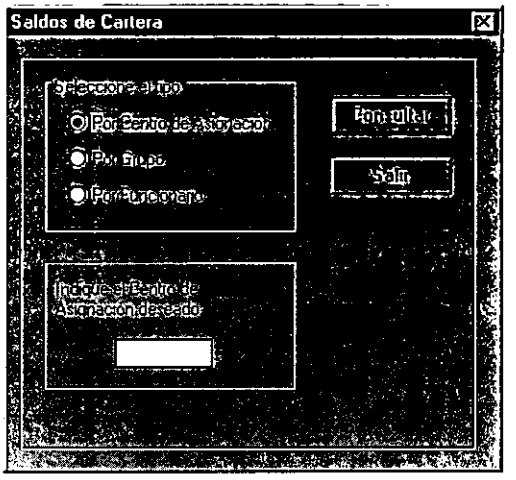


Figura 4.19. Saldos de cartera .

l) *Importes de cartera.* Para mostrar la información sobre importes de cartera se muestra en una sola pantalla que cuenta con las siguientes opciones:

- Importes Generales.- Muestra los importes de la cartera total del cliente.
- Importes de Cartera Vencida.- Muestra los importes vencidos de la cartera (ej. figura 4.20).
- Importes de Cartera Vigente.- Muestra el estado actual de la cartera.
- Importes de Cartera + Intereses.- Muestra los intereses generados de la cartera.
- Importes de Riesgo.- Muestra el importe de riesgo de la cartera del cliente (ej. figura 4.21).

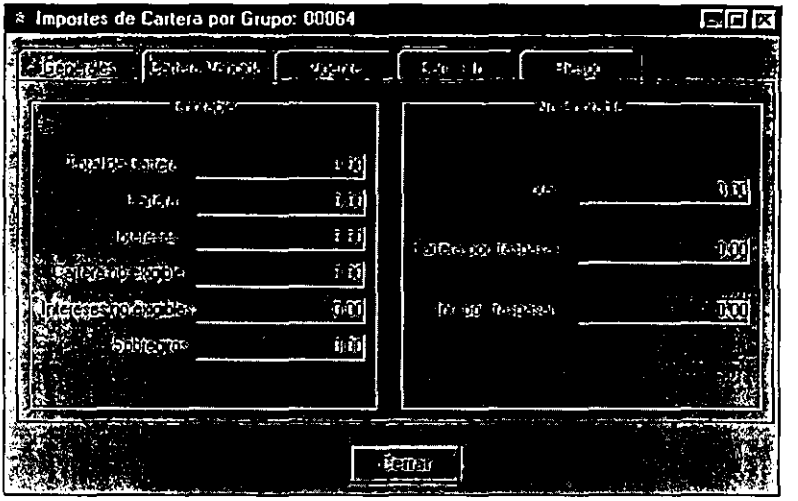


Figura 4.20. Ejemplo 1 de importes de cartera.

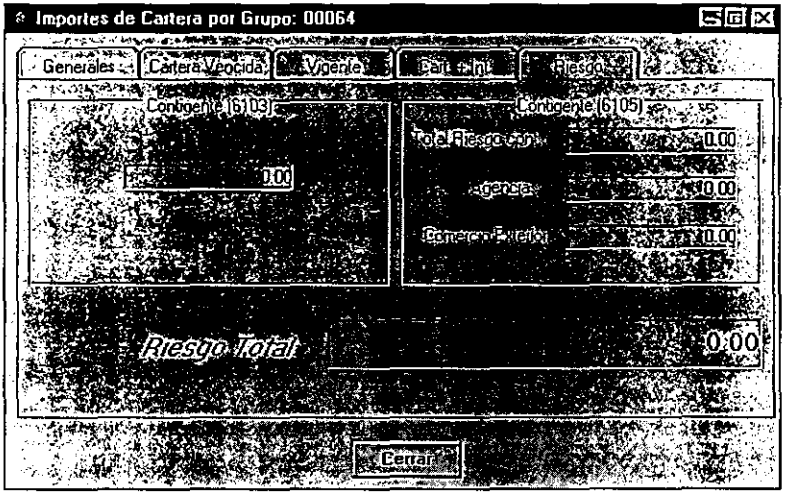


Figura 4.21. Ejemplo 2 de importes de cartera.

Código de los Desarrollos

En este apartado se muestra un ejemplo de código utilizado en cada uno de los procesos más importantes de nuestro desarrollo, los cuales iremos dividiendo por etapas.

A continuación se presenta el código desarrollado para realizar la extracción de información.

```
  0 IDENTIFICATION DIVISION.                                                    000000
100 PROGRAM-ID       CDI-FON-F-CARSAL-010.                                    000100
200 DATE-WRITTEN.    JUL 01 DE 1997                                           000200
250 *AUTHOR.         PCA                                                      000250
260 *                                                                         000260
300 DATE-COMPILED.                                                            000300
400 ************************************************************************* 000400
500 ** ESTE PROGRAMA GENERA EL ARCHIVO PLANO DE FONDOS                        000500
600 ** QUE SERA CARGADO A LA BASE DE DATOS QUE UTILIZA EL                     000600
700 ** SISTEMA DE RIESGOS A PARTIR DE CUENTAS Y FONDOS.                       000700
800 ** SOLO SE ESCRIBIRAN LOS TIPO DE REGISTRO                               000800
900 ** QUE SEAN CUENTAS ACTIVAS=4 Y CUENTA-SEGMENTO=3,8 O 9                  000900
100 ************************************************************************* 000100
1100 ENVIRONMENT DIVISION.                                                    00:100
1200 CONFIGURATION SECTION.                                                   00:200
1300 SOURCE-COMPUTER.     A9-F.                                               00:300
1400 OBJECT-COMPUTER.     A9-F.                                               001400
1500 INPUT-OUTPUT SECTION.                                                    001500
1600 FILE-CONTROL.                                                            001600
1800     SELECT F-OUT-SALD ASSIGN TO DISK.                                    001800
1900 DATA DIVISION.                                                           001900
2000 FILE SECTION.                                                            002000
2100 FD  F-OUT-SALD.                                                          002100
2200 01  REG-OUT-SALD.                                                        002200
2300     02  OUT-RSA-CUENTA.                                                 002300
2400         03 OUT-RSA-OFICINA          PIC  X(3).                         002400
2500         03 OUT-RSA-CONSECUTIVO      PIC  X(7).                         002500
2600     02     OUT-RSA-1                PIC  X.                            002600
2900     02     OUT-RSA-CVESALDO         PIC  X.                            002900
3000     02     OUT-RSA-2                PIC  X.                            003000
3100     02     OUT-RSA-SIGLO            PIC  XX.                           003100
3200     02     OUT-RSA-FECHA-PA         PIC  X(6).                         003200
3500     02     OUT-RSA-3                PIC  X.                            003500
3600     02     OUT-RSA-SALDO            PIC  9(13).99.                     003600
3700                                                                          003700
3800                                                                          003800
7700 DATA-BASE SECTION.                                                       007700
```

```
7800 DB DBCHE.                                                          007800
7900 01  CUENTAS                                                       007900
8000 01  FONDOS USING FON-X-CTA-PTS.                                   008000
8100 WORKING-STORAGE SECTION.                                           008100
8450 77  C2600-SOUT                          PIC X VA "^".              008450
8500 77  S000-EOF                           PIC 9 VA 0.                 008500
8700 01 D000-FECHA-PARAM RECEIVED BY REFERENCE.                        008700
8800    05 D2600-FECHA-PARAMETRO            PIC 999999.                 008800
9400 *                                                                  009400
9500 PROCEDURE DIVISION USING D000-FECHA-PARAM.                         009500
9700 1000-INICIO.                                                       009700
10000    OPEN     INQUIRY DBCHE.                                        010000
10100    OPEN     OUTPUT  F-OUT-SALD                                    010100
10200    PERFORM 2000-PRIMERA-LECTURA                                   010200
10300    PERFORM 2500-GENERA-OTRO ULTIL S000-EOF =1                     010300
10400    CLOSE    F-OUT-SALD                                            010400
10500    CLOSE    DBCHE.                                                010500
10600    STOP RUN.                                                     010600
10700 100-9999-FIN.                                                     010700
10800    EXIT.                                                         010800
10900 ****                                                              010900
11100  2000-PRIMERA-LECTURA.                                            011100
11200     FIND FIRST CUENTAS                                           011200
11300          ON  EXCEPTION                                           011300
11400          IF DMSTATUS (NOTFOUND)                                  011400
11500          MOVE 1 TO S000-EOF.                                     011500
19100 2500-GENERA-OTRO.                                                 019100
19101      IF (CTA-INACTIVA NOT = 4 AND (CTA-SEGMENTO = 3               019101
19104         OR CTA-SEGMENTO = 8 OR CTA-SEGMENTO = 9))                 019104
19107         PERFORM 2600-BUSCA-FONDOS.                               019107
19128       FIND NEXT CUENTAS                                          019128
19131            ON EXCEPTION                                          019131
19134            IF DMSTATUS ( NOTFOUND )                              019134
19137               MOVE 1 TO S000-EOF.                                019137
20700 2600-BUSCA-FONDOS.                                                020700
20800      FIND FON-X-CTA-PTS  WHERE FON-OFICINA = CTA-OFICINA          020800
20900           AND FON-CONSECUTIVO = CTA-CONSECUTIVO                   020900
21000           ON EXCEPTION                                           021000
21010           IF  DMSTATUS ( NOTFOUND)                               021010
21020              MOVE 1 TO S000-EOF.                                 021020
21700      MOVE C2600-SOUT          TO      OUT-RSA-1                  021700
21800      MOVE C2600-SOUT          TO      OUT-RSA-2                  021800
21900      MOVE C2600-SOUT          TO      OUT-RSA-3                  021900
22200      MOVE FON-OFICINA         TO      OUT-RSA-OFICINA            022200
22300      MOVE FON-CONSECUTIVO     TO      OUT-RSA-CONSECUTIVO        022300
22400      MOVE "19"                TO      OUT-RSA-SIGLO              022400
22500      MOVE D2600-FECHA-PARAMETRO TO    OUT-RSA-FECHA-PA           022500
22800      MOVE "5"                 TO      OUT-RSA-CVESALDO           022800
22900      MOVE FON-SALDO           TO      OUT-RSA-SALDO              022900
23000      WRITE REG-OUT-SALD.                                         023000
```

Una vez llevada a cabo la extracción de los datos, se pasa a la etapa de transferencia, de la cual como ejemplo presentamos el siguiente listado.

CDI/FON/J/CARSAL/010 (07/30/97) 7:08PM WEDNESDAY,JULY 30,1997 Pacific Standard Time

```
    0 BEGIN JOB FON/CARSAL(STRING ZONA,STRING FECHA);                          00000000
  100  TASK T;                                                                 00000100
  200  STRING LSDBCH;                                                          00000200
  300  STRING LSNOMFECH;                                                       00000300
  400  STRING LSARCHIVO;                                                       00000400
  500  LSDBCH:="DBCHE"&ZONA;                                                   00000500
  550  LSNOMFECH := FECHA;                                                     00000550
  600  LSNOMFECH := DROP(LSNOMFECH,2);                                         00000600
 7000 %*************************************************************************00007000
 8000 %* JOB QUE CORRE EL PROGRAMA QUE GENERA ARCHIVO                *         00008000
 9000 %* PLANO QUE OBTIENE LOS SALDOS                                *         00009000
 1000 %* CON LOS DATOS DE FONDOS                                     *         00001000
 1100 %* ELABORADO POR PCA EL 30 DE JUN DE 1997                      *         00001100
 1200                                                                          00001200
 1300 %*************************************************************************00001300
 1400                                                                          00001400
 1500 RUN  CDI/FON/O/CARSAL/010(FECHA)  [T];                                   00001500
 1600       DATABASE  DBCHE(TITLE=#LSDBCH);                                    00001600
 1700       FILE F-OUT-SALD(KIND=DISK,                                         00001700
 1800                  TITLE = CDI/#ZONA/FON/#LSNOMFECH/SAL,                   00001800
 2000                  FILETYPE=0,NEWFILE =TRUE,PROTECTION=SAVE);              00002000
 2100                                                                          00002100
 2200  IF T.ISNT.COMPLETEDOK THEN ABORT " ABORTO GENERACION DE ARCHIVO"        00002200
 2300                                                                          00002300
 2400  ELSE BEGIN                                                              00002400
 2500      DISPLAY "TERMINO BIEN LA GENERACION DE ARCHIVO    ";               00002500
 2600      LSARCHIVO :="/home/clientes/archivos/FONSAL"&ZONA&LSNOMFECH;        00002600
 2700      LSARCHIVO := LSARCHIVO&".TXT";                                      00002700
 2800      COPY CDI/#ZONA/FON/#LSNOMFECH/SAL AS #LSARCHIVO                     00002800
 2900      FROM SYS2(PACK) TO DISK(IPADDRESS="10.6.200.38",                    00002900
 3000           USERCODE='sybase'/'sybase');                                   00003000
 3100      END                                                                 00003100
 3200 END JOB.                                                                00003200
```

Una vez transladada la información, el siguiente paso fue subirla al modelo, para esto se ejecutaban procesos de carga, del cual se muestra el siguiente listado como ejemplo de uno de ellos.

```
# cdiBit.sh                                          :Nombre
# PCA                                                :Autor
# -19971001-                                         :Fecha
# Actualización de la Bitácora                       :Objetivo
#       Sistema.3.pos.          -                    :Entrada
#       Fecha(MMDD). 4pos.      -
#       Movimiento(D/E). 1pos.  -
#       Archivos a contar y a sumar -
#       Estatus                 -                    :Salida
# El Movimiento puede ser Diario='D' o
# eventual='E'
# El estatus sera valido en el Shell que lo mando
# llamar
# Este archivo debe ser generado con un 'ls ....'.
# ejem: ls CHEDOCMTY1001.txt>ISCHEDOCMTY1010.TXT
# ====================================================

#!/bin/sh
1.Parametros:
psCdi=$1                    #Sistema
PsMMDD=$2                   #Fecha solo el mes y el dia MMDD
pcMov=$3                    #Movimiento
psFile=$4                   #Archivo Bitácora

#2. Declaración de Variables
lsBtA=wclsBtA.txt           #Archivo de bitácora que se sube a la BD
lsBit=' '                   #nombre del archivo
liErrOr=0                   #Tipo de error
lsPath=/home/clientes/archivo  #Path principal
lsLin=' '                   #Registros del archivo de entrada(wc)
lnCount=0                   #Total por Archivo
lnTot=0                     #Total de registros
lsE=tmpTableSal             #Nombre de la tabla tmp CHE,FON,AHO
lsY=TmpDocumento            #Nombre de la tabla tmp IMD
ldFecha=`date+%D`
ldTime=`date+%T`
echo " \n\n $ldFecha$ldTime Actualizando Bitacora "|tee-a fError.out
#_____________________________________________________
clear
echo "                      $ldFecha"
echo
echo
echo "          BANPAIS"
echo "          ____________________________________"
```

```
echo
echo "          CENTRO DE INFORMACION "
echo "          (CARGA DE DATOS)"
echo "          ____________________________________________"
echo
echo "      Sistema:$pcCdi          Fecha:$psMMDD"
echo
echo
echo
echo
echo " Procesando Cifras de Control..."
echo
echo
#=======================================================
# 3. Construyo el archivo de Bitacora
# lsBit=$lsPath/'BIT'$psCdi$psZona$psMMDD'.TXT'

# if test -s $lsBit
# then
#   #Realizo un APPEND al archivo de Bitácora
#   bcp corpo..Bitacoara in $lsBit -m 1 -c -Ucorpo-Pcorpo1 -t ^ -r \\n --efBi
tError.txt
#   errOr=$?
#   mv $lsBit $lsBit'.OK'
# else
#    #No hay datos en Bajas
#    errOr=2
# fi
# if [$errOr !='0']  #Si no existe el archivo no continuo
# then
#    exit (5)
# fi
#_____Sube los datos a la bitácora _____________________
# 4.Lee linea por linea el archivo de entrada WC
 for lsLin in `cat $psFile`
 do
    #4.1 Cargar los archivos que generan Importe
    lnCount=`wc -l $lsPath/$lsLin |awk '{ print $1}'`
    echo "$lsLin^$lnCount"| tee -a $lsBtA
done

# 4.1 Cargo el archivo que genere, a la base de datos
bcp corpo..BtDArchivo in $lsBtA -m 1 --c -Ucorpo -Pcorpo1 -t^ -r \\n -efBitE
rrOr.txt

# 4.2 Compila el SP de la Bitácora
isql -Ucorpo -Pcorpo < SP_BITUP01.SQL

# 4.3 Construyo la linea que se envia a Sybase
echo "SP_BITUP01""$psMov""\ngo">spBitUpDown.txt
```

```
# 4.4 Ejecuta el Store Procedure
isql -Ucorpo -Pcorpo1 < spBitUpDown.txt | tee -a fError.out

# 4.5 Ejecuta un scrip para la obtención de los errores
isql -Ucorpo -Pcorpo1 < spBitErrOr.txt>fErrOrBit.txt

#4.6 Valido que exista el errOr
if test -s fErrOrBit.txt
Then
   # 4.6.1 elimina las 3 primeras lineas
   lnCount=`wc -l fErrOrBit.txt |awk '{ print $1}'`
   let lnTot=lnCount-4
   tail +3 fErrOrBit.txt>fErrOrBit1.txt
   head -$lnTot fErrOrBit1.txt>fErrOrBit.txt

   For lErrOr in 'cat fErrOrBit.txt'
  do
    stkFileError.xh 35 lError
  done
  exit(1)
fi
#=======================================================
clear
```

Ejemplo de un *Store Procedure*, el cual valida la información de la tabla temporal y la carga en la tabla definitiva.

```
/* SP_BITUPDOWN01.SQL                          :Nombre
PCA                                            :Autor
 -19971010-                                    :Fecha
Actualizar la Bitácora Diario                  :Objetivo
Archivos de Ahorro, Cheques, Fondos e Inversión :Entrada
1= Hubo diferencias en los registros -         :Salida
2= Hubo diferencias en los totales  -          : */

 /* 1. Borra si existe el SP */
IF EXISTS (select name from sysobjects where name ='SP_BITUPDOWN01')
   DROP PROC SP_BITUPDOWN01
go
 /* 2. Crea el SP */
CREATE PROC SP_BITUPDOWN01 @pcMov CHAR(1)
AS
DECLARE
   @lsSys          Varchar(3),
   @lsProd         Varchar(3),
   @lsStrucutra    Varchar(30),
   @lsMMDD         Varchar(4),
   @lsSum          Numeric(10),
```

```
  @lsTot          Numeric(3),
  @lsTotOk        Numeric(3),
  @lsTotWg        Numeric(3)

/* 3. Crea un cursor para bajar toda la información de la tabla de archivo */
BEGIN
  BEGIN TRAN
/*_____________________________________________________*/
   /* 3.1.1 Crea un curso de la taba tmpbtArchivo
        Sistema, producto,fecha,suMa de registros*/
   DECLARE curBit CURSOR
   FOR SELECT substring(btCSArc,1,3),
       Substring (btCSArc,4,3),
       Substring (btCSArc,datalength (btCSArc) -7,4),
       sum (btDNCount)
    FROM BtDArchivo
    GROUP BY substring (btCSArc,1,3),
        Substring(btCSArc,4,3),
       Substring (btCSArc,datalength (btCSArc) -7,4),

/* 3.1.2 Abre el Cursor */
OPEN curBit

/* 3.1.3 Empieza el Ciclo */
FETCH curBit INTO @lsSys
      @lsProd,
      @lsMMDD,
      @lnSum

/ 3.1.4 Inicializa las variables */
SELECT @liTot=0
SELECT @liTotOk=0
SELECT @liTotWg=0

while(@@sqlstatus=0)
BEGIN
   /*3.1.4.1 Incrementa el Total de Bajas */
   SELECT @liTot=@liTot+1

   /*3.1.4.2 Actualiza la variable del producto */
   EXECute SP_BITPROD01 @lsSys, @lsProd,@pcMov, @lsStructura OUT

   /*3.1.4.2 Existe el contrato */
   UPDATE Bitacora

   SET no_registros_carga=@lnSum,
     Importe_carga=importe_origen,
     Diferencia_registro=@lnSum-no_registros_origen,
     Diferencia_importe=importe_origen-importe_carga
   WHERE(SUBSTRING(CONVERT(varchar(6),fecha,12),3,4)=@lsMMDD
```

```
        and sistema=@lsSys
        and estructura=@lsStructura)

    /3.1.4.3 Empieza el Ciclo */
    FETCH curBit INTO @lsSys,
            @lsProd,
            @lsMMDD,
            @lnSum
END
/3.1.5 Desaloja toda la memoria asignada al cursor */
DEALLOCATE CURSOR curBit

/*3.2 Salva todo */
COMMIT TRAN

/* 3.3 Muestra en pantalla y lo envia a un archivo el total de Contratos
el no. de contratos nuevos y los ya existentes */
SELECT @liTot Total_de_Contratos,
       @liTotOk Contratos_Nuevos,
       @liTotWg Contratos_Existentes
END
go
```

Por último, se llevaba a cabo la explotación de la información. Se muestra un ejemplo de código utilizado para este fin.

En base al esquema utilizado, se define el desarrollo por módulos, en este caso:

1. Módulo Inicio

```
'Realizado por : PCA
'Modulo : Procesos para la conversión de numeros a letras
'Objetivo : Manejo de funciones para la conversión de numeros
'          a letras

Option Explicit

Global Titulo$                  'Almacena el titulo del reporte
Global NombreCia$               'Almacena el nombre de la compañia
Global GllaveArch$(10)
Global GSubstMembercode As String
Global GClienteEnAREQUIV, GClienteEnARGROUP As String
'Proceso principal
Public Sub main()
  'Declaración de variables
  Dim i As Integer


  'Datos Generales de la Aplicacion
  PCAPideDatosGenerales
```

```
  'Define archivos a usar
  PCADefinicionArchivos

  'Abre Base de Datos de la Compañia (tablas el sistema)
  If AbrirBDCiaPlt% <> 0 Then
    MsgBox "Existe un error al abrir Base de datos de Compañia de El sistema", vbOKOnly + vbExclamation, "Grupos de
Clientes"
    Exit Sub
  End If

  'Abre Base de Datos del Usuario
  If AbrirBDUsuario% <> 0 Then
    MsgBox "Existe un error al abrir Base de datos de Usuario", vbOKOnly + vbExclamation, "Grupos de Clientes"
    Exit Sub
  End If

  'Abrir Archivos
  If AbrirArchivos% <> 0 Then
    MsgBox "Existe un error al abrir tablas de Btrieve", vbOKOnly + vbExclamation, "Grupos de Clientes"
    Exit Sub
  End If

  Load frmBusArchivo
  frmBusArchivo.Show

End Sub

'Proceso que genera los procesos necesarios para terminar la aplicación
Public Sub TerminaAplicacion()

  'Procesos que cierra archivos de btrieve
  CerrarArchivos%
  'Proceso que cierra base de datos de la compañía
  CerrarBDCia
  'Proceso que cierra base de datos del usuario
  CerrarBDUsuario

End Sub
```

2. Módulo Archivos y Variables

```
'Realizado por : PCA
'Modulo : Procesos para la conversión de numeros a letras
'Objetivo : Manejo de funciones para la conversión de numeros
'          a letras

Option Explicit

'Archivos a utllizar
Global Const AREQUIV1 = 1
Global Const ARGROUP1 = 2
Global Const AREQUIV2 = 3
Global Const ARGROUP2 = 4
Global Const ARHDR = 5
Global Const CLIENTES = 6

'Numero maximo de archivos
Global Const GlNumMaxArchivos = 6

'Variables utilizadas para el manejo de tablas Btrieve
Global GlNombreArchLogico$(GlNumMaxArchivos)        'Almacena el nombre del archivo de btrieve
Global GlNombreIndiceArch$(GlNumMaxArchivos)        'Almacena el nombre de la llave del archivo
Global GlArchCia$(GlNumMaxArchivos)                 'Indica si es un archivo de la compañía
```

```
'Nombre Logico de los registros
Global GlRecordSetArch(GlNumMaxArchivos) As Recordset  'Genera el recordset de los archivos btrieve
'Proceso para el manejo de archivos en operaciones tales
'como apertura y cierre de los mismos
Public Sub PCADefinicionArchivos()
  'Invocación al diccionario de datos
  FileDDF$ = "file.ddf"

'Archivo que almacena las clases equivalentes por account code
  GlNombreArchLogico$(AREQUIV1) = "Arequiv"
  GlNombreIndiceArch$(AREQUIV1) = "Membercode"
  GlArchCia$(AREQUIV1) = "1"

  'Archivo que almacena los grupos de clientes por cliente agrupador
  GlNombreArchLogico$(ARGROUP1) = "ARGROUP"
  GlNombreIndiceArch$(ARGROUP1) = "cliente_agrupador"
  GlArchCia$(ARGROUP1) = "1"

  'Archivo que almacena las clases equivalentes por cliente
  GlNombreArchLogico$(AREQUIV2) = "Arequiv"
  GlNombreIndiceArch$(AREQUIV2) = "Customer_key"
  GlArchCia$(AREQUIV2) = "1"

  'Archivo que almacena los grupos de clientes por cliente
  GlNombreArchLogico$(ARGROUP2) = "ARGROUP"
  GlNombreIndiceArch$(ARGROUP2) = "cliente"
  GlArchCia$(ARGROUP2) = "1"

  GlNombreArchLogico$(Clientes) = "Clientes"
  GlNombreIndiceArch$( Clientes) = ""
  GlArchCia$(CLIENTES) = "1"

End Sub

'Proceso que solicita información general sobre el usuario
Public Sub PCAPideDatosGenerales()
  'Declaración de variables
  Dim spls As Object
  Dim retcode As Integer

  'Inicializa el Login server
  Screen.MousePointer = 11
  On Error Resume Next
  Set spls = CreateObject("pls.login")
  If Err <> 0 Then
    MsgBox "Existe un error al momento de accesar lel CDI."
    On Error GoTo 0
    Exit Sub
  End If
  'Obtiene información sobre el usuario loggeado en ese momento a EL CDI
  retcode = spls.LoggedIn()
On Error GoTo 0
  GlNombreUsuario = spls.LoginName()
  GlUsuario = spls.UserId()
  GlGrupoUsuario = spls.GroupId()
  GlTipoUsuario = spls.UserType()
  GlTipoCliente = spls.Options()
  GlCia = spls.CoKey()
  GlParentId = spls.PID()
  Set spls = Nothing
On Error GoTo 0
  Screen.MousePointer = 0

End Sub
```

3. Módulo Archivos y Variables

```
'Realizado por : PCA
'Modulo : Procesos para la conversión de numeros a letras
' 'Objetivo : Manejo de funciones para el manejo de operaciones con btrieve

Option Explicit

'Variables generales del usuario
Global GlNombreUsuario$
Global GlUsuario$
Global GlGrupoUsuario$
Global GlTipoUsuario%
Global GlTipoCliente%
Global GlCia$
Global GlParentId$
'Global GlLlaveArch$(10)

'Operaciones soportadas
Global Const CGlBtrInsert = 2
Global Const CGlBtrUpdate = 3
Global Const CGlBtrDelete = 4
Global Const CGlBtrGetEqual = 5
Global Const CGlBtrGetNext = 6
Global Const CGlBtrGetPrevious = 7
Global Const CGlBtrGetGrThan = 8
Global Const CGlBtrGetGrEqual = 9
Global Const CGlBtrGetLsThan = 10
Global Const CGlBtrGetLsEqual = 11
Global Const CGlBtrGetFirst = 12
Global Const CGlBtrGetLast = 13
Global Const CGlBtrUnlock = 27
Global Const CGlBtrLock = 100

'Tipo de Base de Datos
Global Const CTypeBD = "SYBASE;"


' Definicion de la Base de Datos
Global GlMPath$
Global GlMPathDDFCia$
Global GlMPathDDFUsuario$

'Nombre Logico de la Base de Datos
Global BDCia As Database
Global BDUsuario As Database
'Encuentra la ruta de acceso a la base de datos de la compañía
'Encuentra la ruta de acceso a la base de datos del directorio del usuario
Private Sub EncuentraPathUsuario()
   'Path que busca la dirección donde se encuentra El sistema
   'y el directorio del usuario
   EncuentraPathGeneral
   'Pasa dirección exacta del directorio del usuario a la función
    GlMPathDDFUsuario$ = GlMPath$ + "\" + GlUsuario + "\"
End Sub
'Busca dentro del directorio de El sistema for Windows a nivel cliente
'el archivo que contiene el path del directorio de El sistema for Windows
Private Sub EncuentraPathGeneral()

  'Declaración de variables para el manejo del canal de salida
  Dim WCanal%

  'Libera memoria del archivo
  WCanal% = FreeFile

  'Abre archivo con path de El sistema for Windows
```

```
  Open "c:\EL CDI\PCApath.plt" For Input As #WCanal%

  'Toma la primera y una linea del archivo para leer la
  'dirección de El sistema for Windows
  Line Input #WCanal%, GlMPath$

  'Cierra archivo
  Close #WCanal%
End Sub
'Realiza operación de busqueda en un registro dependiendo del número
'registros que componen la llave
Private Sub RealizaOperSeek(IndiceArch%, NumMaxCampos%, Operador$)
  'Numero de registros que componen la llave
  Select Case NumMaxCampos%
    Case 1
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1)
    Case 2
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1), GlLlaveArch$(2)
    Case 3
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1), GlLlaveArch$(2), GlLlaveArch$(3)
    Case 4
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1), GlLlaveArch$(2), GlLlaveArch$(3),
GlLlaveArch$(4)
    Case 5
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1), GlLlaveArch$(2), GlLlaveArch$(3),
GlLlaveArch$(4), GlLlaveArch$(5)
    Case 6
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1), GlLlaveArch$(2), GlLlaveArch$(3),
GlLlaveArch$(4), GlLlaveArch$(5), GlLlaveArch$(6)
    Case 7
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1), GlLlaveArch$(2), GlLlaveArch$(3),
GlLlaveArch$(4), GlLlaveArch$(5), GlLlaveArch$(6), GlLlaveArch$(7)
    Case 8
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1), GlLlaveArch$(2), GlLlaveArch$(3),
GlLlaveArch$(4), GlLlaveArch$(5), GlLlaveArch$(6), GlLlaveArch$(7), GlLlaveArch$(8)
    Case 9
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1), GlLlaveArch$(2), GlLlaveArch$(3),
GlLlaveArch$(4), GlLlaveArch$(5), GlLlaveArch$(6), GlLlaveArch$(7), GlLlaveArch$(8), GlLlaveArch$(9)
    Case 10
      GlRecordSetArch(IndiceArch%).Seek Operador$, GlLlaveArch$(1), GlLlaveArch$(2), GlLlaveArch$(3),
GlLlaveArch$(4), GlLlaveArch$(5), GlLlaveArch$(6), GlLlaveArch$(7), GlLlaveArch$(8), GlLlaveArch$(9), GlLlaveArch$(10)
  End Select
End Sub

'Operación que abre los archivos de btrieve
Public Function AbrirArchivos%()

  'Declaración de variables
  Dim i As Integer

  'Inicializa función a valor 0
  AbrirArchivos% = 0

  'Se prepara para el manejo de errores al momento de abrir
  'arhivos de btrieve
  On Error GoTo ErrorAbrirArchivos

  'Inicia proceso de apertura de archivos
  For i = 1 To GlNumMaxArchivos
    'Identifica si es un archivo que pertenece al directorio
    'de la compañia (GlArchCia$(1)) o al directorio del usuario (GlArchCia$(0))
    If GlArchCia$(i) = "1" Then
      Set GlRecordSetArch(i) = BDCia.OpenRecordset(GlNombreArchLogico$(i), dbOpenTable)
    Else
      If GlArchCia$(i) = "2" Then
        Set GlRecordSetArch(i) = BDUsuario.OpenRecordset(GlNombreArchLogico$(i), dbOpenTable)
      Else
```

```
        Set GlRecordSetArch(i) = BDCiaPlt.OpenRecordset(GlNombreArchLogico$(i), dbOpenTable)
      End If
    End If
    GlRecordSetArch(i).Index = GlNombreIndiceArch$(i)
  Next i
  'Si no existe un error sale de la función
  On Error GoTo 0
  Exit Function

'Si hay error regresa el dato como resultado de la función
ErrorAbrirArchivos:
  AbrirArchivos% = Err
  On Error GoTo 0
End Function
'se queda
'Abre base de datos de la compañía de Access

'Abre base de datos de la compañía de El sistema
Public Function AbrirBDCia()

  'Inicializa valor de la función a 0
  AbrirBDCiaPlt% = 0

  'Ejecuta el proceso de busqueda del directorio de la compañía
  EncuentraPathCia
  'Prepara proceso por si existe un error al buscar el directorio
  On Error GoTo ErrorAbrirBDCiaPlt
  'Abre la base de datos de la compañía
  Set BDCiaPlt = Workspaces(0).OpenDatabase(GlMPathDDFCia$ + FileDDF$, False, False, CTypeBD)
  'Si no existe error al abrir la base de datos de la compañía sale de la función
  On Error GoTo 0
  Exit Function
'Si existe error pasa el valor del error como resultado de la función
ErrorAbrirBDCiaPlt:
  AbrirBDCiaPlt% = Err
  On Error GoTo 0
End Function


'Abre base de datos del usuario
Public Function AbrirBDUsuario%()

  'Inicializa a valor 0 la función
  AbrirBDUsuario% = 0

  'Ejecuta la función para encontrar el directorio del usuario
  EncuentraPathUsuario

  'Prepara proceso por si existe un error al abrir la base
  'de datos del usuario
  On Error GoTo ErrorAbrirBDUsuario

  'Abre la base de datos del usuario
  Set BDUsuario = Workspaces(0).OpenDatabase(GlMPathDDFUsuario$ + FileDDF$, False, False, CTypeBD)

  'Si no existe un error al abrir la base de datos de usuario
  'termina la función
  On Error GoTo 0
  Exit Function

'Si existe un error al abrir la base de datos del usuario
'pasa dicho valor como valor de salida  de la función
ErrorAbrirBDUsuario:
  AbrirBDUsuario% = Err
  On Error GoTo 0
End Function
```

```
'Cierra archivos
Public Function CerrarArchivos%()

  'Declaración de variables
  Dim i As Integer

  'Genera proceso de cierre de archivos
  For i = 1 To GlNumMaxArchivos
    GlRecordSetArch(i).Close
  Next i
End Function

'Cierra base de datos de la compañia
Public Sub CerrarBDCia()
  BDCia.Close
End Sub

'Cierra base de datos del usuario
Public Sub CerrarBDUsuario()
  BDUsuario.Close
End Sub

'Función que realiza las diversas operaciones de registros
'existentes en btrieve
Public Function OperReg%(IndiceArch%, Operacion%, NumMaxCampos%)
  'Prepara a la función para el manejo y captura de errores
  On Error GoTo ErrorDeOperLect

  'Ejecuta un select para la selección de la operación que se esta ejecutando
  Select Case Operacion%
    Case CGlBtrInsert
      GlRecordSetArch(IndiceArch%).Update
    Case CGlBtrUpdate
      GlRecordSetArch(IndiceArch%).Update
    Case CGlBtrDelete
      GlRecordSetArch(IndiceArch%).Delete
    Case CGlBtrGetEqual
      Call RealizaOperSeek(IndiceArch%, NumMaxCampos%, "=")
      If GlRecordSetArch(IndiceArch%).NoMatch Then
        OperReg% = 1
      Else
        OperReg% = 0
      End If
    Case CGlBtrGetNext
      GlRecordSetArch(IndiceArch%).MoveNext
      If GlRecordSetArch(IndiceArch%).EOF Then
        OperReg% = 1
      Else
        OperReg% = 0
      End If
    Case CGlBtrGetPrevious
      GlRecordSetArch(IndiceArch%).MovePrevious
      If GlRecordSetArch(IndiceArch%).BOF Then
        OperReg% = 1
      Else
        OperReg% = 0
      End If
    Case CGlBtrGetGrThan
      Call RealizaOperSeek(IndiceArch%, NumMaxCampos%, ">")
      If GlRecordSetArch(IndiceArch%).NoMatch Then
        OperReg% = 1
      Else
        OperReg% = 0
      End If
    Case CGlBtrGetGrEqual
      Call RealizaOperSeek(IndiceArch%, NumMaxCampos%, ">=")
      If GlRecordSetArch(IndiceArch%).NoMatch Then
```

```
        OperReg% = 1
      Else
        OperReg% = 0
      End If
    Case CGIBtrGetLsThan
      Call RealizaOperSeek(IndiceArch%, NumMaxCampos%, "<")
      If GIRecordSetArch(IndiceArch%).NoMatch Then
        OperReg% = 1
      Else
        OperReg% = 0
      End If
    Case CGIBtrGetLsEqual
      Call RealizaOperSeek(IndiceArch%, NumMaxCampos%, "<=")
      If GIRecordSetArch(IndiceArch%).NoMatch Then
        OperReg% = 1
      Else
        OperReg% = 0
      End If
    Case CGIBtrGetFirst
      GIRecordSetArch(IndiceArch%).MoveFirst
      If GIRecordSetArch(IndiceArch%).BOF Then
        OperReg% = 1
      Else
        OperReg% = 0
      End If
    Case CGIBtrGetLast
      GIRecordSetArch(IndiceArch%).MoveLast
      If GIRecordSetArch(IndiceArch%).EOF Then
        OperReg% = 1
      Else
        OperReg% = 0
      End If
  End Select

  'Si no existe error da por terminada la función
  On Error GoTo 0
  Exit Function

'Si existe error emite el valor del error como resultado
'de la función
ErrorDeOperLect:
  OperReg% = Err
  On Error GoTo 0
End Function
'Prepara operación de actualización de uno o unos registros
Public Sub PCAPreparaActualizacion(IndiceArch%, Operacion%)
  'Identifica si la operación es un insert o si es un update
  'para generar los procesos necesarios en su configuración
  If Operacion% = CGIBtrInsert Then
    GIRecordSetArch(IndiceArch%).AddNew
  Else
    GIRecordSetArch(IndiceArch%).Edit
  End If
End Sub
```

4. Módulo Proceso

```
'Realizado por : PCA
'Modulo : Procesos para la conversión de numeros a letras
' 'Objetivo : Manejo de funciones para el manejo de operaciones con btrieve


Sub PCAVerificaEquivalencias(LClienteArchivo As String)
  'proceso que busca, a partir del cliente que se encuentra en el archivo
  'de texto, el cliente equivalente en arequiv, buscando a partir de
```

```
    'arequiv, a argroup y el equivalente en arequiv
    Dim LCliente As String
    LCliente = Mid(LClienteArchivo, 3, 10)
    PCABuscaAREQUIV (LCliente)
    If GClienteEnAREQUIV <> "" Then
      'encontre cliente_equivalente por cliente_member_code en AREQUIV, busco en ARGROUP
      PCABuscaARGROUP (GClienteEnAREQUIV)
      If GClienteEnARGROUP <> "" Then
        'encontre cliente_que_agrupa por cliente en ARGROUP
        PCABuscaEnAREQUIVCIteEquivalente (GClienteEnARGROUP)
        If GClienteEnAREQUIV <> "" Then
          'encontre cliente_membre_code por cliente_equivalente
          Mid(LClienteArchivo, 3, 10) = GClienteEnAREQUIV
        Else
          Mid(LClienteArchivo, 3, 10) = LCliente
        End If
      Else
        'no encontre en argroup
        Mid(LClienteArchivo, 3, 10) = LCliente
      End If
    Else
      'no encontre cliente en arequiv
      Mid(LClienteArchivo, 3, 10) = LCliente
    End If
    GSubstMembercode = LClienteArchivo
  End Sub
  'para argroup.bas
  Sub PCABuscaAREQUIV(LCliente As String)
    Dim ResultadoBusqueda As Integer
    'verifica en arequiv, busco con el cliente (arch.texto.linea 02)
    'y recupero el cliente_equivalente
    GlLlaveArch$(1) = LCliente
    ResultadoBusqueda = OperReg%(AREQUIV1, CGIBtrGetEqual, 1)
    If ResultadoBusqueda <> 0 Then
      'no encontre equivalencia lo dejo igual
      GClienteEnAREQUIV = ""
    Else
      'encontre equivalencia en arequiv
      GClienteEnAREQUIV = GlRecordSetArch(AREQUIV1)!Customer_key
    End If
  End Sub
  'para argroup.bas
  Sub PCABuscaARGROUP(LClienteEnAREQUIV As String)
    Dim ResultadoBusqueda As Integer
    'con el cliente_equivalente recupero el cliente_que_agrupa
    GlLlaveArch$(1) = LClienteEnAREQUIV   'busco por indice 2
    ResultadoBusqueda = OperReg%(ARGROUP2, CGIBtrGetEqual, 1)
    If ResultadoBusqueda <> 0 Then
      'no encontre equivalencia lo dejo igual
      GClienteEnARGROUP = ""
    Else
      GClienteEnARGROUP = GlRecordSetArch(ARGROUP2)!cliente_agrupador
    End If
  End Sub
  'para argroup.bas
  Sub PCABuscaEnAREQUIVCIteEquivalente(GClienteEnARGROUP As String)
    Dim ResultadoBusqueda As Integer
    'con el cliente que agrupa busco en arequiv, en cliente_equivalente,
    'y obtengo cliente_member_code, que es el que se guarda en el arreglo
    'y a su vez en el temporal
    Dim LEncontreAREQUIV As Byte
    GlLlaveArch$(1) = GClienteEnARGROUP
    ResultadoBusqueda = OperReg%(AREQUIV2, CGIBtrGetEqual, 1)
    If ResultadoBusqueda <> 0 Then
      'no encontre equivalencia lo dejo igual
      GClienteEnAREQUIV = ""
    Else
```

```
      'encontre equivalencia
      GClienteEnAREQUIV = GlRecordSetArch(AREQUIV2)!Membercode
    End If
End Sub
'para argroup.bas
Sub PCAInsertaTmpFacturas(LArrClientes() As String, NoLineasDetFactura As Integer)
    'verifica si la factura que existe en el arreglo ya fue insertada
    'en la tabla temporal, si es nueva la  inserta, si no la actualiza
    Dim i As Integer
    For i = 3 To NoLineasDetFactura
      'valida por cada linea porque cada una tiene un servicio diferente
      If Not PCAValidaCliteRelServInsertado(Mid(LArrClientes(2), 3, 10), Mid(LArrClientes(i), 3, 10)) = 0 Then  'le quite al
principio Mid(LArrClientes(1), 3, 8),
        PCAInsertaFacturaTmp LArrClientes(), i
      Else
        PCAActualizaFacturas LArrClientes(), i
      End If
    Next i
End Sub
'para argroup.bas
Function PCAValidaCliteRelServInsertado(LCli As String, LServ As String) 'se lo quite LCite As String,
    Dim ResultadoBusqueda As Integer
    'verifica si la combinacion cliente, relacion, servicio ya fue insertada
    'en la tabla temporal
    GlLlaveArch$(1) = LCli
    GlLlaveArch$(2) = LServ
    ResultadoBusqueda = OperReg%(ARFACTMP, CGlBtrGetEqual, 2)
    PCAValidaCliteRelServInsertado = ResultadoBusqueda%
End Function
'para argroup.bas y este proceso tambien

Sub PCALimpiaArregloClientes(LArrClientes() As String, LLineaFactura As Integer)
    'inicializo el arreglo donde guardo cada factura que se guarda en el temporal
    Dim i As Integer
    For i = 1 To LLineaFactura - 1
      LArrClientes(i) = ""
    Next i
EndSub


Sub PCAAvanzaForma2()
    frmBusArchivo.Hide
    Load frmUbicacionINLOC
    frmUbicacionINLOC.Show
End Sub

Sub PCAAvanzaForma3()
    frmUbicacionINLOC.Hide
    Load frmProcesando
    frmProcesando.Show
End Sub


Sub PCALeeLinea(GCanalArchivo As Integer)
    Input #GCanalArchivo, GLineaLeida
End Sub

Sub PCACierraArchivos()
    Close #GCanalArcProceso%
    Close #GCanalArcCostServ%
    Close #GCanalArcOrdCostServ%
End Sub

Sub PCATerminaAplicacion()
    PCACierraArchivoEntrada
    PCACierraArchivos
End Sub
```

```
Sub PCADescargaFormas()
   Unload frmParIntPoliza
   Unload frmBusArchivo
   Unload frmUbicacionINLOC
End Sub

Sub PCACierraArchivoEntrada()
   Close #GCanalArcEntrada%
End Sub

Sub PCACierraArchivoCostServ()
   Close #GCanalArcCostServ%
End Sub

Sub PCALimpiaTextos()
   frmParIntPoliza.txtCodServOrigen.Text = ""
   frmParIntPoliza.txtCodServDestino.Text = ""
   frmParIntPoliza.txtCosUnitServicio.Text = 0
   frmParIntPoliza.txtNumMinServicios.Text = 0
   frmParIntPoliza.txtCarMinFacturado.Text = 0
   frmParIntPoliza.txtMonMinXFactura.Text = 0
End Sub


Function PCALeeArchivo()
   'lee el archivo factura.txt linea por linea
   Dim LAbriArchivo As Integer
   LAbriArchivo = PCAAbreArchivoEntrada
   If LAbriArchivo = 1 Then
      'obtiene el numero de lineas que tiene el archivo factura.txt
      'y al mismo tiempo abre un arreglo donde se van a validar
      'los servicios
      GNoLineas = PCACuentaLineasXTipo
   End If
   PCALeeArchivo = LAbriArchivo
End Function

Sub PCAObtenCliente(i As Integer)
   'obtiene los clientes en la segunda posicion del arreglo
   Dim LLineaInvalida As Long
   LLineaInvalida = 0
   If Trim(Mid(GLineaLeida, 3, 10)) <> "" Then
      ReDim Preserve GArrRelaciones2(i)
      GArrRelaciones2(i) = Trim(Mid(GLineaLeida, 3, 10))
   Else
      LLineaInvalida = 1
   End If
   If LLineaInvalida = 1 Then
      MsgBox "Proceso terminado, linea 02 inválida en el archivo " + frmBusArchivo.txtUbicArchivo.Text, vbOKOnly +
vbExclamation, "Grupos de Clientes"
      PCAEscribeError "Proceso terminado, linea 02 inválida en el archivo " + frmBusArchivo.txtUbicArchivo.Text
      PCADescargaFormas
      PCATerminaAplicacion
   End If
End Sub
```

El diseño utilizado como modelo, se muestra en la figura 4.22 , en la que observamos tanto las tablas principales que intervienen en el mismo, como sus relaciones.

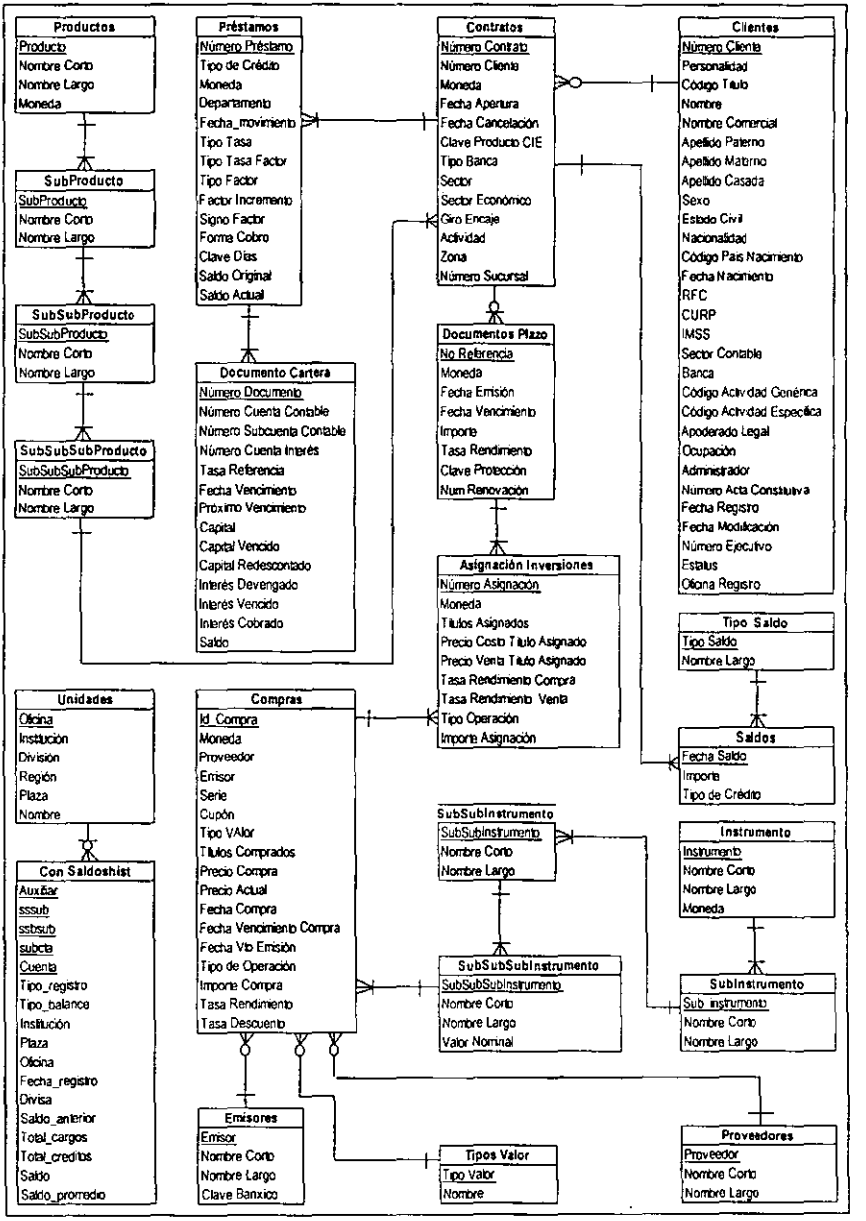


Figura 4.22. Diagrama Relacional del CDI.

A continuación se presenta el diseño final conformado a partir de múltiples entrevistas con los usuarios, de recabar sus necesidades y de verificar como se estaba organizando la información, así como una manera fácil de explotarla, en base a la organización y a creación de índices y relaciones entre las tablas que lo conforman.

Cabe mencionar que este es un diseño aproximado al real, debido a la confidencialidad de la información y que únicamente se mostrarán las estructuras de las tablas y sus índices, mientras que las relaciones entre las mismas deberán verse en el diagrama relacional contenido en el cuerpo de este trabajo.

Tablas

En este apartado del trabajo de tesis podremos ver un acercamiento al modelo final, resultado de la etapa de diseño, que conforma al CDI , el diseño, los *layouts* y los índices, así como las relaciones entre las tablas, se podrán observar a detalle como se muestra a continuación.

Tabla: Clientes

Columnas

Nombre	Tipo	Tamaño
clave_cliente	Texto	8
personalidad	Texto	6
cod_titulo	Texto	5
nombre	Texto	30
nombre_comercial	Texto	50
ap_paterno	Texto	30
ap_materno	Texto	50
ap_casada	Texto	50
sexo	Texto	1
edo_civil	Texto	7
nacionalidad	Texto	15
cod_pais_nacimiento	Texto	5
fecha_nacimiento	Fecha/Hora	8
rfc	Texto	10
IMSS	Texto	20
sector_contable	Texto	10

clave_institucion	Texto	10
cod_actividad	Texto	10
apoderado_legal	Texto	50
ocupacion	Texto	20
administrador	Texto	50
no_acta_constitutiva	Texto	20
fecha_registro	Fecha/Hora	8
fecha_modificacion	Fecha/Hora	8
no_ejecutivo	Texto	15
estaus	Texto	8

Indices de tabla

Nombre	Número de campos
clave_cliente	1
Campos:	clave_cliente, Ascendente
clave_institucion	1
Campos:	clave_institucion, Ascendente
PrimaryKey	1
Campos:	clave_cliente, Ascendente

Tabla: Compras

Columnas

Nombre	Tipo	Tamaño
clave_compra	Texto	10
moneda	Texto	5
proveedor	Texto	7
emision	Texto	10
serie	Texto	10
cupon	Texto	10
tipo_valor	Texto	5
titulos_comprados	Número (largo)	4
precio_compra	Moneda	8
precio_actual	Moneda	8

fecha_compra	Fecha/Hora	8
fecha_venc_compra	Fecha/Hora	8
fecha_venc_emision	Fecha/Hora	8
tipo_operacion	Texto	5
importe_compra	Moneda	8
tasa_rendimiento	Número (simple)	4
tasa_descuento	Número (simple)	4
observaciones	Texto	50

Indices de tabla

Nombre	Número de campos
clave_compra	1
Campos:	clave_compra, Ascendente
PrimaryKey	1
Campos:	clave_compra, Ascendente

Tabla: Contratos

Columnas

Nombre	Tipo	Tamaño
clave_contrato	Texto	11
no_cliente	Texto	8
moneda	Texto	5
fecha_apertura	Fecha/Hora	8
fecha_cancelacion	Fecha/Hora	8
clave_prod_cie	Texto	10
sector	Texto	10
sector_economico	Texto	10
giro	Texto	20
clave_zona	Texto	6
clave_institucion	Texto	10

Indices de tabla

Nombre	Número de campos
clave_contrato	1
Campos:	clave_contrato, Ascendente
clave_prod_cie	1
Campos:	clave_prod_cie, Ascendente
PrimaryKey	1
Campos:	clave_contrato, Ascendente

Tabla: DocCartera

Columnas

Nombre	Tipo	Tamaño
no_documento	Texto	8
no_cta_contable	Texto	15
no_subcta_contable	Texto	8
no_cta_interes	Texto	8
tasa_referencia	Texto	5
fecha_vencimieno	Fecha/Hora	8
prox_vencimiento	Fecha/Hora	8
capital	Moneda	8
capital_vencido	Moneda	8
interes_devengado	Número (simple)	4
interes_vencido	Número (simple)	4
interes_cobrado	Número (simple)	4
saldo	Número (doble)	8

Indices de tabla

Nombre	Número de campos
PrimaryKey	1
Campos:	no_documento, Ascendente

Tabla: DocPlazo

Columnas

Nombre	Tipo	Tamaño
no_referencia	Texto	8
moneda	Texto	5
fecha_emision	Fecha/Hora	8
fecha_vencimiento	Fecha/Hora	8
importe	Moneda	8
tasa_rendimiento	Número (simple)	4
no_renovacion	Texto	8

Indices de tabla

Nombre	Número de campos
PrimaryKey	1
Campos:	no_referencia, Ascendente

Tabla: Emisores

Columnas

Nombre	Tipo	Tamaño
clave_emisor	Texto	10
nombre_corto	Texto	10
nombre_largo	Texto	30
clave_banxico	Texto	10
atencion	Texto	30
telefono	Texto	15

Indices de tabla

Nombre	Número de campos
clave_banxico	1
Campos:	clave_banxico, Ascendente
clave_emisor	1
Campos:	clave_emisor, Ascendente
PrimaryKey	1
Campos:	clave_emisor, Ascendente

Tabla: Instrumentos

Columnas

Nombre	Tipo	Tamaño
clave_intrumento	Texto	10
nombre_corto	Texto	10
descripcion	Texto	30
moneda	Texto	5

Indices de tabla

Nombre	Número de campos
PrimaryKey	2
Campos:	clave_intrumento, Ascendente
	moneda, Ascendente

Tabla: Inversiones

Columnas

Nombre	Tipo	Tamaño
no_asignacion	Texto	8
moneda	Texto	5
fecha_emision	Fecha/Hora	8
fecha_vencimiento	Fecha/Hora	8
titulos_asignados	Número (largo)	4
precio_costo_tit_asignado	Moneda	8
precio_venta_tit_asignado	Moneda	8
tasa_rendimiento_compra	Número (simple)	4
tasa_rendimiento_venta	Número (largo)	4
tipo_operacion	Texto	5
importe_asignacion	Moneda	8

Indices de tabla

Nombre	Número de campos
PrimaryKey	1
Campos:	no_asignacion, Ascendente

Tabla: Prestamos

Columnas

Nombre	Tipo	Tamaño
no_prestamo	Texto	12
tipo_credito	Texto	7
moneda	Texto	5
departamento	Texto	10
fecha_movimiento	Fecha/Hora	8
tipo_tasa	Texto	8

tipo_tasa_factor	Número (simple)	4
factor_incremento	Número (simple)	4
forma_cobro	Texto	10
clave_dias	Texto	5
saldo_original	Moneda	8
saldo_actual	Moneda	8

Indices de tabla

Nombre	Número de campos
clave_dias	1
Campos:	clave_dias, Ascendente
PrimaryKey	1
Campos:	no_prestamo, Ascendente

Tabla: Productos

Columnas

Nombre	Tipo	Tamaño
codigo_producto	Texto	5
descripcion	Texto	25
moneda	Texto	5

Indices de tabla

Nombre	Número de campos
nombre_corto	1
Campos:	codigo_producto, Ascendente
PrimaryKey	1
Campos:	codigo_producto, Ascendente

Tabla: Proveedores

Columnas

Nombre	Tipo	Tamaño
clave_proveedor	Texto	8
descripcion	Texto	50
direccion	Texto	50

Indices de tabla

Nombre	Número de campos
clave_proveedor	1
Campos:	clave_proveedor, Ascendente
PrimaryKey	1
Campos:	clave_proveedor, Ascendente

Tabla: Saldos

Columnas

Nombre	Tipo	Tamaño
clave_tipo_saldo	Texto	11
fecha_saldo	Fecha/Hora	8
importe	Moneda	8
tipo_credito	Texto	3

Indices de tabla

Nombre	Número de campos
clave_tipo_saldo	1
Campos:	clave_tipo_saldo, Ascendente

PrimaryKey	1
Campos:	clave_tipo_saldo, Ascendente

Tabla: SaldosHistoricos

Columnas

Nombre	Tipo	Tamaño
clave_subsubsubproducto	Texto	5
clave_subsubproducto	Texto	5
clave_subcuenta	Texto	10
cuenta	Texto	11
tipo_registro	Texto	2
tipo_balance	Texto	10
institucion	Texto	50
oficina	Texto	5
fecha_registro	Fecha/Hora	8
moneda	Texto	5
saldo_anterior	Moneda	8
total_cargos	Moneda	8
total_creditos	Moneda	8
saldo	Moneda	8
saldo_promedio	Moneda	8

Indices de tabla

Nombre	Número de campos
saldos_historicos	4
Campos:	clave_subsubsubproducto, Ascendente clave_subsubproducto, Ascendente clave_subcuenta, Ascendente cuenta, Ascendente

Tabla: SubInstrumentos

Columnas

Nombre	Tipo	Tamaño
clave_subinstrumento	Texto	10
nombre_corto	Texto	10
descripcion	Texto	30

Indices de tabla

Nombre	Número de campos
clave_subinstrumento	1
Campos:	clave_subinstrumento, Ascendente
PrimaryKey	1
Campos:	clave_subinstrumento, Ascendente

Tabla: SubProductos

Columnas

Nombre	Tipo	Tamaño
clave_subproducto	Texto	5
descripcion	Texto	25

Indices de tabla

Nombre	Número de campos
clave_subproducto	1
Campos:	clave_subproducto, Ascendente
PrimaryKey	1
Campos:	clave_subproducto, Ascendente

Tabla: SubSubInstrumentos

Columnas

Nombre	Tipo	Tamaño
subsubinstrumento	Texto	5
descripcion	Texto	20

Indices de tabla

Nombre	Número de campos
PrimaryKey	1
Campos:	subsubinstrumento, Ascendente

Tabla: SubSubProductos

Columnas

Nombre	Tipo	Tamaño
clave_subsubproducto	Texto	5
descripcion	Texto	25

Indices de tabla

Nombre	Número de campos
clave_subsub_producto	1
Campos:	clave_subsubproducto, Ascendente
PrimaryKey	1
Campos:	clave_subsubproducto, Ascendente

Tabla: SubSubSubProductos

Columnas

Nombre	Tipo	Tamaño
clave_subsubsubproducto	Texto	5
descripcion	Texto	25

Indices de tabla

Nombre	Número de campos
clave_subsubsubproducto	1
Campos:	clave_subsubsubproducto, Ascendente
PrimaryKey	1
Campos:	clave_subsubsubproducto, Ascendente

Tabla: Sucursales

Columnas

Nombre	Tipo	Tamaño
clave_institucion	Texto	10
division	Texto	20
region	Texto	6
plaza	Texto	20
nombre	Texto	50

Indices de tabla

Nombre	Número de campos
PrimaryKey	1
Campos:	clave_institucion, Ascendente

Tabla: TipoSaldo

Columnas

Nombre	Tipo	Tamaño
clave_tipo_saldo	Texto	5
descripcion	Texto	30

Indices de tabla

Nombre	Número de campos
clave_tipo_saldo	1
Campos:	clave_tipo_saldo, Ascendente
PrimaryKey	1
Campos:	clave_tipo_saldo, Ascendente

Tabla: TiposValor

Columnas

Nombre	Tipo	Tamaño
clave_tipo_valor	Texto	10
descripcion	Texto	30

Indices de tabla

Nombre	Número de campos
clave_tipo_valor	1
Campos:	clave_tipo_valor, Ascendente
PrimaryKey	1
Campos:	clave_tipo_valor, Ascendente

A continuación se presenta la definición para cada una de las tablas, de los datos que deberán ser introducidos en ellas. Se presenta el nombre de la tabla, el nombre del campo y una breve descripción del dato que deberá ser introducido en cada uno de ellos.

Tabla: Clientes

clave_cliente
Descripción: número único por cliente
personalidad
Descripción: tipo de personalidad del cliente(física o moral)
cod_titulo
Descripción: tipo de profesión del cliente
Nombre
Descripción: nombre del cliente
nombre_comercial
Descripción: nombre comercial del cliente
ap_paterno
Descripción: apellido paterno del cliente

ap_materno
Descripción: apellido materno del cliente
ap_casada
Descripción: apellido de casada del cliente
sexo
Descripción: sexo del cliente
edo_civil
Descripción: estado civil del cliente
nacionalidad
Descripción: nacionalidad
cod_pais_nacimiento
Descripción: código del país de nacimiento del cliente
fecha_nacimiento
Descripción: fecha de nacimiento del cliente
rfc
Descripción: RFC del cliente
IMSS
Descripción: número de seguro social del cliente
sector_contable

Descripción: sector al que pertenece el cliente

clave_institucion

Descripción: clave de la sucursal bancaria del cliente

cod_actividad

Descripción: código de actividad del cliente

apoderado_legal

Descripción: apoderado legal en caso de existir

ocupacion

Descripción: ocupación del cliente

administrador

Descripción: administrador del cliente en caso de existir

no_acta_constitutiva

Descripción: número de cata constitutiva del cliente si es persona moral

fecha_registro

Descripción: fecha de registro del cliente

fecha_modificacion

Descripción: fecha de última modificación de datos

no_ejecutivo

Descripción: clave del ejecutivo que atiende al cliente

estaus

Descripción: estatus del cliente (activo o inactivo)

clave_compra

Descripción: clave única de la compra

moneda

Descripción: moneda en que se realizó la operación

proveedor

Descripción: clave del proveedor al que se le realizó la compra

Tabla: Emisiones

emision

Descripción: emisión de la compra

serie

Descripción: número de serie de la compra

cupon

Descripción: cupón en el que se afecta la compra

tipo_valor

Descripción: tipo de moneda en que la compra es realizada
titulos_comprados
Descripción: número de títulos comprados en la transacción
precio_compra
Descripción: precio en el que la moneda fué comprada
precio_actual
Descripción: precio en el que la compra se cotiza actualmente
fecha_compra
Descripción: fecha en la que se realiza la transacción
fecha_venc_compra
Descripción: fecha en la que termina la operación de compra
fecha_venc_emision
Descripción: fecha en la que la emisión llega a su vencimiento
tipo_operacion
Descripción: tipo de operación (compra o venta)
importe_compra
Descripción: importe con el que se realiza la transacción
tasa_rendimiento
Descripción: tasa a la que se fija el rendimiento de la compra
tasa_descuento
Descripción: tasa en la que se fija el descuento de la compra
observaciones
Descripción: observaciones

Tabla: Contratos

clave_contrato
Descripción: número de cuenta del contrato
no_cliente
Descripción: número único por cliente
moneda
Descripción: moneda del contrato
fecha_apertura
Descripción: fecha de apertura del contrato en la sucursal
fecha_cancelacion
Descripción: fecha de cancelación del contrato
clave_prod_cie
Descripción: clave que describe el producto del CIE
sector

Descripción: sector al que pertenece el contrato

sector_economico

Descripción: clave del sector económico al que pertenece el contrato según INEGI

giro

Descripción: giro al que se aplica el contrato

clave_zona

Descripción: clave de zona a la que pertenece el contrato

clave_institucion

Descripción: clave de cada una de las sucursales bancarias

Tabla: DocCartera

no_documento

Descripción: número único con el que se identifica el documento

no_cta_contable

Descripción: cuenta contable a la que afecta

no_subcta_contable

Descripción: subcuenta contable a la que pertenece

no_cta_interes

Descripción: cuenta de interés

tasa_referencia

Descripción: tasa de referencia

fecha_vencimieno

Descripción: fecha en la que el documento termina

prox_vencimiento

Descripción: fecha de m s próximo vencimiento

capital

Descripción: capital del documento

capital_vencido

Descripción: capital del documento vencido a la fecha

interes_devengado

Descripción: interés aplicado al documento

interes_vencido

Descripción: interés vencido del doumento a la fecha

interes_cobrado

Descripción: interés que ha sido cobrado a la fecha de la consulta

saldo

Descripción: saldo a la fecha

no_referencia

Descripción: número asignado a cada documento

moneda

Descripción: moneda en la que se realiza la transacción

fecha_emision

Descripción: fecha de emisión del documento

fecha_vencimiento

Descripción: fecha de vencimiento del documento

importe

Descripción: importe del documento

tasa_rendimiento

Descripción: tasa de rendimiento a la que se fija el documento

no_renovacion

Descripción: número asignado a cada documento cuando se renueva

Tabla: Emisores

clave_emisor

Descripción: clave única por emisor

nombre_corto

Descripción: nombre descriptivo manejado por el banco

nombre_largo

Descripción: nombre descriptivo general

clave_banxico

Descripción: clave asignada por Banco de México

atencion

Descripción: persona con la cual se tiene el contacto

telefono

Descripción: teléfono de la atención

Tabla: Instrumentos

clave_intrumento

Descripción: clave única por instrumento bancario

nombre_corto

Descripción: nombre del instrumento dentro del banco

descripcion
Descripción: nombre del instrumento
moneda
Descripción: moneda del instrumento

Tabla: Inversiones

no_asignacion
Descripción: número único de asignación por inversión
moneda
Descripción: clave de moneda en la que se realiza la inversión
fecha_emisión
Descripción: fecha de emisión de la inversión
fecha_vencimiento
Descripción: fecha de vencimiento de la inversión
titulos_asignados
Descripción: número de títulos asignados a la inversión
precio_costo_tit_asignado
Descripción: precio del costo al cual se asignan los títulos
precio_venta_tit_asignado
Descripción: precio de la venta al cual se asignan los títulos
tasa_rendimiento_compra
Descripción: tasa a la que se fija el rendimiento de la compra
tasa_rendimiento_venta
Descripción: tasa a la que se fija el rendimiento de la venta
tipo_operación
Descripción: tipo de operación
importe_asignación
Descripción: cantidad con la que se asigna la inversión

Tabla: Prestamos

no_prestamo
Descripción: número único definido para cada préstamo otorgado
tipo_credito
Descripción: tipo de crédito al que pertenece el préstamo
moneda

Descripción: moneda en la que se realizó el préstamo
departamento
Descripción: departamento al que se carga el préstamo
fecha_movimiento
Descripción: fecha en la que el préstamo es otorgado
tipo_tasa
Descripción: tipo de tasa en la que el préstamo es otorgado (fija o variable)
tipo_tasa_factor
Descripción: factor en el que el préstamo es otorgado
factor_incremento
Descripción: incremento fijado al factor del préstamo
forma_cobro
Descripción: periodicidad y tipo de cobro del préstamo
clave_dias
Descripción: período en que es pagadero el préstamo
saldo_original
Descripción: saldo en el que el prestamo es iniciado
saldo_actual
Descripción: saldo en el que se encuentra el préstamo al momento de realizar la consulta

Tabla: Productos

codigo_producto
Descripción: definición de la clave del producto
descripcion
Descripción: descripción del nombre del producto
moneda
Descripción: clave de moneda en la que se maneja el producto

Tabla: Proveedores

clave_proveedor
Descripción: clave del proveedor
descripcion
Descripción: nombre del proveedor
direccion

Descripción: dirección del proveedor

Tabla: Saldos

clave_tipo_saldo
Descripción: clave de la cuenta
fecha_saldo
Descripción: fecha a la que se refleja el saldo
importe
Descripción: importe del saldo
tipo_credito
Descripción: tipo de sistema del saldo

Tabla: SaldosHistoricos

clave_subsubsubproducto
Descripción: clave del triple subproducto bancario al que nos referimos
clave_subsubproducto
Descripción: clave del doble subproducto bancario a registrar
clave_subcuenta
Descripción: clave de la subcuenta a la que pertenece el dato
cuenta
Descripción: código de la cuenta del cliente
tipo_registro
Descripción: tipo de registro en cuanto a activo o desactivado
tipo_balance
Descripción: periodicidad del balance con el que se registra el dato
institucion
Descripción: clave de la institución a la que pertenece el movimiento
oficina
Descripción: clave de la institución en la que se afecta el movimiento
fecha_registro
Descripción: fecha en la que el registro o la transacción es generado
moneda
Descripción: moneda en la que se realiza la transacción
saldo_anterior
Descripción: saldo del cliente antes del movimiento actual
total_cargos

Descripción: total de los abonos registradas en este movimiento
total_creditos
Descripción: total de los cargos registradas en este movimiento
saldo
Descripción: saldo del cliente después de la transacción
saldo_promedio
Descripción: saldo promedio del cliente al momento de realizar la transacción

Tabla: SubInstrumentos

clave_subinstrumento
Descripción: clave única del subinstrumento bancario
nombre_corto
Descripción: nombre del subinstrumento dentro del banco
descripcion
Descripción: nombre del subinstrumento

Tabla: SubProductos

clave_subproducto
Descripción: clave del subproducto bancario
descripcion
Descripción: descripción del subproducto bancario

Tabla: SubSubInstrumentos

subsubinstrumento
Descripción: clave del subsubinstrumento
descripcion
Descripción: descripción del subsub instrumento bancario

Tabla: SubSubProductos

clave_subsubproducto
Descripción: clave del subsub producto bancario

descripcion
Descripción: descripción del subsub producto bancario

Tabla: SubSubSubProductos

clave_subsubsubproducto
Descripción: clave del triple subproduto bancario
descripcion
Descripción: descripción del triple subproducto bancario

Tabla: Sucursales

clave_institucion
Descripción: clave de cada una de las sucursales bancarias
division
Descripción: tipo de institución clasificada por activada y captación
region
Descripción: región física de las 6 a la que pertenece geográficamente
plaza
Descripción: ciudad de ubicación
nombre
Descripción: descripción del nombre de la institución

Tabla: TipoSaldo

clave_tipo_saldo
Descripción: clave del tipo de saldo
descripcion
Descripción: descripción del tipo de saldo

Tabla: TiposValor

clave_tipo_valor
Descripción: clave única del tipo valor
descripcion
Descripción: descripción del tipo valor

Una vez definido el modelo, se lleva a cabo la implementación y las pruebas finales, trabajo que se describirá a detalle en el siguiente capítulo.

CAPITULO 5

PRUEBAS, LIBERACION Y MANTENIMIENTO

En el presente capítulo se describirán principalmente cuáles fueron las validaciones que se hicieron de la información, el tipo de ayuda al operador en el momento de la ejecución, los medios de utilización como son el manual técnico y la ayuda en línea, para la explotación de la información y los resultados presentados a los usuarios finales.

5.1 Presentación de pruebas que validen la información transmitida y recuperada

Implementación

Dentro de la puesta en marcha de la configuración del CDI se realizaron diversos reportes mediante la herramienta GQL Windows, los cuales nos permitieron validar tanto la cantidad como la calidad de la información, pero de la misma manera nos hicieron ver la importancia de tener puntos de control dentro del proceso.

Sin embargo, antes de realizar reportes, se hicieron verificaciones propias de áreas técnicas para validar los resultados de los desarrollos, un ejemplo de lo anterior se muestra en el siguiente punto.

Pruebas de operación

Una de las validaciones más importantes utilizada como parte del desarrollo fue el generar tablas de resultados ó matrices de prueba, como las que se muestran a continuación.

En la primera tabla (véase Tabla 5.1), se observa el resultado obtenido de pruebas realizadas, esto es requerido antes de llevarse a cabo la liberación. El encabezado de la tabla describe el sistema donde se aplicó la prueba, el módulo del sistema al que pertenece el programa o proceso que se va a utilizar y el nombre del proceso del cual se obtuvieron los resultados.

Definición de entradas, descripción de columnas:

ID	Número de la prueba realizada.
Nombre	Parámetro de entrada al proceso.
Tabla Asociada	Tablas afectadas durante la ejecución del proceso.
Tipo de Dato	Definición del tipo de información recibida.

Definición de salidas, descripción de columnas:

ID	Número del resultado de la prueba.
Nombre	Estado del resultado de la prueba.
Tabla Asociada	Tabla (en este caso archivo) donde se almacenó el resultado final de la ejecución del proceso.

Sistema	Carga CDI
Módulo	Carga cheques
Programa	CDI/CHE/F/CARDIA/020

Definición de Entradas

ID	Nombre	Tabla Asociada	Tipo de Dato
1	saldo diario y T+1 cta-activa=4	Dbchemty Dbchesal	char *
2	saldo diario y T+1 cta-activa<>4	Dbchemty Dbchesal	char *
3	contrato cta-apertura >= fecha-parametro y cta-activa=4	Dbchemty Dbchesal	char *
4	contrato cta-apertura < fecha-parametro y cta-activa=4	Dbchemty Dbchesal	char *
5	contrato cta-apertura >= fecha-parametro y cta-activa<>4	Dbchemty Dbchesal	char *
6	contrato cta-apertura < fecha-parametro y cta-activa<>4	Dbchemty Dbchesal	char *
7	contrato cta-apertura < fecha-parametro y cta-activa<>4 y bandera fin de mes	Dbchemty Dbchesal	char *

Definición de Salidas

ID	Nombre	Tabla Asociada
1	Aceptado	CDI/*ZONA*/ CHE/*MMDD*/ /SAL
2	Rechazado	CDI/*ZONA*/ CHE/*MMDD*/ /SAL
3	Aceptado	CDI/*ZONA*/ CHE/*MMDD*/ /CON (CLI,DIR)
4	Rechazado	CDI/*ZONA*/ CHE/*MMDD*/ /CON (CLI,DIR)
5	Rechazado	CDI/*ZONA*/ CHE/*MMDD*/ /CON (CLI,DIR)
6	Rechazado	CDI/*ZONA*/ CHE/*MMDD*/ /CON (CLI,DIR)
7	Aceptado	CDI/*ZONA*/ CHE/*MMDD*/ /SAL

Tabla 5.1. Matriz de pruebas I.

En la segunda tabla (véase Tabla 5.2) bajo el mismo esquema de prueba, se determina si el caso se presentó en datos reales y la descripción de sus columnas es como sigue:

Valor de entradas, descripción de columnas:

ID	Número de la prueba realizada.
Caso Dato	Confirmación de que se efectuó la prueba.
Valor / Comentario	Descripción de la entrada al proceso.

Valor de salidas, descripción de columnas:

ID	Número del resultado de la prueba.
Valor / Comentario	Resultado final de la prueba, cumplió sí o no.
RF	Confirmación de que se recibió el resultado.

Valor de **Entradas**			Valor de **Salidas**		
ID	Caso Dato	Valor / Comentario	ID	Valor / Comentario	RF
1	✓	saldo diario y T+1 cta-inactiva=4	1	S	✓
2	✓	saldo diario y T+1 cta-activa<>4	2	N	✓
3	✓	contrato cta-apertura >= fecha-parametro y cta-activa=4	3	S	✓
4	✓	contrato cta-apertura < fecha-parametro y cta-activa=4	4	N	✓
5	✓	contrato cta-apertura >= fecha-parametro y cta-activa<>4	5	N	✓
6	✓	contrato cta-apertura < fecha-parametro y cta-activa<>4	6	N	✓
7	✓	contrato cta-apertura < fecha-parametro y cta-activa<>4 y bandera fin mes	6	S (se generan además los saldos mensuales)	✓

Tabla 5.2. Matriz de pruebas II.

Una vez que se había hecho la carga de las dos primeras zonas (MTY y SAL) dentro del CDI, se lograron obtener los primeros resultados propios del modelo, los cuales, fueron comparados contra el sistema de *UNISYS*. Cabe mencionar que de aquí en adelante, el CDI comenzó a ser utilizado como herramienta de manejo y explotación diaria por parte de los principales involucrados en el proyecto, aún realizando reportes de la información procesada con herramientas primitivas como GQL Windows, ya que la facilidad que el proyecto otorgaba en cuanto a flexibilidad y tiempo de respuesta era inmejorable.

Como ya se ha mencionado, dentro de los procesos especiales, se lleva a cabo la generación de una tabla que mantiene las bitácoras de carga del sistema diariamente (véase Tabla 5.3). Esto fue con el fin de tener una referencia, al mismo tiempo que un control, sobre los datos que han ingresado en el sistema. De igual forma, sirve para tener un medio con el cual determinar si el proceso de transferencia de datos concluye satisfactoriamente o si presenta algún problema. Dentro de los archivos de bitácora, se tienen 5 líneas de carga diariamente, donde se determina tanto el sistema (Cheques, Ahorros, etc.) como el tipo de información que se está ingresando (Dirección, Contratos, Clientes, etc.).

Por ejemplo, a cada producto del sistema, con su tipo de información, le corresponde un archivo de bitácora, y la suma de todos los datos referentes a los Clientes ingresados en esa fecha deben ser la totalidad de los Clientes que se ingresan al sistema.

Es por lo anterior que en el proceso de extracción de datos se generan múltiples archivos de bitácora, y el poner la información de todos ellos en una tabla del CDI, nos permite tener completo conocimiento de lo que hay en el sistema.

La tabla 5.3 muestra uno de los archivos de bitácora, en donde se tiene el resultado del número de registros referentes a contratos, clientes, direcciones y saldos, siendo los tres primeros iguales.

Sistema	Registros
CON	75
CLI	75
DIR	75
SAL	4500

Tabla 5.3 Archivo de bitácora para el sistema de Cheques.

Una vez que cada uno de los archivos generados en el proceso de extracción ha sido cargado en las tablas del modelo, se procede a verificar si esa carga ha sido realizada satisfactoriamente en UNIX, mediante el uso nuevamente, de programación *shell.*

Cuando ambas cantidades se reportan como iguales, la generada en UNISYS como la transmitida a UNIX, se asegura que el sistema tiene toda la información generada dentro del modelo.

En la figura 5.1 se muestra, como resultado de la generación de la bitácora, un reporte tomado de la tabla del modelo, en donde podemos verificar que toda la información generada durante un mes, desde los procesos de carga inicial, los procesos mensuales y los procesos diarios, la información se ha introducido correctamente al modelo de datos. El número de registros que fueron introducidos al sistema en la fecha indicada se muestra en la columna para cada archivo especificado por renglón. La primera columna presenta cargas iniciales de contratos (CON), clientes (CLI), direcciones (DIR) y saldos (SAL), mientras que las restantes son los números que representan la carga diaria.

Archivo	Lu 0720	Ma 0721	Mi 0722	Ju 0723	Vi 0724	Lu 0727	Ma 0728	Mi 0729	Ju 0730	Vi 0731	Lu 0803	Ma 0804	Mi 0805	Ju 0806	Vi 0807	Lu 0810	Ma 0811	Mi 0812	Ju 0813	Vi 0814
	INI																			
CHESALMTYXXXX.TXT	24934	24954	22850	22866	22898	22924	22948	22986	23010	23044	22050	22724	22820	22838	22840	22860	22874	22910	22922	22938
CHESALSALXXXX.TXT	10998	4008	6556	6564	6568	6574	6576	6578	6580	6586	3424	6438	6446	6454	6452	6450	6456	6452	6450	6426
AHOSALMTYXXXX.TXT	24060	24083	23889	23890	23886	23894	23878	23884	23876	23880	23881	23874	23877	23890	23888	23885	23893	23908	23921	14208
FONSALMTYXXXX.TXT	2946	2958	2970	2981	2992	3001	3009	3019	3629	3041	3051	3067	3079	3088	3094	3104	3108	3119	3129	3140
FONSALSALXXXX.TXT	998	1002	1007	1089	1012	1015	1016	1020	1021	1024	1036	1030	1032	1034	1035	1036	1037	1038	1039	1036
CHECONMTYXXXX.TXT	-	-	27	19	19	19	14	22	25	21	15	26	26	15	16	14	10	23	18	15
AHOCONMTYXXXX.TXT	24060	23	17	7	12	17	10	14	7	14	11	13	14	26	8	10	22	22	17	1
CHECONSALXXXX.TXT	-	-	7	4	6	4	2	7	2	4	4	6	3	5	2	2	5	3	6	2
AHOCLIMTYXXXX.TXT	24060	X	X	x	x	x	x	x	7	10	11	13	14	26	8	10	22	22	17	1
CHECLISALXXXX.TXT	8767	X	X	x	x	x	x	x	2	4	4	6	3	5	2	2	5	3	6	2
CHECLIMTYXXXX.TXT	23857	X	X	x	x	x	x	x	29	14	15	26	20	15	16	14	10	23	18	15
CHEDIRMTYXXXX.TXT	23857	X	X	x	x	x	x	x	25	21	15	26	20	15	16	14	10	23	18	15
CHEDIRSALXXXX.TXT	8767	X	X	x	x	x	x	x	2	4	4	6	3	5	2	2	5	3	6	2
AHODIRMTYXXXX.TXT	24060	X	X	x	x	x	x	x	7	14	11	13	14	26	0	10	22	22	17	1
CHECONINIMTY0721	23842																			
CHECONINISAL0721	8767																			
AHOCONINIMTY0721	24060																			
CHERELMTY0721	23857																			
AHORELMTY0721	24060																			
CHERELSAL0721	8767																			

(x) No se registró la cuenta.

Figura 5.1 Cantidad de Registros Mensuales, en la Carga del Archivo de Unisys.

Generación de Validaciones dentro del ambiente UNIX

Una vez que la información transmitida se encuentra dentro del directorio de recuperación de datos, tenemos la obligación de verificar que sea montada completamente en el modelo sin problema, en caso de presentarse alguna contingencia durante la generación ó transmisión de información, se pueden originar inconsistencias de mayor importancia al momento de explotar la información.

En caso de presentar problemas, en algún proceso, ya sea de generación o transmisión, se realizan procesos que limpian la información de las tablas que se afectan durante la carga diaria.

Estos procedimientos se ejecutan aun dentro del sistema automatizado, y tienen la capacidad de recuperar la información que se ha ingresado al CDI hasta antes del momento del error, además de eliminar los registros que presentan la inconsistencia en las tablas, esto permite que las mismas puedan ser cargadas nuevamente con información correcta.

Los procesos de inicialización solamente aplican para algunas de las cargas, como son: Contratos, Clientes, Direcciones, Saldos y Tesorería, no así para Inversiones, Compra Venta de Divisas, Posición, Precios Tasa Mercado o Emisiones Mercado, ya que éstos, al momento de ser cargados, se encargan de verificar que no exista alguna fecha diferente a la que se está procesando, debido a que en estos puntos la extracción de datos que se realiza es más particular y más controlable por el valor de fecha.

Los procesos de inicialización no afectan a los datos que ya han sido cargados anteriormente al modelo; sin embargo, es necesario verificar en cada carga diaria, que el reporte de registros se haya completado, ya que de no hacerlo y en caso de ocurrir algún error, éste no se corregirá posteriormente y se almacenará con la información correcta.

5.2 Realización de Pruebas que validen el Funcionamiento de la Aplicación

Una vez que dentro del modelo de datos se procede a extraer información, por medio de reportes generados vía *GQL Windows* o por aplicaciones de *Visual Basic*, se obtienen los primeros resultados propios del modelo(CDI), los cuales nos presentan información especifica, la cual es cotejada contra reportes obtenidos de la manera anterior en los sistemas de UNISYS.

Los reportes obtenidos por nuestros procesos muestran coherencia en los datos y resultados que coinciden con los obtenidos en los otros sistemas. Para fines de ejemplo, se muestran los reportes 5.1 y 5.2.

En el reporte 5.1 que es generado con el sistema anterior de Unisys, nos muestra el tipo de producto (Cheque e Inversión) con el monto, según sea de cargo o abono y su número de control, ésta información pertenece al cliente Manufacturas Kaltex, SA de CV. Nótese la estructura del reporte, aunque las operaciones se han realizado en la misma fecha, está limitado a mostrar primero todos los cargos y después todos los abonos, y para el ejecutivo que desea conocer los movimientos diarios de la cuenta, se le dificulta cotejar los movimientos de un día de operación.

El reporte 5.2 es presentado por el CDI, el cual muestra una estructura más eficiente para presentar la información del cliente, además de incluir saldos totales e información adicional como es la cobertura crediticia, la cual no era nada sencillo obtenerla con el sistema anterior, las columnas de cargos y abonos pueden ser fácilmente cotejadas como operaciones diarias, y al pie del reporte aparece el nombre o las iniciales del usuario que hizo la consulta.

Si se observan las cifras presentadas en ambos reportes, son iguales, esto nos da seguridad en cuanto a la consistencia que mantiene la información, desde su extracción hasta la explotación de la misma.

BANPAIS, INSTITUCION DE BANCA MULTIPLE

A9 JOB DATE 17 MAY 98
SPOOL 01 PAG 001
CLIENTE: MANUFACTURAS KALTEX, SA DE CV

Operación	No. Control	Cargos
CHEQUE ...	76107	(1,824)
CHEQUE ...	75797	(2,098)
CHEQUE ...	76068	(2,185)
CHEQUE ...	76133	(2,300)
CHEQUE ...	75239	(2,639)
CHEQUE ...	76078	(2,789)
CHEQUE ...	76160	11,195
CHEQUE ...	76056	11,200
CHEQUE ...	76162	12,967
CHEQUE ...	566	18,685
CHEQUE ...	76138	21,025
CHEQUE ...	76043	24,150
CHEQUE ...	76144	26,564

BANPAIS, INSTITUCION DE BANCA MULTIPLE

A9 JOB DATE 17 MAY 98
SPOOL 02 PAG 001
CLIENTE: MANUFACTURAS KALTEX, SA DE CV

INVERSION	75545	3,123
INVERSION	76089	3,423
INVERSION	76077	4,847
INVERSION	76146	5,010
INVERSION	76065	7,613
INVERSION	76102	11,033
INVERSION	75982	(27,000)
INVERSION	76062	(30,350)
INVERSION	76069	(39,380)
INVERSION	76126	44,074
INVERSION	76076	53,130

Reporte 5.1. Explotación de datos de Cliente por Unisys.

BANPAIS
INSTITUCIÓN DE BANCA MÚLTIPLE

20/12/98

CENTRO DE INFORMACION

Número de Cliente: 001104983540
Nombre: MANUFACTURAS KALTEX, SA DE CV

Saldo Disponible:	$ 23,129,152.82	Cobertura Crediticia:	$ 3,000,000.00		
Saldo Buen Fin:	$ -	Pendiente de Abonar:	$ 9,209,504.62		

Fecha	Ref	Operación	Cargos	Abonos
17-Mar		CHEQUE ...	$ (1,824.00)	
17-Mar	566	CHEQUE ...	$ (2,097.60)	
17-Mar	76133	CHEQUE ...	$ (2,185.00)	
17-Mar	6	CHEQUE ...	$ (2,300.00)	
17-Mar	87	CHEQUE ...	$ (2,639.25)	
17-Mar	1681	CHEQUE ...	$ (2,788.75)	
17-Mar	564	INVERSION		$ 3,123.40
17-Mar	562	INVERSION		$ 3,423.00
17-Mar	76221	INVERSION		$ 4,846.68
17-Mar	75982	INVERSION		$ 5,010.32
17-Mar	177471	INVERSION		$ 7,613.07
17-Mar	222359	INVERSION		$ 11,032.69
17-Mar	98358	CHEQUE ...		$ 11,195.06
17-Mar	150504	CHEQUE ...		$ 11,199.93
17-Mar	625006	CHEQUE ...		$ 12,966.77
17-Mar	76173	CHEQUE ...	$ 287.50	$ 18,684.60
17-Mar	674193	CHEQUE ...	$ 28.75	$ 21,024.89
17-Mar	333333	CHEQUE ...		$ 24,150.00
17-Mar	333333	CHEQUE ...		$ 26,563.71
17-Mar	473299	INVERSION	$ (27,000.00)	
17-Mar	333333	INVERSION	$ (30,350.34)	
17-Mar	333333	INVERSION	$ (39,380.00)	
17-Mar	697069	INVERSION		$ 44,074.00
17-Mar	697070	INVERSION		$ 53,130.00

Usuario: E.L.C. Hoja No. 1

Reporte 5.2. Explotación de datos de Cliente por CDI.

5.3 Generación de Manuales de Usuario y Capacitación

Una vez que el producto se ha terminado, se entrega a los usuarios la documentación necesaria para que éstos puedan operar los procedimientos que conforman el producto sin ninguna complicación.

Esta documentación se elabora de acuerdo al tipo de procedimiento que se está describiendo, y al tipo de usuario que va a operar dicho procedimiento, en ambos casos la información debe ser clara y entendible.

Actualmente la mayoría de las aplicaciones elaboradas para usuarios que trabajan en ambiente *Windows* no cuentan con lo que propiamente se llama un manual de usuario, en su lugar cuentan con lo que se conoce como *Ayuda en Línea*, la cual hace referencia a un archivo de extensión ".hlp" (archivo de ayuda o *help*).

Este tipo de archivos se desarrollan con *software* generador de HiperTexto (Robohelp), el cual se reconoce fácilmente por su característica más común: un texto subrayado con color verde, con lo cual se le indica al usuario que si presiona el *mouse* en dicho texto, aparecerá una definición, un vídeo, algún sonido, una gráfica, etc.

Estos archivos por lo general son consultados mediante el menú *Ayuda* (en otras aplicaciones suele aparecer como el signo ?) de la aplicación y la mayoría de ellos muestra gráficas e imágenes exactamente iguales a la aplicación que se está explicando, de tal manera que el usuario tenga la confianza de conocer la operación del sistema con ejemplos gráficos, lo cual disminuye el tiempo de capacitación considerablemente.

Los archivos *help* al igual que los manuales, deben explicar de forma clara y mediante ejemplos, el funcionamiento del tema consultado por el usuario, ya que un archivo mal diseñado puede confundir al usuario y hacer que se pierda en el contenido de la ayuda de su aplicación.

Por otra parte, para aplicaciones que no se encuentran en ambiente *Windows*, los manuales de usuario suelen ser en extremo útiles, ya que por carecer de una *interface* amigable, deben conformarse con un documento que los guíe en su trabajo.

Para ejemplificar este tipo de manuales, a continuación se mostrará el documento que se entregó a los operadores del *Mainframe* de BANPAIS para que llevar a cabo el proceso de Extracción de Datos para un sistema de Cheques.

Manual Técnico para carga de cheques

El manual técnico se presenta en un formato de fichas, cada una de las cuales consta de 5 partes principales: Objetivo, Diagrama de procesos, Matriz de pruebas, Información técnica relevante y un ejemplo de ejecución.

Como parte inicial vemos un encabezado, donde se definen de manera rápida el sistema, el módulo, el lenguaje del proceso y el nombre del programa.

1. **Objetivo**, donde se explica brevemente la función del programa.

2. **Diagrama de flujo de procesos**, que consiste de 4 apartados:

   2.1 **Descripción de proceso y una tabla** que resume el sistema, el lenguaje, el módulo, el nombre del programa, las entidades (productos), la base de datos y la proyección (entrada y salida del proceso).
   2.2 **Simbología**, que explica cada icono usado en la plantilla.
   2.3 **Plantilla**, la cual explica el flujo del proceso de manera gráfica.
   2.4 **Descripción del proceso**, que paso a paso nos define el desarrollo del proceso.

3. **Matriz de pruebas**, donde se le da al operador o usuario final, un muestreo, precisamente realizado por nosotros, donde se definen las posibles entradas más representativas y las salidas ocurridas de cada entrada.

4. **Información técnica relevante**, que especifica las condiciones o requerimientos indispensables para la ejecución del proceso.

5 **Ejecución**, es un ejemplo de como se ejecutaría el proceso de manera manual y aislada.

En las siguientes páginas se muestra como ejemplo, el contenido de una ficha técnica (manual de usuario).

El operador localiza el encabezado de la ficha técnica como sigue:

SISTEMA	Centro de Información Extracción de Datos	**MÓDULO**	Carga cheques
LENGUAJE	COBOL	**PROGRAMA**	CDI/CHE/J/CARDIA/020

1. Objetivo:

Obtiene los datos a partir de la base de cheques, con base en éstos se construirán cuatro archivos planos que serán cargados en el sistema de GQL, conteniendo datos de Contratos Nuevos, Saldos diarios, Clientes Nuevos y Direcciones con base a los parámetros dados al programa.

2. Diagrama de Flujo de Procesos

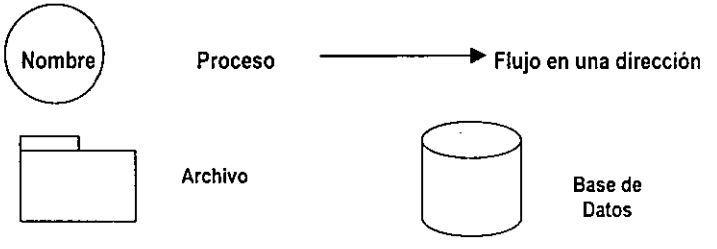
2.1 Descripción:

SISTEMA	UNISYS	**MODULO**	Limpieza cheques
LENGUAJE	COBOL	**PROGRAMA**	CDI/CHE/J/ CARDIA/020
ENTIDADES	CUENTAS	**BASE DE DATOS**	DBCHE*ZONA*
PROYECCION	CDI/*ZONA*/CHE/*MMDD*/CON CDI/*ZONA*/CHE/*MMDD*/SAL CDI/*ZONA*/CHE/*MMDD*/CLI CDI/*ZONA*/CHE/*MMDD*/DIR (UNISYS)	CHECON*ZONAMMDD*.TXT CHESAL*ZONAMMDD*.TXT CHECLI*ZONAMMDD*.TXT CHEDIR*ZONAMMDD*.TXT (Mercurio)	

Se generan cuatro archivos planos en formato ASCII, a partir de la base de cheques del sistema Unisys, el primero de los cuales tendrá información de saldos de la base y el segundo contendrá información de contratos dados de alta desde la fecha pasada como parámetro; el tercero Clientes y el cuarto Direcciones, serán los complementos al archivo de Contratos, conteniendo los nuevos datos.

Los archivos deberán generarse, como ya se indicó, bajo dos parámetros que serán dados por el usuario, siendo el primero las 3 letras que representen a la zona y el segundo la fecha con la que el programa discriminará la información requerida por el usuario únicamente.

2.2 Simbología:

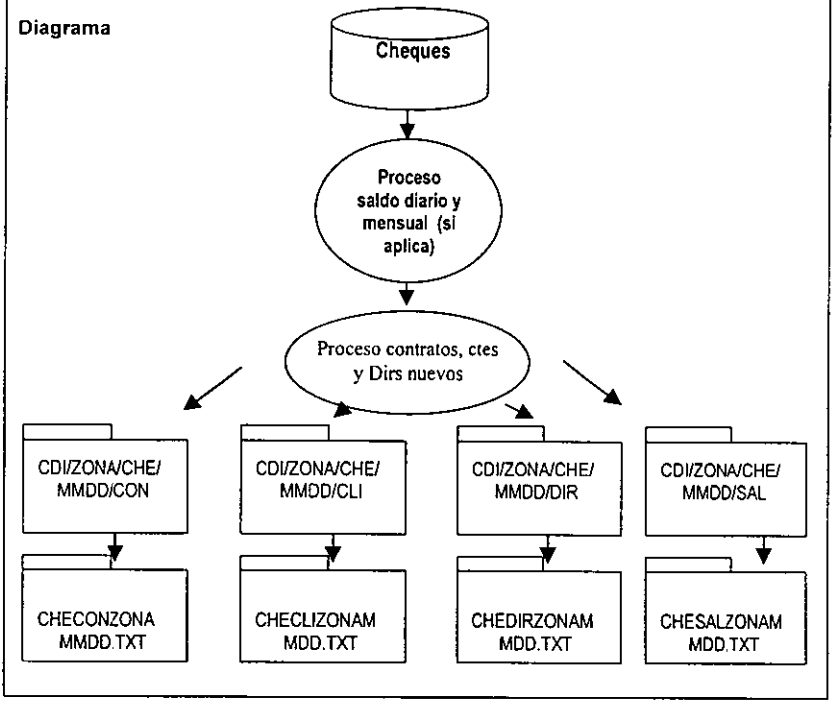


2.3 Plantilla:

Diagrama de Flujo de Proceso

Sistema	Extracción de datos
Módulo	Carga cheques
ID Programa	CDI/CHE/F/CARDIA/020

Diagrama



2.4 Descripción del Proceso:

1. Se definen las estructuras con las que crearemos los archivos planos, tanto para el archivo de contratos como para los archivos de saldos, clientes y direcciones.
2. Proceso: Se ejecuta el programa CDI/CHE/J/CARDIA/020 ("zon","aammdd"), el proceso preguntará si se procesa información mensual, en caso de ser así se activará una bandera que generará dos saldos adicionales, que posteriormente se definirán.
3. Arranca el proceso de saldos, se escribirán en registro los datos *cta-oficina, cta-consecutivo, cta-saldo,* obtenidos a partir de la base de datos; *fecha-saldo*, se tomará a partir de la fecha parámetro de la ejecución y los campos *siglo* y *cve-saldo* (igual a "1") se escribirán como estándares. Sólo se escribirá el registro a la salida *si cta-inactiva* es diferente de 4.
4. Con el mismo registro que fue armado en el paso anterior, se le moverá un "2" y se obtendrá el saldo de **T+1,** sumando algebraicamente los depósitos pendientes y los cargos pendientes y de igual forma se escribirá a la salida únicamente si *cta-inactiva* no es igual a 4.
   * En caso que se haya contestado "S" al solicitar el programa para saber si se define información mensual, el proceso generará los saldos mensuales Promedio y de Fin de Mes; a este último se le pondrá la fecha de fin de mes y se escribirá en el archivo de saldos, mientras que para obtener el saldo promedio se pasará al campo el saldo con ese dato y de igual manera tendrá la fecha de fin de mes.
5. Arranca el proceso de contratos, donde los campos cta-oficina, cta-*consecutivo* y *cta-fech-apertura (cta-oficina* también para *no-sucursal*) se toman de la base de datos y se escriben en el registro para llenar número de contrato, participa, y número de sucursal, mientras que *siglo, producto, sproducto, ssproducto, sssproducto, fecha-cancel, no-contrato-alborada, no-ejecutivo, tipo-banca, sector, sector-eco, giro-encaje, moneda y actividad* se toman como parámentros de programa (ver especificación del usuario). El registro se escribe a la salida solamente si la fecha de apertura es mayor o igual a la fecha de parámetro de la ejecución además de que el campo cuenta-inactiva sea diferente de 4.
6. Enseguida se llena el registro de clientes, donde *cta-oficina, cta-consecutivo, cta-título, cta-nombre, cta-apellidos, cta-fecha-nac, cta-rfc, cta-sector-contable y cta-oficina* (para *oficina-registro*) se toman de la base de datos, mientras que *personalidad, codigo-titulo, nombre-comercial, apellido materno, apellido de casadasexo, estado-civil, nacionalidad codigo-pais-nacimiento, siglo, curp, banca, cod-act-gen, cod-act-esp, apoderado-legal, ocupacion, administrador, acta-constitutiva, fecha-proceso, no-ejecutivo, no-cliente-alb y estatus* son

parámetros determinados por el programa. El registro se escribe a la salida solamente si la fecha parámetro del programa es menor que la fecha de apertura y que la cta esté activa. Nota: La fecha de nacimiento pasa por un proceso de validación.

7. Por último se llena el registro de contratos, *donde cta-oficina, cta-consecutivo,cta-direccion, cta-poblacion, cta-codigo-posta, cta-num-telefonico* se toman de la base de datos, *no-de domicilio, numeros interior y exterior, colonia, municipio-delegacion estado, codigo-estado, codigo-pais apartado-postal, telefono2, e-mail, fecha_registro y fecha-modificacion* son completados por el programa. De igual forma el registro se escribe si y sólo si la fecha de apertura es mayor o igual que la fecha enviada como parámetro al programa.
8. Después de haber generado todos los archivos en Unisys, el programa los copiará en el servidor Mercurio (vía FTP) en el directorio /home/clientes/archivos con los siguientes nombres: CHECON*ZONAMMDD*.TXT, CHEDIR*ZONAMMDD*.TXT, CHECLI*ZONAMMDD*.TXT, CHESAL*ZONAMMDD*.TXT.

3. Matriz de Pruebas:

Sistema	**Centro de Información**
Módulo	**Carga cheques**
Programa	**CDI/CHE/F/CARDIA/020**

Definición de Entradas

ID	Nombre	Tabla Asociada	Tipo de Dato
1	saldo diario y T+1 cta-activa=4	Dbchemty Dbchesal	char *
2	saldo diario y T+1 cta-activa<>4	Dbchemty Dbchesal	char *
3	contrato cta-apertura >= fecha-parametro y cta-activa=4	Dbchemty Dbchesal	char *
4	contrato cta-apertura < fecha-parametro y cta-activa=4	Dbchemty Dbchesal	char *
5	contrato cta-apertura >= fecha-parametro y cta-activa<>4	Dbchemty Dbchesal	char *
6	contrato cta-apertura < fecha-parametro y cta-activa<>4	Dbchemty Dbchesal	char *
7	contrato cta-apertura < fecha-parametro y cta-activa<>4 y bandera fin de mes	Dbchemty Dbchesal	char *

Definición de Salidas

ID	Nombre	Tabla Asociada
1	Aceptado	CDI/*ZONA*/ CHE/*MMDD*//SAL
2	Rechazado	CDI/*ZONA*/ CHE/*MMDD*//SAL
3	Aceptado	CDI/*ZONA*/ CHE/*MMDD*//CO N (CLI,DIR)
4	Rechazado	CDI/*ZONA*/ CHE/*MMDD*//CO N (CLI,DIR)
5	Rechazado	CDI/*ZONA*/ CHE/*MMDD*//CO N (CLI,DIR)
6	Rechazado	CDI/*ZONA*/ CHE/*MMDD*//CO N (CLI,DIR)
7	Aceptado	CDI/*ZONA*/ CHE/*MMDD*//SAL

Valor de Entradas			Valor de Salidas		
ID	Caso Dato	Valor / Comentario	ID	Valor / Comentario	RF
1	✓	saldo diario y T+1 cta-inactiva=4	1	S	✓
2	✓	saldo diario y T+1 cta-activa<>4	2	N	✓
3	✓	contrato cta-apertura >= fecha-parametro y cta-activa=4	3	S	✓
4	✓	contrato cta-apertura < fecha-parametro y cta-activa=4	4	N	✓
5	✓	contrato cta-apertura >= fecha-parametro y cta-activa<>4	5	N	✓
6	✓	contrato cta-apertura < fecha-parametro y cta-activa<>4	6	N	✓
7	✓	contrato cta-apertura < fecha-parametro y cta-activa<>4 y bandera fin mes	6	S (se generan además los saldos mensuales)	✓

4. Información Técnica Relevante:

Los archivos necesarios para la ejecución de este programa son los siguientes:

CDI/CHE/F/CARDIA/020	fuente
CDI/CHE/O/CARDIA/020	objeto
CDI/CHE/J/CARDIA/020	job ejecutable

5. Ejemplo de ejecución:

?SO MESS para desplegar mensajes del programa

para ejecutar el programa
ST CDI/CHE/F/CARDIA/020(**"MTY","970415"**)

Accept: Se va a procesar mensual (S/N) ?
pregunta el programa
?#### AX S
para procesar mensual
?#### AX N
para procesar diario

Para el caso de los operadores de la aplicación desarrollada en *Visual Basic*, se elaboró un archivo *help* con las características mencionadas anteriormente, y debido a que se emplearon las mismas gráficas descritas en el apartado de *Pantallas* en el capítulo 4, se mostrará a continuación los pasos a seguir para activar el archivo de ayuda y realizar una consulta para un Cliente de tipo Grupo (varias empresas).

Ayuda en Línea para el Usuario

1.- Estando dentro de la aplicación, se puede activar el archivo *help de 2 maneras diferentes:*

a) Activando con el *mouse* el menú de *Ayuda.*
b) Oprimiendo simultáneamente las teclas "Alt" + "y".

Para efectos de ejemplificar la ayuda, mencionamos sólo la función de dos instrucciones que aparecen en la barra de menú (ver Figura 5.2). Estas instrucciones pertenecen a la función de los botones (a) y (h), en el capítulo anterior, en el apartado de pantallas se han explicado a detalle las funciones restantes.

2.- Aparecerá una pantalla con las barras de herramientas de la aplicación tal como lo muestra la figura 5.2.

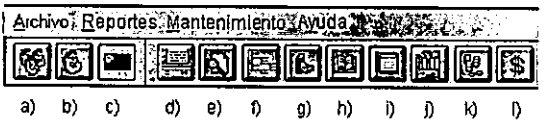


Figura 5.2. Ayuda de la Aplicación.

3.- Presionar el botón identificado con el inciso "a)" para solicitar la introducción del cliente deseado.

4.- Introducir en la pantalla que muestra la figura 5.3. cualquiera de los siguientes datos y oprimir el botón de *Buscar*:

- Código de Grupo.
- Nombre de Grupo.

Nota: Mientras más datos se introduzcan, más eficiente será la búsqueda.

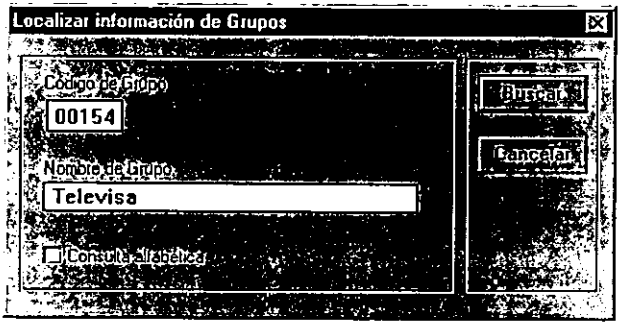


Figura 5.3. Consulta por Grupos.

5.- El resultado de la búsqueda es un listado de las empresas que conforman el Grupo solicitado, si se desea ver el tipo de cuentas con las que cuenta cualquiera de sus empresas, solamente hay que hacer un doble *click* con el *mouse* sobre la empresa deseada y el resultado se muestra en la figura 5.4.

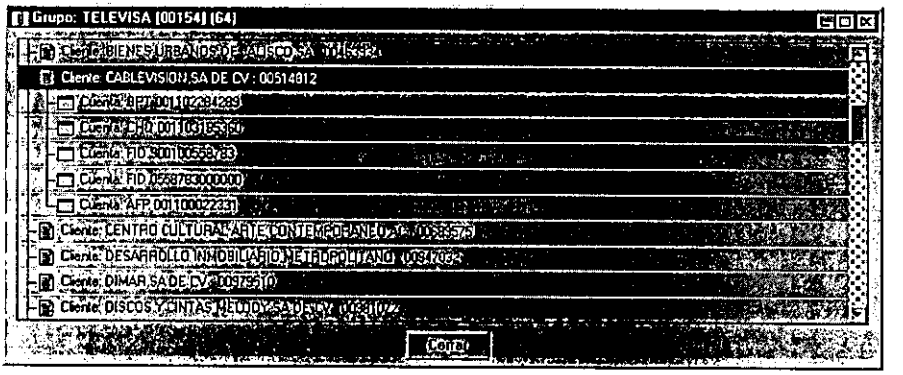


Figura 5.4. Resultado de la Consulta.

6.- Por último, si se desea tener información general del cliente que se está consultando, solamente hay que presionar el botón marcado con el inciso "h)" de la barra de herramientas mostrada en la figura 5.2.

El resultado de esta operación se muestra a continuación en la figura 5.5.

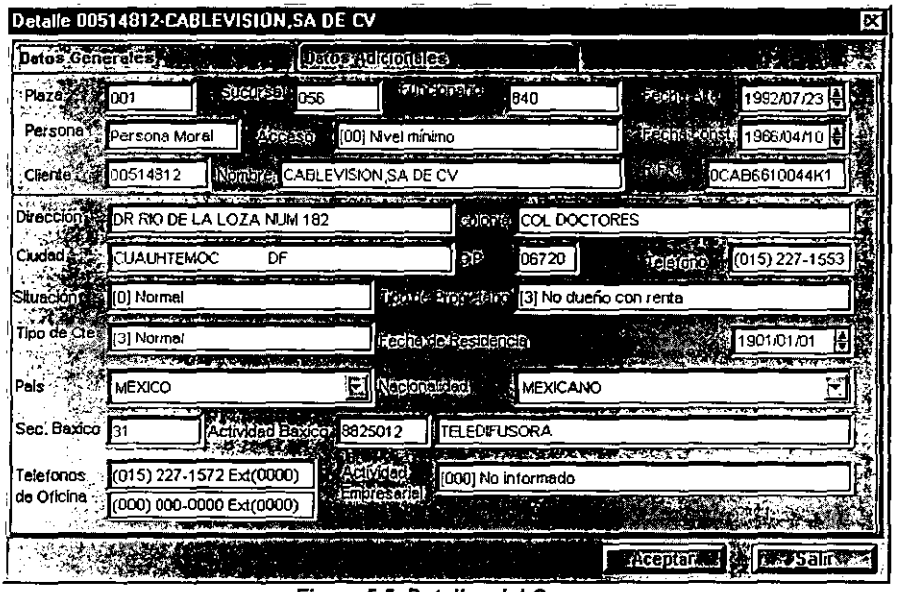


Figura 5.5. Detalles del Grupo.

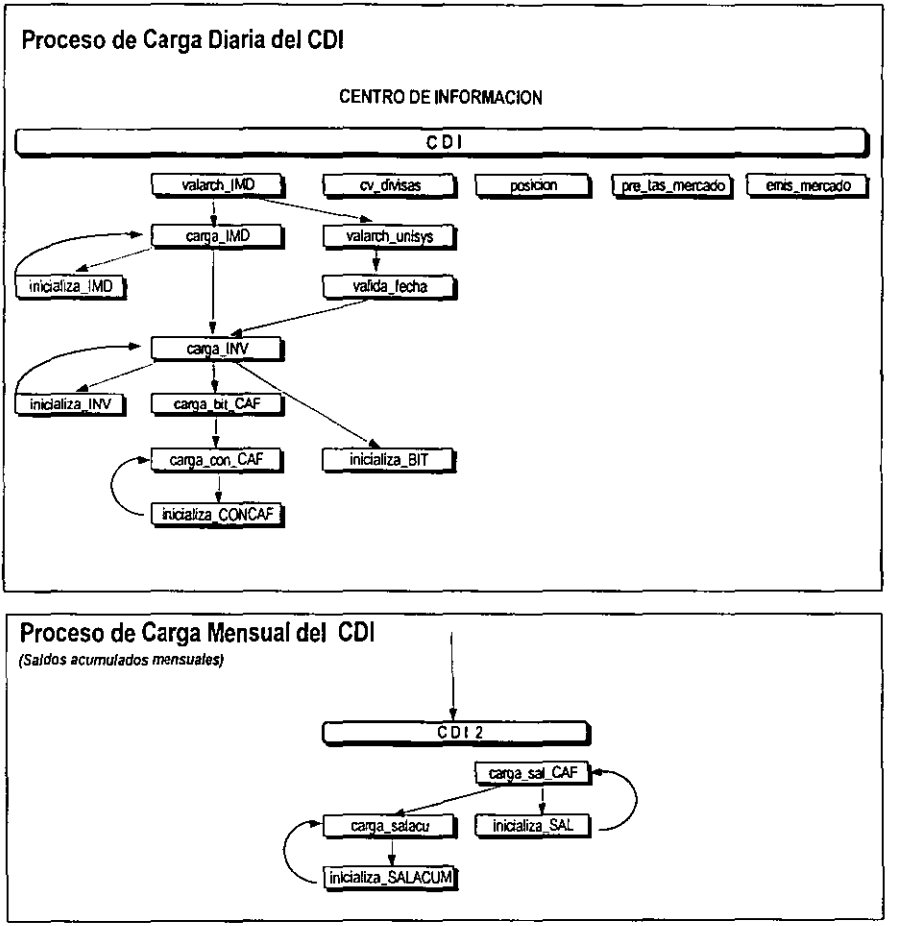
5.4 Procesos automáticos

Como parte de la automatización del CDI, nos enfocaremos a la parte en UNIX, debido a que los procesos de Extracción y Transferencia de información son llevados en UNISYS por parte del personal de operación, y estos no pueden tener una automatización como tal.

El proceso automático diario del CDI se muestra en la figura 5.6 que contiene dos figuras, las cuales se muestran dos procesos, cada uno con las funciones que realiza y la secuencia en que las lleva acabo.

El primer proceso se ejecuta a las 5 de la mañana, y lleva a cabo la carga de información de los sistemas de Inversión (Mesa de Dinero y Tradicional), incluyendo

la parte referente a Clientes, Contratos y Direcciones de los demás sistemas; mientras que la parte de carga de Saldos, se lleva a cabo a las 19 horas.



Figuras 5.6. Procesos Automáticos del CDI.

Los horarios seleccionados para la ejecución de los procesos, se determinaron de tal forma que en ese tiempo, el servidor tuviese la menor carga de trabajo posible, además de no interferir con horas críticas de operación normal, ya que son autónomos en su ejecución.

Paralelamente a la ejecución de los procesos anteriores, se inician otros procesos definidos como:

ValidaIMD, el cual se encarga de revisar cada 30 minutos que los archivos necesarios para la carga de información de Inversión Mesa de Dinero, se encuentren en los directorios especificados para de ahí tomarlos, validarlos y dar paso a dos procesos más. Estos procesos son: CargaIMD y ValidaINV.

CargaIMD, que propiamente lleva a cabo la carga de la información sobre la base de datos del modelo, por medio de programas como los mencionados en el capítulo anterior.

En caso de que se presente un contratiempo en el proceso **CargaIMD**, se ejecuta el proceso **InicializaIMD**, el cual se encarga de dejar todas las tablas posiblemente afectadas como estaban antes de iniciar con la carga y de volver a ejecutar el proceso **CargaIMD**; en caso que nuevamente se presente algún problema, se da aviso al operador de Unisys, reportando los errores en la generación del archivo que se procesa y se detiene completamente el proceso de carga del CDI para su revisión.

ValidaINV, que como el primero, se encarga de revisar que los archivos necesarios para cargar la información de Inversión Tradicional se encuentren en el directorio especificado.

Una vez que el proceso **ValidaINV** haya finalizado, se procede a ejecutar el **ValidaFecha**, que se encarga de validar que todas las fechas de los archivos de bitácora sean las mismas que la fecha del proceso, ya que pudo haberse corrido un proceso con la fecha del día anterior.

Por ejemplo, cuando han finalizado los procesos **CargaIMD** y **ValidaFecha**, se ejecuta el proceso de **CargaINV**, que llena la información referente a Inversión Tradicional. De igual forma que **CargaIMD**, éste cuenta con un proceso que en caso de presentarse algún problema con la carga, se activará y dejará en estado inicial las tablas para poder llevar a cabo el proceso de carga nuevamente, nos referimos a **InicializaINV**.

Cuando se ha realizado la carga total de la parte referente a Inversiones, se ejecuta la carga de los demás sistemas por medio de los siguientes procesos:

CargaBitácora, que propiamente se encarga de poner los archivos de bitácora dentro de las tablas de SYBASE, para poder tener un control más exacto de lo que se está cargando diariamente.

Si la carga de los archivos de bitácora presenta algún problema, se ejecuta el proceso **InicializaBitácora**, que se encarga de determinar cuales son las diferencias, y detiene el proceso, al mismo tiempo que reinicia las tablas correspondientes para la nueva carga.

CargaDatos, se dispara únicamente si el proceso anterior fue terminado satisfactoriamente y se encarga de montar la información referente a Contratos, Clientes y Direcciones de los demás sistemas en el modelo de datos. Si éste presentara algún problema, existe de igual forma el procesos **InicializaDatos**, que se encargará de dejar todas las tablas utilizadas como fueron encontradas antes de la ejecución del proceso anterior.

Al mismo tiempo que ValidaIMD, se ejecutan los siguiente procesos:

CVDivisas, que almacena la información referente a los tipos de cambio diarios.

Posición, que almacena la información referente a este rubro.

PreTasMercado, que guarda los datos referentes a precios y Tasas del Mercado por día.

EmiMercado, que sube la información referente a las Emisiones de Mercado hechas diariamente.

Estos últimos procesos no tienen necesidad de reiniciarse limpios para volverse a ejecutar, únicamente es necesario volver a ejecutar el proceso después de haber puesto los archivos necesarios en el directorio de recuperación (/home/clientes/archivos) trayéndolos del directorio de respaldos y restaurándolos con la extensión ".txt".

Una vez que se ha realizado este proceso, la carga diaria del CDI se puede dar por concluida, y la explotación de la información se verá reflejada con la información más reciente.

5.5 Liberación del Sistema

Una vez que toda la carga se ha automatizado, el proceso restante consiste en dar capacitación a los usuarios, tanto a los operadores de nivel técnico, como a los directores a nivel usuario.

La capacitación consta de cursos a usuarios finales por una semana y la capacitación a los operadores de Unisys ejecutando junto con ellos los procesos por una semana.

Es importante, como parte de este proceso, utilizar los manuales descritos anteriormente (manual técnico de usuario y la ayuda en línea), que serán de vital importancia para el personal de BANPAIS, ya que ésta será su única herramienta cuando el sistema se haya completado y la explotación tenga que hacerse de manera cotidiana.

Se llevaron a cabo pruebas de capacidad del CDI, como: Saldos de Monterrey, los cuales son grandes listados, obteniendo información por alto volumen y concurrencias de usuarios realizando varias sesiones de consulta simultáneamente.

Se hicieron pruebas de impresión, locales y en red, salvar información de archivos de texto, los cuales se podían abrir en otras aplicaciones, por ejemplo Microsoft Word o Microsoft Excel.

Se hicieron simulaciones de pérdida de información y recarga de la misma, etc.

Es importante hacer notar que el sistema se ha automatizado únicamente en Monterrey (MTY) y Saltillo (SAL), posteriormente se incorporarán los demás sistemas y se hablará de ellos en el capítulo de resultados y conclusiones.

5.6 Mantenimiento

Una vez que se ha implementado el CDI, hay varios pasos que deben tomarse en cuenta para garantizar que funcione siempre de manera efectiva y que el tiempo fuera de servicio se reduzca al mínimo. El CDI necesita de cuidados continuos para asegurar que funcione con eficiencia y sin problemas.

Entre los procesos de mantenimiento más importantes que se tienen que llevar a cabo constantemente dentro de la formación del CDI, se tienen los siguientes:

Mantenimiento preventivo. Es un conjunto de actividades que se realizan en un período de tiempo para evitar que el CDI pueda presentar problemas debido a la actividad diaria, la mayor parte de este mantenimiento es con respecto al *hardware,* y consiste principalmente en la limpieza y revisión de los diferentes componentes. Además de considerar el posible crecimiento de los equipos para adaptarse a las condiciones de trabajo futuras.

Dentro de las actividades periódicas de este tipo de mantenimiento se encuentra la realización de respaldos que contengan la información de manera diaria, en el transcurso de una semana, semanal en el transcurso de un mes y mensual por cada período anual que genere información.

Propiamente dentro de la formación del CDI se realiza la depuración de la base de datos, ésta se realiza sobre las tablas de carga con volumen importante, tales como la tabla de saldos, la cual tiene un crecimiento de miles de registros diariamente (ésta en particular deberá permanecer con la historia de un año, tentativamente, para verificar su funcionamiento y operabilidad con volúmenes de información de millones de registros).

Mantenimiento adaptativo. Este fue parte primordial de nuestro diseño, debido a que se consideró un crecimiento exponencial en algunos aspectos de la información, como el manejo de saldos, que incluye saldos diarios y mensuales por cada contrato, para todas las zonas y productos incluidos en el sistema. Así, la selección del *hardware*, no solo se hizo aprovechando recursos disponibles, sino además buscando un fácil crecimiento en cuanto a recursos físicos, como son los tamaños (en capacidad) de los discos SUN, la facilidad de instalación y seguridad de que no es necesario detener el sistema para realizar nuevas instalaciones de *hardware*.

Mantenimiento correctivo. Se refiere a la reparación o compostura de algún dispositivo o elemento dañado o al cambio de piezas deterioradas para que funcione adecuadamente.

Mantenimiento perfectivo. Es la optimización del CDI con mejores componentes para aumentar su rendimiento realizando la migración hacia nuevas tecnologías.

Dentro de este tipo de mantenimiento se han llevado a cabo rectificaciones en los procesos de carga diaria, los cuales constituyen pequeñas mejoras en la carga de información. Además se ha realizado una actualización de la información técnica y de los manuales de usuario, en los casos que ha sido necesario.

A continuación se hará la presentación de estadísticas, resultados finales y en general, resultados del producto terminado en operación a nivel nacional.

Presentación de cargas iniciales y de cargas diarias

En la figura 5.7. se muestran los datos estadísticos que se obtuvieron a partir de las tablas de Bitácora, y que reflejan en números, la cantidad de registros que han sido cargados en el sistema. En la primer columna se presentan únicamente las cargas iniciales y en las otras columnas las cargas diarias de cada archivo.

El reporte de la figura 5.7 muestra un ejemplo por producto y por zona para un rango de fechas específico, únicamente se despliega la información para dos regiones de la zona norte, Monterrey (MTY) y Saltillo (SAL), como resultado parcial y validado para autorizar iniciar la marcha del CDI a nivel nacional.

En la figura 5.8, se muestra el complemento de la estadística de carga para las zonas restantes, como parte de la implementación a nivel nacional del CDI.

Archivo	Mi 1015	Mi 1015	Ju 1016	Vi 1017	Lu 1020	Ma 1021	Mi 1022	Ju 1023	Vi 1024	Lu 1027	Ma 1028	Mi 1029	Ju 1030	Vi 1031	Lu 1103	Ma 1104	Mi 1105	Ju 1106	Vi 1107
CHESALMTYXXXX.TXT	XXX	23408	23436	23470	23484	23494	23518	23540	23540	23516	23532	23574	23616	23620	23396	23300	23438	23450	23480
CHECONMTYXXXX.TXT	XXX	17	24	23	13	14	20	17	19	23	18	28	26	25	25	25	20	30	17
CHECLIMTYXXXX.TXT	XXX	17	24	23	13	14	20	17	19	23	18	28	26	25	25	25	20	30	17
CHEDIRMTYXXXX.TXT	XXX	17	24	23	13	14	20	17	19	23	18	28	26	25	25	25	20	30	17
FONSALMTYXXXX.TXT	XXX	3485	3496	3510	3519	3523	4530	3544	3554	3683	3692	3716	3731	3750	3610	3630	3641	3658	3670
AHOSALMTYXXXX.TXT	XXX	23714	23924	22923	23927	23937	23949	23948	23459	24010	24010	24008	24005	24005	23486	23990	23990	24002	24009
AHOCONMTYXXXX.TXT	XXX	11	16	10	16	21	20	12	19	18	6	9	7	10	17	15	9	20	11
AHOCLIMTYXXXX.TXT	XXX	11	16	10	16	21	20	12	19	18	6	9	7	10	17	15	9	20	11
AHODIRMTYXXXX.TXT	XXX	11	16	10	16	21	20	12	19	18	6	9	7	10	17	15	9	20	11
Contratos y Saldos	XXX																		
Saldos Acumulados	XXX																		
Saldos Inv. Acumulados	XXX																		
CHESALSALXXXX.TXT	XXX	6394	6490	6416	6422	6392	6398	6396	6392	6210	6218	6192	6208	6222	6240	6242	6245	6230	12464
CHECONSALXXXX.TXT	XXX	5	5	6	4	4	6	6	8	6	4	8	7	5	14	4	6	4	4
CHECLISALXXXX.TXT	XXX	5	5	6	4	4	6	6	8	6	4	8	7	5	14	4	6	4	4
CHEDIRSALXXXX.TXT	XXX	5	5	6	4	4	6	6	8	6	4	8	7	5	14	4	6	4	4
No aplica Fondos	XXX	1110	1113	1116	1119	1123	1129	1126	1128	1151	1153	1158	1165	1170	1141	1142	1147	1147	1147
Contratos y Saldos																			
Saldos Acumulados																			
Saldos Inv. Acumulados																			

Figura 5.7. Estadística de la Carga de Archivo de Unisys.

Archivo	Mi 1015	Mi 1015	Ju 1016	Vi 1017	Lu 1020	Ma 1021	Mi 1022	Ju 1023	Vi 1024	Lu 1027	Ma 1028	Mi 1029	Ju 1030	Vi 1031	Lu 1103	Ma 1104	Mi 1105	Ju 1106	Vi 1107
	INI																		
CHESALMEXXXXX.TXT	101462	37172	37200	37316	37264	37322	37316	37341	37330	37320	37318	37574	37601	75780	37182	37252	37306	37366	37406
CHECONMEXXXXX.TXT	94731	36	21	25	28	30	30	31	33	34	32	42	43	37	36	43	32	40	38
CHECLIMEXXXXX.TXT	94731	36	21	25	28	30	30	31	33	34	32	42	43	37	36	43	32	40	38
CHEDIRMEXXXXX.TXT	94731	36	21	25	28	30	30	31	33	34	32	42	43	37	36	43	32	40	38
FONSALMEXXXXX.TXT		7378	7393	7416	7436	7452	7472	7489	7780	7476	7474	7602	7738	7671	7443	7675	7699	7725	7754
AHOSALMEXXXXX.TXT	7799	7715	7724	7720	7726	7721	7722	7723	7731	7726	7724	7116	7718	7420	7772	7725	7724	7726	7727
AHOCONMEXXXXX.TXT	7799	7	15	12	10	7	8	6	6	12	10	7	8	4	8	3	4	6	6
AHOCLIMEXXXXX.TXT	7799	7	15	12	10	7	8	6	6	12	10	7	8	4	8	3	4	6	6
AHODIRMEXXXXX.TXT	1799	7	15	12	10	7	8	6	6	12	10	7	8	4	8	3	4	6	6
Contratos y Saldos																			
Saldos Acumulados																			
Saldos Inv. Acumulada																			
CHESALFOMEXXXXX.TXT	25112	19880	19918	19842	19954	19990	19974	19981	23780	19978	19976	20090	28461	40241	19996	20330	20056	20080	20136
CHECONFOMEXXXXX.TXT	12556	17	20	18	20	26	13	19	24	17	15	25	24	80	33	29	20	22	28
CHECLIFOMEXXXXX.TXT	12556	17	20	18	20	26	13	19	24	17	15	25	24	80	33	29	20	22	28
CHEDIRFOMEXXXXX.TXT	12556	17	20	18	20	26	13	19	24	17	15	25	24	80	33	29	20	22	28
FONSALFOMEXXXXX.TXT		4093	4106	4119	4133	4147	4154	4167	4174	4158	4156	4223	4244	4223	4262	4289	4300	4314	4336
AHOSALFOMEXXXXX.TXT	14319	14241	14236	14172	14178	14185	14181	14180	14183	14185	14183	14109	14110	14104	14101	14159	14189	14250	14256
AHOCONFOMEXXXXX.TXT	14319	9	6	8	12	14	7	10	6	11	9	4	10	4	10	66	39	71	17
AHOCLIFOMEXXXXX.TXT	14319	9	6	8	12	14	7	10	6	11	9	4	10	4	10	66	39	71	17
AHODIRFOMEXXXXX.TXT	14319	9	6	8	12	14	7	10	6	11	9	4	10	4	10	66	39	71	17
Contratos y Saldos																			
Saldos Acumulados																			
Saldos Inv. Acumulados																			

Figura 5.8. Estadística de la Carga de Archivo de Unisys (Continúa).

Archivo	Mi 1015	Mi 1015	Ju 1016	Vi 1017	Lu 1020	Ma 1021	Mi 1022	Ju 1023	Vi 1024	Lu 1027	Ma 1028	Mi 1029	Ju 1030	Vi 1031	Lu 1103	Ma 1104	Mi 1105	Ju 1106	Vi 1107
	INI																		
CHESALGUAXXXX.TXT	32808	14176	14186	14212	14258	14284	14302	14334	14360	14668	14674	14424	14442	2846	14296	14320	14352	14356	14372
CHECONGUAXXXX.TXT	16404	14	9	22	21	19	13	20	15	15	9	16	16	12	16	14	18	9	12
CHECLIGUAXXXX.TXT	16404	14	9	22	21	19	13	20	15	15	9	16	16	12	16	14	18	9	12
CHEDIRGUAXXXX.TXT	16404	14	9	22	21	19	13	20	15	15	9	16	16	12	16	14	18	9	12
FONSALGUAXXXX.TXT		3341	3347	3462		3398	3403	3424	3434	3694	3701	3469	3429	3484	3483	3493	3507	3512	3520
AHOSALGUAXXXX.TXT	4151	4115	4118	4120	4121	4123	4121	4127	4127	4134	4140	4124	4124	4128	4127	4130	4127	4127	4126
AHOCONGUAXXXX.TXT	4151	1	6	4	4	5	2	7	2	6	2	1	3	6	4	5	4	5	2
AHOCLIGUAXXXX.TXT	4151	1	6	4	4	5	2	7	2	6	2	1	3	6	4	5	4	5	2
AHODIRGUAXXXX.TXT	4151	1	6	4	4	5	2	7	2	6	2	1	3	6	4	5	4	5	2
Contratos y Saldos																			
Saldos Acumulados																			
Saldos Inv. Acumulados																			
CHESALHERXXXX.TXT	25112	10712	10730	10736	10720	10732	10742	10754	10764	10826	10850	10786	10810	11652	10578	10612	10638	10658	10661
CHECONHERXXXX.TXT	12556	13	12	9	8	9	11	7	5	11	10	5	13	11	7	17	13	15	8
CHECLIHERXXXX.TXT	12556	13	12	9	8	9	11	7	5	11	10	5	13	11	7	17	13	15	8
CHEDIRHERXXXX.TXT	12556	13	12	9	8	9	11	7	5	11	10	5	13	11	7	17	13	15	8
FONSALHERXXXX.TXT		1349	1353	1355		1360	1367	1372	1379	1488	1499	1380	1388	1395	1382	1391	1400	1412	1415
AHOSALHERXXXX.TXT	6430	6394	6397	6399	6403	6406	6401	6397	6403	6437	6442	6409	6411	6413	6411	6413	6416	6414	6413
AHOCONHERXXXX.TXT	6430	4	4	3	7	4	1	0	5	3	3	4	5	4	4	2	4	3	2
AHOCLIHERXXXX.TXT	6430	4	4	3	7	4	1	0	5	3	3	4	5	4	4	2	4	3	2
AHODIRHERXXXX.TXT	6430	4	4	3	7	4	1	0	5	3	3	4	5	4	4	2	4	3	2
Contratos y Saldos																			
Saldos Acumulados																			
Saldos Inv. Acumulados																			

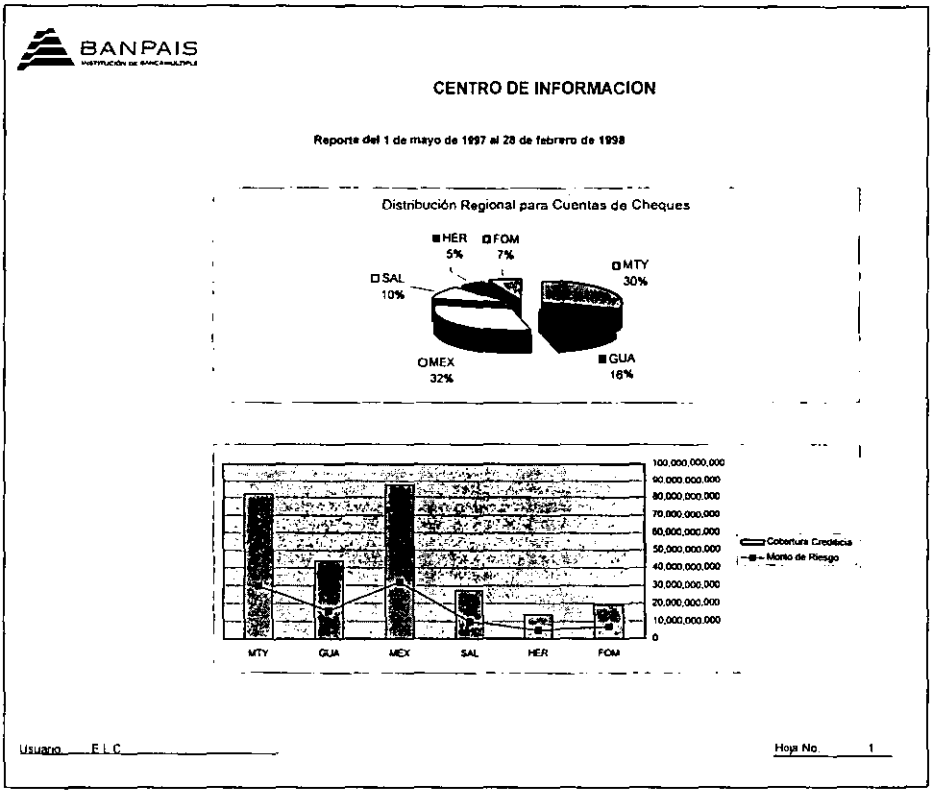
Figura 5.8. Estadística de la Carga de Archivo de Unisys.

Generación de Reportes basándose en resultados finales

Una vez que se han cargado todos los datos dentro del modelo, es posible obtener resultados globales que reflejen el comportamiento total de la institución, por ejemplo el reporte 5.3, que muestra el nivel de captación de ingresos para el sistema de cheques, dividido por regiones.

Es de suma importancia el poder hablar de estos reportes, ya que no solamente integran las áreas del banco, sino que además tiene la capacidad de conjuntar información de diferentes productos dentro del mismo reporte, además de reflejar gráficamente su comportamiento a través del tiempo, lo que nos da un panorama más claro de la situación actual, lo que es imprescindible para la toma de decisiones.

El reporte 5.3 nos muestra el comportamiento del sistema de Cheques por saldos a nivel nacional: Hermosillo, Foráneas México, Monterrey, Saltillo, Guadalajara y México, también nos muestra su cobertura crediticia y el monto de riesgo de cada una de ellas.



Reporte 5.3. Estatus del Banco (Reporte de Resultados Finales por Regiones).

Otro ejemplo se muestra en el reporte 5.4, donde se explotan los datos de cartera por grupo, en donde se despliega de cada cliente, su número, el monto de la cartera vencida, el de cartera vigente, su total de ambas y la fecha de la consulta.

BANPAIS

20/12/98

CENTRO DE INFORMACION

DATOS DE CARTERA - GRUPO EJEMPLO

Nota: Los importes de cartera se presentan en cifras naturales e incluyen moneda extranjera valorizada.

Cliente	Número de Cliente	Cartera Vencida	Cartera Vigente	Cartera Total	Fecha de Consulta	Calif. CNBV
MANUFACTURAS KALTEX, SA DE CV	CAR 001104983540	$ 0.09	$ 451,423,406.52	$ 451,423,406.61	17/05/98	A
MANUFACTURAS KALTEX, SA DE CV	CAR 001201007540	$ 383,010.57	$ 620,522,195.16	$ 620,905,205.73	17/05/98	A
TELEVISA, SA DE CV	CAR 001103892007	$ -	$ 81,673,818.97	$ 81,673,818.97	17/05/98	B
ALUMINIO CONESA, SA DE CV	CBI 005101972673	$ 2,645.00	$ -	$ 2,645.00	17/05/98	C
CIA SIDERURGICA DE GUADALAJARA, SA DE CV	CBI 005101883805	$ -	$ 389,561,038.70	$ 389,561,038.70	17/05/98	
TOTAL GRUPO		$ 385,655.66	$ 1,543,200,259.35	$ 1,543,585,915.01		

Usuario: E L C

Hoja No. 1

Reporte 5.4 . Explotación de datos de Cartera por Grupo.

Carga del sistema a escala nacional

Una vez que se ha demostrado la eficiencia de los procesos, es decir que la información pasa sin problemas desde su extracción hasta su nueva carga en el CDI, para lo cual nos apoyamos en los archivos de bitácora y el funcionamiento de los reportes, se autoriza proceder a implementar las funciones del CDI a nivel nacional.

En el siguiente capítulo se mostrarán más de los aspectos que obtuvimos como resultados del proyecto, las conclusiones personales, así como el valor agregado del mismo.

CAPITULO 6

RESULTADOS Y CONCLUSIONES

En este capítulo se comentará sobre las experiencias personales que nos dejó este trabajo, así como de los objetivos cumplidos y del valor agregado que este proyecto puso sobre los planes iniciales.

6.1 El CDI como base para la migración de Información a BANORTE

Es importante mencionar que durante el desarrollo del proyecto se llevó a cabo la venta de BANPAIS a la también regiomontana institución bancaria BANORTE, a quien se le presentó el CDI como un medio organizado que permitiría hacer la migración de la información de los sistemas incluidos en el modelo a los sistemas propios del nuevo dueño debido a las siguientes características:

- El CDI provee una fuente de información confiable, oportuna, veraz y eficaz que es producto de la extracción de datos de varios sistemas del banco.

- Cuenta con los datos necesarios para obtener reportes de manera diaria con información fresca, y al mismo tiempo con una historia que antes no se podía reportar de manera ágil.

- Es el principio de la estandarización de los sistemas del banco en un solo esquema, lo cual posiblemente, madurando el proyecto, pudiera dar origen al *DataWare Housing* del Banco, del cual carece y es de gran importancia.

- La principal ventaja del CDI dentro de los sistemas anteriores del banco es que es el único producto en el que se han agrupado los datos de las diferentes cuentas y explotado la información a nivel cliente.

- Es el soporte informático de algunos de los sistemas del banco a nivel nacional.

- El CDI sirve para dar un esquema de la situación actual del banco, así como una historia del comportamiento de los sistemas incluidos en él.

- Tiene la capacidad de ser explotado por herramientas de fácil uso tales como *GQLWindows* o por lenguajes más completos como *Visual Basic* que permiten realizar aplicaciones más específicas.

- Puede incrementarse tantas veces como sea necesario.

Cabe mencionar que aunque no se solicito presentar un estudio de **Análisis Costo Beneficio** a BANPAIS, se realizó como una prueba del beneficio del sistema desarrollado, con el propósito de mostrar a los usuarios del nuevo sistema, así como al personal de la organización que realizará la inversión, que los beneficios que se esperan obtener superan a los costos estimados durante el desarrollo del sistema. (para detalles ver apéndice D "Análisis Costo-Beneficio").

Para BANPAIS no fue necesario el recorte ni la contratación de personal debido a que el sistema anterior seguía corriendo paralelamente y sin interferir con el nuevo sistema, y en beneficio a la operación, se contó con una aplicación amigable que permitía desarrollar consultas en línea sobre la situación actual del banco, lo que resultaba sumamente difícil con el sistema anterior, debido a la complejidad de operación del mismo, ya que dependían del personal a cargo del *mainframe* para poder hacer extractos de la información.

Con el nuevo sistema se pueden realizar reportes, cuya información permite a los funcionarios y ejecutivos tomar decisiones estratégicas que conduzcan a la operación bancaria de acuerdo a las exigencias de la competencia en el mercado financiero.

Como parte de valor agregado, debe mencionarse que el CDI está diseñado para soportar sin problemas la situación del año 2000. Esto es otra ventaja sobre la mayoría de los sistemas implementados en el banco, ya que muchos de ellos aún trabajan en sistemas *Mainframe*, y no tiene la capacidad, por sí mismos y hasta este momento, de obtener resultados del nuevo milenio sin causar conflicto con los actuales.

Previendo la posibilidad de que el CDI se convierta en el *DataWare Housing* de la institución, dejamos varios aspectos abiertos, tal como el no cerrar definitivamente el modelo y siempre pensar en la inclusión de nuevos módulos al mismo, haciendo que nuestro diseño tenga amplias posibilidades de crecimiento y expansión. Es por esto que nuestro sistema es un fuerte candidato a la migración o incorporación de sistemas alternos para trabajar sin problema dentro del siguiente milenio.

Como conclusiones personales podemos mencionar que es de gran satisfacción el poder aportar nuestros conocimientos para crear un sistema de software que permita resolver un problema real en una situación real. El cual satisface los siguientes aspectos:

- Cumplir con los requerimientos del usuario.
- Comprobar su eficiencia en los momentos críticos.
- Entregarlo en el tiempo estimado.
- Que su vida útil sea de un amplio periodo.
- Que no sobrepase el presupuesto asignado a su elaboración.
- Presentar un valor agregado al usuario.

En la realización del presente trabajo logramos observar que la elaboración de un producto de software está considerado como un producto de ingeniería, con igual valor y utilidad como cualquier otro proyecto en el que se requiera la habilidad de transformar materia prima en un producto tangible.

Concluimos también que, como ingenieros, tenemos la capacidad de observar un problema con un panorama más amplio y con una visión objetiva del mismo, es decir, no estamos limitados a la utilización de una herramienta determinada, al contrario, podemos emplear a nuestro favor las condiciones del entorno para realizar un plan de acción que permita cubrir con satisfacción las necesidades de dicho problema.

Como ingenieros en computación, tenemos la obligación de mantenernos actualizados día a día en cuanto a nuevas tecnologías, debido a que nuestro medio avanza a grandes pasos y que los problemas a los que nos enfrentamos requieren ser resueltos con una calidad y excelencia que permita seguir adelante con nuestro trabajo diario.

6.2 Análisis de los Beneficios

Beneficios Tácticos

Un beneficio táctico es aquel que permite que la organización continúe realizando la misma actividad pero a menor costo (o con mayor ganancia). Suelen asociarse con reducciones en el personal administrativo o de oficina. Otro ejemplo es el ahorro que resulta de poder procesar transacciones de negocios más rápidamente.

Para BANPAIS no fue necesario el recorte ni la contratación de personal debido a que el sistema anterior seguía corriendo paralelamente y sin interferir con el nuevo sistema, y en beneficio a la operación , se contó con una aplicación amigable que permitía desarrollar consultas en línea sobre la situación actual del banco, lo que resultaba sumamente difícil con el sistema anterior debido a la complejidad de operación del mismo.

Beneficios estratégicos del nuevo sistema

Un beneficio estratégico es el que permite comenzar a realizar un tipo de negocio totalmente nuevo, o a hacerlo en un área totalmente nueva. Los beneficios estratégicos permiten a la organización realizar operaciones que le serían imposibles con el sistema actual; para BANPAIS particularmente se pueden mencionar los siguientes beneficios:

- No depender directamente del personal a cargo del *mainframe* para poder realizar extractos de la información de la base de datos de los sistemas de Cartera, Ahorro, Inversión, etc.

- Contar con la información mencionada anteriormente en un tiempo de récord, lo cual resultaba imposible con el sistema anterior.

- Elaborar reportes en base a las consultas realizadas con el nuevo sistema, cuya información permita a los funcionarios y ejecutivos tomar decisiones estratégicas que conduzcan la operación bancaria de acuerdo a las exigencias de la competencia en el mercado financiero.

- Como Beneficio adicional y sin costo alguno para el Banco, el nuevo sistema se desarrollo para poder funcionar sin ningún problema durante la transición del año 1999 al año 2000.

En la tabla 6.1 se aprecia una síntesis de lo señalado anteriormente con el fin de hacer un poco más comprensible los beneficios del nuevo sistema contra la inversión realizada en el desarrollo del mismo.

Costo		Beneficio
Concepto	**Inversión**	
Salarios Totales.	$ 252,000	Dentro del Presupuesto del Banco.
Viajes y Viáticos.	$ 18,000	Contacto directo con usuarios encargados de la información.
Capacitación.	$ 0	Incluida en Salarios.
Contratación de Personal.	$ 0	No necesario para operar el Nuevo Sistema.
Equipo de desarrollo e instalaciones.	$ 0	No necesario pues se aprovecharon recursos del Banco.
Hardware y *Software* para la operación del sistema.	$ 0	No necesario pues se aprovecharon recursos del Banco (Equipo y Licencias Corporativas)
Inversión Total	$ 270,000	• Consultas de Información en línea. • Conocimiento de la situación de la Banca. • Facilidad para la Elaboración de Reportes y Tomas de decisiones .

Tabla 6.1 Análisis Costo Beneficio.

Apéndice A
Metodologías de Programación

Metodología sobre Programación Orientada a Objetos

El modelo y diseño orientado a objetos es una nueva forma de pensar en problemas usando modelos organizados en conceptos del mundo real. La construcción fundamental es el objeto, que combina tanto los datos como el comportamiento de una solo entidad.

Para la representación de conceptos orientados a objetos se utiliza una metodología de desarrollo orientada a objetos así como una notación gráfica. La metodología consiste en construir un modelo de aplicación y después agregar detalles de implementación durante el diseño de sistema. A esto se llama OMT (*Object Modeling Technique*, por James Rumbaugh). La cual consta de las siguientes etapas:

- **Análisis**. Es una abstracción concisa y precisa de lo que un sistema deseado debe hace y la forma en que debe realizarlo.

- **Diseño del sistema.** En esta etapa el objetivo del sistema se organiza en susbsistemas basados en el análisis de la estructura y la arquitectura propuesta.

- **Diseñador de objetos**. El objetivo del diseño de objeto son las estructuras de datos y algoritmos requeridos para implementar cada clase.

- **Implementación**. Los objetos clases y las relaciones desarrolladas durante el diseño de objetos finalmente se traducen a un lenguaje de programación particular o base de datos.

La metodología OMT usa tres tipos de modelos para describir un sistema: el modelo de objetos, el modelo dinámico o orientado a eventos y el modelo funcional o estructural. Una descripción completa del sistema requiere los tres modelos.

- **El modelo de objetos** describe la estructura estática de los objetos en un sistema y sus relaciones. El modelo de objetos contiene diagramas de objetos. Un diagrama de objetos es un gráfico cuyos nodos son objetos clases y cuyos arcos son relaciones entre clases.

- **El modelo dinámico y orientado a eventos** describe aspectos del sistema que cambian en el tiempo. Este modelo se usa para especificar e implementar aspectos de control del sistema, por lo que contiene diagramas de estado. Un diagrama de estado es un gráfico cuyos nodos son estados y cuyos arcos son transiciones entre estos estados causados por eventos.

- **El modelo funcional o estructural** describe las transformaciones de los datos en el sistema. El modelo funcional contiene un diagrama de flujo de datos. Un diagrama de flujo de datos representa un cálculo. Un diagrama de flujo de datos es un gráfico cuyos nodos son procesos y cuyos arcos son flujos de datos.

En este trabajo hacemos énfasis en el modelo de objeto, que es el más fundamental, ya que los conceptos de orientación a objetos: identidad, clasificación, polimorfismo, herencia, etc. se aplican a través de todo el ciclo de desarrollo.

Ya que la metodología OMT de Rumbaugh, se refiere a tres tipos de modelos, mencionaremos conceptos de Booch, y Yourdon, los cuales están relacionados el desarrollo de CDI.

Significado de "orientado a objetos"

Quiere decir que organizamos el software como una colección de entidades (objetos) discretos que contienen tanto estructuras de datos como comportamiento.

Las características de los objetos son:

- Identidad: los datos están cuantificados en entidades discretas y distinguibles denominadas objetos.

- Clasificación: los objetos con la misma estructura de datos (atributos) y comportamiento (operaciones) forman un tipo o clase de objetos. Es una abstracción que describe propiedades importantes para una aplicación.

- Instancias: objeto particular de una clase. Posee su propia identidad y su propio valor para cada uno de los atributos pero comparte los nombres de los atributos y las operaciones con las demás instancias de la clase.

- Polimorfismo: una mima operación puede comportarse de modo distinto en distintas clases.

- Herencia: compartir atributos y operaciones entre clases tomando como base una relación jerárquica.

Las técnicas orientadas a objetos mejoran la capacidad del profesional de la computación en varios aspectos. Algunas características de la Tecnología Orientada a Objetos (OOT) incluyen:

- Reutilización
- Estabilidad
- El diseñador piensa en términos del comportamiento de objetos y no en detalles de bajo nivel.
- Se construyen clases más complejas.
- Confiabilidad.
- Integridad.
- Mantenimiento más sencillo.
- Mejor comunicación entre profesionales de los sistemas de información y los empresarios.
- Independencia del diseño.
- Computación cliente-servidor.
- Computación de distribución masiva.
- Mejores herramientas CASE. Ingeniería de software asistido por computadora.
- Bibliotecas de clases para las industrias.
- Bibliotecas de clases para las empresas.

Dentro de la notación gráfica se utilizan varios diagramas durante el análisis de requerimientos y el diseño:

El diagrama de Flujo de Datos nos dice:

- Recursos de datos del sistema.
- Flujos de datos en el sistema.
- Funciones las cuales transforman datos en el sistema.
- Funciones las cuales causan transacciones de datos en el sistema.

El diagrama de Función nos dice:

- Funciones en el sistema.
- Secuencia y desempeño de la función.

Un ejemplo de un diagrama de función se presenta en la figura 4.1, en ella se puede distinguir la aplicación de las sentencias principales.

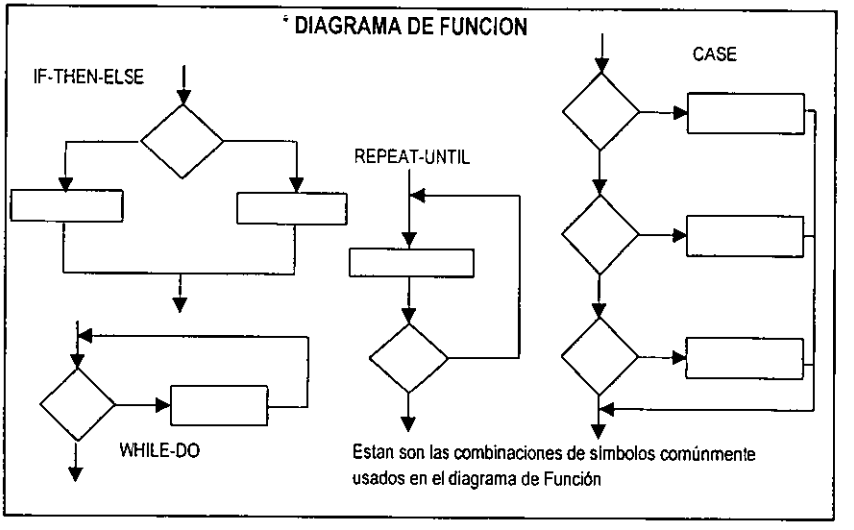


Figura A.1. Diagrama de Función.

El diagrama de Estado de Transición (DET) nos dice:

- Estado del sistema.
- Relación entre los estados en el sistema.
- Eventos que causan la transición de estados en el sistema.
- Desempeño de los resultados de acción en respuesta a los eventos.

El diagrama Entidad-Relación (DER) nos dice:

- Entidades del sistema.
- Relación entre estas entidades.

Los diagramas de Interacción de Objetos nos dice:

- Objetos y clases en el sistema
- Relación entre objetos
- Interface de objetos
- Flujo de datos entre objetos
- Método llamado
- Secuencia de llamadas (opcional)

El diagrama de Booch nos dice:

- Dependencia relacionada entre las clases

Definición de Objeto

Un objeto es una entidad integral que puede:

- Cambiar de estado
- Se comporta en ciertos caminos discernibles
- Ser manipulado por varias formas de estímulos
- Están en relación con otros objetos

Objetos:

- Existen, ocupan un espacio y asumen un estado
- Tienen atributos
- Exhiben conductas

La definición formal de un objeto desde la perspectiva del Diseño Orientado a Objetos (DOO) es: **Objeto** es cualquier entidad que tiene estado, comportamiento e identidad; la estructura y comportamiento de objetos similares se definen en la clase que tienen en común; los términos instancia y objeto son intercambiables.

Un objeto NO es :

Atributos como tiempo, belleza o color.
Emociones, como amor o coraje.
Entidades que normalmente son objetos sin embargo, se tienen la idea de que son atributos de una clase cuando ocurre algún problema en particular

De la definición de objeto, se desprenden las siguientes definiciones:

- **Operaciones o métodos**. Procesos que actúan con un objeto para transformar la estructura interna de datos o proveer de información a la estructura interna.
- **Mensaje**. Solicitud a un objeto para derivar una operación.
- **Clase**. Conjunto de objetos que tienen características similares. Un objeto no es una clase, pero una clase puede ser un objeto.
- **Instancia**. Objeto individual de una clase.

Programación, diseño y análisis orientados a objetos, definiciones:

Programación Orientada a Objetos (POO). Método de implementación en el cual los programas son organizados como colecciones cooperativas de objetos, cada uno de los cuales representa una instancia de alguna clase y sus clases, a su vez, son miembros de una jerarquía de clases unidad con relaciones de herencia.

Partes claves de la definición:

- POO utiliza objetos, no algoritmos, como la construcción de bloques fundamentales.
- Cada objeto es miembro de una clase.
- Las clases se relacionan unas a otras vía relaciones de herencia.

Diseño Orientado a Objetos (DOO). Método de diseño que abarca la descomposición de un proceso orientado a objetos y un anotación para descripciones lógicas y físicas, así como modelos estáticos y dinámicos del sistema que se esta diseñando.
Partes claves de la definición:

- DOO se dirige a una descomposición orientada a objetos

- DOO utiliza diferentes notaciones para expresar diferentes modelos de los diseños lógicos (estructura de clases y objetos) y físico (arquitectura de modelos y procesos)

Análisis Orientado a Objetos (AOO). Un método de análisis que examina los requerimientos desde la perspectiva de clases y objetos encontrados en el vocabulario del domino del problema.

Elementos del modelo orientado a objetos.

Elementos primarios: abstracción, encapsulado, modularidad y jerarquía.
Elementos secundarios: tipificación, concurrencia y persistencia.

Abstracción. Denota las características esenciales que distinguen a un objeto de todos los otros tipos de objetos. De esta forma nos provee de fronteras conceptualmente definidas, relativas a la expectativa del observador.

Decidir un correcto conjunto de abstracciones para el dominio de un problema es el punto central en el diseño orientado a objetos.

Encapsulado. "Ocultamiento de información", es el proceso que esconde todos los detalles de un objeto que no contribuyen a sus características esenciales.

Abstracción y encapsulado son conceptos que se complementan.

Modularidad. La propiedad de un sistema que ha sido descompuesta en un conjunto de módulos y pobremente acoplados. Las clases y objetos son implementadas en módulos para producir la arquitectura de un sistema.

Jerarquía. El lugar u ordenamiento de abstracciones. Las dos jerarquías más importantes en un sistema complejo:

- La estructura de la clase (el "tipo de" jerarquía), herencia
- La estructura del objeto (la "parte de" jerarquía), agregación

Tipificación. El enforzamiento de la clase de un objeto, tal que los objetos de diferentes tipos no pueden ser intercambiados, a lo más, pueden ser intercambiados en formas muy restringidas.

Un tipo es muy similar a una clase. La tipificación permite que las abstracciones sean expresadas en una forma tal que el lenguaje de programación usado para implementar el diseño puede ser usado para forzar las decisiones de diseño.

Concurrencia. La propiedad que distingue un objeto activo de uno que no esta activo, es decir se refiere a la ejecución de tareas al mismo tiempo.

Persistencia. Propiedad de un objeto cuya existencia trasciende tiempo (por ejemplo, el objeto continua existiendo después que su creador cesa su existencia) y/o espacio (por ejemplo, la localización del objeto se mueve de la dirección de espacio en que fue creado).

Un objeto toma cierta cantidad de espacio y existe por una cantidad particular de tiempo. Ambos, su estado y su clase deben persistir.

Metodología sobre Técnicas de Ingeniería de Software Aplicadas a Programación (TISAP)

A continuación se presenta un compendio de lo que es esta metodología TISAP, la cual se ha utilizado a través del desarrollo del presente trabajo:

Objetivos:

- Diseñar y desarrollar programas usando técnicas de Ingeniería de Software.
- Producir programas que no generen ningún problema después de ser liberados.
  - Sin Fallas.
  - Sin modificaciones requeridas por falta de entendimiento o deficiencias de diseño.
  - Sin problemas de desempeño y/o rendimiento.
- Producir programas sencillos, legibles y fáciles de modificar.
- Evitar que los proyectos de programación se excedan en presupuesto.
- Minimizar el costo de mantenimiento de los sistemas.

Como características de un sistema se considera que debe ser:

- Comprensible.
- Modificable.
- Confiable.

Dentro del Diseño de la Lógica, la metodología TISAP considera el uso de:

- Estructuras de Control.
- Modularidad.
- Diseño Top – Down.
- Diseño Bottom – Up.
- Estructuras de Datos.

En las fases de Implementación y Verificación, se considera:

- Depuración de sintáxis.
- Preparación de pruebas.

– Preparación de check-list para pruebas.
– Preparación de casos de prueba.
– Preparación de procedimientos de pruebas.

- Revisión de lógica.
- Depuración de lógica.
- Revisión final.

Apéndice B
Código Fuente

En este apéndice se pretende mostrar únicamente las partes más representativas del código para todos y cada uno de los procesos involucrados. Se distinguen entre ellos, al igual que en el cuerpo de la tesis, 4 tipos diferentes de programación, cada una representando a las etapas principales del proyecto.

Los desarrollos aquí mostrados representan únicamente las partes más importantes de los programas codificados, debido a que se debe guardar la confidencialidad de la información, además de que no se pretende llenar este trabajo con código, que finalmente no será parte representativa del mismo, sino únicamente mostrar un ejemplo de cómo se llevaron a cabo los procesos.

A continuación se presenta el código generado por uno de los procesos de extracción de información, correspondiente al primer módulo de nuestro proceso.

```
  0 IDENTIFICATION DIVISION.                                              000000
100 PROGRAM-ID   CDI-FON-F-CARSAL-010.                                    000100
200 DATE-        JUL 01 DE 1997                                           000200
    WRITTEN.
250 *AUTHOR.     PCA                                                      000250
260 *                                                                     000260
300 DATE-COMPILED.                                                        000300
400 ************************************************************************ 000400
    ************
500 ** ESTE PROGRAMA GENERA EL ARCHIVO PLANO DE                           000500
    FONDOS
600 ** QUE SERA CARGADO A LA BASE DE DATOS QUE                           000600
    UTILIZA EL
700 ** SISTEMA DE RIESGOS A PARTIR DE CUENTAS Y                          000700
    FONDOS.
800 ** SOLO SE ESCRIBIRAN LOS TIPO DE REGISTRO                           000800
900 ** QUE SEAN CUENTAS ACTIVAS=4 Y CUENTA-                             000900
    SEGMENTO=3,8 O 9
100 ************************************************************************ 000100
    ************
1100 ENVIRONMENT DIVISION.                                                001100
1200 CONFIGURATION SECTION.                                               001200
1300 SOURCE-COMPUTER.    A9-F.                                            001300
1400 OBJECT-COMPUTER.    A9-F.                                            001400
1500 INPUT-OUTPUT SECTION.                                                001500
1600 FILE-                                                                001600
     CONTROL.
1800      SELECT F-OUT-SALD ASSIGN TO DISK.                               001800
1900 DATA                                                                 001900
     DIVISION.
2000 FILE SECTION.                                                        002000
2100 FD   F-OUT-SALD.                                                     002100
2200 01   REG-OUT-SALD.                                                  002200
2300      02   OUT-RSA-CUENTA.                                           002300
2400           03 OUT-RSA-OFICINA        PIC  X(3).                      002400
2500           03 OUT-RSA-               PIC  X(7).                      002500
     CONSECUTIVO
2600      02        OUT-RSA-1            PIC  X.                         002600
2900      02        OUT-RSA-CVESALDO     PIC  X.                         002900
3000      02        OUT-RSA-2            PIC  X.                         003000
3100      02        OUT-RSA-SIGLO        PIC  XX.                        003100
3200      02        OUT-RSA-FECHA-PA     PIC  X(6).                      003200
```

```
3500      02      OUT-RSA-3                PIC  X.                    003500
3600      02      OUT-RSA-SALDO            PIC  9(13).99.           003600
3700                                                                 003700
3800                                                                 003800
7700 DATA-BASE SECTION.                                              007700
7800 DB DBCHE.                                                       007800
7900 01  CUENTAS                                                     007900
8000 01  FONDOS USING FON-X-CTA-                                    008000
       PTS.
8100 WORKING-STORAGE SECTION.                                        008100
8450 77  C2600-                           PIC X VA "^".              008450
       SOUT
8500 77  S000-EOF                        PIC 9 VA 0.                 008500
8700 01 D000-FECHA-PARAM RECEIVED BY REFERENCE.                       008700
8800    05 D2600-FECHA-PARAMETRO          PIC 999999.                008800
9400 *                                                               009400
9500 PROCEDURE DIVISION USING D000-FECHA-PARAM.                      009500
9700 1000-INICIO.                                                    009700
10000     OPEN     INQUIRY DBCHE.                                    010000
10100     OPEN     OUTPUT  F-OUT-SALD                                010100
10200     PERFORM 2000-PRIMERA-LECTURA                               010200
10300     PERFORM 2500-GENERA-OTRO ULTIL S000-EOF =1                 010300
10400     CLOSE    F-OUT-SALD                                        010400
10500     CLOSE    DBCHE.                                            010500
10600     STOP RUN.                                                 010600
10700 100-9999-FIN.                                                  010700
10800     EXIT.                                                     010800
10900 ****                                                           010900
11100  2000-PRIMERA-LECTURA.                                         011100
11200      FIND FIRST CUENTAS                                        011200
11300           ON  EXCEPTION                                        011300
11400           IF DMSTATUS (NOTFOUND)                               011400
11500           MOVE 1 TO S000-EOF.                                  011500
19100 2500-GENERA-OTRO.                                              019100
19101       IF (CTA-INACTIVA NOT = 4 AND (CTA-SEGMENTO = 3           019101
19104          OR CTA-SEGMENTO = 8 OR CTA-SEGMENTO = 9))             019104
19107          PERFORM 2600-BUSCA-FONDOS.                            019107
19128        FIND NEXT CUENTAS                                       019128
19131             ON EXCEPTION                                       019131
19134             IF DMSTATUS ( NOTFOUND )                           019134
19137                MOVE 1 TO S000-EOF.                             019137
20700 2600-BUSCA-FONDOS.                                             020700
20800       FIND FON-X-CTA-PTS  WHERE FON-OFICINA = CTA-             020800
       OFICINA
20900             AND FON-CONSECUTIVO = CTA-CONSECUTIVO              020900
21000             ON EXCEPTION                                       021000
```

```
21010          IF  DMSTATUS ( NOTFOUND)                              021010
21020              MOVE 1 TO S000-EOF.                               021020
21700      MOVE C2600-SOUT          TO OUT-RSA-1                     021700
21800      MOVE C2600-SOUT          TO OUT-RSA-2                     021800
21900      MOVE C2600-SOUT          TO OUT-RSA-3                     021900
22200      MOVE FON-OFICINA         TO OUT-RSA-OFICINA               022200
22300      MOVE FON-CONSECUTIVO        OUT-RSA-                      022300
     TO                                CONSECUTIVO
22400      MOVE "19"                TO  OUT-RSA-SIGLO                022400
22500      MOVE D2600-FECHA-           OUT-RSA-FECHA-PA              022500
     PARAMETRO TO
22800      MOVE "5"                 TO  OUT-RSA-CVESALDO             022800
22900      MOVE FON-SALDO           TO OUT-RSA-SALDO                 022900
23000      WRITE REG-OUT-SALD.                                       023000
```

El siguiente listado corresponde a uno de los procesos uitlizados para la transferencia de información entre los diferentes ambientes (UNISYS a UNIX) que pertenece a la segunda etapa del proyecto.

CDI/FON/J/CARSAL/010 (07/30/97) 7:08PM WEDNESDAY,JULY 30,1997
Pacific Standard Time

```
    0 BEGIN JOB FON/CARSAL(STRING ZONA,STRING FECHA);                00000000
  100  TASK T;                                                       00000100
  200  STRING LSDBCH;                                                00000200
  300  STRING LSNOMFECH;                                             00000300
  400  STRING LSARCHIVO;                                             00000400
  500  LSDBCH:="DBCHE"&ZONA;                                         00000500
  550  LSNOMFECH := FECHA;                                           00000550
  600  LSNOMFECH := DROP(LSNOMFECH,2);                               00000600
 7000 %*************************************************************** 00007000
      **
 8000 %* JOB QUE CORRE EL PROGRAMA QUE GENERA ARCHIVO                 00008000
      *
 9000 %* PLANO QUE OBTIENE LOS SALDOS                                 00009000
      *
 1000 %* CON LOS DATOS DE FONDOS                                      00001000
      *
 1100 %* ELABORADO POR PCA EL 30 DE JUN DE 1997          *           00001100
 1200                                                                 00001200
 1300 %*************************************************************** 00001300
      **
 1400                                                                 00001400
```

```
1500 RUN  CDI/FON/O/CARSAL/010(FECHA)  [T];                          00001500
1600        DATABASE  DBCHE(TITLE=#LSDBCH);                          00001600
1700        FILE F-OUT-SALD(KIND=DISK,                               00001700
1800                    TITLE = CDI/#ZONA/FON/#LSNOMFECH/SAL,        00001800
2000                    FILETYPE=0,NEWFILE                           00002000
    =TRUE,PROTECTION=SAVE);
2100                                                                 00002100
2200  IF T ISNT COMPLETEDOK THEN ABORT " ABORTO GENERACION           00002200
    DE ARCHIVO"
2300                                                                 00002300
2400  ELSE BEGIN                                                     00002400
2500      DISPLAY "TERMINO BIEN LA GENERACION DE ARCHIVO     ";      00002500
2600      LSARCHIVO                                                  00002600
    :="'/home/clientes/archivos/FONSAL"&ZONA&LSNOMFECH;
2700      LSARCHIVO := LSARCHIVO&".TXT'";                            00002700
2800      COPY CDI/#ZONA/FON/#LSNOMFECH/SAL AS #LSARCHIVO            00002800
2900      FROM SYS2(PACK) TO DISK(IPADDRESS="10.6.200.38",           00002900
3000            USERCODE='sybase'/'sybase');                         00003000
3100      END                                                        00003100
3200 END JOB.                                                        00003200
```

El que sigue es un listado presentado para ejemplificar la carga de información al modelo, tercera etapa de nuestro proceso.

Programa: SP_AHOCYDDIA01.SQL
Autor: PCA
Fecha: 11/08/1997

Objetivo: Archivo que localiza y da de alta en la base de datos y carga los datos del cliente y de su dirección, llamado desde SP_AHOCARDIA01.SQL.

Entrada: Los archivos planos de la base de datos AHORROS, Clientes, Direcciones y el parámetro CUENTA que será un numérico con la cuenta a actualizar en las tablas, después de haberse verificado en el programa que hace la llamada.

Salida: Las tablas Direcciones y Clientes del Modelo de CDI actualizadas con la información diaria y con los nuevos contratos.

```
/* Borra si existe el STORE_PROCEDURE*/
IF EXISTS (Select name from sysobjects where name = 'SP_AHOCYDDIA01')
  DROP PROC SP_AHOCYDDIA01
go

/*Crea el Store Procedure */
/* Se declaran las variables que se usaran en el modelo a semenjanza de
  los registros de Contratos, Clientes, Direcciones que se daran de alta*/
CREATE PROC SP_AHOCYDDIA01 @lnCuentaTrabajo numeric(10)
AS
DECLARE

  @lnCCuenta                numeric(10),
  @lnCPersonalidad          varchar(3),
  @lsCCodigo          varchar(4),
  @lnCCodigoTitulo    numeric(1),
  @lnCNombre                varchar(20),
  @lnCNombreComercial  varchar(1),
  @lnCApellidoPat     varchar(20),
  @lnCApellidoMat     varchar(3),
  @lnCApellidoCas     varchar(3),
  @lnCSexo            varchar(1),
  @lnCEstadoCivil     varchar(1),
  @lnCNacionalidad    varchar(1),
  @lnCCodPaisNac            numeric(3),
  @lnCFechaNac        datetime,
  @lnCRFC             varchar(15),
  @lnCCURP            varchar(3),
  @lnCSectorCont      numeric(2),
  @lnCBanca           varchar(2),
  @lnCCodActGen             numeric(3),
  @lnCCodActEsp       numeric(3),
  @lnCApoderadoLegal  varchar(3),
  @lnCOcupacion       varchar(3),
  @lnCAdministrador   varchar(3),
  @lnCNoActaConst           varchar(3),
  @lnCFechaReg        datetime,
  @lnCFechaModif      datetime,
  @lnCNoEjecutivo     numeric(3),
  @lnCEstatus         varchar(1),
  @lnCNoAlborada      numeric(3),
  @lnCOficinaRegistro numeric(3),
  @lnDCuenta          numeric(10),
  @lnDNoDomicilio     numeric(1),
  @lsDCalle           varchar(35),
  @lnDNoInterior      numeric(1),
```

```
  @lnDNoExterior      numeric(1),
  @lnDColonia         varchar(1),
  @lnDMpoDeleg        varchar(35),
  @lnDCodMpoDeleg     numeric(1),
  @lnDEstado          varchar(1),
  @lsDCodEstado       numeric(1),
  @lsDCodPais         numeric(1),
  @lsDCP              numeric(5),
  @lsDApdoPostal      varchar(1),
  @lsDTelefono1       numeric(7),
  @lsDTelefono2       varchar(1),
  @lnDEMail           varchar(1),
  @lnDFechaReg        datetime,
  @lnDFechaModif      datetime

BEGIN

  BEGIN TRAN

  /* Se declara el cursor de la tabla de Clientes
     nuevos (tabla temporal) para insertar el registro
     ojo, solo se localiza el registro con la cuenta solicitada */
     DECLARE curCliNue CURSOR
     FOR SELECT  id_cliente, personalidad, titulo, codigo_titulo,
           nombre, nombre_comercial, apellido_paterno,
           apellido_materno, apellido_casada, sexo,
           estado_civil, nacionalidad, codigo_pais_nacimiento,
           fecha_nacimiento, rfc, curp, sector_contable, banca,
           codigo_actividad_generica, codigo_actividad_especifica,
           apoderado_legal, ocupacion, administrador,
           no_acta_constitutiva, fecha_registro, fecha_modificacion,
           no_ejecutivo, estatus, no_cliente_alborada, oficina_registro
     FROM tmpTableCli WHERE  id_cliente = @lnCuentaTrabajo


  /* Se declara el cursor de la tabla de Direcciones
     nuevos (tabla temporal) para insertar el registro
     ojo, solo se localiza el registro con la cuenta solicitada */
     DECLARE curDirNue CURSOR
     FOR SELECT id_cliente, no_domicilio, calle, numero_interior,
           numero_exterior, colonia, municipio_delegacion,
           codigo_municipio_delegacion, estado, codigo_estado,
           codigo_pais, codigo_postal, apartado_postal, telefono1,
           telefono2, e_mail, fecha_registro, fecha_modificacion
     FROM tmpTableDir WHERE  id_cliente = @lnCuentaTrabajo
```

```
/*Ahora abrimos los cursores */
  OPEN curCliNue
  OPEN curDirNue

/* y cargamos el  cursor  de datos del cliente */
  FETCH curCliNue
  INTO  @lnCCuenta, @lnCPersonalidad, @lsCCodigo, @lnCCodigoTitulo,
      @lnCNombre, @lnCNombreComercial, @lnCApellidoPat,
      @lnCApellidoMat, @lnCApellidoCas, @lnCSexo, @lnCEstadoCivil,
      @lnCNacionalidad, @lnCCodPaisNac, @lnCFechaNac, @lnCRFC,
      @lnCCURP, @lnCSectorCont, @lnCBanca, @lnCCodActGen,
      @lnCCodActEsp, @lnCApoderadoLegal, @lnCOcupacion,
      @lnCAdministrador, @lnCNoActaConst, @lnCFechaReg, @lnCFechaModif,
      @lnCNoEjecutivo, @lnCEstatus, @lnCNoAlborada, @lnCOficinaRegistro

/* e insertamos el registro en la tabla de Clientes */
  INSERT INTO Clientes
  VALUES (@lnCCuenta, @lnCPersonalidad, @lsCCodigo, @lnCCodigoTitulo,
      @lnCNombre, @lnCNombreComercial, @lnCApellidoPat,
      @lnCApellidoMat, @lnCApellidoCas, @lnCSexo, @lnCEstadoCivil,
      @lnCNacionalidad, @lnCCodPaisNac, @lnCFechaNac, @lnCRFC,
      @lnCCURP, @lnCSectorCont, @lnCBanca, @lnCCodActGen,
      @lnCCodActEsp, @lnCApoderadoLegal, @lnCOcupacion,
      @lnCAdministrador, @lnCNoActaConst, @lnCFechaReg, @lnCFechaModif,
      @lnCNoEjecutivo, @lnCEstatus, @lnCNoAlborada, @lnCOficinaRegistro)


/* ahora cargamos el  cursor  de datos  de la direccion del cliente */
  FETCH curDirNue
  INTO @lnDCuenta, @lnDNoDomicilio, @lsDCalle, @lnDNoInterior,
     @lnDNoExterior, @lnDColonia, @lnDMpoDeleg, @lnDCodMpoDeleg,
     @lnDEstado, @lsDCodEstado, @lsDCodPais, @lsDCP, @lsDApdoPostal,
     @lsDTelefono1, @lsDTelefono2, @lnDEMail, @lnDFechaReg,
     @lnDFechaModif

/* e insertamos los datos en la tabla de Direcciones */
  INSERT INTO Direcciones
  VALUES (@lnDCuenta, @lnDNoDomicilio, @lsDCalle, @lnDNoInterior,
     @lnDNoExterior, @lnDColonia, @lnDMpoDeleg, @lnDCodMpoDeleg,
     @lnDEstado, @lsDCodEstado, @lsDCodPais, @lsDCP, @lsDApdoPostal,
     @lsDTelefono1, @lsDTelefono2, @lnDEMail, @lnDFechaReg,
     @lnDFechaModif)

/* Liberamos los cursores */
DEALLOCATE CURSOR curCliNue
```

```
  DEALLOCATE CURSOR curDirNue

  COMMIT TRAN
  /* y regresamos a la llamada desde SP_AHOCARDIA01 */
END
go
```

Por último se muestra el código utilizado dentro de los procesos de explotación de información, quedando representados así todos los pasos de nuestro ciclo.

Módulo Proceso

```
'Realizado por : PCA
'Modulo : Procesos para la conversión de numeros a letras
' 'Objetivo : Manejo de funciones para el manejo de operaciones con btrieve


Sub PCAVerificaEquivalencias(LClienteArchivo As String)
   'proceso que busca, a partir del cliente que se encuentra en el archivo
   'de texto, el cliente equivalente en arequiv, buscando a partir de
   'arequiv, a argroup y el equivalente en arequiv
   Dim LCliente As String
   LCliente = Mid(LClienteArchivo, 3, 10)
   PCABuscaAREQUIV (LCliente)
   If GClienteEnAREQUIV <> "" Then
     'encontre cliente_equivalente por cliente_member_code en AREQUIV, busco en ARGROUP
     PCABuscaARGROUP (GClienteEnAREQUIV)
     If GClienteEnARGROUP <> "" Then
       'encontre cliente_que_agrupa por cliente en ARGROUP
       PCABuscaEnAREQUIVCIteEquivalente (GClienteEnARGROUP)
       If GClienteEnAREQUIV <> "" Then
         'encontre cliente_membre_code por cliente_equivalente
         Mid(LClienteArchivo, 3, 10) = GClienteEnAREQUIV
       Else
         Mid(LClienteArchivo, 3, 10) = LCliente
       End If
     Else
       'no encontre en argroup
       Mid(LClienteArchivo, 3, 10) = LCliente
     End If
   Else
     'no encontre cliente en arequiv
     Mid(LClienteArchivo, 3, 10) = LCliente
```

```
    End If
    GSubstMembercode = LClienteArchivo
  End Sub
  'para argroup.bas
  Sub PCABuscaAREQUIV(LCliente As String)
    Dim ResultadoBusqueda As Integer
    'verifica en arequiv, busco con el cliente (arch.texto.linea 02)
    'y recupero el cliente_equivalente
    GlLlaveArch$(1) = LCliente
    ResultadoBusqueda = OperReg%(AREQUIV1, CGlBtrGetEqual, 1)
    If ResultadoBusqueda <> 0 Then
      'no encontre equivalencia lo dejo igual
      GClienteEnAREQUIV = ""
    Else
      'encontre equivalencia en arequiv
      GClienteEnAREQUIV = GlRecordSetArch(AREQUIV1)!Customer_key
    End If
  End Sub
  'para argroup.bas
  Sub PCABuscaARGROUP(LClienteEnAREQUIV As String)
    Dim ResultadoBusqueda As Integer
    'con el cliente_equivalente recupero el cliente_que_agrupa
    GlLlaveArch$(1) = LClienteEnAREQUIV   'busco por indice 2
    ResultadoBusqueda = OperReg%(ARGROUP2, CGlBtrGetEqual, 1)
    If ResultadoBusqueda <> 0 Then
      'no encontre equivalencia lo dejo igual
      GClienteEnARGROUP = ""
    Else
      GClienteEnARGROUP = GlRecordSetArch(ARGROUP2)!cliente_agrupador
    End If
  End Sub
  'para argroup.bas
  Sub PCABuscaEnAREQUIVCIteEquivalente(GClienteEnARGROUP As String)
    Dim ResultadoBusqueda As Integer
    'con el cliente que agrupa busco en arequiv, en cliente_equivalente,
    'y obtengo cliente_member_code, que es el que se guarda en el arreglo
    'y a su vez en el temporal
    Dim LEncontreAREQUIV As Byte
    GlLlaveArch$(1) = GClienteEnARGROUP
    ResultadoBusqueda = OperReg%(AREQUIV2, CGlBtrGetEqual, 1)
    If ResultadoBusqueda <> 0 Then
      'no encontre equivalencia lo dejo igual
      GClienteEnAREQUIV = ""
    Else
      'encontre equivalencia
      GClienteEnAREQUIV = GlRecordSetArch(AREQUIV2)!Membercode
```

```
  End If
End Sub
'para argroup.bas
Sub PCAInsertaTmpFacturas(LArrClientes() As String, NoLineasDetFactura As Integer)
  'verifica si la factura que existe en el arreglo ya fue insertada
  'en la tabla temporal, si es nueva la  inserta, si no la actualiza
  Dim i As Integer
  For i = 3 To NoLineasDetFactura
    'valida por cada linea porque cada una tiene un servicio diferente
    If Not PCAValidaCliteRelServInsertado(Mid(LArrClientes(2), 3, 10), Mid(LArrClientes(i), 3, 10)) =
0 Then  'le quite al principio Mid(LArrClientes(1), 3, 8),
      PCAInsertaFacturaTmp LArrClientes(), i
    Else
      PCAActualizaFacturas LArrClientes(), i
    End If
  Next i
End Sub
'para argroup.bas
Function PCAValidaCliteRelServInsertado(LCli As String, LServ As String) 'se lo quite LClte As
String,
  Dim ResultadoBusqueda As Integer
  'verifica si la combinacion cliente, relacion, servicio ya fue insertada
  'en la tabla temporal
  GlLlaveArch$(1) = LCli
  GlLlaveArch$(2) = LServ
  ResultadoBusqueda = OperReg%(ARFACTMP, CGlBtrGetEqual, 2)
  PCAValidaCliteRelServInsertado = ResultadoBusqueda%
End Function
'para argroup.bas y este proceso tambien

Sub PCALimpiaArregloClientes(LArrClientes() As String, LLineaFactura As Integer)
  'inicializo el arreglo donde guardo cada factura que se guarda en el temporal
  Dim i As Integer
  For i = 1 To LLineaFactura - 1
    LArrClientes(i) = ""
  Next i
EndSub


Sub PCAAvanzaForma2()
  frmBusArchivo.Hide
  Load frmUbicacionINLOC
  frmUbicacionINLOC.Show
End Sub

Sub PCAAvanzaForma3()
```

```
    frmUbicacionINLOC.Hide
    Load frmProcesando
    frmProcesando.Show
End Sub


Sub PCALeeLinea(GCanalArchivo As Integer)
    Input #GCanalArchivo, GLineaLeida
End Sub

Sub PCACierraArchivos()
    Close #GCanalArcProceso%
    Close #GCanalArcCostServ%
    Close #GCanalArcOrdCostServ%
End Sub

Sub PCATerminaAplicacion()
    PCACierraArchivoEntrada
    PCACierraArchivos
End Sub

Sub PCADescargaFormas()
    Unload frmParIntPoliza
    Unload frmBusArchivo
    Unload frmUbicacionINLOC
End Sub

Sub PCACierraArchivoEntrada()
    Close #GCanalArcEntrada%
End Sub

Sub PCACierraArchivoCostServ()
    Close #GCanalArcCostServ%
End Sub

Sub PCALimpiaTextos()
    frmParIntPoliza.txtCodServOrigen.Text = ""
    frmParIntPoliza.txtCodServDestino.Text = ""
    frmParIntPoliza.txtCosUnitServicio.Text = 0
    frmParIntPoliza.txtNumMinServicios.Text = 0
    frmParIntPoliza.txtCarMinFacturado.Text = 0
    frmParIntPoliza.txtMonMinXFactura.Text = 0
End Sub


Function PCALeeArchivo()
```

```
    'lee el archivo factura.txt linea por linea
    Dim LAbriArchivo As Integer
    LAbriArchivo = PCAAbreArchivoEntrada
    If LAbriArchivo = 1 Then
      'obtiene el numero de lineas que tiene el archivo factura.txt
      'y al mismo tiempo abre un arreglo donde se van a validar
      'los servicios
      GNoLineas = PCACuentaLineasXTipo
    End If
    PCALeeArchivo = LAbriArchivo
End Function

Sub PCAObtenCliente(i As Integer)
    'obtiene los clientes en la segunda posicion del arreglo
    Dim LLineaInvalida As Long
    LLineaInvalida = 0
    If Trim(Mid(GLineaLeida, 3, 10)) <> "" Then
      ReDim Preserve GArrRelaciones2(i)
      GArrRelaciones2(i) = Trim(Mid(GLineaLeida, 3, 10))
    Else
      LLineaInvalida = 1
    End If
    If LLineaInvalida = 1 Then
      MsgBox "Proceso terminado, linea 02 inválida en el archivo " +
frmBusArchivo.txtUbicArchivo.Text, vbOKOnly + vbExclamation, "Grupos de Clientes"
      PCAEscribeError "Proceso terminado, linea 02 inválida en el archivo " +
frmBusArchivo.txtUbicArchivo.Text
      PCADescargaFormas
      PCATerminaAplicacion
    End If
End Sub
```

Apéndice C
Glosario

3Com *3Com Corporation* provee una amplia variedad de *hardware* y *software* para redes de área local.

ANSI (*American National Standars Institute*) instituto americano de normas nacionales. Organización de afiliados privados sin fines de lucro, fundada en 1918, que coordina el desarrollo de normas nacionales voluntarias en Estados Unidos tanto en el sector privado como en el público. Las normas de tecnología de la información atañen al análisis, control y distribución de la información, lo cual incluye lenguajes de programación, intercambio electrónico de datos (*EDI*), telecomunicaciones y propiedades físicas de disquetes, cartuchos y cintas magnéticas.

Batch Lote o grupo. Programa por lotes o trabajo por lotes se refiere a un programa que procesa un conjunto entero de datos, tal como un programa de informes o de clasificación. El procesamiento remoto por lotes implica la transmisión de un archivo completo a través de una red. Las operaciones por lotes son también llamadas operaciones fuera de línea (*offline*).

Binary Large Object (BLOB) (*Binary Large Object*) objeto grande binario. Acuñado por *Borland*, es un campo de base de datos, que mantiene cualquier tipo de información digitalizada en formato binario.

Buffer Una porción reservada de la memoria que se utiliza para almacenar los datos mientras son procesados. En un programa se crean "*buffers*" para contener algunos datos de cada uno de los archivos que van a ser leídos o grabados. Un *buffer* puede ser también un pequeño banco de memoria usado para fines especiales.

Bulkcopy Copia en masa. Copia que no es utilizada para un procesamiento de alta velocidad. Puede referirse a copias completas del contenido de la memoria o de los datos almacenados en otro medio.

CASE (*Computer Aided Software Engineering o Computer Aided Systems Engineering*) ingeniería de *software* asistida por computadora o ingeniería de sistemas asistida por computadora. *Software* que se utiliza en una, o cualquiera de las fases del desarrollo de un sistema de información, incluyendo análisis, diseño y programación. Por ejemplo, los diccionarios de datos y herramientas de diagramación ayudan en las fases de análisis y diseño, mientras que los generadores de aplicaciones aceleran la fase de programación.

Las herramientas *CASE* proporcionan métodos automáticos para diseñar y documentar las técnicas tradicionales de programación estructurada. La meta última de *CASE* es proveer un lenguaje para describir el sistema completo, que sea suficiente para generar todos los programas necesarios.

CDI Centro De Información. Siglas para identificar el proyecto desarrollado para la Institución Bancaria en el presente trabajo.

COBOL 74 (*Common Business Oriented Language*) lenguaje común orientado a los negocios. Un lenguaje de programación de alto nivel orientado a los negocios, que ha sido el principal lenguaje de aplicaciones comerciales en mini y macro computadoras.

Driver Controlador, conductor. También llamado *device driver* (controlador de dispositivos), es una rutina de programa que contiene las instrucciones necesarias para controlar la operación de un dispositivo periférico. Contienen el código de máquina preciso para activar todas las funciones de cada dispositivo.

Ethernet *LAN* estándar 802.3 de *IEEE* originalmente desarrollada por *Xerox*, Digital e Intel que utiliza el método de acceso *CSMA/CD*, transmite 10 Mbps y puede concetar en total hasta 1.024 nodos. *Ethernet* estándar (también llamado *Ethernet "thick"* denso) utiliza una topología de bus con una longitud de cable con un máximo de 1.640 pasos sin utilizar un repetidor.

FTP (*File Transfer Protocol*) protocolo de transferencia de archivos. Un protocolo *TCP/IP* que es usado para conectarse a la red, listar directorios y copiar archivos. También puede traducir entre *ASCII* y *EBCDIC*.

Hardware Toda la maquinaria y el equipamiento. Contrástese con *software*, el cual es un conjunto de instrucciones que le dicen a la computadora que hacer. En operación, una computadora es a la vez *hardware* y *software*. Uno es inútil sin el otro, y cada uno regula al otro. El diseño de hardware especifica que instrucciones puede seguir, y luego las instrucciones le dicen qué hacer. "Si puedes tocarlo es *hardware*"

Interfaces Una conexión e interacción entre *hardware*, *software* y usuario. Las interfaces de hardware son los conectores y cables que transportan las señales eléctricas en un orden prescrito. Las interfaces de *software* son los lenguajes, códigos y mensajes que utilizan los programas para comunicarse unos con otros, tal como entre un programa de aplicación y el sistema operativo. El diseño y construcción de interfaces constituye una parte principal del trabajo de los ingenieros, programadores y consultores.

Job Trabajo, tarea. Una unidad de trabajo que se ejecuta en la computadora. Una tarea puede ser solo un programa o grupo de programas que deben trabajar juntos.

Mainframe Macrocomputadora. Una computadora grande. A medidados de los años sesenta, las épocas antiguas de las computadoras, todas las computadoras eran mainframes (literalmente "gabinete principal"), ya que el término se refería al gabinete que contenía el *CPU*. Aunque mainframe aún significa gabinete principal, usualmente se refiere a un gran sistema de computación y toda la experiencia asociada que va con él.

ODBC (*Open DataBase Connectivity*) Conectividad Abierta para Bases de Datos. Los *ODBC* son métodos estándar de acceso a bases de datos desarrollados por *Microsoft Corporation*. La meta de los *ODBC* es hacer posible el acceso a los datos desde cualquier aplicación, independientemente del sistema de administración de bases de datos que se esté utilizando.

Performance Desempeño, comportamiento, rendimiento. En computación se utiliza este término para evaluar el desempeño ya sea de un equipo de *hardware* o la eficiencia de un sistema de *software*.

Query Consulta. Una consulta a una base de datos que permite al usuario contar, sumar y listar registros seleccionados contenidos en ella. Nótese la diferencia con *report* (reporte, informe) que es generalmente una salida impresa más elaborada con encabezados y números de página.

RDBMS (*Relational Database Management System*) Sistema de Administración de Bases de datos Relacionales. Está formado por una colección de programas que nos dan la capacidad de almacenar, modificar, y extraer información de una base de datos relacional.

SCSI (*Small Computer System Interface*) interfaz pequeña de sistemas de computadoras. Interfaz para más de siete periféricos. Es una interfaz de bus de 8 bits y permite que dos dispositivos se comuniquen a la vez (de principal a periféricos, de periférico a periférico).

Shell Script Guión *Shell*. Archivo de órdenes UNIX ejecutables creado por un editor de texto y ejecutable por la orden chmod.

SmallTalk Sistema Operativo y lenguaje de programación orientado a objetos que fue desarrollado en el Centro de Investigación de *Xerox Corporation* en Palo Alto. Como entorno integrado, elimina la distinción entre lenguaje de programación y sistema operativo. Smalltalk fue el primer lenguaje de programación orientado a objetos; originalmente se empleaba para crear prototipos de lenguajes de programación más simples, y las interfaces gráficas que son tan populares hoy.

Software Instrucciones de un computadora. Una serie de instrucciones que realizan una tarea en particular se llama programa o programa de *software*. Las dos categorías principales son *software* de sistemas y *software* de aplicaciones.
El *software* de sistemas se compone de programas de control, incluyendo el sistema operativo, *software* de comunicaciones y administrador de bases de datos.
El *software* de aplicaciones es cualquier programa que procesa datos para el usuario (inventario, nómina, hoja de cálculo, procesador de texto, etc.)

SPARC (*Scalable Performance ARChitecture*) arquitectura de rendimiento graduable. Computadora *RISC* de 32 bits creada por *Sun Microsystems, Inc.*)

SQL (*Structured Query Language*) lenguaje de consulta estructurado. Lenguaje utilizado para solicitar y procesar datos en una base de datos relacional. Desarrollado originalmente por IBM para sus macrocomputadoras, ha habido muchas implementaciones creadas para aplicaciones de bases de datos en mini y microcomputadoras. Las instrucciones de SQL se pueden utilizar para trabajar interactivamente con una base de datos o puede incluirse en un lenguaje de programación para servir de interfaz a una base de datos.

STORED PROCEDURES Conjunto de Procesos por lotes, de intrucciones *SQL* que es almacenado en el servidor de Base de Datos para tenerlo siempre disponible y ejecutarlo cuando sea necesario.

SUN (*Sun Microsystems, Inc.*) Fabricante de las estaciones de trabajo de alto rendimiento, basadas en redes, fundado en 1982. Sun se ajusta a un modelo de computación informático de sistemas abiertos a lo largo de toda su línea de productos, lo cual le permite interactuar en redes de sistemas de computación de otros fabricantes.

Sybase Familia de herramientas de desarrollo *SQL* de Sybase, Inc., que incluye un servidor *SQL*, juego de herramientas (*Toolset*) SQL (diseño, desarrollo y control) e interfaces cliente/servidor (arquitectura distribuida de bases de datos).

TCP/IP (*Transmission Control Protocol/Internet Protocol*) protocolo de control de transmisiones/protocolo internet. Conjunto de protocolos de comunicaciones desarrollado por la *DARPA* (*Defense Advanced Research Projects Agency* – agencia de proyectos de investigación avanzada de defensa) para intercomunicar sistemas diferentes. Se ejecuta en un gran número de computadoras *VAX* y basadas en *UNIX* y es utilizado por muchos fabricantes de *hardware*, desde los de computadoras personales hasta los de macrocomputadoras.

TELNET Protocolo de emulación de terminales desarrollado originalmente por *ARPAnet*.

Triggers Disparadores. Procesos de activación inmediata que generalmente son utilizados para mantener la integridad de una base de datos en un servidor.

Unisys (*Unisys Corporation*). Empresa de informática formada en 1986 como resultado de la fusión de las corporaciones *Burroughs y Sperry*. *Unisys* ha concentrado sus esfuerzos en proveer soluciones integradas para los mercados verticales, tales como el financiero, de aerolíneas y de comunicaciones.

UNIX Sistema Operativo multiusuario y multitarea de *AT&T* que se ejecuta en una amplia variedad de sistemas de computación de micro a macrocomputadoras. El *UNIX* está escrito en C (también desarrollado por *AT&T*) que es un lenguaje diseñado para programación a nivel de sistemas. Es la transportabilidad inherente al C lo que permite que *UNIX* pueda ejecutarse en tal cantidad de computadoras diferentes.

Visual Basic Versión de *BASIC* de *Microsoft* utilizado para desarrollar aplicaciones de *Windows*. Su fácil utilización se encuentra entre el complejo kit de desarrollo de *software* de *Windows (SDK)* para los programadores de C y *Word BASIC*, el lenguaje de macros de Word para Windows.

Windows 95 Versión de 32 bits para el entorno operativo para gráficos de *Microsoft*, proporciona un ambiente similar al de *Macintosh*, en el cual cada aplicación activa se visualiza en una pantalla movible y redimensionable sobre una pantalla principal.

Con el objeto de usar todas las funciones del *Windows*, las aplicaciones deben escribirse especificamente para él. Sin embargo, *Windows* también ejecuta aplicaciones del DOS y se puede usar como el entorno operativo desde el que se ejecutan todos los programas.

Windows NT (*Windows New Technology*) nueva tecnología de *Windows*. Sistema Operativo avanzado de 32 bits para 386 y superiores de *Microsoft* previsto para 1993. Ejecuta aplicaciones previstas para *DOS*, *Windows* 3.x y *NT* (*Win* 32). También se ha planificado el soporte *PoSix*. La compatibildad con *OS/2* no está clara, *NT* no utiliza *DOS*, es un sistema operativo autónomo.

Apéndice D
Análisis Costo Beneficio

Objetivo

El propósito de un análisis costo-beneficio es mostrar a los usuarios del nuevo sistema, así como al personal de la organización que realizará la inversión, que los beneficios que se esperan obtener superan a los costos estimados durante el desarrollo del sistema.

A continuación se realizará una breve descripción (así como su aplicación práctica) de los puntos que se trataron en el análisis costo-beneficio presentado a BANPAIS.

Análisis de los costos

Costo de Construcción del sistema

- Salarios y gastos extra para todo el personal relacionado con el proyecto.
- 4 Personas tiempo completo (8 horas) durante 6 meses $ 8,000 por persona al mes.
- Un líder de proyecto $ 10,000 al mes.

- Tiempo de computadora y herramientas de desarrollo para el personal.
- No existen ya que se aprovecharon recursos existentes del banco tanto en *mainframe* como en UNIX.

- Costos de reclutamiento del personal nuevo.
- No se requirieron nuevas contrataciones para operar el nuevo sistema

- Espacio de Oficina y equipo para el personal nuevo.
- Se habilitaron 2 oficinas equipadas para el personal encargado de operar el sistema, el costo fue nulo ya que el banco ya contaba con el equipo.

- Gastos de viaje para visitar usuarios ajenos.
- 6 Viajes México - Monterrey - México para 2 personas $ 1,500 cada uno.

Costo de Instalación

- Capacitación a usuarios.
- 2 Personas por 10 días (Incluido en salarios).

- Conversión de Bases de Datos.
- En este caso el costo de las licencias de las bases de datos fue nulo, ya que el banco ya contaba con las mismas.

- Instalación comercial.
- El banco contaba ya con las licencias del *Software* así como del *Hardware* con las cuales se desarrolló el sistema, por lo que no existió costo por instalación. Estas licencias son:

    1. *Burroughs.*
    2. *Solaris.*
    3. *Sybase.*
    4. *Visual Basic.*
    5. Windows 95.

- Ejecuciones paralelas.
- Es nulo ya que el sistema anterior sigue trabajando transparentemente al nuevo sistema.
- Equipo de desarrollo durante la instalación.
- No se generó ningún gasto ya que se utilizaron equipos existentes que no estaban siendo utilizados por el personal del banco.

Costo del Dinero

Existe un costo asociado con el uso del dinero. Dependiendo de la organización, se puede expresar en términos del costo del dinero prestado, o de los intereses que se ganarían si se tuviera invertido en lugar de estarse usando para el proyecto.

En el caso de BANPAIS, como cualquier institución bancaria que se preste de competir en un mercado difícil, se cuenta con un presupuesto destinado para el desarrollo de sistemas para la institución, por lo que el costo del desarrollo del sistema, a final de cuentas se encontraba dentro de ese presupuesto, por lo que no se vio en la necesidad de alterarlo para cubrir la totalidad de la inversión realizada en el proyecto.

BIBLIOGRAFIA

Rivero E., "Bases de Datos Relacionales", 1ª. Edición, Editorial Paraninfo, Madrid, 1988.

Poolet Michell and Reilly Michael, "ACCESS 95 Client / Server Development", 1ª. Edición, Editorial QUE Corp., USA, 1996.

Gorman Michael, "Enterprise Database in a Client / Server Enviroment", 1ª. Edición, Editorial Wiley - QED, USA, 1994.

Lefebvre Alain, "Intranet Cliente / Servidor Universal", 1ª. Edición, Editorial Eyrolles, París, 1997.

Fairley Richard, "Ingeniería de Software", 1ª. Edición, Editorial Mc.Graw Hill, USA, 1985.

Freedman Allan, "Diccionario de Computación", 1ª. Edición, Editorial Mc.Graw Hill, Madrid, 1993.

Jourdan Edward, "Análisis estructurado Moderno", 1ª. Edición, Editorial Prentice Hall, México, 1993.

Rumbauhg James, "Modelo y Diseño Orientado a Objetos, Metodología OMT", 1ª. Edición, Editorial Prentice Hall, Gran Bretaña, 1996.

Booch Grandy, "Objetc Oriented Design with Applications", 1ª. Edición, Editorial Benjamin / Cummings, Estados Unidos, 1994.

Manuales

"Técnicas de Ingeniería de Software Aplicadas a la Programación", Softek, 1994.

Páginas WEB

1.- http://banpais.com.mx Información sobre BANPAIS y sus servicios.

2.- http://banorte.com.mx Información sobre BANORTE, nuevo dueño de BANPAIS