23
2 ej

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

SISTEMA DE COMPENSACION ELECTRONICA DE CHEQUES PARA PC

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A N :

MARCO ANTONIO GARCIA FLORES

ARACELI GONZALEZ GUTIERREZ

NORMA HERNANDEZ ESPEJEL

GUSTAVO PEREZ MIGUEL

GUADALUPE ARTURO SANTAMARIA MURILLO

DIRECTOR DE TESIS: M. EN I. JUAN CARLOS ROA BEIZA

MEXICO, D. F. 1999

TESIS CON
FALLA DE ORIGEN

28021D

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

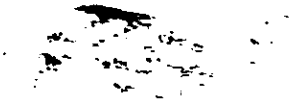
UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

PAGINACION DISCONTINUA



Gracias

A Dios, *por la familia que me ha dado, por el consuelo y la fe que me ha confortado siempre.*

A mis padres, *por su amor, sus consejos. su formación. apoyo y por no haber perdido la confianza en mí. Los quiero mucho.*

A mi esposo, *por apoyarme incondicionalmente en todo, por su amor, respeto y comprensión.*

A mis hijos, *por ser la luz de mi vida y por permitirme robarles mucho de su tiempo.*

A mis hermanas, *por su gran ayuda, por el gran amor que le brindan a mis hijos, por su confianza. no tengo como pagarles.*

A mi hermano Quique, *por que siempre estará en mi corazón.*

A mis suegros, ***tíos, sobrinos, cuñados y concuños***. *por su cariño y apoyo.*

A mis amigas, *por sus consejos, por compartir su amistad conmigo y darme ánimos para continuar.*

A mis profesores y al M.I. Juan Carlos Roa Beiza, *por toda la enseñanza en mi formación como profesionista.*

A todos ellos, *por enseñarme que en esta vida todo se puede, si uno se lo propone, por alentarme a seguir adelante y no rendirme. Por que gracias a todos he logrado realizar una de las metas de mi vida.*

Araceli

AGRADECIMIENTOS

El presente trabajo es de una manera un tributo a todas estas personas que gracias a su apoyo y amor me alentaron a continuar mis estudios y terminar esta etapa profesional en mi vida.

Mi familia

En la vida, todos tenemos la oportunidad de crecer tanto espiritualmente como profesionalmente, gracias a dios, yo tuve el apoyo de mi familia que en todo momento me apoyo y animó a seguir adelante en mis estudios.

Mi madre:

Le doy gracias a mi madre por su amor y apoyo incondicional, por su desvelo y preocupación y sobre todo por alentarme a conseguir mis metas. Gracias mami, por ser lo más bello en mi vida.

Mi padre:

A mi padre por que me formó la idea de lograr desarrollarme profesionalmente. Este fue uno de mis grandes retos y lo he cumplido. espero que te sientas muy orgulloso de mí. Te quiero mucho y me haces mucha falta.

Mis hermanos:

A mis hermanos con los que he compartido alegrías y tristezas y muchas otras cosas que nos han hecho crecer como seres humanos. Gracias por preocuparse por mí, y por el cariño que me tienen, y sobre todo, por hacer mi vida muy hermosa.

Wences:

Y te doy gracias a ti cielo, por que llegaste a mi vida en esta etapa, me has motivado ha terminar este trabajo, gracias por tu apoyo, amor y paciencia. Gracias, por que aparte que eres el amor de mi vida, eres mi mejor amigo. Te amo.

Mis amigos:

Le doy gracias a todos mis amigos, que de alguna manera han sido para mi una fuente de aprendizaje y experiencias nuevas, gracias por compartir conmigo mis sueños y estar ahi cuando he necesitado un consejo. Me siento muy afortunada de tenerlos como amigos.

Mis profesores:

Doy gracias a todos mis profesores, que me han transmitido todos sus conocimientos para desarrollar mi vida profesionalmente. Especialmente al Ing. Juan Carlos Roa, por que gracias a su apoyo, experiencia y conocimientos nos ayudó a lograr nuestra titulación.

Universidad Nacional Autónoma de México

Y finalmente, gracias a que existe una institución como nuestra Alma Mater, en la cual tenemos una fuente de conocimientos ilimitada, en la que tenemos la oportunidad de formarnos integralmente y crecer día con día.

NORMA ANGÉLICA HERNÁNDEZ ESPEJEL

A mis padres:

Por todo el amor y apoyo que de ellos he recibido en todos y cada uno de los momentos de mi vida, por creer en mí. por darme el ejemplo de ser siempre mejor, por que además de ser mis padres son mis mejores amigos, por la fe inquebrantable que siempre tuvieron en mí y con ello infundirme el valor y la confianza para seguir adelante. logrando así la culminación de mi más grande anhelo.

A mis hermanos:

Por todo el apoyo y comprensión que siempre he recibido de cada uno de ellos.

A mi esposa:

Por todo el amor, paciencia, apoyo, confianza y comprensión que de ella he recibido y a quien amo tanto.

A mis dos adorables hijos:

Lizbeth Aileen y Carlos David, quienes son la alegría de mi vida, a quienes amo tanto y por que ahora son el motivo y la razón que me hacen ser:
un mejor padre,
un mejor esposo,
un mejor hijo,
un mejor hermano,
un mejor amigo,
un mejor ser humano.

A mis amigos:

A todos y cada uno de mis amigos que de manera incondicional y desinteresada me han alentado a seguir siempre adelante.

A la U.N.A.M.

Por todos los años de albergue socio-cultural y profesional que nos brindo sin esperar más, que proporcionar a nuestra patria profesionales forjados por y para México.

A la Facultad de Ingeniería:

Por la oportunidad de ser nuestra segunda casa y la creadora de ética profesional en nosotros, siempre te recordaremos poniendo tu nombre muy en alto en cada una de nuestras acciones hacia la sociedad.

A mis profesores:

Por su esfuerzo, dedicación y paciencia para ser de nosotros, profesionales comprometidos con México y la sociedad.

Gustavo.

Dedico esta tesis con mucho cariño y respeto, a mis padres.

Hago manifiesto mi agradecimiento a mis padres, hermanas, esposa, hija, profesores, cuñados y amigos por su apoyo y contribución, que me ayudó a lograr la culminación de este ciclo profesional de estudios.

Mi agradecimiento también para la Universidad Nacional Autónoma de México por todo lo bueno que tiene para sus estudiantes.

Guadalupe Arturo Santamaria Murillo.

ÍNDICE

CAPÍTULO 3. DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA.

CAPÍTULO 1.

MARCO TEORICO.

1.1 CONCEPTOS BÁSICOS DE LA COMPENSACIÓN BANCARIA.

1.1.1 Proceso de compensación.

Tomemos como base las siguientes definiciones:

Compensación. (del latín compensatio) sust. fem. Acción y efecto de compensar.

Compensar (del latín compensare = pesar juntamente dos cosas hasta igualarlas) verbo trns. Neutralizar con una cosa el efecto de otra.

Con esto nos podemos dar una idea de en qué consiste la compensación de cualquier cosa, pero veamos los detalles de como se aplica a operaciones entre bancos.

Dentro de los servicios normales que presta cualquier banco, se encuentra el de permitir a sus clientes la apertura de cuentas. Para los fines de este trabajo, nos concentraremos en lo que son las cuentas de cheques. Una vez que el cliente llena la solicitud y cubre los requisitos necesarios, su cuenta es dada de alta y a más tardar en una semana le es entregada una chequera conteniendo de 20 a 50 cheques, con los cuales puede efectuar el pago por la adquisición de bienes y/o servicios en los lugares que aceptan este tipo de documentos como forma de pago.

Hagamos ahora la suposición de que tenemos un escenario en el cual existen sólo dos instituciones, el banco 1 y el banco 2. También supongamos que tenemos dos personajes, la persona 1 que tiene contratada una cuenta en el banco 1, y la persona 2, que recibe un pago con cheque, de la persona 1 (véase la figura 1.1.1.1).

Cuando la persona 1 le paga a la persona 2 con uno de estos documentos (a esto se le llama girar un cheque), tenemos varias situaciones posibles:

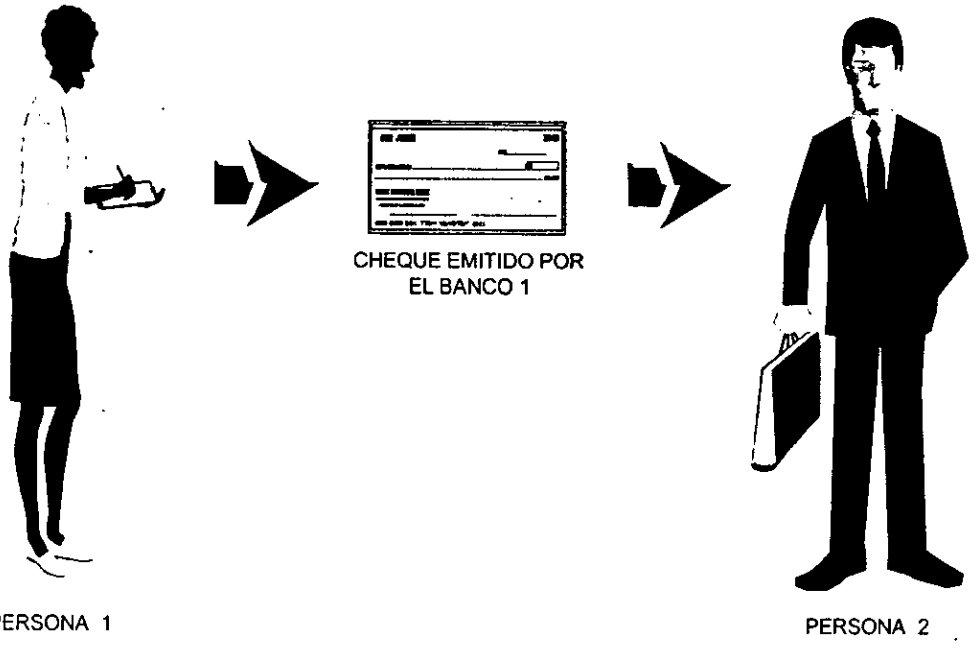


Figura 1.1.1.1 Pago con cheque.

a) Que la persona 2, que es la que recibe el cheque, se presente a cobrarlo en alguna sucursal del banco 1.

Cuando ocurra lo anterior, el cajero del banco 1, después de pedirle a la persona 2 que se identifique y demás, accesa el sistema propio del banco 1 y determina si el cheque que se esta cobrando tiene fondos suficientes, si la firma del cheque coincide con la registrada en el sistema y cualquier otra cuestión de rutina que hacen los cajeros antes de pagar un cheque. Una vez que determina que todo esta correcto, el cajero le paga a la persona 2 el importe del cheque en efectivo, la persona 2 se va y el cajero guarda el cheque, que como pertenece al propio banco 1, no saldrá fuera de la institución sino que irá directamente a los archivos del banco 1, para que se pueda efectuar cualquier aclaración o proceso posterior con el cheque. En este caso todavía no tenemos nada que ver con la compensación de cheques (véase la figura 1.1.1.2).

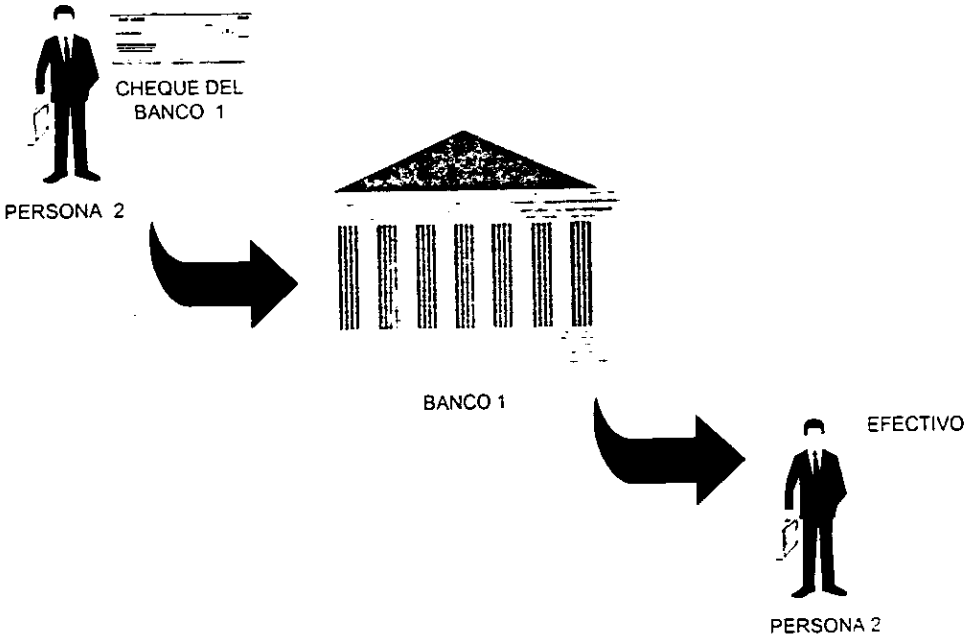


Figura 1.1.1.2 Cobro del cheque en efectivo, en el mismo banco que lo emitió.

b) Que la persona 2, deposite el documento en su cuenta de cheques que fue abierta también en el mismo banco 1.

Puede ser que la persona 2 no desee llevarse el efectivo por el momento y prefiera depositar el cheque en su propia cuenta. En este caso, como el deposito lo hace en su cuenta, que por coincidencia pertenece al mismo banco 1, el procedimiento para su cobro es igual que en el inciso a), ya que estos cheques una vez que se verifica dentro del mismo banco que son válidos, son tratados como si fuesen efectivo, sólo que en este caso al mismo tiempo que se hace el pago (abono, por usar el término técnico) en la cuenta de la persona 2, también se descuenta esta misma cantidad (técnicamente se dice que se efectúa un cargo) de la cuenta de la persona 1, que fue la que giro el cheque originalmente. Igual que en el inciso anterior, el cheque sigue estando dentro de la misma institución bancaria 1 y otra vez, esto no tiene que ver con lo que se conoce como compensación bancaria de cheques (véase la figura 1.1.1.3).

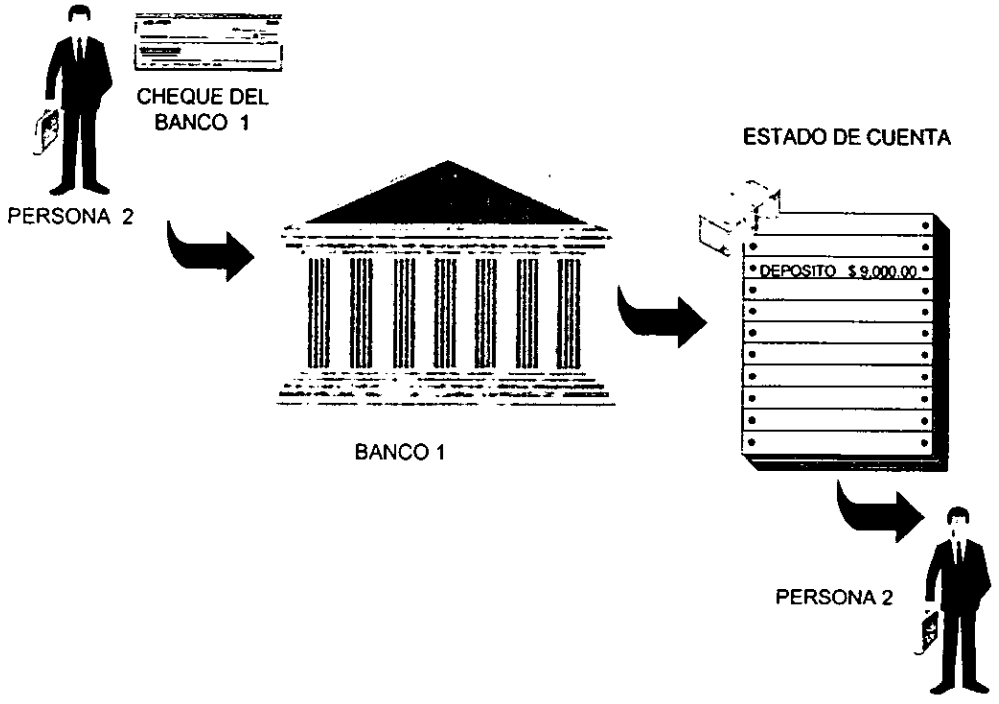


Figura 1.1.1.3 Depósito del cheque en cuenta, en el mismo que banco que lo emitió.

c) Que la persona 2 se presente a cobrar el documento en alguna sucursal del banco 2, donde ella tiene contratada una cuenta de cheques.

En este caso, el cajero del banco 2 no puede saber si la firma que está en el cheque es válida, ni siquiera sabe si la cuenta existe o no. ¿Por qué? Porque esta información sólo la posee el banco 1 que es a donde pertenece el cheque que la persona 2 quiere cobrar. Este tipo de cobro no se puede realizar directamente, y lo que le queda a la persona 2, es llevar a cabo lo que se explica en el siguiente inciso.

d) Que la persona 2 deposite el cheque para su cobro en la cuenta de cheques que tiene en el banco 2.

Dado que la persona 2 no puede cobrar directamente el cheque emitido por el banco 1 en el banco 2, la persona 2 lo deposita en su cuenta de cheques que tiene en el banco 2, para que el banco 2 trate de cobrarlo por ella. ¿Cómo que para que trate de cobrarlo? Sí, el banco 2 determinará, en conjunto con el banco 1 si este cheque que la persona 2 esta tratando de hacer efectivo en realidad lo es. ¿Y cómo se hace esto? Bueno, esto se hace por medio del proceso de compensación bancaria, que explicaremos en un momento. Por lo que toca a la persona 2, al efectuar el depósito del cheque, se le da un comprobante del depósito del mismo, con una leyenda que dice "SALVO BUEN COBRO" o "S.B.C.", y se le dice que al otro día puede tener disponible el dinero en su cuenta, "si todo sale bien" (véase la figura 1.1.1.4).

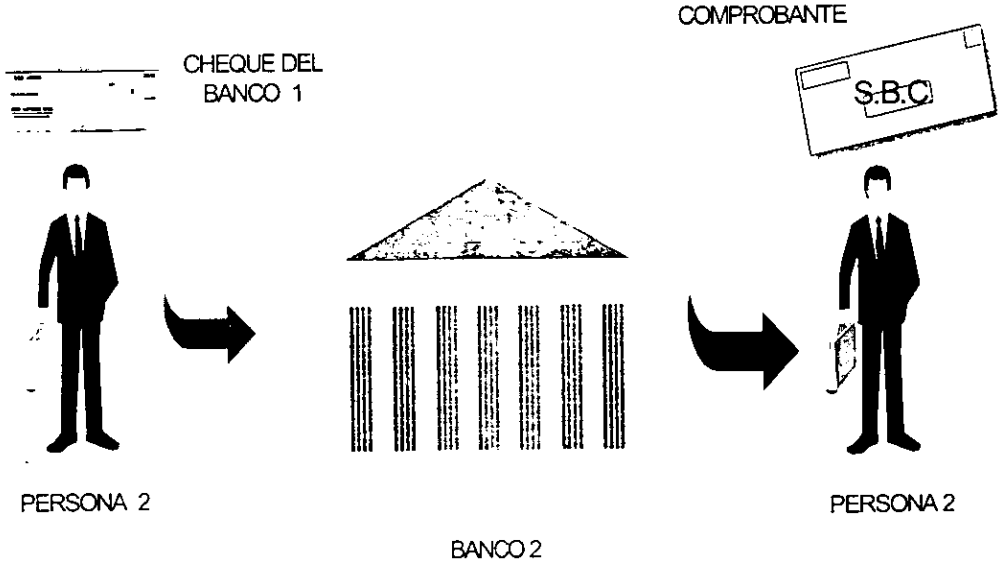


Figura 1.1.1.4 Depósito del cheque en cuenta de otro banco, diferente al emisor.

Haciendo una pausa aquí, la persona 2 podría hacer lo siguiente. Primero podría ir a una sucursal del banco 1 cobrar el cheque como se describe en el inciso a) y después trasladarse hasta la sucursal del banco 2, donde tiene su cuenta, en la que podría depositar el dinero en efectivo y habría logrado el mismo efecto. Bueno, esto no siempre resulta muy práctico y mucha gente prefiere depositar los cheques en su cuenta, aunque estos cheques así depositados, no siempre pertenecen al mismo banco en el que se hace el depósito del documento, como en el caso de este inciso. Existen además, cheques que no pueden ser cobrados en efectivo y sólo pueden ser depositados en una cuenta, este es el caso de los cheques marcados con la leyenda "PARA DEPÓSITO EN LA CUENTA DEL BENEFICIARIO", en esta situación, la persona que quiere cobrar el cheque no necesariamente debe abrir una cuenta en el banco que emitió el documento para así poder cobrarlo, sino que puede depositarlo en alguna otra cuenta de otro banco, que vuelve a ser el caso de nuestro inciso d).

Siguiendo dentro del marco de nuestro escenario descrito en los párrafos anteriores, tenemos ahora que el banco 2, una vez que ha concluido su horario de atención al público, manda a su delegado (una persona autorizada por el banco), al banco 1 para que verifique si ese cheque que su cliente quiere cobrar es válido, y se le puede pagar o no. Esto lo hace para darle el servicio a su cliente y evitarle que él se dirija personalmente primero. a cobrar el cheque al banco 1 y luego de regreso al banco 2 a realizar el depósito. como se explicó en el párrafo anterior.

El delegado del banco 2. se presenta en el banco 1 con el cheque del cliente del banco 2 (la persona 2). Se verifica que la cuenta existe, que la firma es válida, que el documento no ha sido alterado, que la cuenta tiene fondos suficientes, etc. De esta manera se determina que el cheque es válido y se puede abonar el importe del cheque, a la cuenta de la persona 2 en el banco 2. El procedimiento que se llevaría a cabo para realizar el cobro del cheque podría ser de esta manera: que el importe del cheque se le diera al delegado del banco 2 y después el mismo delegado regresara a su banco y se hiciera el deposito en la cuenta del cliente. Sin embargo esto sería riesgoso para el

banco 2 porque el delegado podría huir si la cantidad ameritara el riesgo. Lo que se puede hacer, es que el banco 2 abriera una cuenta en el banco 1. Cuando el delegado del banco 2, se presentara con el cheque de su cliente para cobrarlo, el importe del cheque se depositaría en la cuenta que el banco 2 tiene en el banco 1, luego el banco 2 depositaría el importe del cheque en la cuenta de su cliente (la persona 2). De esta manera el banco 2 recibe un abono por parte del banco 1 y a su vez existe un cargo a su cuenta por concepto del depósito realizado en la cuenta de su cliente. Con esto las cantidades operadas, quedan compensadas, y esto, es lo que en términos generales se conoce como compensación bancaria (véase la figura 1.1.1.5).

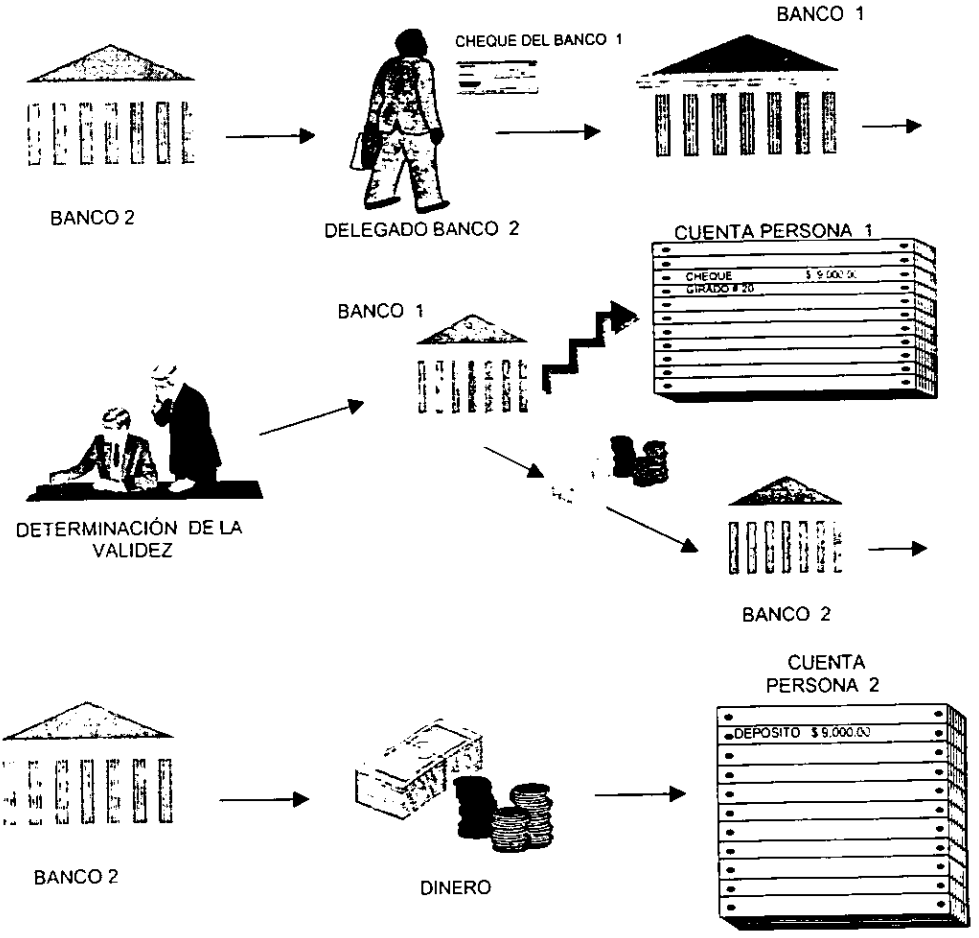


Figura 1.1.1.5 Compensación directa de un cheque.

Haciendo referencia a la definición que se encuentra al principio de esta sección, vemos aquí que el banco 1 realizó un abono en la cuenta que el banco 2 tiene en el banco 1, y que consistió en el pago del importe del cheque cobrado por la persona 2. También podemos observar que el banco 1 realizó un cargo a la cuenta de su cliente (la persona 1) que fue quien originalmente giró el cheque. Por otra parte, el banco 2 realizó un abono en la cuenta de su cliente (la persona 2). Este último abono fue hecho con fondos del propio banco 2, en este sentido, podemos decir, que el banco 2 recibió un cargo por la misma cantidad que abono a la cuenta de la persona 2. De esta manera, tenemos una compensación entre los montos correspondientes a los cargos y abonos, contablemente diríamos que las cantidades operadas están balanceadas (véase la figura 1.1.1.6).

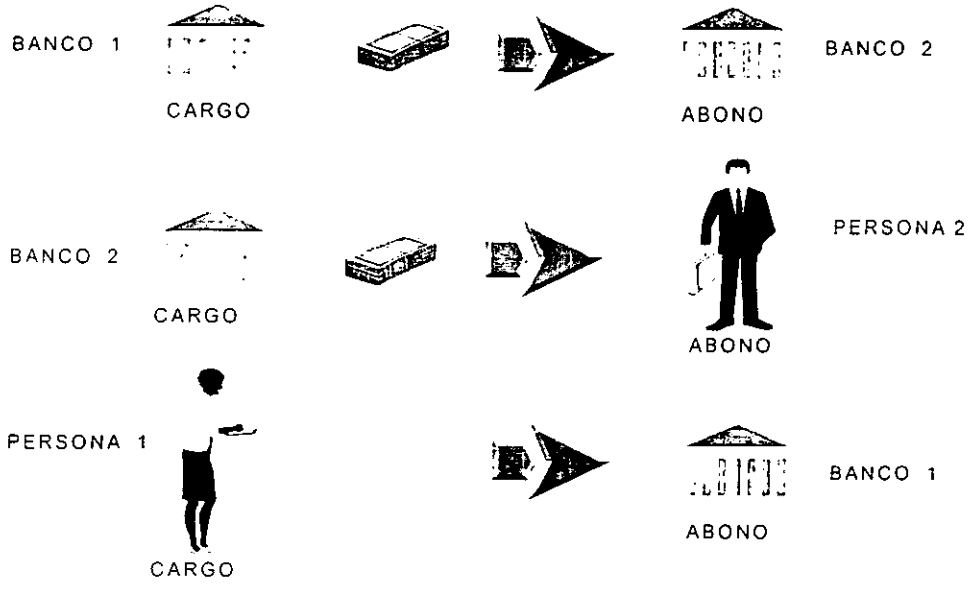


Figura 1.1.1.6 Resumen contable de operaciones.

Cabe señalar que aunque el ejemplo usado aquí se hizo utilizando cuentas de cheques, se puede dar el caso de compensación de valores, compensación de tarjetas de crédito, compensación de abonos, compensación de cargos, etc.; y todas ellas

involucrarían la compensación de las cantidades cargadas y abonadas entre los distintos participantes.

1.1.1.1 Compensación de cheques.

Después de la explicación dada en la sección anterior (1.1.1 Proceso de compensación), lo que resta por decir, es que la compensación de cheques es un subconjunto de los diferentes tipos de compensación que pueden existir, pero, aplicado en este caso particular, a los cheques. El ejemplo dado en la sección anterior sirve perfectamente para explicar lo que es en sí la compensación bancaria de cheques.

Ahora ampliemos un poco más nuestro panorama, en la sección anterior teníamos sólo dos bancos, dos personas y un cheque a cobrar. Supongamos ahora que tenemos cinco bancos. Como se pudo notar en la sección anterior, el banco 2 tenia una cuenta en el banco 1. Si el banco 1 quisiera prestar el mismo servicio que el banco 2 en cuanto al cobro de cheques, el banco 1 tendría también que tener una cuenta el banco 2, para que en ella le efectuaran los pagos de los cheques cobrados por él, al banco 2. En la situación de cinco bancos, tendríamos a cada banco con la necesidad de tener cinco cuentas, además de la necesidad de llevar el control de cada una de ellas. Si tomamos en cuenta que en México existen actualmente alrededor de cincuenta bancos entre nacionales y extranjeros, resultaría impráctico llevar a cabo la compensación de la manera en que se ilustró en la sección anterior, donde se hizo de esa manera con fines explicativos. También cabe señalar que la cantidad de cheques compensada diariamente esta en el orden de los cientos de miles de cheques, con lo cual el control de las cuentas sería todavía más complicado.

¿Como se lleva a cabo entonces la compensación de cheques en la realidad? Se lleva a cabo de la siguiente manera. Los bancos no abren una cuenta en cada uno de los otros bancos que participan en la compensación de cheques, simplemente se ponen de acuerdo y cada banco abre una cuenta en el "banco liquidador". El banco liquidador, es

el que realizará los cargos y los abonos correspondientes según los importes de los cheques operados por cada banco. El delegado de un banco cualquiera, no tiene que ir a cada uno de los otros bancos participantes a presentar los cheques para su cobro, sino que se presenta con todos los cheques de todos los otros bancos en la "cámara de compensación", que es donde se lleva a cabo el proceso de compensación (véase la sección 1.1.3 ¿Qué es la cámara de compensación?). De manera general, podemos decir que la cámara de compensación es el lugar donde se realiza el proceso de la información contenida en los cheques, para que se pueda realizar posteriormente la compensación de los cargos y abonos, por parte del banco liquidador (véase la figura 1.1.1.1.1).

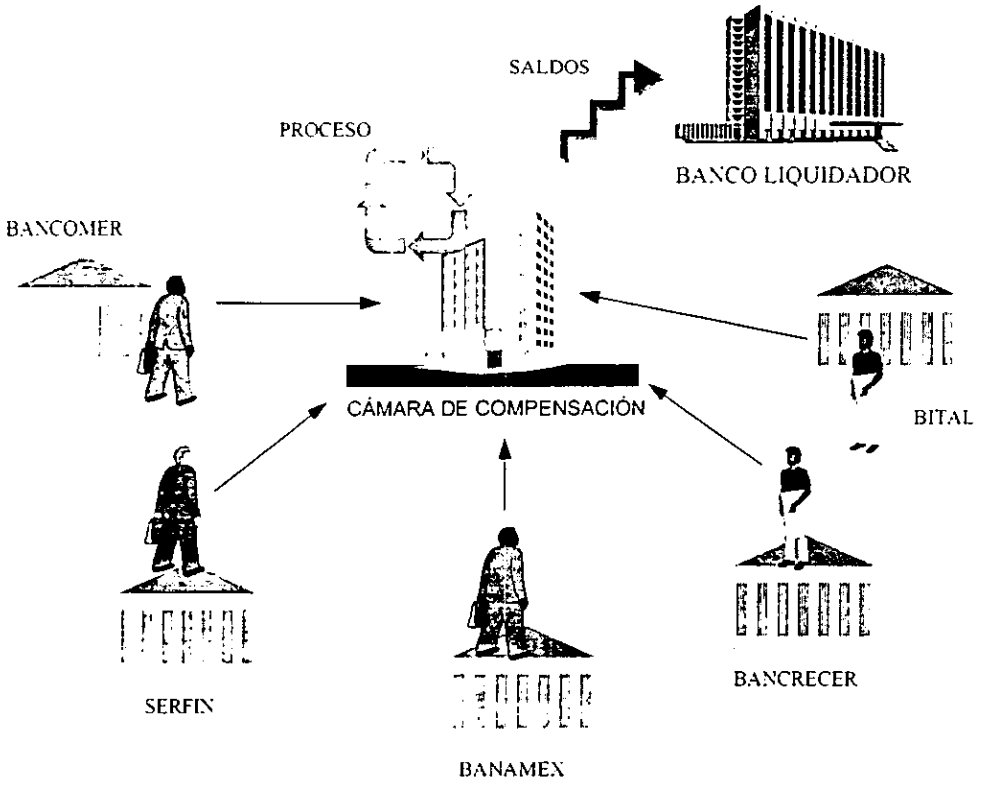


Figura 1.1.1.1.1 Compensación de cheques.

1.1.1.2 Intercambio.

El intercambio es el cambio físico de documentos entre bancos que se produce en la cámara de compensación. en el caso de la compensación de cheques, el tipo de documentos intercambiados son obviamente los cheques (véase la figura 1.1.1.2.1).

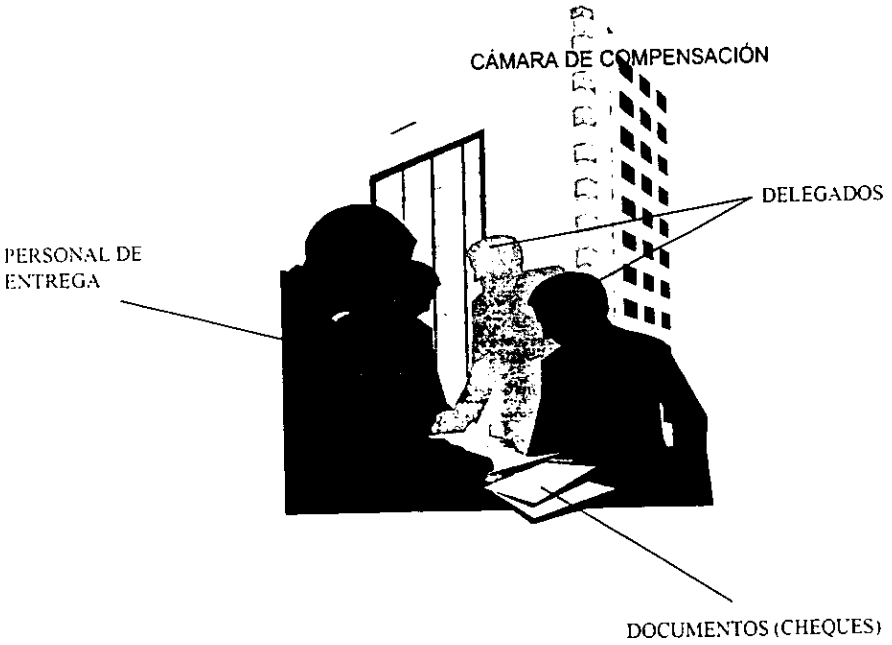


Figura 1.1.1.2.1 Intercambio de documentos.

Cada banco se presenta a la cámara de compensación con los cheques a cargo de los otros bancos, el banco que presenta o cede los cheques para su compensación, recibe el nombre de banco cedente. Los cheques presentados (cedidos) por un banco cualquiera corresponden a uno o más bancos llamados "girados", también se puede decir que el banco girado está "recibiendo" cheques por parte del banco cedente, es por esto que el banco girado también recibe el nombre de banco receptor. Resumiendo

lo expresado en este párrafo, podemos decir que en el proceso de compensación, un banco cualquiera, por un lado presenta cheques, y por otro lado, recibe cheques a través de otros bancos. Podemos decir, que las operaciones que se dan dentro del proceso de compensación a nivel global son: presentar cheques a cargo de otros bancos y recibir los cheques a cargo del propio banco. Los cheques presentados por un banco, reciben en conjunto el nombre de cheques cedidos, y los cheques recibidos por el banco reciben el nombre de cheques a su cargo (del banco que los recibe). Se dice que los cheques son "a su cargo" porque el banco girado debe pagar por ellos, es decir, el banco será cargado por concepto de estos documentos.

Durante el proceso de compensación hay muchos bancos cedentes y muchos bancos girados. Cualquier banco puede ser cedente por una parte y por otra parte puede ser girado por los demás bancos, de la diferencia entre la suma de los importes de los cheques presentados y la suma de los importes de los cheques recibidos, se obtiene un saldo o diferencia, que será la cantidad resultante de la compensación de cheques para ese banco en particular.

1.1.1.3 Truncamiento.

Dentro de las modalidades de la compensación de cheques, se encuentra el concepto del truncamiento. En el caso del truncamiento, existe un intercambio de información, en lugar de un intercambio de documentación. La documentación, es decir el papel físico, permanece en las manos del que lo presenta, que hemos llamado banco cedente, y no llega a las manos a las que esta destinado, que hemos llamado banco girado, o lo que es lo mismo, banco receptor (véase la figura 1.1.1.3.1).

En nuestro país no existe el truncamiento ya que como se han introducido muchas medidas de seguridad en los cheques, el banco receptor siempre prefiere recuperar los documentos a su cargo, para asegurarse de su autenticidad.

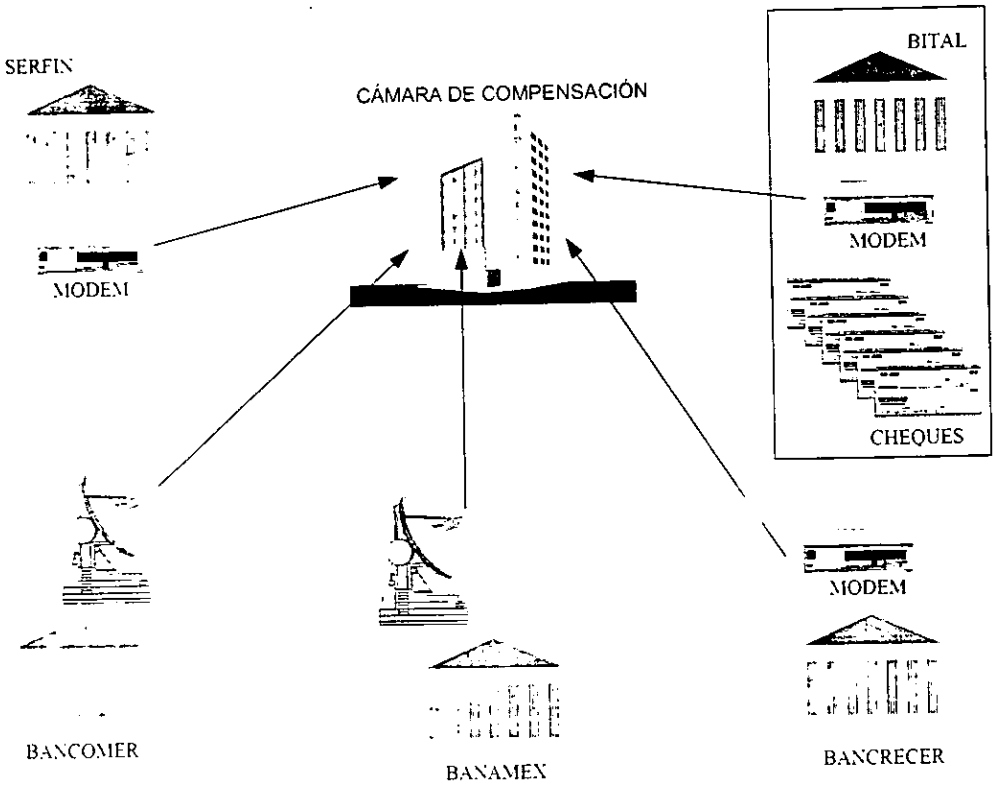


Figura 1.1.1.3.1 Truncamiento.

1.1.1.4 Intercambio y compensación directos.

Dado que la cámara de compensación cobra por cada cheque procesado, hay bancos que para reducir sus costos por concepto de compensación de cheques, han optado por hacer acuerdos y llevar a cabo la compensación entre ellos. de manera similar a como se explicó en el punto 1.1.1 (Proceso de compensación).

Aunque existen muchos bancos, como sucede en todos los tipos de negocios, existen empresas lideres que son las que tienen una mayor participación del mercado. En el

caso de las instituciones de crédito, es de esperarse que los bancos más importantes tengan la mayor cantidad de clientes y por lo tanto el mayor volumen de cheques operados durante un día cualquiera. Pongamos la situación de que tenemos muchos bancos, pero que dos de ellos son los más importantes de todos. Al final de un día normal de labores. esos dos bancos importantes tendrían cada uno, una gran cantidad de cheques del otro banco importante, puesto que los dos son bancos con muchos clientes y por lo tanto manejan muchos cheques diariamente. Aunque estos dos bancos importantes podrían llevar los cheques correspondientes a cada otro a la cámara de compensación. como lo hacen con los cheques de cualquier otro banco, pueden optar por hacer un acuerdo para intercambiar sus respectivos cheques y pagarse entre ellos, sin la intervención de la cámara de compensación ni del banco liquidador. La responsabilidad por el intercambio y compensación de estos cheques, queda entre los dos bancos, así como también cualquier arreglo de diferencias originadas por este proceso de intercambio y compensación directos.

1.1.2 Requerimientos para llevar a cabo la compensación.

Se puede decir que los requerimientos para llevar a cabo la compensación de cheques son, por lo menos la existencia de dos bancos que operen cuentas de cheques, como en el ejemplo de la sección 1.1.1 (Proceso de compensación). Ampliando esto un poco más. a una situación en la que existiera una mayor cantidad de bancos participantes, necesitaríamos. por conveniencia, un banco liquidador y una cámara de compensación. Obviamente necesitaríamos clientes en los bancos y personal que laborara tanto en los bancos como en el banco liquidador y en la cámara de compensación, además de la infraestructura necesaria para operar dentro de cada una de estas entidades. Dentro de la infraestructura necesaria en la cámara de compensación, es deseable la existencia de sistemas que lleven a cabo de manera automatizada, el proceso de la compensación de cheques.

1.1.3 ¿Qué es la cámara de compensación bancaria?

Tomemos la siguiente definición:

cámara (del latín vulgar camara < del griego kamara = bóveda, cuarto abovedado) sust. fem. cuarto especial de un edificio, destinado a algo importante.

En si la cámara de compensación es el edificio y conjunto de instalaciones donde se lleva a cabo el proceso de la información contenida en los cheques y la obtención de las cantidades resultantes para poder realizar la compensación de los cargos y abonos, en las cuentas de los bancos participantes, a través del banco liquidador. En México, hasta hace poco no existían cámaras de compensación que no fueran para compensación de cheques, y tal vez el nombre más adecuado para estas entidades sería: "cámara de compensación bancaria de cheques" (véase la figura 1.1.3.1).

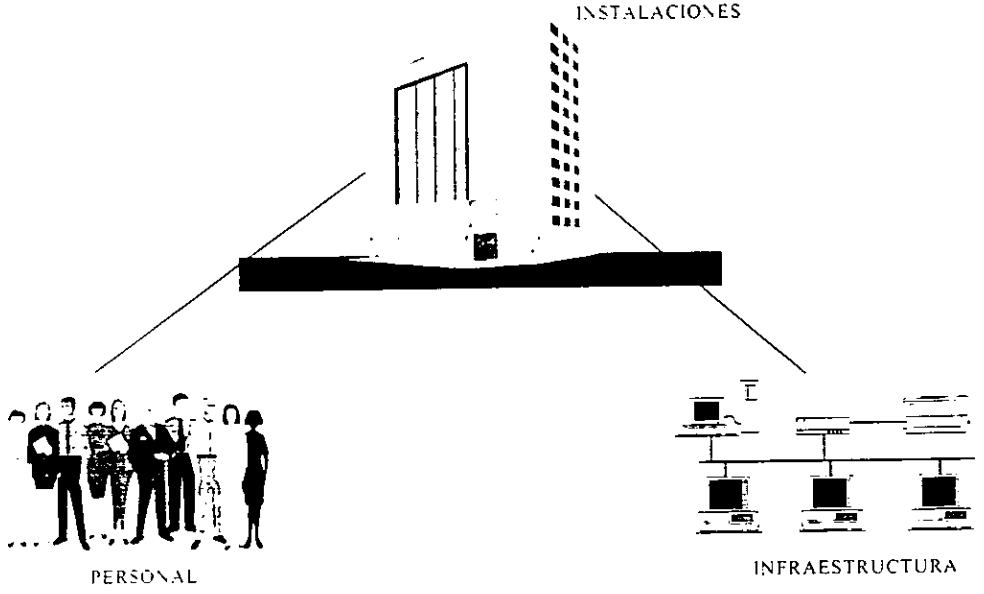


Figura 1.1.3.1 La cámara de compensación de cheques.

Es en la cámara de compensación, donde todas las tardes, después de que los bancos han dado por concluida su atención al público, llegan los delegados de cada banco participante en la compensación con todos los cheques que recibieron durante sus actividades de ese día, a cargo de otros bancos. En este lugar los cheques son procesados, por lo general, de manera automatizada, y al final del proceso se obtienen los resultados (en cifras) de la compensación, mismos que se pueden resumir en cargos y abonos que serán cobrados o pagados por el banco liquidador, a cada una de las instituciones participantes.

1.1.4 Funciones y atribuciones de la cámara de compensación bancaria.

Como es de esperarse, la principal función de la cámara de compensación bancaria es llevar a cabo el proceso de compensación de cheques y la obtención de las cifras que se cargaran o abonaran según sea el caso, a cada una de las instituciones participantes a través del banco liquidador. Dentro de esta función general se encuentran otras subfunciones, por llamarlas así, como son: la recepción de la documentación y de los cheques presentados por cada banco para su compensación; el proceso automatizado de los cheques; la generación de los resultados del proceso de compensación, dentro de los que se encuentran estados de cuenta, archivos con el detalle de los cheques recibidos por cada banco; obtención de las cantidades a compensarse por medio del banco liquidador; entre otros.

Otra de las funciones de la cámara de compensación es la de servir de apoyo a los bancos participantes en caso de aclaraciones con respecto a un cargo indebido en la cuenta de algún banco o como mediadora entre los bancos para solucionar dudas o arreglar diferencias entre los mismos, cuando se presenta un problema derivado del proceso de compensación. Dentro de esta función de apoyo a la banca, se encuentra la de prestar asesoría para los nuevos bancos que desean incorporarse a la compensación bancaria, así como de proveerles un ambiente de pruebas y validación para este fin (véase la figura 1.1.4.1).

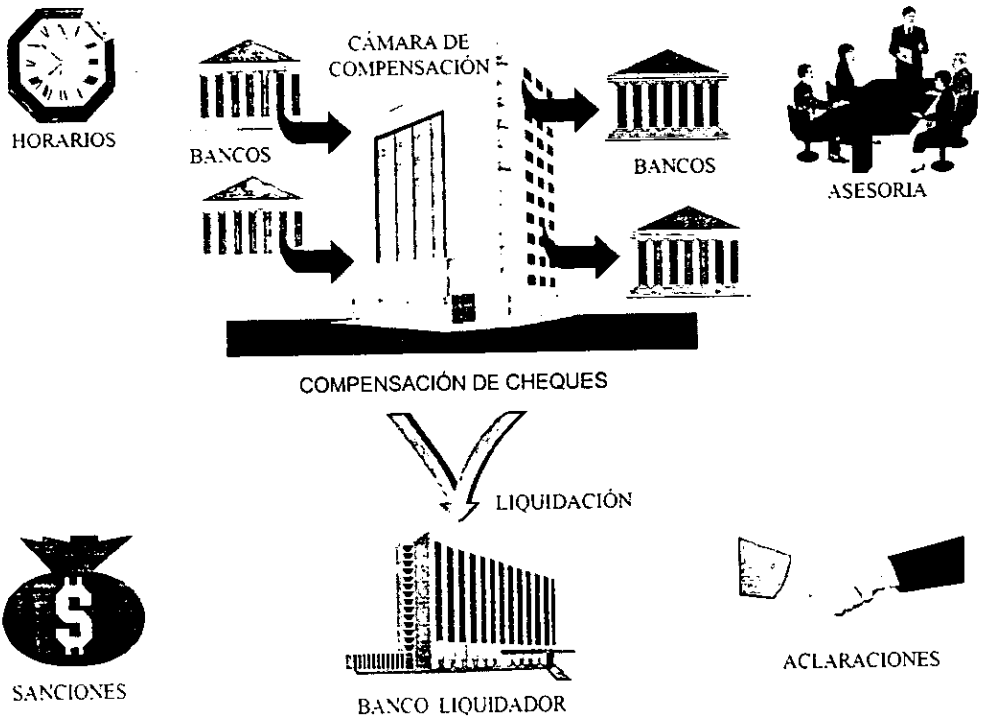


Figura 1.1.4.1 Funciones y atribuciones de la cámara de compensación.

Dentro de las atribuciones que tiene la cámara están la de hacer cumplir los acuerdos celebrados entre los bancos y la cámara de compensación, en cuanto a los horarios de recepción y entrega de los cheques; el formato en que se debe presentar la información; restringir el acceso a las instalaciones de la cámara, quedando permitido únicamente a los delegados de los bancos; el cobro de sanciones económicas cuando alguno de los bancos incurre en faltas a los lineamientos establecidos, como sería por ejemplo, por llegar más tarde del horario máximo permitido.

Otra atribución que tiene la cámara de compensación es la de no procesar la información presentada por un banco a causa de inconsistencias dentro de la misma, por ejemplo, si un banco presenta información para su proceso en un medio magnético,

y dice que contiene mil cheques por un valor de 10 millones de pesos y el sistema de software instalado en la cámara de compensación determina que estas cifras son inconsistentes con el contenido del archivo, entonces la cámara de compensación tiene derecho a no aceptar ese archivo presentado por el banco y le exigirá que presente otro archivo con la información correcta, o que corrija y se responsabilice por las cantidades que dice estar presentando para su proceso. También tiene la atribución de ampliar el horario en situaciones especiales como días de alto volumen de transacciones, como por ejemplo en los días de quincena o en los dias de "puente".

Supongamos que un banco se queda fuera por no presentarse dentro del horario establecido, en este caso, el banco tiene que arreglarse con cada otro de los bancos a los cuales les presenta documentos. Para solucionar esto, los bancos involucrados realizan una compensación entre ellos, de una manera similar a la que se ilustró en el ejemplo de la sección 1.1.1 (Proceso de compensación). En una situación como la anterior, no interviene ni la cámara de compensación, ni el banco liquidador, quedando la responsabilidad entre el banco que llego tarde y los bancos con los cuales se arreglo "por fuera", como se dice en el argot de la compensación.

1.1.5 Estructura orgánica de la cámara de compensación bancaria.

En México, la cámara de compensación bancaria se creó como un organismo perteneciente al banco central (Banco de México), que a la vez fungía como el banco liquidador de la compensación de las operaciones realizadas con cheques tanto en moneda nacional como en dólares.

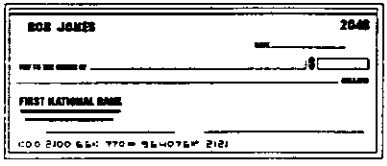
En este sentido, la cámara de compensación bancaria, heredó una estructura orgánica similar a la del Banco de México, que consiste en una estructura jerárquica que comienza en la dirección general, siguiendo con las subdirecciones por área, dentro de las subdirecciones tenemos oficinas y por último están los empleados de oficina.

En la actualidad la cámara de compensación se ha establecido a nivel nacional, con la apertura de numerosas sucursales en todo el país. No obstante, la administración de dichas sucursales, se lleva a cabo desde la matriz y no se han creado nuevas categorías dentro de la estructura jerárquica.

1.2 SISTEMAS AUTOMATIZADOS DE COMPENSACIÓN DE CHEQUES.

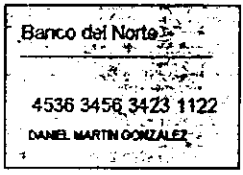
1.2.1 Antecedentes.

Fundamentalmente, los cheques son un instrumento de pago que satisface, de manera razonable, la necesidad de trasladar fondos en el ejercicio de la actividad económica (véase la figura 1.2.1.1).



CHEQUE

TARJETA DE CRÉDITO



EFECTIVO

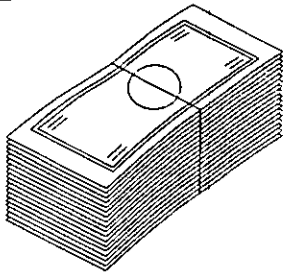


Figura 1.2.1.1 Instrumentos de pago comunes.

Los cheques cumplen con su objetivo gracias a las operaciones que realizan los bancos, de entre las cuales las principales serían las siguientes: preparación y entrega de chequeras, pago de cheques en ventanilla, recepción de depósitos, preparación de cheques para su envió a la cámara de compensación, presentación y recepción de cheques en la cámara de compensación, proceso de cargo de cheques en las cuentas

de los giradores, preparación de devoluciones, recepción de devoluciones y abonos en las cuentas de los depositantes.

Las instituciones bancarias realizan estas operaciones diariamente en grandes cantidades y para efectuar este proceso cuentan con diferentes sistemas, los cuales pueden clasificarse en los siguientes grupos, de acuerdo con el tipo de equipo de procesamiento empleado:

a) Sistemas computarizados con reconocimiento de caracteres magnetizables.
b) Sistemas computarizados.
c) Sistemas manuales.

a) Los sistemas computarizados con reconocimiento de caracteres magnetizables son, indudablemente, los más avanzados, ya que la captura de los datos se hace del documento mismo usando maquinas lectoras-clasificadoras (véase la sección 1.2.2.2.1 lecto-clasificadores).

b) En los sistemas computarizados los datos se transfieren manualmente de los cheques a los equipos de procesamiento electrónico de datos. Estos sistemas, presentan los problemas propios de los sistemas manuales, a saber: lentitud en la captura de los datos, captura sujeta a verificaciones laboriosas, pérdida de documentos mayor que cuando se trata de sistemas de captura automática y agudización de estos problemas cuando hay picos en la carga de trabajo.

c) Los sistemas manuales además de no ser compatibles con los anteriores, tienen menos recursos para solucionar esencialmente los mismos problemas, aunque operan volúmenes más pequeños de documentos.

En años anteriores existían, además de estas diferencias entre los sistemas usados por los bancos, problemas para automatizar los procesos de cheques que se originaban

por las diferencias que presentaban los propios documentos, dentro de las cuales las más notables eran: los diferentes diseños de chequeras, datos en los cheques ubicados en posiciones distintas, varios tamaños de cheques, empleo de una codificación diferente en los datos de los cheques ya que unos bancos usaban el código CMC7 y otros el E13B (véase la sección 1.3 Tecnología de caracteres MICR).

Finalmente, como complemento a la actividad de las instituciones bancarias, se realiza una función esencial para el sistema de cheques, que es, la compensación de cheques presentados por las instituciones en la cámara de compensación.

La cámara de compensación seguía un procedimiento manual debido a las diferencias entre los documentos compensables. La falta de automatización en la cámara de compensación genera en las instituciones bancarias, actividades muy laboriosas para la preparación de documentos compensables y para la captura de los datos.

Resumiendo lo anterior, vemos que aunque las ventajas del cheque como instrumento de pago son ciertamente indiscutibles, era un hecho que su manejo en ese entonces provocaba a la banca nacional, serios problemas para el procesamiento e intercambio de la información. Problemas que se debían esencialmente a los diferentes sistemas de procesamiento y a la existencia de diferentes tipos de cheques, por falta de estándares.

Podemos ver según lo anterior, que la compensación de cheques es uno de los procesos que hacen deseable su automatización, porque tiene beneficios de repercusión general para la banca.

Entre los beneficios más destacados podemos señalar los siguientes:

a) Simplicidad en la operación de cheques a todos los niveles, liberando de cargas administrativas y de trabajo manual al personal involucrado.

b) Optimo aprovechamiento de la capacidad instalada de los equipos de reconocimiento de caracteres magnetizables.

c) Instalación de equipos y sistemas automatizados en la cámara de compensación para, de igual manera, agilizar el proceso mismo de la compensación y por lo tanto el proceso de la información obtenida de los cheques que realizan los bancos a partir de los resultados que reciben de la cámara de compensación.

d) Como consecuencia de los puntos anteriores, tenemos la ventaja de una mejor atención y servicio para los usuarios finales de la banca, que son las personas que pagan y cobran los cheques.

Con esta problemática en mente, se empezaron a realizar esfuerzos tendientes a solucionarla, por medio del uso de sistemas automatizados de compensación de cheques, para lo cual se realizaron los estudios y evaluaciones pertinentes para la adquisición de los mismos.

Esencialmente, un sistema automatizado de compensación de cheques, consiste de máquinas que pueden leer la banda de caracteres magnetizables contenida en los cheques, transmitir dicha información a una computadora, donde posteriormente se puede hacer el proceso de la información capturada, hacer los cálculos pertinentes y obtener como salida principal un archivo de computadora que contiene los cheques a cargo de los diferentes bancos participantes así como el cómputo de las cantidades a cargo o a favor de cada banco, según la diferencia entre los importes de los cheques presentados y recibidos. Aparte de la lectura de los documentos, la cual se realiza a una gran velocidad, las máquinas que llevan a cabo la lectura de los cheques, realizan al mismo tiempo la clasificación de los mismos, según el banco girado del que se trate, lo cual facilita en mucho el manejo de los documentos, tomando en cuenta que esta separación física de los documentos se hacia en forma manual y representaba una gran cantidad de tiempo, esfuerzo y posibilidad de errores. En las secciones posteriores

entraremos más en detalle, en cada una de las fases que comprende el sistema automatizado de cheques.

Para los interesados en la historia de la compensación automatizada en nuestro país, tenemos los siguientes párrafos, hasta el inicio de la sección 1.2.2 (Preparación de la información).

Los Estados Unidos de América probablemente cuentan con una de las más vastas experiencias en la compensación automatizada de cheques. La American Bankers Association (Asociación Americana de Banqueros), publicó en 1959, The Common Machine Language (El Lenguaje Común de Máquina), que contiene el informe del Comité Técnico de Mecanización del Proceso de Cheques, que completó la fase del Common Machine Language dentro del proyecto de mecanización de cheques. La publicación proporciona especificaciones detalladas del caracter magnetizable E13B y da guías generales a los bancos, proveedores de equipos para proceso de cheques e impresores, para la implantación del proyecto.

Con el auxilio de la opinión de técnicos calificados, la Comisión Permanente de Automatización de la Asociación de Banqueros de México, llegó, por acuerdo unánime, a la conclusión de que era conveniente para la banca mexicana que se unificara el uso de caracteres magnetizables, y determinó que para el efecto se usara el código americano E13B, dadas sus características prácticas y de operación y considerando también los altos volúmenes de transacciones que manejan las instituciones mexicanas con bancos del extranjero. Simultáneamente, emitió los estándares en cuanto a tamaño, calidad de papel y distribución de datos en los cheques.

La Asociación de Banqueros de México después de evaluar, a través de su Comisión Permanente de Automatización, las posibilidades de automatizar en una primera etapa la cámara de compensación en la zona metropolitana de la cuidad de México, una vez lograda la estandarización del caracter magnetizable E13B, los campos y los códigos

que debe contener el cheque, solicitó al Banco de México su intervención y apoyo oficial, en su calidad de banco central, para la implantación de la cámara de compensación automatizada de la zona metropolitana.

Atendiendo a esta solicitud, el Banco de México estructuró un grupo de trabajo que se dio a la tarea de iniciar, en colaboración con la ABM (Asociación de Banqueros de México), la organización del centro automatizado de compensación de cheques, que se denominó Centro de Compensación Bancaria, CECOBAN.

Para determinar las posibilidades de implantar dicha automatización, se hizo una encuesta entre los principales proveedores de equipos de procesamiento electrónico de datos establecidos en México en aquel entonces. Esta encuesta se orientó a la identificación de equipos (hardware) y programas (software) que pudieran ser útiles al propósito de la automatización de la compensación de cheques.

Los proveedores incluidos en la encuesta, efectuada en 1976, fueron Burroughs, Honeywell Sistemas de Información, Control Data de México, IBM de México, NCR de México, y Sperry Univac Mexicana. De esta encuesta se desprendió que con la excepción de Control Data de México y de Univac, los proveedores tenían, en principio, las facilidades de hardware y de software para automatizar el servicio de compensación de cheques.

Se invitó a los proveedores a un concurso para la implantación del sistema de compensación de cheques mismo que, después de diversos análisis y consideraciones, se le adjudicó a la compañía Honeywell Sistemas de Información, que fue la que propuso el sistema que cumplía mejor con las premisas fundamentales establecidas por el grupo encargado de la automatización de la cámara de compensación que fueron: seguridad de operación, capacidad de crecimiento y eficiencia. Por cierto, este sistema, al igual que otros de los finalistas, eran sistemas que ya habían sido probados ampliamente en producción real en cámaras de compensación de Estados Unidos, sin

embargo tuvo que ser adecuado a las necesidades específicas de la cámara de compensación de la zona metropolitana, que hasta 1998 fue la única en México que proporcionaba el servicio de compensación automatizada de cheques. En realidad las "otras" cámaras de compensación que prestan este servicio a nivel nacional, son sucursales de esta misma cámara de la zona metropolitana, pero en este momento estamos hablando de cámaras de compensación y no de empresas de compensación.

El sistema antes mencionado, aunque tuvo cierto éxito, duro sólo unos cuantos años y fue sustituido más tarde, por el equipo minicomputador Honeywell DPS 6 que en conjunto con el software denominado MICR Plus II, realizaba el proceso de la compensación automatizada de cheques. Este sistema fue muy exitoso y estuvo en operación por poco más de una década (el software era una aplicación desarrollada en COBOL). Permaneció en operación hasta finales de 1998.

Finalmente el sistema en uso a partir del año 1999, esta conformado por el software Image Visión (escrito en C++), el cual opera sobre una red LAN de microcomputadoras que corren bajo Windows NT.

Cabe destacar aquí que existe una parte de hardware del sistema que se ha mantenido en operación desde el principio. Se trata de las máquinas lecto-clasificadoras de cheques (véase la sección 1.2.2.2.1 Lecto-clasificadores).

En esencia estos distintos sistemas de compensación automatizada de cheques llevan a cabo las mismas funciones, cada nueva generación proporciona nuevas facilidades tanto en su operación como en su mantenimiento debido a que aprovechan las ventajas que traen consigo los últimos avances tecnológicos en materia de computación. Sin embargo, los conceptos sobre la compensación y de lo que debe hacer un sistema de compensación automatizada, permanecen sin cambio.

1.2.2 Preparación de la información.

Hemos hablado anteriormente de introducir en una computadora la información contenida en la banda de caracteres magnetizables de los cheques, para su posterior procesamiento. Para realizar lo anterior, se siguen varios pasos que se describen a continuación.

Primeramente, cada vez que el cajero de una sucursal bancaria recibe un cheque que no es propio del banco, lo separa, ya que sabe que estos cheques son para compensación. Periódicamente, alguna persona dentro de la sucursal, pasa a cada caja a recoger los cheques a cargo de otros bancos y los va reuniendo en pequeños grupos a los cuales se antepone una tarjeta de papel grueso como de cartulina, la cual contiene datos para control como son: el número de la sucursal, el número de cheques agrupados y el importe total de los mismos. Este grupo de cheques, reciben en conjunto, el nombre de lote. Cada lote consiste de hasta 200 cheques aproximadamente, esto ayuda a tener un mejor control y poder localizar algún error en caso de alguna cifra errónea, que se detecte posteriormente en el proceso de los cheques.

La sucursal del banco, una vez que termina su horario de atención al público, envía los cheques que presentará para su compensación a una oficina concentradora del banco, que es donde se reúnen todos los cheques de las sucursales ubicadas dentro de una zona geográfica que se denomina plaza de compensación. En esta oficina concentradora del banco, se lleva a cabo la preparación de la información que será enviada posteriormente a la cámara de compensación.

Brevemente, mencionaremos aquí algunos datos pertinentes a la banda de caracteres magnetizables presente en los cheques. El banco imprime la chequera para el cliente sólo con parte de la banda de caracteres magnetizables. Dentro de los datos que se

imprimen, se encuentran el número de cuenta, el banco que emite la chequera (codificado en tres dígitos), el número de cheque, un código de seguridad, entre otros.

Adicionalmente el cheque tiene, dentro de su banda de caracteres magnetizables, un espacio destinado a la impresión del importe del mismo. Debido a que al momento de imprimir la chequera, no se sabe por cuanto importe será girado cada cheque, este campo se deja en blanco y se imprimen en el cheque sólo los datos conocidos en ese momento, los cuales reciben en conjunto el nombre de premarcaje o premarcado. Cuando el cheque llega a la oficina concentradora del banco, ya tiene escrito con letra y número, el importe del cheque. El primer paso dentro de la oficina concentradora del banco consiste entonces, en realizar lo que se conoce como el postmarcaje o postmarcado de los cheques. En esta fase se imprime en el campo correspondiente, el importe del cheque con caracteres magnetizables, quedando de esta manera completada la impresión de la banda de información codificada en dichos documentos (véase la figura 1.2.2.1).

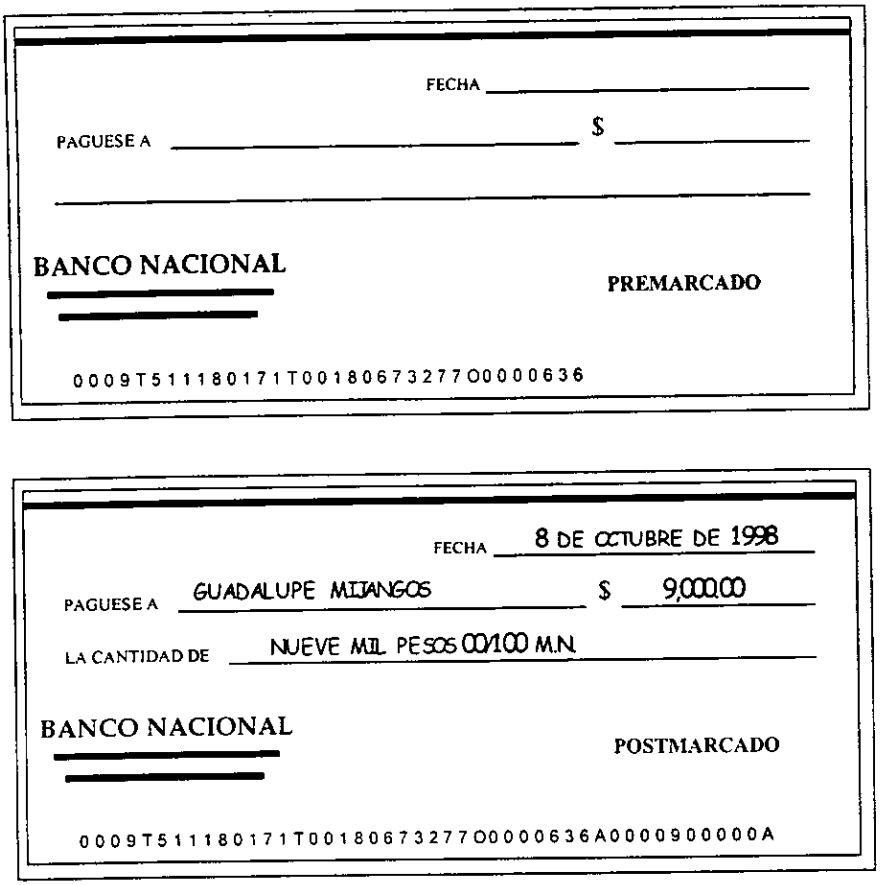


Figura 1.2.2.1 Cheque premarcado y cheque postmarcado.

El postmarcaje se realiza en maquinas conocidas como postmarcadoras, y aunque en realidad, estas máquinas pueden imprimir toda la banda del cheque, el premarcado generalmente se lleva a cabo en máquinas de imprenta especiales para ese fin. Las máquinas postmarcadoras pueden llevar un registro, en una tira de papel, de los importes capturados, similar a la manera en que lo hace una sumadora normal de oficina. Al final del postmarcado de los cheques de un lote, se obtiene, de la misma postmarcadora, la suma de los importes de los cheques portmarcados, misma que se utiliza como una cifra de control de este proceso de postmarcación. La tira de papel con

los importes de los cheques postmarcados y la suma total, se conocen como tira de postmarcadora.

Por lo general y dependiendo del número de documentos, son varias personas las que llevan a cabo esta fase de postmarcado. Cada persona se ocupa de un lote a la vez, el cual le es asignado para su postmarcaje. Después de postmarcar cada lote, el capturista verifica que la suma total de los documentos que acaba de postmarcar, coincida con la suma obtenida manualmente por la sucursal que envió el lote de cheques. En caso de alguna diferencia se determina donde ocurrió el error, si en la sucursal o en el postmarcado y se corrige cualquiera de ellas. Una vez que se obtiene la cifra correcta, se produce, haciendo uso del equipo postmarcador, un volante (un documento parecido a un cheque) de control para el lote, el cual contiene el importe total de los documentos incluidos en el lote, una clave que lo identifica como documento de control y una clave que lo identifica como volante de lote. Lo anterior, se hace con el fin de que el sistema, después de realizar su propio cálculo, pueda encontrar si existe una diferencia entre las dos cantidades: la marcada en el volante de lote y las cantidades sumadas directamente a partir de los documentos por parte del sistema de compensación automatizada de cheques (véase la figura 1.2.2.2).

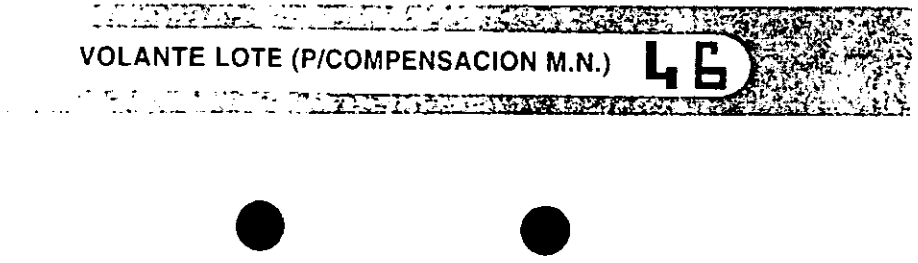


Figura 1.2.2.2 Volante de lote.

Supongamos ahora una situación en la que pudieran existir diferencias entre lo que sumó la sucursal y lo que reporta el equipo de postmarcación. Digamos que el error se produjo en la sucursal a la hora de hacer la suma. En este caso, dado que los importes postmarcados en los cheques son correctos y dado que la suma obtenida en la sucursal no es enviada por el banco a la cámara de compensación sino solamente la tira de postmarcadora (que también esta correcta), entonces, el problema se reduce a verificar cuál fue el cheque mal sumado; verificar que fue bien postmarcado y decirles a los de la sucursal que tengan más cuidado. En el caso de que el error sea en el postmarcado de algún cheque, al final, se hace una anotación en la tira de postmarcadora, del cheque mal capturado y se anota a mano el total correcto del lote. En esta situación, debido a que no es conveniente corregir el cheque una vez postmarcado, existirá una diferencia entre la cantidad calculada a partir de los cheques por parte del sistema de compensación y la cantidad reportada por el banco, diferencia que será detectada posteriormente en la cámara de compensación. Retomaremos más adelante, lo que se hace en el sistema de compensación automatizada, cuando nos encontramos ante una de estas situaciones de diferencia de cifras.

Con los lotes de cheques ya postmarcados adecuadamente, se forman grupos de lotes que reciben el nombre de remesa. Una remesa consta de hasta 15 lotes, y aunque este no es un número de lotes fijo, en la práctica, una cantidad muy grande de lotes reduce el desempeño del sistema al realizar búsquedas, correcciones, etc., además de que se pone en riesgo una mayor cantidad de datos, en el caso de una caída del sistema y de la posible pérdida de información.

A toda la remesa se le antepone otro volante de control adicional, llamado volante de remesa, el cual contiene un número que identifica a la remesa, el importe total de los documentos incluidos en la remesa, una clave que lo identifica como documento de control y una clave que lo identifica como volante de remesa (véase la figura 1.2.2.3).

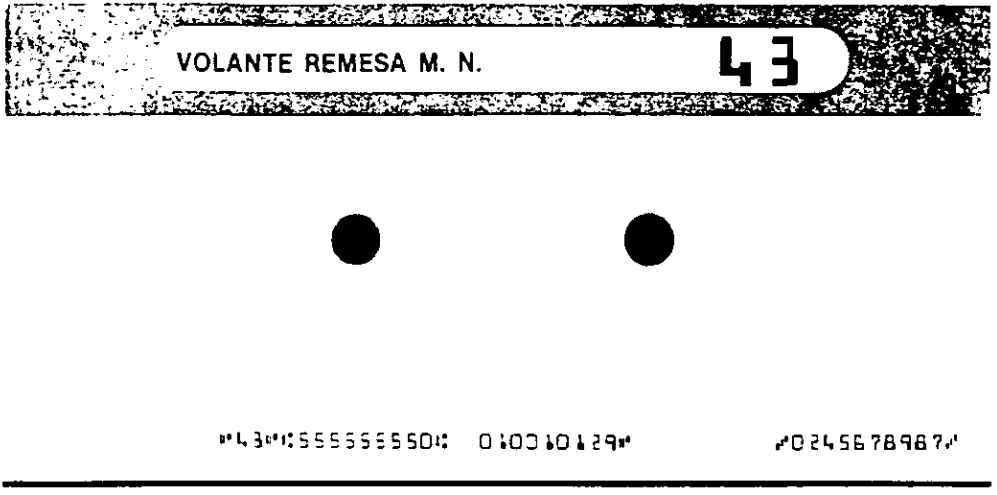


Figura 1.2.2.3 Volante de remesa.

El banco prepara además de lo anterior, un resumen de los lotes contenidos en la remesa, el número de documentos y el importe correspondiente a cada lote, así como el total global de documentos e importes de la remesa. Dicho resumen, es presentado en un formato especial, conocido como carta de remesa o carta remesa. La carta remesa y las tiras de postmarcadora de cada lote se introducen en una bolsa de plástico, que a su vez se introduce en una caja que contiene los cheques ordenados como se describió antes, y esto es lo que se necesita para poder presentar los documentos para su proceso automatizado en la cámara de compensación (véase la figura 1.2.2.4).

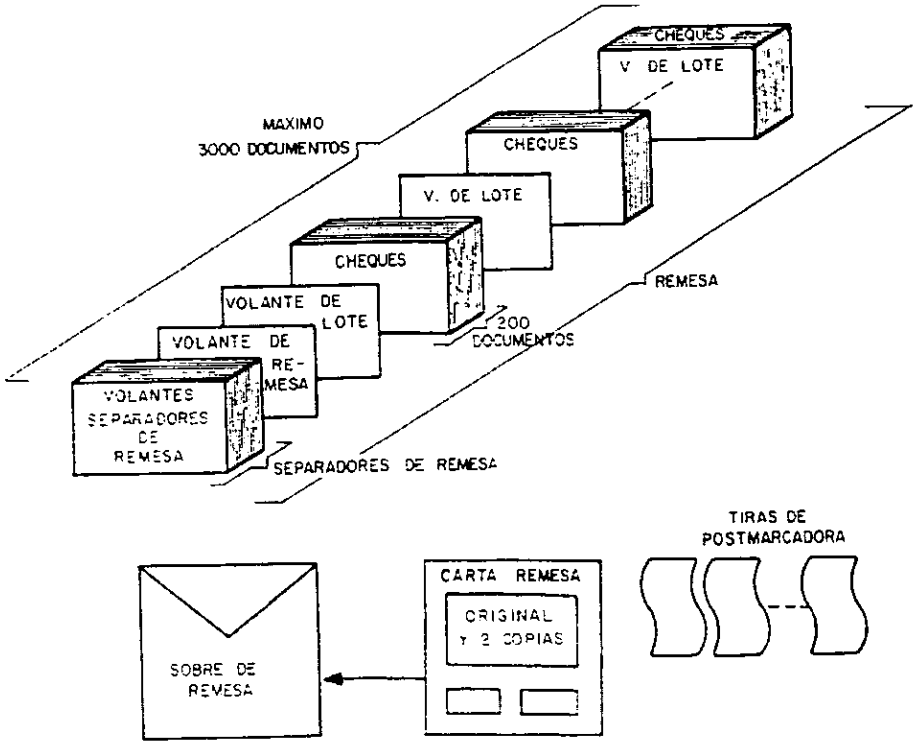


Figura 1.2.2.4 Preparación de la remesa de entrada al sistema.

Veamos a continuación la siguiente fase del proceso, que es la lectura de cheques.

1.2.2.1 Lectura de la información.

Hemos visto en las secciones anteriores, que una parte importante dentro de todo el proceso de compensación automatizada de cheques, consiste en capturar la información contenida en la banda de caracteres magnetizables de los cheques de una manera ágil y confiable. A causa de los altos volúmenes que se manejan, no sería conveniente que la información contenida en los cheques fuera capturada manualmente haciendo uso de un teclado, ya que sería un proceso muy tardado y sujeto a una gran cantidad de errores. Es por esto que la entrada de los datos a la

computadora se hace por medio de una "lectura", la cual consiste en un reconocimiento de caracteres. En este caso, el reconocimiento de los caracteres, no se realiza por medio de la técnica conocida como Optical Character Recognition (Reconocimiento Óptico de Caracteres), en la cual, se leen patrones de puntos, los cuales se comparan con patrones previamente definidos, para saber que fue lo que se leyó. Lo que se usa aquí, es la técnica MICR o Magnetic Ink Character Recognition (Reconocimiento de Caracteres de Tinta Magnética), en la cual, cada caracter, según su forma, produce una señal que se compara con patrones de señales previamente definidos, para llevar a cabo la identificación del caracter o símbolo leído.

En la sección 1.2.2 (Preparación de la información), se menciona que los bancos presentan los cheques para su compensación, dentro de cajas que contienen lo que se conoce como remesa. Posteriormente, en la cámara de compensación, estas remesas son llevadas al área de lectura de documentos, donde son alimentadas al sistema. En esta área de lectura, las remesas se leen en grupos de hasta siete de ellas y este conjunto, a su vez, recibe el nombre de entry (entrada). De esta manera los cheques quedan agrupados dentro del sistema en entries (entradas). Resumiendo, un entry se compone de varias remesas, cada una de las cuales contiene varios lotes.

La lectura se lleva a cabo de la siguiente manera: se van tomando los documentos de las cajas presentadas por los bancos, se tiene cuidado de que todos los documentos de control estén presentes y en su posición correcta, el operador de la lectora coloca los cheques en una pequeña máquina conocida como "vibrador", que precisamente, por medio de vibraciones, hace que los documentos se acomoden uniformemente con respecto a su lado inferior, para facilitar el proceso de su lectura. Una vez vibrados, los documentos se colocan en una especie de depósito mismo que tiene un declive y una pesa que presiona los cheques contra una rueda que impulsa los cheques a través de una serie de rodillos que los transportan hasta donde se encuentra una cabeza lectora sensible al magnetismo, similar a una cabeza lectora de cintas magnéticas (véase la figura 1.2.2.1.1). Esta cabeza lectora capta las señales originadas a causa del paso,

por debajo de la misma, de los caracteres magnetizables impresos en los cheques. Aquí haremos la aclaración de que los caracteres no tienen magnetismo de por sí, sino que están impresos con tinta magnetizable, es decir que en presencia de un campo magnético, las partículas de la tinta responden al estímulo del campo y se orientan siguiendo las leyes físicas del magnetismo. De acuerdo con la forma del caracter impreso con este tipo de tinta, y una vez de que se ha sensibilizado, se genera una señal distinta para cada caracter. Dicha señal, es traducida en un caracter específico, por medio de una combinación de hardware y software dentro de la lectora. Existe antes de la cabeza lectora un dispositivo que activa, por medio de la aplicación de un campo magnético, la tinta impresa en los caracteres magnetizables de los cheques y por eso, estos caracteres se llaman magnetizables y no magnéticos.

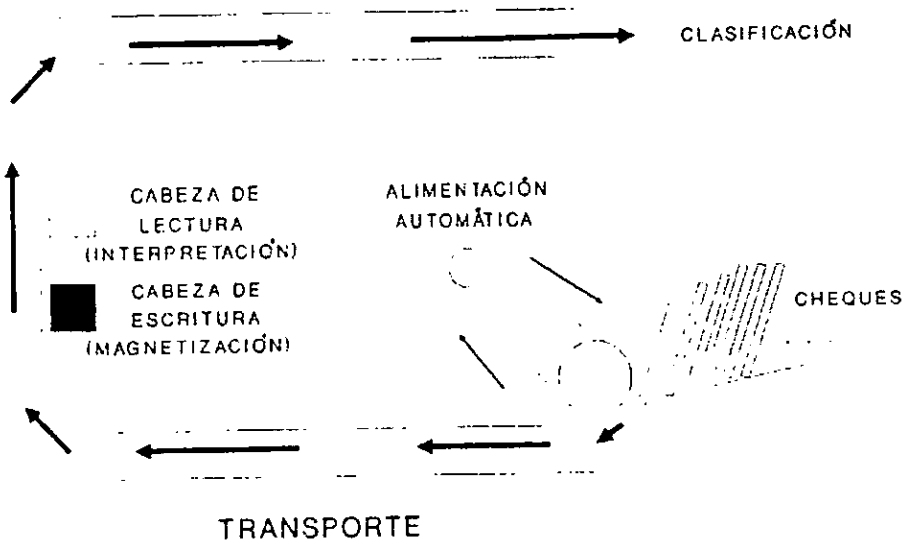


Figura 1.2.2.1.1 Lectura automatizada de cheques.

Hasta aquí, la máquina ya ha leído los caracteres impresos en el cheque a través de su cabeza lectora de caracteres magnetizables según lo anterior. Ahora, el cheque ha pasado más allá del punto en que se encuentra la cabeza lectora y en fracciones de segundo debe ser clasificado. La clasificación consiste en que la máquina lecto-clasificadora interprete la información contenida en el cheque y basándose en dicha

información, envia el documento a un casillero. Haciendo una analogía con un sistema manual, se le pediría a una persona que los cheques del banco 1 los colocara en el casillero 1, los del banco 2 en el casillero 2, y así sucesivamente. Para este fin, existe un programa que se carga en la memoria de la lectora, el cual, usando un dato impreso en el cheque (el número del banco girado), determina en que casillero debe caer. La máquina entonces, activa electrónicamente diversos mecanismos que guían el cheque hasta su casillero. Este camino a través del cual se mueve el cheque desde que es alimentado hasta que llega a su casillero, se conoce como transporte.

Durante el proceso de lectura se van produciendo archivos de computadora con la información correspondiente a los cheques leídos. Cada archivo corresponde a un entry.

Al momento de la lectura pueden existir errores en la captura automática de la información contenida en la banda del cheque, mismos que son detectados por el software de la lectora. El campo correspondiente al caracter con error, es marcado lógicamente en el archivo como un campo erróneo. Estos errores se deben principalmente a deficiencias en la impresión de la banda de caracteres magnetizables del cheque. Los cheques con error o errores son enviados por la lecto-clasificadora a un casillero especial, llamado casillero de rechazos. Los cheques con error, reciben en conjunto el nombre de rechazos y se les da un tratamiento adicional de corrección, como se verá en secciones posteriores. Los cheques que son leídos sin errores detectados por el sistema, reciben el nombre de cheques killed (muertos), como si los hubiéramos matado de un sólo tiro.

Los archivos que contienen los registros de los cheques leídos, recibirán un procesamiento adicional en las siguientes fases de la compensación automatizada de cheques, como veremos más adelante.

1.2.2.2 Lectores de cheques.

Como se había mencionado brevemente en la sección 1.2.1 (Antecedentes), uno de los elementos presentes en todos los sistemas automatizados de compensación de cheques, es una máquina que lee la información de la banda de caracteres magnetizables de los cheques. Con este tipo de máquinas, la información capturada puede introducirse directamente a un equipo de procesamiento de datos de una manera confiable y rápida, las cuales, son dos características deseables de los equipos lectores de cheques. Existen básicamente dos tipos de estas máquinas, mismos que se mencionan a continuación.

1.2.2.2.1 Lecto-clasificadores.

La palabra compuesta lecto-clasificadora, es una contracción de la palabra lectora-clasificadora. Lectora, por que se trata de una máquina que interpreta, como si la hubiera leído, la información impresa en la banda de caracteres magnetizables y clasificadora, porque de acuerdo a la información obtenida del cheque durante el proceso de su lectura, la máquina realiza una clasificación física de los documentos (véase la figura 1.2.2.2.1.1).

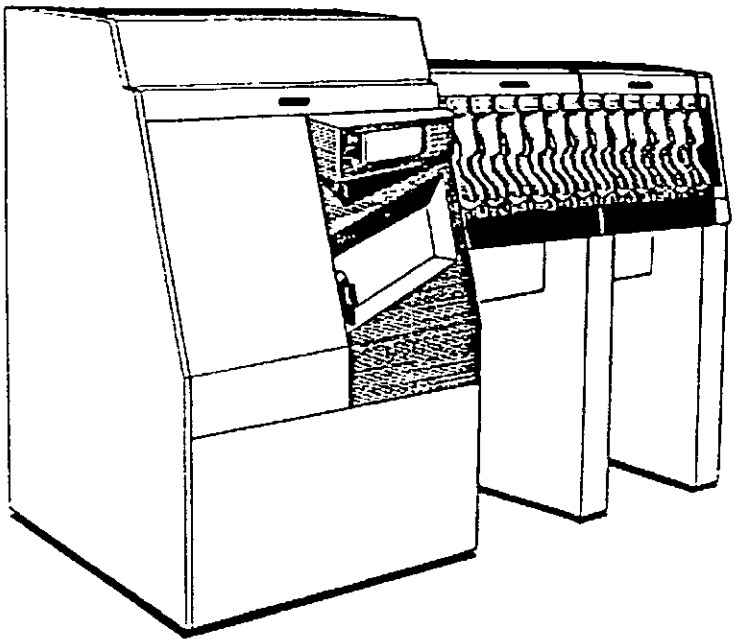


Figura 1.2.2.2.1.1 Lecto-clasificadora de documentos.

Estas máquinas se fabrican en varios modelos y capacidades de procesamiento. Para los fines de una cámara automatizada de compensación que maneje una cantidad de cheques respetable. una máquina aceptable, sería una que procesara arriba de 2000 documentos por minuto. Esta capacidad de procesamiento de información, podría repartirse entre varias máquinas, por ejemplo, tres máquinas que leyeran cada una, alrededor de 700 documentos por minuto. De hecho, no es recomendable tener sólo un equipo lecto-clasificador de 2000 documentos por minuto, debido a que el costo de estos equipos aumenta proporcionalmente de acuerdo al número de los documentos procesados por unidad de tiempo. Además, en el caso de que fallara el único equipo disponible en la cámara de compensación, tendríamos que esperar un tiempo razonable para su compostura y nos quedaríamos sin proceso de compensación automatizada. lo cual simplemente no puede ser posible. ¿Le gustaría ir al banco a preguntar por qué no puede disponer del dinero que deposito hace tres días, y que le

dijeran que es porque se descompuso la única lecto-clasificadora de la cámara de compensación o alguna otra parte del sistema?

1.2.2.2.2 Lectores unitarios

Existen en el mercado, pequeños equipos que realizan la lectura de la banda de caracteres magnetizables de los cheques. Esta función, la llevan a cabo de la misma manera que las lecto-clasificadoras grandes, es decir, los caracteres son sensibilizados por medio de un pequeño imán, después los caracteres son leídos y traducidos en un código que identifica a cada símbolo. Aquí, la persona que usa el lector alimenta manualmente, uno por uno, los cheques de los cuales quiere capturar la banda de caracteres magnetizables. Debido a que los documentos se leen uno a uno, estos equipos reciben el nombre de lectores unitarios. Estos dispositivos son de dimensiones reducidas, por esta razón, no cuentan con un módulo de clasificación de cheques por casillero. Por lo general, este tipo de lectores se utiliza en las ventanillas de las sucursales bancarias, como un dispositivo de entrada automática de datos. De esta manera, se evitan errores al capturar la banda de caracteres magnetizables de los cheques (véase la figura 1.2.2.2.2.1).

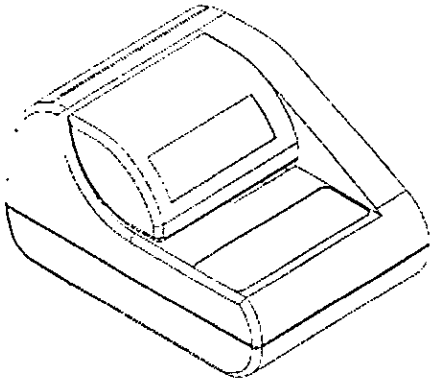


Figura 1.2.2.2.2.1 Lector unitario de cheques.

1.2.4 Proceso de la información.

El sistema de compensación automatizada de cheques opera dando énfasis a la velocidad y continuidad del proceso, sin detrimento de la función de control de la información. Para el efecto, se sirve de recursos tanto administrativos como de hardware y software, que permiten alcanzar esta meta. En esta sección se indica la forma en que se lleva a cabo el proceso de la información que se ha introducido al sistema por medio de las lecto-clasificadoras de documentos.

Lo primero que ocurre después de la lectura de los documentos, que ahora se encuentran registrados en archivos de computadora, es lo que se conoce como separación de remesas. Una vez que el operador de la lecto-clasificadora da por concluida la lectura del entry mediante un comando, el sistema de compensación, automáticamente toma el archivo correspondiente a ese entry y empieza a separarlo en remesas. Sobre las remesas se realizaran operaciones de corrección de cheques y conciliación de cifras, como veremos a continuación.

Una vez que tenemos el entry separado en remesas, el sistema de compensación automatizada comienza a separar los cheques de la remesa que tuvieron algún error al momento de su lectura. Entre los errores más comunes están, por ejemplo, el no reconocimiento de algún o alguno de los caracteres de la banda del cheque, originados por una mala calidad de la impresión.

Dado que en el proceso de lectura los campos con errores fueron identificados en el archivo como erróneos, el sistema separa ahora todos estos cheques y genera otro archivo por remesa, que contiene solamente los cheques con errores de lectura. Los cheques rechazados en una remesa en particular, se llevan a otra área diferente a la de lectura, que se llama área de captura y conciliación de cifras. Aquí, un operador capturista recibe los cheques rechazados de la remesa. Con el número que identifica al entry al que pertenece la remesa y con el número de remesa, el capturista solicita al

sistema que le muestre, en una terminal de vídeo, los registros correspondientes a los cheques rechazados que le fueron entregados para su corrección (véase la figura 1.2.4.1).

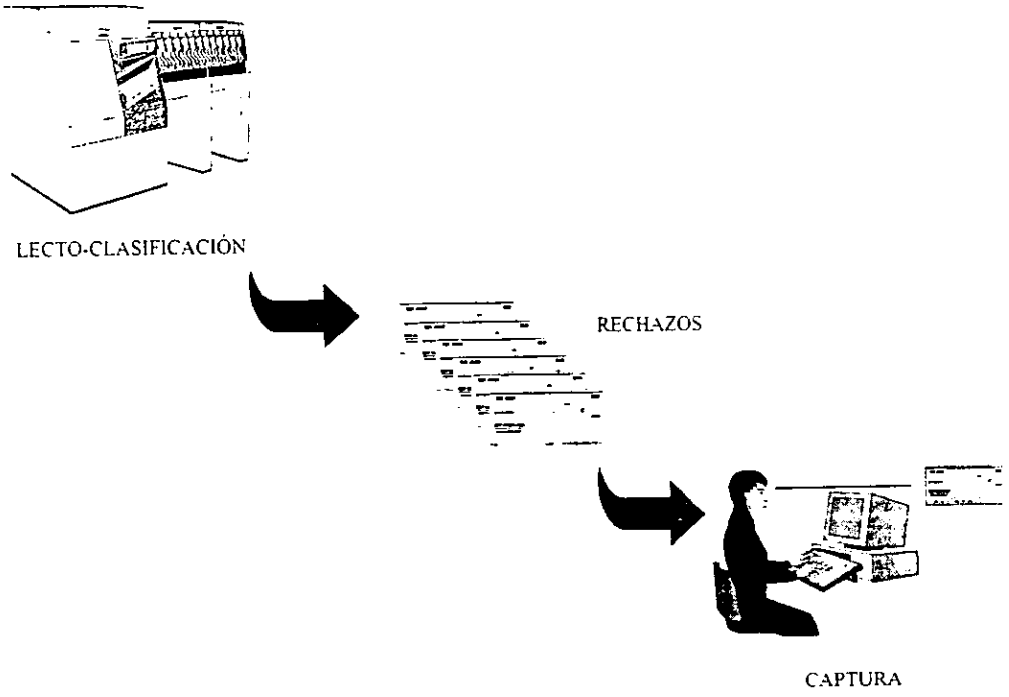


Figura 1.2.4.1 Captura de cheques rechazados.

Supongamos que en el primer cheque erróneo de la remesa, la lecto-clasificadora no reconoció uno de los dígitos del número de cuenta del cheque. Dado que el banco utilizará este dato para hacer un cargo automático a su cliente en el sistema propio del banco, un cambio en el número de cuenta ocasionaría que el cliente cargado fuera otro diferente al que pagó el cheque. Cuando el operador capturista solicite al sistema la remesa para su corrección, el sistema le mostrara el primer rechazo y el cursor se posicionará en el campo erróneo, en este caso el número de cuenta, el operador observa el cheque y ve que el número que la lecto-clasificadora no pudo leer fue un cinco, el cual, a simple vista no parece tener problemas. El operador captura la

información correcta, la verifica y confirma la modificación. El sistema, se coloca automáticamente en el siguiente cheque erróneo. En el caso de varios campos con error dentro del mismo cheque, sólo se confirma la modificación del registro al final del último campo corregido. Si se analizara el dígito erróneo más a detalle en un laboratorio de cheques, con el equipo adecuado, se podría notar que existe un bajo nivel de la señal generada o alguna otra anomalía originada por una impresión deficiente (véase la sección 1.4 Estándares de los cheques con especificaciones E13B).

Continuando con nuestro proceso de corrección de rechazos, el capturista iría corrigiendo uno a uno los cheques hasta llegar al último cheque de esa remesa. En ese momento, el sistema, al no encontrar más registros en el archivo de errores, cerraría la sesión de corrección y generaría un archivo con los registros ya corregidos de esa remesa. En realidad, genera dos registros por cada cheque, uno con el registro antes de su corrección y otro con el registro ya corregido.

Después de que la remesa ha sido corregida, si es que tuvo rechazos por supuesto, el sistema toma dos archivos: uno que corresponde a la remesa original y otro que corresponde a los cheques ya corregidos. Con los archivos mencionados, el sistema realiza un merge (mezcla) de archivos. El sistema localiza secuencialmente los cheques erróneos, compara que el registro original corresponda con el que se encuentra en el archivo de correcciones y luego sustituye ese registro, por el que ya esta corregido.

Después de que se tiene un archivo de la remesa sin errores, se lleva a cabo lo que se conoce como balanceo de cifras. Aquí el sistema verifica que la suma obtenida a partir de los importes de los cheques, corresponda con la que fue leída de los documentos de control preparados por el banco para la presentación de los cheques (véase la figura 1.2.4.2). Esta validación se hace hasta este momento, debido a que algunos cheques pueden haber tenido errores en el campo de importe en el momento de su lectura, y no se hubiera podido obtener un cálculo confiable del importe global de la remesa en estas

circunstancias. Después de la correcciones se puede llevar a cabo esta validación, para verificar que las cifras reportadas por el banco, son correctas.

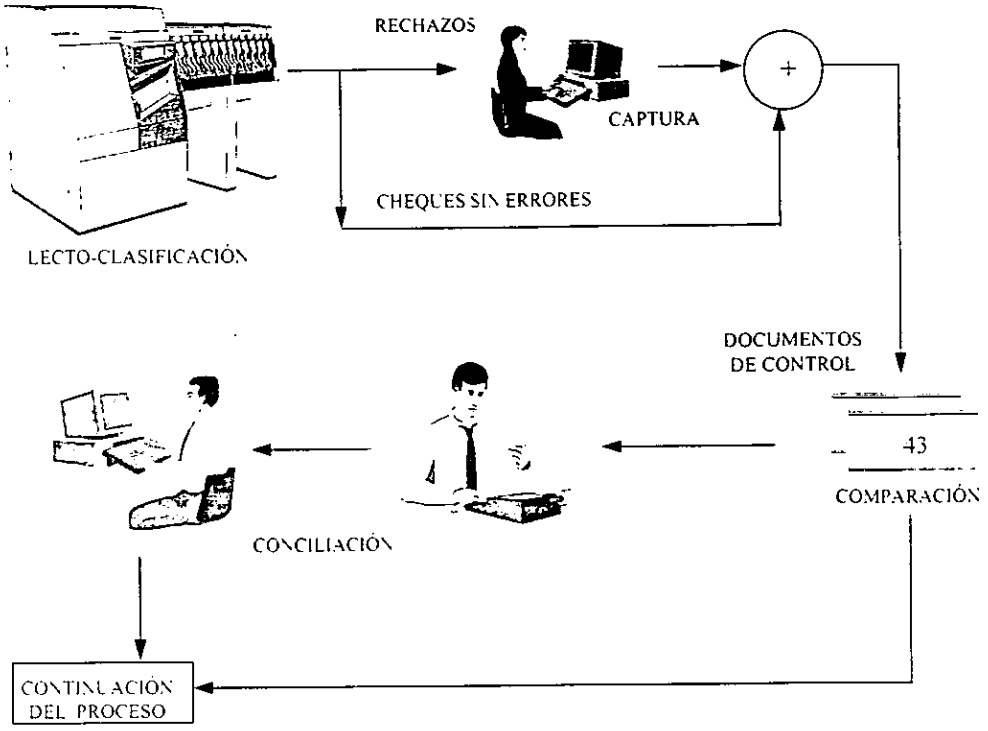


Figura 1.2.4.2 Conciliación de cifras.

Tomemos en cuenta también, que en el caso de remesas sin errores detectados en la lectura, se puede dar el caso de que existan diferencias en las cifras reportadas y las cifras calculadas por el sistema, esto se puede dar tanto por un cheque postmarcado incorrectamente como por cifras mal reportadas en el volante de remesa. Es en esta fase de balanceo de cifras donde se detectan este tipo de errores.

En caso de algún error en el balanceo de cifras, el sistema permite, igual que en el caso anterior, la modificación de las cifras de control o la introducción de ajustes en las cifras. Supongamos que el banco generó mal el volante de remesa, esto se puede verificar visualmente a partir de la carta remesa, donde se puede observar que el banco

capturó algo diferente de lo que dice que vale la remesa. En este caso, se corrige por terminal el volante de remesa. El sistema hace la validación otra vez y dice que ahora todo esta correcto y con esto terminaría el balanceo de esa remesa.

Supongamos ahora que el volante de remesa esta bien, pero en esta remesa había un cheque mal postmarcado, valía 1,000 pesos y fue postmarcado por 100 pesos, vamos por las tiras de postmarcadora y leemos la anotación que se hizo al respecto, entonces, como nos faltan 900 pesos, hay que hacer un ajuste "de más". Para llevar a cabo el ajuste. se inserta dentro de la remesa un registro que en realidad no corresponde a un cheque pero que nos sirve para compensar la cantidad faltante, como en este caso, o sobrante, en la situación de que el cheque de nuestro ejemplo se hubiera postmarcado por 10.000 pesos, en cuyo caso tendríamos que hacer un ajuste "de menos".

Ahora supongamos que tanto el volante de remesa como los cheques estaban bien, pero que la lecto-clasificadora interpretó erróneamente uno de los dígitos del importe de algún cheque. Resulta que era un cheque postmarcado por 300 pesos, pero por deficiencias en la impresión fue interpretado como si fuera por 500 pesos. Aquí el dígito 3 fue leído como 5, esto se llama misread (mala lectura), en este caso no se produjo un error en el proceso de lectura, puesto que la lecto-clasificadora no tuvo dudas al leer el dígito 5 que en realidad era un 3. Esta diferencia es detectada en esta fase de balanceo de cifras. En este caso, también se realiza un ajuste como en el caso anterior, pero en este ejemplo, se trataría de un ajuste de menos. Nótese que en esta fase de balanceo los registros correspondientes a cheques no se modifican, puesto que el rastro de si fueron corregidos o no, se lleva en la fase de captura de rechazos y no en la de balanceo. Aquí sólo podemos modificar los volantes de control y recurrir al uso de ajustes. De las operaciones realizadas a la remesa durante esta fase de balanceo de cifras. se generan algunos archivos de control con el rastro de las modificaciones efectuadas sobre la remesa. Hay remesas que no tienen ningún problema y balancean automáticamente, sin la intervención del operador humano.

Una vez que se han balanceado todas las remesas de un entry en particular el sistema automáticamente empieza a realizar un proceso de auditoria al entry. Recordemos que el entry fue inicialmente separado en remesas después del proceso de lectura. El sistema verifica ahora que todas las remesas pertenecientes al entry existan todavía dentro del sistema, para lo cual vuelve a armar el entry a partir de las remesas ya corregidas y balanceadas (en su caso). A continuación, tomando como base el archivo original del entry y usando los archivos de control generados en las fases de captura de rechazos y en la de balanceo de cifras, el sistema realiza todas las operaciones que fueron realizadas en las remesas separadas, sólo que esta vez lo hace sobre el archivo original del entry. El sistema, por medio del proceso anterior, debe llegar al mismo archivo que resultó de unir las remesas ya balanceadas, con lo cual terminaría esta fase de auditoria.

Una vez que el sistema se asegura de que todo lo que se hizo fue legal y que no se perdió o alteró información en el camino, procede a hacer la separación, por banco girado, de los registros de los archivos correspondientes a cada entry. Para lo anterior, los archivos correspondientes a cada entry son recorridos secuencialmente y cada registro de cada entry es "enviado" a otro archivo, que contiene registros correspondientes únicamente para ese banco girado en específico. Con esto, se concluye la separación de los registros por banco girado o receptor.

1.2.5 Salidas del proceso.

Retomando la secuencia de la sección anterior (1.2.4 Proceso de la información), una vez que los registros se encuentran en archivos separados por banco girado, se procede a la generación de resultados de la compensación. Como la salida más importante del sistema, se obtiene un medio magnético, que contiene un archivo con los cheques a cargo de cada banco. Este archivo recibe el nombre de archivo de cheques a su cargo, ya que el banco que los recibe es el que tiene que pagar por el importe de los mismos. Generalmente se usa una cinta magnética para los bancos de

mayor volumen de cheques recibidos, y un disco flexible para el caso de los que reciben un volumen bajo. También se puede transmitir este archivo hacia el banco receptor a través del uso de telecomunicaciones.

Con el archivo de cheques a su cargo mencionado arriba, el banco realiza el proceso posterior del mismo, que consiste en la aplicación automática de la información contenida en el archivo. Al realizar este proceso, el banco carga a sus clientes el importe de los cheques que giraron.

Otra de las salidas del proceso que recibe cada banco participante en la compensación automatizada de cheques, consiste en un reporte que da un detalle tanto de las operaciones presentadas a las demás instituciones como de las operaciones recibidas de cada una de ellas. Este reporte es conocido comúnmente como HCOM (Hoja de COMpensación) y le proporciona al banco, un panorama general de qué tantos cheques compensó con cada una de las otras instituciones y la diferencia entre los importes correspondientes a esta compensación de cheques. (véase la figura 1.2.5.1). Si no existiera el banco liquidador, el banco tendría que pagar o cobrar a cada banco la cantidad mostrada en la columna del reporte titulada "DIFERENCIA", pero cuando existe un banco liquidador, como es el caso de nuestro país, el banco solo tiene que realizar un movimiento global por concepto de la compensación de cheques. Esta cantidad a operar es el total que aparece en la misma columna de "DIFERENCIA" antes mencionada.

En el esquema de liquidación, tenemos que cada banco participante en la compensación de cheques, tiene una cuenta a su nombre, en el banco liquidador. A través de esta cuenta le es cargado o abonado, según sea el caso, el importe total correspondiente al resultado de la compensación de cheques. Digamos que un banco tuvo un resultado deudor como resultado de la compensación, entonces recibirá un cargo en su cuenta a causa de esto. Como en todas las cuentas, para que se efectúe un cargo, primero debe haber fondos suficientes en la cuenta, o de lo contrario se

entraría en una situación de carencia de fondos. Ahora tomemos el caso de un banco que resulta con un saldo acreedor en el proceso de compensación de cheques. En este caso este banco tiene un panorama más favorable ya que simplemente esta esperando a que le depositen el dinero en su cuenta para hacer uso de los fondos.

COMPENSACION ELECTRÓNICA PLAZA : MEXICO, D.F. FECHA : 1999/04/06
HOJA DE COMPENSACION PARA : 002 BANCO NACIONAL DE MEXICO, S.A. HORA : 14:33

	PRESENTADO		RECIBIDO		DIFERENCIA
INSTITUCION	DOCTOS	IMPORTE	DOCTOS	IMPORTE	IMPORTE
007 CITIBANK	1,897	37,532,591.27	1,989	24,126,068.15	13,406,523.12
011 CONFIA	2,063	35,215,965.14	0	0.00	35,215,965.14
013 INDUSTRI	11	143,945.00	0	0.00	143,945.00
014 SANT-MEX	3,640	73,983,774.10	2,664	48,178,600.62	25,805,173.48
017 BBV	4,316	68,695,123.86	0	0.00	68,695,123.86
019 BANEFA	259	2,220,847.18	103	4,190,870.92	-1,970,023.74
021 BITAL	9,695	139,058,586.29	0	0.00	139,058,586.29
025 SURESTE	101	2,930,443.89	0	0.00	2,930,443.89
030 BAJIO	53	3,186,564.46	73	435,509.94	2,751,054.52
032 IXE	261	2,768,554.27	0	0.00	2,768,554.27
036 INBURSA	306	16,657,577.54	0	0.00	16,657,577.54
037 INTERAC	4	11,745.10	0	0.00	11,745.10
042 MIFEL	108	2,069,783.45	0	0.00	2,069,783.45
044 INVERLAT	3,787	112,018,356.24	0	0.00	112,018,356.24
059 INVEX	3	65,690.25	0	0.00	65,690.25
062 AFIRME	154	3,157,155.18	0	0.00	3,157,155.18
068 PROMEX	1,063	12,785,692.17	735	6,855,526.38	5,930,165.79
071 BANPAIS	733	13,189,344.63	0	0.00	13,189,344.63
072 BANORTE	1,379	56,825,628.14	0	0.00	56,825,628.14
086 BANCEN	71	1,310,412.24	0	0.00	1,310,412.24
106 AMERICA	2	3,530.36	0	0.00	3,530.36
107 BOSTON	9	175,363.60	0	0.00	175,363.60
108 TOKIO	9	175,187.59	0	0.00	175,187.59
113 DRESDNER	5	424,006.00	0	0.00	424,006.00
119 REPUBLIC	14	501,271.46	0	0.00	501,271.46
149 BANRURAL	40	250,560.09	0	0.00	250,560.09
161 BANCRECE	2,421	34,182,550.17	0	0.00	34,182,550.17
T O T A L :	32,404	619,540,249.67	5,564	83,786,576.01	535,753,673.66

Figura 1.2.5.1 Reporte "hoja de compensación" (HCOM).

Bien, después de la explicación de lo que se hace para saldar las cuentas resultado de la compensación de cheques, diremos que esta aplicación de cargos y abonos se realiza también de manera automatizada por medio de un archivo con un formato acordado entre la cámara de compensación y el banco liquidador. Este archivo también es producido por el sistema de compensación automatizada de cheques y se conoce como archivo de saldos o archivo de liquidación.

Otras salidas producidas por el sistema de compensación automatizada de cheques, son los archivos para obtener microfichas. Una microficha es un medio de almacenamiento de información en el que se graba en una película de celuloide, por

medio de un rayo láser, el equivalente a 207 páginas impresas de 11 x 14 pulgadas. Las dimensiones de la microficha son de 10 x 15 centímetros. Para poder revisar visualmente la información contenida en la microficha, se utiliza un dispositivo que cuenta con unos lentes de aumento que proyectan sobre una pantalla los datos almacenados en la microficha. De esta manera, la información puede ser visualizada fácilmente. Las microfichas contienen, en forma de reportes formateados, los datos de cada uno de los cheques presentados a compensación por un banco en particular. El conjunto mencionado de microfichas, recibe el nombre de microfichas de cheques cedidos. También se le entrega a cada banco las microfichas con los datos de los cheques recibidos a su cargo, estas microfichas reciben en conjunto, el nombre de microfichas de cheques a su cargo.

Un producto interno, es decir, para uso de la cámara de compensación, es la obtención de medios magnéticos de respaldo de la información procesada durante un día en particular. Este respaldo, se realiza con el fin de poder aclarar cualquier dificultad o reclamación posterior.

Existen otros reportes que se obtienen como salida del sistema de compensación de cheques y que son para uso interno de la cámara de compensación. Por ejemplo, existe un reporte que se conoce como resumen de compensación (véase la figura 1.2.5.2 Resumen de compensación). Este reporte contiene un renglón por cada institución participante en la compensación, donde se pueden observar los totales de los cheques presentados y recibidos por cada banco, así como la diferencia entre estas dos cantidades, que es en sí, el saldo resultado de la compensación de cheques para cada banco. Aquí puede observarse que el total de la columna de lo presentado debe ser igual al total de la columna de lo recibido, esto es porque todos los cheques que se presentaron a compensación, a su vez, deben ser recibidos por algún banco girado, y la diferencia entre todo lo presentado menos todo lo recibido debe ser igual a cero, lo que indica que todos los cheques fueron compensados.

RESUMEN DE COMPENSACION ELECTRONICA PLAZA : MEXICO, D.F. FECHA : 1999/04/12 HORA : 21:09

CLAVE		ACEPTADO DOCTOS	ACEPTADO IMPORTE	RECIBIDO DOCTOS	RECIBIDO IMPORTE	DIFERENCIA
002	BANAMEX	46,355	644,279,001.34	32,007	465,200,981.52	179,078,019.82
003	SERFIN	7,838	145,642,459.75	15,180	192,079,017.65	-46,436,557.90
007	CITIBAN	6,369	90,432,462.60	5,174	103,821,512.45	-13,389,049.85
011	CONFIA	9,363	127,192,283.76	9,293	112,930,155.41	14,262,128.35
012	BANCOME	42,637	589,470,761.20	30,143	353,656,043.41	235,814,717.79
013	INDUSTR	52	793,088.67	32	126,443.13	666,645.54
014	SANT-ME	14,901	207,965,630.41	16,290	214,456,091.14	-6,490,460.73
017	BBV	17,495	215,185,512.01	19,642	275,330,639.88	-60,145,127.87
019	BANEFA	399	12,902,408.71	1,043	8,957,867.20	3,944,541.51
021	BITAL	48,807	604,538,769.45	38,532	415,766,886.63	188,771,882.82
025	SURESTE	278	2,286,269.12	829	20,062,979.06	-17,776,709.94
030	BAJIO	522	5,838,815.85	160	3,268,794.52	2,570,021.33
032	IXE	920	25,564,115.39	1,299	10,700,849.98	14,863,265.41
036	INBURSA	155	6,298,618.37	1,148	37,394,870.50	-31,096,252.13
037	INTERAC	0	0.00	75	935,969.62	-935,969.62
042	MIFEL	0	0.00	733	19,753,961.59	-19,753,961.59
044	INVERLA	14,680	181,310,611.53	17,062	273,664,388.58	-92,353,777.05
059	INVEX	0	0.00	38	368,863.01	-368,863.01
062	AFIRME	673	14,842,953.02	660	10,342,371.66	4,500,581.36
068	PROMEX	4,941	56,646,191.26	5,049	58,104,291.51	-1,458,100.25
072	BANORTE	0	0.00	10,252	207,078,896.06	-207,078,896.06
102	ABN	0	0.00	5	972,149.24	-972,149.24
106	AMERICA	0	0.00	11	242,417.45	-242,417.45
107	BOSTON	0	0.00	16	416,444.97	-416,444.97
108	TOKIO	0	0.00	56	656,770.63	-656,770.63
113	DRESDNE	0	0.00	13	189,628.97	-189,628.97
116	REPUBLI	0	0.00	97	6,987,181.16	-6,987,181.16
149	BANKBOA	0	0.00	176	1,400,351.77	-1,400,351.77
161	BANCREC	0	0.00	11,470	136,323,133.74	-136,323,133.74
T O T A L		216,385	2,931,189,952.44	216,385	2,931,189,952.44	0.00

TOTAL NEGATIVO : -644,471,803.93 TOTAL POSITIVO : 644,471,803.93

Figura 1.2.5.2 Reporte "resumen de compensación".

Se obtiene otro reporte que sirve para verificar que el archivo de cheques a su cargo, que recibe el banco, esta correcto. Para producir este reporte, el sistema de compensación recorre los archivos a su cargo de todos los bancos y va reportando en cada renglón los totales de documentos y el importe correspondiente para cada banco (véase la figura 1.2.5.3). De esta manera se verifica que el banco recibirá en su archivo, la misma cantidad de operaciones e importe reportada por el sistema en la hoja de compensación (HCOM), para ese banco en particular. Este impreso se conoce como reporte de control.

REPORTE DE CIFRAS DE CONTROL DE LA COMPENSACION ELECTRONICA PLAZA : MEXICO, D.F.
FECHA : 1999/04/12

CLAVE	INSTITUCION	DOCTOS	IMPORTE
002	BANCO NACIONAL DE MEXICO, S.A.	32,007	465,200,981.52
003	BANCA SERFIN, S.A.	15,180	192,079,017.65
007	CITIBANK MEXICO, S.A.	5,174	103,821,512.45
011	BANCA CONFIA, S.A.	9,293	112,930,155.41

012	BANCOMER, S.A.	30,143	353,656,043.41
013	BANCO INDUSTRIAL, S.A.	32	126,443.13
014	BANCO SANTANDER MEXICANO, S.A.	16,290	214,456,091.14
017	BANCO BILBAO VIZCAYA MEXICO, S.A.	19,642	275,330,639.88
019	BANCO DEL EJER, FZA. AEREA Y ARM, S.N.C.	1,043	8,957,867.20
021	BANCO INTERNACIONAL, S.A.	38,532	415,766,886.63
025	BANCO DEL SURESTE, S.A.	829	20,062,979.06
030	BANCO DEL BAJIO, S.A.	160	3,268,794.52
032	IXE BANCO, S.A.	1,199	10,700,849.98
036	BANCO INBURSA, S.A.	1,148	37,394,870.50
037	BANCO INTERACCIONES, S.A.	75	935,969.62
042	BANCA MIFEL, S.A.	733	19,753,961.59
044	BANCO INVERLAT, S.A.	17,062	273,664,388.58
059	BANCO INVEX, S.A.	38	368,863.01
062	BANCA AFIRME, S.A.	660	10,342,371.66
068	BANCA PROMEX, S.A.	5,049	58,104,291.51
071	BANPAIS, S.A.	4,013	47,600,642.03
072	BANCO MERCANTIL DEL NORTE, S.A.	6,068	157,651,940.87
086	BANCO DEL CENTRO, S.A.	171	1,826,313.16
102	ABN AMRO BANK (MEXICO), S.A.	5	972,149.24
106	BANK OF AMERICA MEXICO, S.A.	11	242,417.45
107	BANCO DE BOSTON, S.A.	16	416,444.97
108	BANK OF TOKIO MEXICO, S.A	56	656,770.63
113	DRESDNER BANK MEXICO, S.A.	13	189,628.97
119	REPUBLIC NATIONAL BANK OF N.Y. MEX, S.A.	97	6,987,181.16
143	BANCO DE CREDITO RURAL DEL NORTE, S.N.C.	1	14,192.20
144	BANCO DE CREDITO RURAL DEL CENTRO, S.N.C.	13	163,998.10
146	BANCO DE CREDITO RURAL DEL GOLFO, S.N.C.	6	27,570.99
147	BANCO DE CREDITO RURAL DEL CENTRO SUR S.N.C	40	352,817.78
148	BANCO DE CREDITO RURAL DEL NOROESTE, S.N.C.	1	2,265.67
149	BANCO NACIONAL DE CREDITO RURAL, S.N.C.	99	789,781.94
158	BANCO DE CREDITO RURAL DEL ITSMO, S.N.C.	4	11,996.23
160	BANCO DE CREDITO RURAL DEL PENINSULAR, S.N.C	1	3,286.51
161	BANCRECER, S.A.	11,470	136,323,133.74
162	BANCO DE CRED. RURAL DEL CENTRO NORTE S.N.C	1	8,587.00
165	BANCO DE CRED. RURAL DEL PACIFICO SUR S.N.C	10	25,855.35
	T O T A L :	216,385	2,931,189,952.44

Figura 1.2.5.3 Reporte de control de cifras.

Se obtiene además, como otra salida de uso interno, un archivo para fines estadísticos. Estos archivos son alimentados como entrada a otro sistema que realiza cálculos que sirven para observar el comportamiento y los volúmenes de información procesados por día. Estos datos se van acumulando y al final del mes se obtienen cifras de todas las operaciones realizadas por los bancos durante dicho período. De la misma manera, se obtienen datos que nos dan cifras anuales de lo procesado por cada banco durante el año. Sin embargo, ese es otro sistema y no del que nos estamos ocupando en este trabajo.

1.2.6 Esquema general de un caso práctico.

Esta sección nos servirá para resumir las fases que componen el sistema automatizado de compensación de cheques.

Podemos decir que la compensación automatizada de cheques se compone de cinco etapas o fases:

a) Lectura de cheques.
b) Corrección de cheques rechazados.
c) Balanceo o conciliación de cifras.
d) Auditoría.
e) Separación de registros por banco girado, o dispersión.

Cada una de estas fases ha sido ya explicada anteriormente. En esta sección simplemente se hace un resumen del las fases comprendidas en el proceso de compensación automatizada, y se muestra en forma gráfica, como se ve en la figura 1.2.6.1.

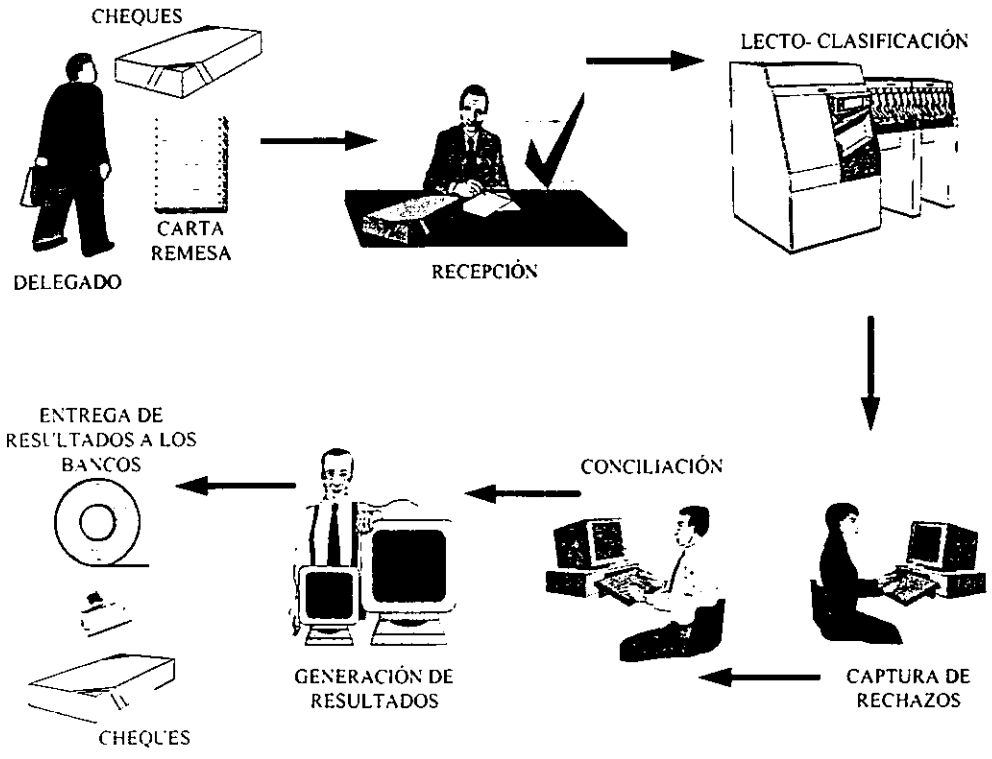


Figura 1.2.6.1 Flujo de una remesa en el sistema de compensación automatizada.

1.2.7 Compensación electrónica.

Se puede decir, que la compensación electrónica de cheques, es un subconjunto de la compensación automatizada de cheques. También puede decirse, que la compensación electrónica es una mezcla entre lo que es la compensación automatizada de cheques y el truncamiento, ya que en la compensación electrónica se incluyen aspectos de ambas. Por ejemplo, en la compensación electrónica de cheques, los documentos no son leídos por la cámara de compensación, sino que son leídos por el banco o por una empresa que lo hace por él. Sin embargo, existe una lectura de cheques dentro del proceso global, en este aspecto la compensación electrónica tiene una similitud con la compensación automatizada. Por otra parte, tenemos una similitud

en cuanto al truncamiento, en que la información presentada como entrada al sistema no son los cheques físicos sino los datos correspondientes a los cheques, en un formato procesable por computadora, o en forma electrónica, de donde toma su nombre este tipo de compensación (electrónica). Sin embargo, los cheques sí se intercambian entre los bancos y no permanecen con la institución cedente como ocurre en el truncamiento. Vemos pues, de esta manera, que la compensación electrónica es un compuesto de los otros dos tipos (automatizada y truncamiento).

Entrando un poco más en detalle, diremos que en la compensación electrónica, los cheques son leídos en lecto-clasificadoras que se encuentran en las instalaciones del propio banco cedente o en alguna empresa maquiladora de la lectura de cheques. Como resultado de la lectura de los documentos, se obtiene un archivo que contiene un registro por cada cheque leído, además de otros registros de control. Este archivo, perteneciente a un banco dado, junto con los archivos preparados por cada uno de los otros bancos participantes en la compensación, es la entrada al sistema de compensación electrónica de cheques. Este sistema, usando cada uno de estos archivos, realiza una validación y dispersión de los mismos, para obtener un archivo de cheques a cargo de cada banco girado. Los bancos participantes se presentan en la cámara de compensación, con los documentos a cargo de los otros bancos y realizan el intercambio de los mismos. Este intercambio se lleva a cabo dentro de las instalaciones de la misma cámara. Más tarde, el banco realiza el proceso de afectación de las cuentas de sus clientes y verifica que las cifras obtenidas por su propio sistema sean iguales a las reportadas por el sistema de compensación electrónica. Este proceso de validación de cifras, por parte del banco, también se realiza en la compensación automatizada.

En el sistema de compensación electrónica, no existe la fase de captura de rechazos, ya que sólo deben presentarse cheques sin errores de lectura, como una regla de este proceso. Sin embargo, a pesar de esta regla, ocasionalmente los bancos presentan su información con este tipo de errores, además de otras anomalías en la información, que

no son propiamente errores de lectura. Es por lo anterior, que se hace necesaria una fase de validación de cada uno de los registros presentados. En esta fase de validación, se verifica que el banco haya preparado correctamente la información presentada a la cámara de compensación. Para este fin, se valida cada campo de cada registro. Cada campo tiene definidos el tipo de datos que debe contener y un rango de valores permitidos. Si existiera alguna inconsistencia en cuanto al tipo o al contenido de uno o más campos del registro, el registro sería rechazado, y el cheque correspondiente a ese registro no sería compensado automáticamente por el sistema. El cheque correspondiente a un registro rechazado, tiene que ser compensado "por fuera" como se menciona en la sección 1.1.1 (Proceso de compensación).

En el sistema de compensación electrónica, tampoco existe la fase de balanceo de remesas propiamente dicha, aunque existe una validación parecida como veremos un poco más adelante.

Podemos decir que en la compensación electrónica, cada archivo presentado, corresponde a una remesa de la compensación automatizada. Un banco cedente puede presentar uno o más archivos. Estos archivos, debido a que dentro del concepto de compensación electrónica, se tiene contemplado que el archivo llegue al sistema vía telecomunicaciones, recibe el nombre común de transferencia. Sin embargo, un archivo presentado al sistema en medio magnético (cinta o disquete), también es llamado "transferencia".

Cada archivo o transferencia consta de tres tipos de registros.

a) Registro encabezado.
b) Registro de detalle.
c) Registro sumario.

a) Cada transferencia consta de un sólo registro encabezado, en este registro se encuentran codificados datos como el banco cedente, el número de transferencia (que es único y sirve para identificar a la transferencia) y la fecha del proceso, entre otros. En la fase de validación se verifica que cada uno de estos campos sea correcto y si alguno de ellos no lo es, entonces toda la transferencia es rechazada por el sistema como medida de seguridad. En este caso, el banco deberá corregir los datos incorrectos y volver a presentar el archivo para su validación, hasta que sea aceptado por el sistema.

b) Pueden existir dentro del archivo uno o más registros de detalle. Cada registro de detalle contiene información correspondiente a un cheque. Los registros de detalle tienen un formato diferente a los registros encabezado y sumario, y contienen entre otros campos: número consecutivo de secuencia del registro, banco emisor que es lo mismo que banco cedente, banco receptor que es lo mismo que banco girado, número de la cuenta del cheque, número de cheque, fecha del proceso, etc. Existen campos dentro del registro que para su validación, dependen del valor de otros campos. Este es el caso de los llamados dígitos verificadores. Dentro de la validación de cada registro de detalle, se incluye la validación de los dígitos verificadores mencionados. También se verifica que el importe del cheque sea mayor a cero, entre otras validaciones. En caso de algún error en el registro de detalle, este es marcado lógicamente como un registro "rechazado". Si pasa exitosamente todas las validaciones, el registro es identificado como "aceptado".

c) El registro sumario tiene también un formato diferente a los formatos de los registros de encabezado y detalle. Aquí se incluyen datos de cifras de control como son: el total de registros de detalle (cheques) presentados en el archivo y el importe total correspondiente a los mismos. El sistema de compensación electrónica, realiza su propio conteo de los cheques e importes que vienen dentro del archivo, y los compara con los reportados por el banco en el registro sumario. Si existe alguna diferencia entre las cifras calculadas y reportadas, todo el archivo es rechazado y el banco debe

corregirlo y enviarlo de nuevo, hasta que sea aceptado por el sistema. En este sentido, esta parte de la validación correspondería en cierto modo, al balanceo de remesas que se da en la compensación automatizada. En el caso de la compensación automatizada de cheques la remesa no continua su proceso hasta que no esta balanceada, en este caso, si la remesa (transferencia) no esta balanceada, ni siquiera puede entrar al sistema.

La fase de auditoria que existe en la compensación automatizada, no existe en la compensación electrónica. Lo anterior es debido a que la auditoría tiene su origen en el hecho de que, en la compensación automatizada, se realizan tanto la captura de rechazos como el balanceo de remesas, fases que no existen en la compensación electrónica.

Finalmente, al igual que en el caso de la compensación automatizada, tenemos una fase en la que los registros son enviados a su "casillero" correspondiente. Esta fase es conocida, dentro de la compensación electrónica de cheques, como dispersión de operaciones.

Tenemos entonces dentro de la compensación electrónica las siguientes fases:

a) Lectura de cheques en el banco.
b) Generación, por parte del banco, del archivo electrónico de cheques.
a) Validación de los archivos electrónicos de cheques.
b) Dispersión de las operaciones por banco girado.

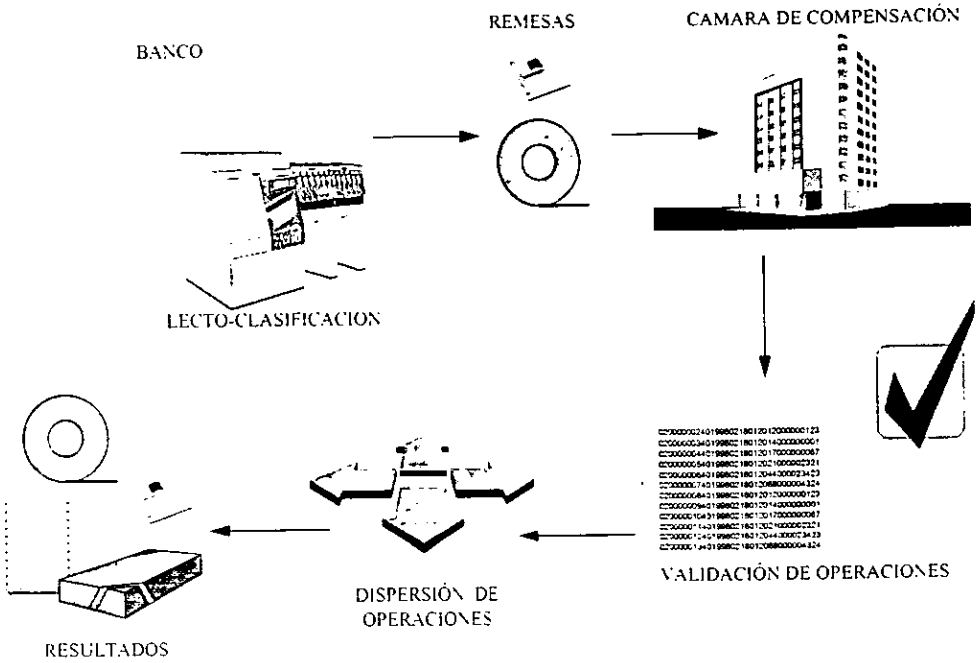


Figura 1.2.7.1 Fases de la compensación electrónica de cheques.

Las salidas obtenidas del proceso de compensación electrónica de cheques, son muy similares a las obtenidas en el proceso de compensación automatizada, y se pueden resumir en las siguientes:

a) Archivos de cheques a su cargo para los bancos girados.
b) Hoja de compensación para cada banco participante (HCOM).
c) Microfichas con el detalle de los cheques presentados y recibidos.
d) Reporte del resumen de la compensación.
e) Reportes de control de las cifras en los archivos a su cargo.
f) Respaldo de la información procesada durante un día en particular.
f) Archivo de saldos.
g) Archivo para estadísticas.

La descripción de cada una de estas salidas puede consultarse en la sección 1.2.5 (Salidas del proceso).

Cabe mencionar que en el caso de la compensación electrónica de cheques, debido a que los documentos no son leídos en la cámara de compensación, el horario máximo de recepción de los archivos presentados por los bancos cedentes, es aproximadamente una hora más tarde, que en el caso de la compensación automatizada de cheques. Lo anterior al menos, es lo que sucede en la empresa de compensación que opera con el sistema descrito en el presente trabajo, que por cierto, es la única empresa de compensación bancaria que existe actualmente en el país.

1.3 TECNOLOGÍA DE CARACTERES MICR.

Estudios exhaustivos efectuados en los Estados Unidos y en otros países condujeron al desarrollo de la técnica para codificar y leer caracteres impresos con tinta magnetizable. Estos estudios demostraron que los caracteres magnetizables son el medio más eficaz y confiable para leer documentos que han sido manipulados por el público.

1.3.1 Definición y técnicas MICR.

La técnica para leer en forma automatizada documentos codificados con caracteres magnetizables es conocida como **MICR** (Magnetic Ink Character Recognition), que traducida al español significa (Reconocimiento de caracteres impresos con tinta magnetizable).

Existen en el mercado varios equipos que procesan documentos en forma mecanizada los cuales evitan el costoso proceso de transcribir toda la información de los documentos, por medio de un teclado, a tarjetas perforadas o a algún medio de almacenamiento magnético. La opción más usual para poder leer documentos directamente en los equipos electrónicos son los caracteres magnetizables MICR . Esta técnica permite la captación de la información de los documentos a las velocidades que ofrecen los equipos de cómputo electrónico.

1.3.2 Fonts MICR.

Son el conjunto de caracteres utilizados en la impresión de cheques.

1.3.2.1 El conjunto de caracteres E13-B.

Los caracteres MICR usados por los cheques se conocen como fonts E13-B. Este font de caracteres consta de diez números (0 al 9) y de cuatro símbolos de control (importe, a nuestro cargo, tránsito y guión), como se muestran en la figura 1.3.2.1.1

El font E13-B se encuentra especificado en el **ANSI** (American National Standards Institute) (Instituto Nacional Estadounidense de Estándares).

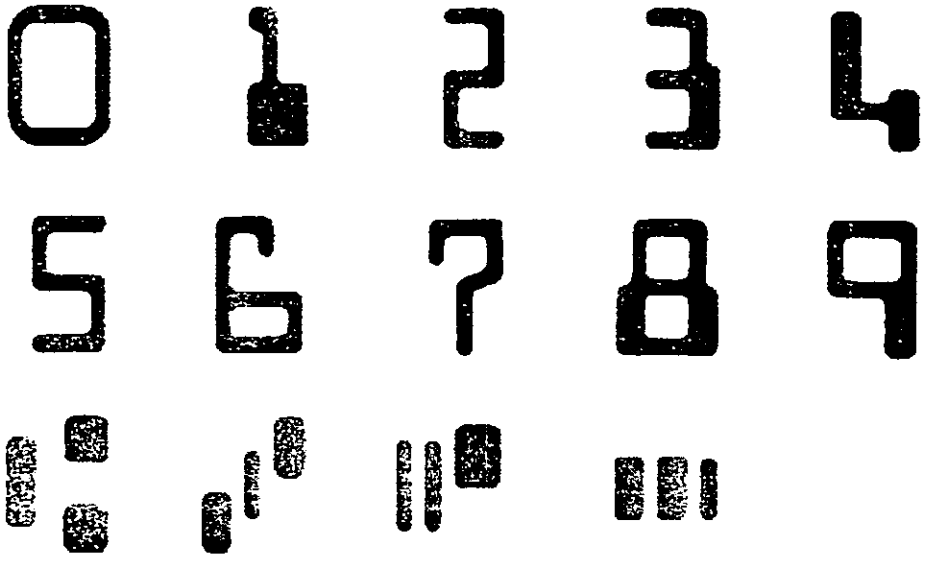


Figura 1.3.2.1.1 Caracteres E13-B

1.3.2.2 El conjunto de caracteres CMC-7.

Existen también los fonts de caracteres MICR CMC-7, la técnica que se utiliza para leer estos caracteres es idéntica a la usada para leer los caracteres E13-B. Este font de caracteres consta de diez números (0 al 9) y de cinco símbolos de control (dos puntos, punto y coma, menor que, igual y mayor que), como se muestran en la figura 1.3.2.2.1

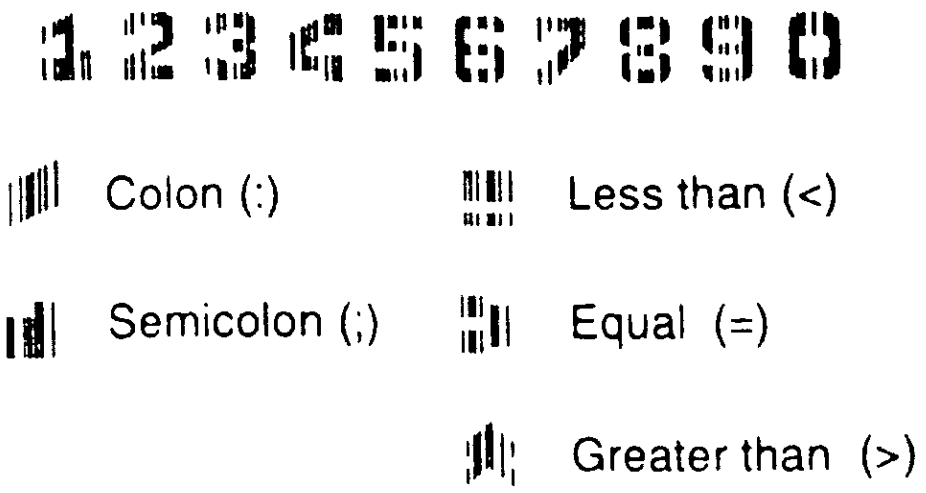


Figura 1.3.2.2.1 Caracteres CMC-7.

1.3.3 Lectura magnética y lectura óptica.

Existen básicamente tres clases de equipos para leer electrónicamente los caracteres magnetizables. De ellos, dos leen los caracteres magnéticamente y se les denomina lectoras MICR o lectoras magnéticas. El tercer tipo reconoce los caracteres ópticamente y se les denomina lectoras **OCR** (Optical Character Recognition) o lectoras ópticas (Reconocimiento Optico de Caracteres).

Los caracteres E13-B se imprimen con una tinta que contiene un alto contenido de óxido de hierro. Este material tiene la característica de que puede ser magnetizado.

Las lectoras magnéticas transportan los documentos a través de un magneto, que provoca que los caracteres E13-B se magneticen. Los caracteres magnetizados pasan frente a la cabeza lectora, y el campo magnético que producen genera un impulso

eléctrico que es recibido por la cabeza lectora. La intensidad y duración de este impulso permite descifrar el carácter que esta siendo leído.(ver figura 1.3.3.1)

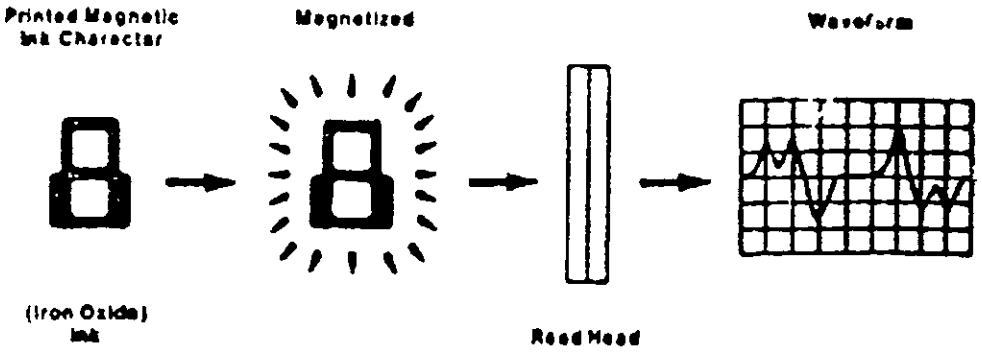


Figura 1.3.3.1 Lectura de caracteres magnéticos.

Se utilizan dos tipos de cabezas lectoras en el reconocimiento magnético: de pista sencilla y de pista múltiple o matricial.

La primera es una cabeza con una sola abertura que detecta el impulso eléctrico generado por el carácter. Cuando el carácter se desplaza frente a la abertura de la cabeza lectora, el impulso eléctrico que éste genera produce una onda con un perfil único para cada carácter.

La lectora matricial, en cambio, esta compuesta de pequeñas cabezas verticalmente. Conforme un carácter magnético se desplaza frente a ella, las pequeñas cabezas van cortando los planos del carácter, detectando la presencia del impulso eléctrico. Así se produce un patrón matricial único para cada carácter.

La principal diferencia, entonces, entre la cabeza de pista sencilla y la cabeza matricial, es que la primera reconoce el carácter midiendo la intensidad del impulso eléctrico y la

segunda, la presencia de ese impulso eléctrico. En cambio, en una lectora OCR las propiedades magnéticas de los caracteres E13-B no son utilizadas para la lectura. En vez de esto, la lectora OCR utiliza un sensor para detectar la cantidad de luz reflejada por el carácter y la cantidad de luz reflejada por el fondo del documento. Cuando existe un contraste suficiente, el sensor puede distinguir la forma del carácter. La técnica utilizada con más frecuencia en el reconocimiento óptico utiliza una columna de fotoceldas para detectar la presencia de áreas claras y obscuras. Este método reconoce simultáneamente una banda vertical del carácter, conforme este es transportado frente a la cabeza lectora. (ver figura 1.3.3.2)

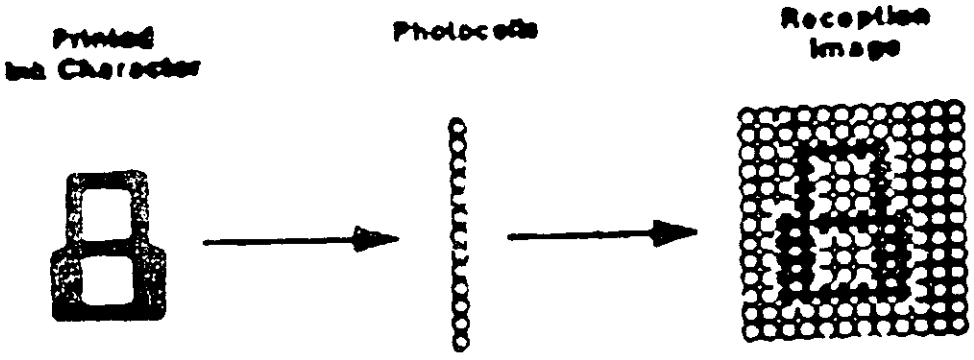


Figura 1.3.3.2 Lectura óptica de caracteres.

Las especificaciones MICR han sido diseñadas para garantizar que los caracteres E13-B puedan ser leídos por máquinas electrónicas. Al respetar estas especificaciones el usuario puede asegurarse que los caracteres magnetizables tendrán una buena probabilidad de ser leídos por un equipo lector, si es que este ha sido calibrado adecuadamente.

Una lectora magnética puede o no leer caracteres que están fuera de especificaciones. La capacidad de un carácter de ser leído está en función de su grado de imperfección y de la tecnología utilizada en la lectura.

1.3.4 Reconocimiento de caracteres impresos con tinta magnetizable.

La eficiencia y confiabilidad del proceso de lectura, depende fundamentalmente de la calidad en la impresión de los caracteres MICR, por lo que es conveniente describir en términos generales las fases en que se efectúa la impresión de dichos caracteres, así como también, en que consiste la técnica MICR y para comprender la importancia de observar los estándares para la impresión de cheques con caracteres magnetizables.

Impresión de cheques.

Los datos del cheque que deben ser registrados para su proceso automatizado, deben codificarse utilizando los caracteres MICR dentro de una banda situada en la parte inferior del documento, denominada "banda libre", la cual se describe a detalle en el capitulo 1.4. Dicha impresión se efectúa generalmente en dos fases perfectamente diferenciadas por el tiempo, lugar y equipo en que se realiza.

Premarcado

La primera fase, denominada "premarcado", consiste en la impresión de los datos del cheque que lo identifican en el momento en que éste es personalizado; tales como el tipo de documento, la clave del banco, el número de cuenta y el número de cheque. Esta operación se realiza antes de emitir los cheques a la circulación.

Posmarcado.

La segunda fase, denominada "posmarcado", consiste en la impresión del importe por el que fue expedido el cheque y se realiza una vez que el cheque ha entrado a cobro al sistema bancario.

Lectura electrónica de cheques

El proceso de lectura de un cheque, se efectúa desplazando el documento a través de un dispositivo lector, compuesto básicamente por 2 estaciones de paso y un elemento identificador de caracteres: en la primera estación está instalada una "cabeza magnetizadora", la cual magnetiza los caracteres que van pasando frente a ella; en la segunda estación está instalada una "cabeza lectora" que genera, por cada carácter que pasa frente a ella, un impulso eléctrico cuyo patrón e intensidad dependerá de la forma de dicho carácter; finalmente, el elemento identificador compara el impulso generado, contra los patrones "tipo" almacenados en su memoria. a fin de determinar a cual de ellos corresponde.

La forma del impulso eléctrico que se genera en la lectura de un carácter magnético es única. lo que permite el reconocimiento de cada uno de ellos. (ver figura 1.3.4.1)

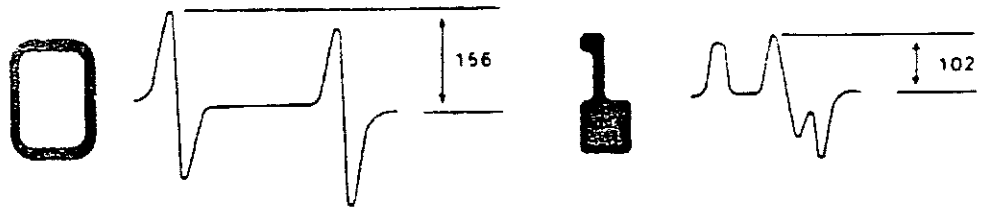


Figura 1.3.4.1 Formas de onda

Una vez efectuado el reconocimiento de los caracteres, la lógica del equipo efectúa el reconocimiento de cada campo, el cual se hace en base a símbolos delimitadores de cada campo. Cualquier deficiencia en la impresión de los caracteres, como puede ser: falta. manchas o exceso de tinta, alteraciones en la forma y/o dimensiones de los caracteres. etc.; provocará deformaciones en el impulso eléctrico y en consecuencia el carácter y/o el campo correspondiente no será reconocido. Esta lectura se inicia en el extremo derecho del cheque, por lo que es indispensable que el posicionamiento de los

diferentes grupos de caracteres con respecto a la orilla derecha del cheque, estén dentro de las medidas especificadas.

Puede verse la importancia de que los campos y los caracteres estén siempre dentro de la misma área de impresión y en las posiciones adecuadas en cada cheque, para que los equipos lectores puedan reconocer la información de que se trate.

1.3.5 Breve historia del E13-B.

Desde 1945, el consumo de cheques en los Estados Unidos se incrementó fuertemente, hasta llegar a convertirse en una verdadera "explosión". Esta explosión continuó en los años cincuenta, llegándose a expedir, hacia finales de década, un billón de cheques por año.

Como consecuencia, los bancos se vieron abrumados de trabajo. Desde 1910 habían confiado en los mismos procedimientos administrativos y en las mismas máquinas de contabilidad, cuando repentinamente el costo de la clasificación manual, debido al mayor volumen, alcanzó niveles desproporcionados. Para nadie fue una sorpresa que los banqueros empezaran a soñar con el día en que el cheque pasara por el banco sin ser tocado por la mano del hombre.

Con el perfeccionamiento de los procesadores de datos a principios de los años cincuenta, posteriormente conocidos como computadoras, este sueño no fue sino el primer paso a una realidad y la palabra mágica que se oía en todas las reuniones donde asistían los banqueros no era otra que "automatización".

Lograr realizar el sueño de la automatización bancaria requeriría de una colaboración sin precedente entre los cientos de bancos, entre las firmas que fabricaban equipos de oficina y entre los impresores de cheques, la mayoría de ellos impresores de cobertura regional.

En abril de 1954 la **ABA** (American Bankers Association) (Asociación de Banqueros de América) creó un subcomité técnico para el manejo mecanizado de cheques. Cinco años mas tarde, este subcomité presentaria sus resultados habiendo desarrollado la tecnología que permitiria la clasificación de los cheques a través de la lectura electrónica de una serie de números de apariencia extraña que eran impresos en la parte inferior del documento.

Con la implementación del programa de "Reconocimiento de Caracteres de tinta magnetizable" en 1959 llegó el día del gran cambio en la automatización bancaria.

Una vez formado el comité de la ABA en 1954, compañías como IBM y General Electric iniciaron la carrera para desarrollar la tecnología de lectura de cheques. Siempre se asumió un activo papel en el desarrollo de la automatización bancaria.

Fue muy fácil para los miembros del comité decidir que información necesitaba ser impresa en el cheque para el proceso automatizado, (número de cuenta, número de cheques e importe) pero, en cambio, fueron muchas las preguntas que no tuvieron respuesta inmediata. ¿Cómo se imprimiría la banda magnetizable? ¿Cómo sería leída? ¿Cuál sería el tamaño de los caracteres? ¿Y su forma? ¿Y su localización en el documento?

Cada fabricante de equipo tenía respuestas diferentes. A principios de 1954 muchos fabricantes, entre ellos IBM, Pitney_Bowes, Sperry-Rand y General Electric, recurrieron a Deluxe para evaluar sus cheques experimentales. Algunos de estos cheques tenían impreso un código binario, otros códigos de barras o códigos de bits impresos tanto en tinta común como en tinta magnetizable y tinta fluorescente. Las codificaciones aparecían en el frente, en el reverso en la parte superior y en la parte inferior de los cheques, lo que reflejaba las diferentes preferencias de los fabricantes de equipo de oficina.

Los miembros del comité, preocupados por la aceptación por parte del publico de codificaciones complejas, se inclinaron inicialmente hacia la codificación con tinta fluorescente impresa en el reverso de los cheques.

Con cada fabricante de equipo de oficina promoviendo un lenguaje diferente, la ABA pronto se percato de que ese no era el camino para desarrollar un sistema de automatización común. En enero de 1955, la ABA pidió a los fabricantes de equipo que desarrollaran un lenguaje común. En julio de 1956, de las propuestas sometidas, la ABA escogió E13-B como el sistema más apropiado para el manejo mecanizado de cheques. Para ese entonces cualquier impresor de cheques sabía que el apelativo de tinta magnética era erróneo. Lo cierto es que la tinta contenía un alto porcentaje de óxido de hierro y esto la hacía magnetizable. Pero la tinta en sí nunca fue ni seria magnética.

Debido al trabajo desarrollado por compañías como IBM con los primeros cheques experimentales. Deluxe ganó una experiencia invaluable en las incipientes etapas del E13-B. Aprendió sobre lo complejo del manejo de la tinta magnetizable; empezó a diseñar modificaciones en las que permitieran controlar confiablemente las variables del proceso de producción. Su asociación con los fabricantes de equipo de oficina introdujo a la compañía a las increíblemente estrictas tolerancias requeridas por la tecnología de la lectura automatizada.

En septiembre de 1956. la ABA invitó a los fabricantes de equipo y a los impresores de cheques a decidir si la codificación debería aparecer en la parte superior o inferior del cheque.

La ABA formó un subcomité integrado por fabricantes de equipo de oficina e impresores para resolver los problemas técnicos del MICR. Los fabricantes de equipo de oficina perfeccionaban la tecnología de lectoclasificación necesaria para el proceso automatizado de cheques. La persona acostumbrada a las operaciones manuales de clasificación quedaba gratamente sorprendida al ver la inventiva de los nuevos equipos.

En marzo de 1957, la ABA formó otro subcomité de ingenieros de los fabricantes de equipo de oficina y de los impresores de cheques, para desarrollar una tipografía que pudiese ser leída por estas nuevas máquinas. En diciembre de 1958, el comité de diseño se decidió por una tipografía de forma robótica y más bien cuadrada, llamada E13-B.

La "E" proviene de ser éste el quinto diseño propuesto, el 13 del ancho del carácter (0.013") y la "B" de la segunda revisión del diseño.

1.3.6 Especificaciones OCR.

Las especificaciones son el conjunto de estándares seleccionados para usarse en el diseño de los cheques. En el capítulo 1.4 se amplían y describen con más detalle las especificaciones para los cheques. Las especificaciones OCR son las siguientes:

a) Material de los cheques.
   Los cheques deben estar impresos en papel seguridad y de características especiales con el propósito de asegurar que sean capaces de soportar adecuadamente los rigores del manejo manual, del proceso de lectura y clasificación (Ver apéndice A. para mayor información acerca de las especificaciones, definiciones y características del papel para cheques.).

b) Legibilidad de la información.
   Toda la información contenida en el cheque debe aparecer de una manera legible y fácilmente localizable a simple vista.

c) Tamaño de los cheques.
   Todos los cheques deben tener dimensiones máximas y mínimas de acuerdo a los estándares aprobados para este fin, los cuales se describen a detalle en el capítulo 1.4

d) Banda libre.

Todos los cheques deben tener en su parte inferior una banda (espacio) destinada a codificar en ella la información necesaria para su proceso.

e) Banda de Impresión.

Es la zona dentro de la banda libre donde deben quedar impresos los caracteres correspondientes a la información del cheque.

1.4 ESTÁNDARES DE LOS CHEQUES CON ESPECIFICACIONES E13-B.

Estudios exhaustivos efectuados en los Estados Unidos y en otros países condujeron al desarrollo de la técnica para codificar y leer caracteres impresos con tinta magnetizable. Estos estudios demostraron que los caracteres magnetizables son el medio más eficaz y confiable para leer documentos que han sido manipulados por el público. Por esta razón el uso de caracteres magnetizables se ha generalizado en todo el mundo y se han establecido estándares en los países que procesan cheques en forma automatizada.

En las siguientes secciones se presentan los detalles técnicos y especificaciones que deben cumplirse en los cheques, en la codificación de los campos y en los caracteres magnetizables y así facilitar en lo futuro el manejo de cheques entre los bancos.

1.4.1 Especificaciones para el diseño de los cheques.

Los bancos llegaron al acuerdo de establecer especificaciones estándar para el diseño de los cheques que se tramitan en el sistema bancario. Para llegar a estas especificaciones se tomaron en cuenta todos los factores que intervienen en el proceso de cheques, desde su impresión, personalización de chequeras de clientes y expedición de cheques, hasta su cobro. Como el trámite de los cheques en los sistemas internos de los bancos y en las cámaras de compensación del país van desde procesos manuales hasta procesos automatizados, los estándares establecidos sirven tanto para los procesos manuales como para los procesos automatizados.

Las ventajas de las especificaciones del diseño de cheques tienden a:

- Facilitar los procesos electrónicos de compensación de cheques incorporando elementos que garanticen la calidad y confiabilidad de su registro electrónico.

- Propiciar la evolución de los sistemas de compensación de cheques a nuevos y más eficientes esquemas de operación.
- Facilitar el desarrollo de aplicaciones de intercambio electrónico de información.
- Incorporar medidas lógicas y físicas de seguridad que dificulten realizar fraudes.

Para que los cheques sean procesados por los lectores electrónicos, es necesario que cumplan los requisitos de diseño que se detallan a continuación:

a) Material de los cheques
b) Legibilidad de la información.
c) Tamaño de los cheques.

1.4.1.1 Material de los cheques.

Los cheques deben estar impresos en papel seguridad para cheques (ver apéndice A , para mayor información acerca de las especificaciones, definiciones y características del papel para cheques.).

1.4.1.2 Legibilidad de la información.

Toda la información contenida en el cheque referente a banco, sucursal, localidad o plaza, número de cheque, número de cuenta, importes con letra y número, y el número de clave de tránsito, debe aparecer de una manera legible y fácilmente localizable a simple vista, y de acuerdo con la Ley General de Títulos y Operaciones de Crédito.

1.4.1.3 Tamaño de los Cheques.

Debido a limitaciones de los lectores de caracteres magnetizables el tamaño de los cheques deberá estar dentro de las siguientes dimensiones:

Largo. Entre 16.35 cm. mínimo y 22.22 cm. máximo (6 7/16" Mínimo y 8 3/4" Máximo).

Ancho. Entre 6.98 cm. mínimo y 9.31 cm. Máximo (2 3/4" Mínimo y 3 2/3" Máximo). (ver figura 1.4.1.3.1).

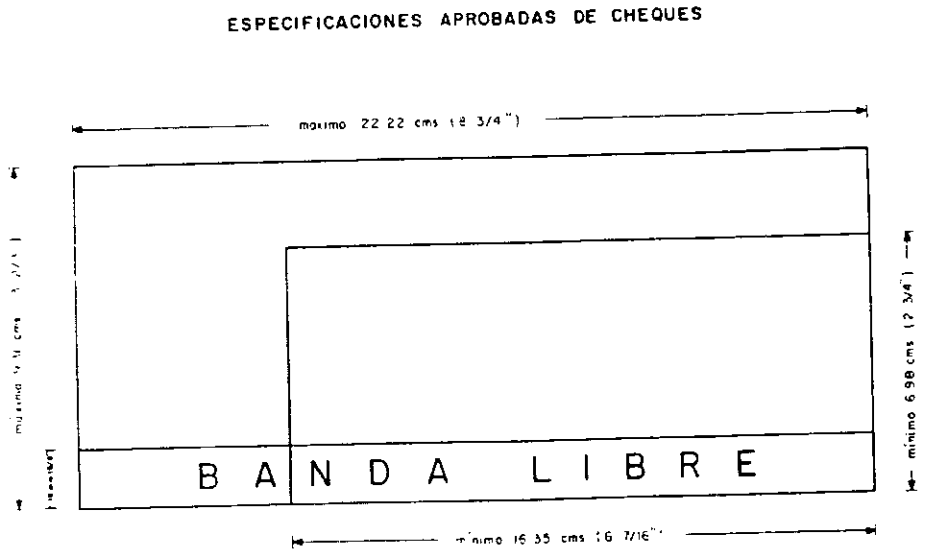


Figura 1.4.1.3.1 Dimensiones del cheque y banda libre

Banda libre.

Todos los cheques deben tener en su parte inferior una banda de 16 mm. (5/8"), medida a partir del extremo inferior del documento. el largo de la banda se extiende a todo lo ancho del documento, a este espacio se le denomina banda libre. (ver figura 1.4.1.3.1.)

La impresión de los caracteres magnetizables está limitada a una porción dentro de la banda libre. A ésta zona se le llama banda de impresión y tiene una altura total de ¼ ". La banda de impresión se encuentra a 3/16" sobre el extremo inferior del cheque. Debe existir un espacio mínimo de 1/8" entre el último carácter de la banda libre y el extremo izquierdo del documento y, ¼" del extremo derecho. (ver figura 1.4.1.3.2)

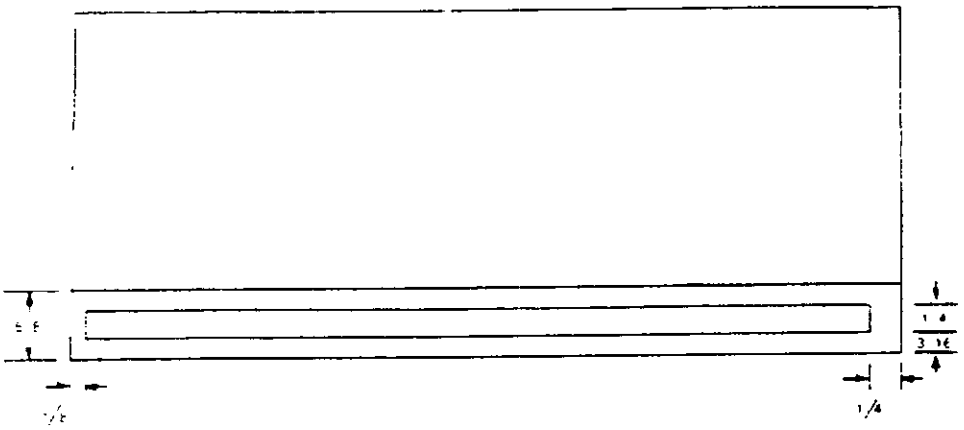


Figura 1.4.1.3.2 Banda de impresión

1.4.2 Especificaciones de Lenguaje Común.

El estándar seleccionado para usarse en los cheques que se procesarán electrónicamente dentro del sistema bancario, es el carácter magnetizable E13-B. Este carácter ha sido probado en los Estados Unidos desde 1959. También ha sido probada la distribución y contenido de los campos codificados con dichos caracteres en los cheques, que manejan los bancos.

1.4.2.1 Lenguaje Común.

Las características de los caracteres E13-B es lo que constituyen el lenguaje común.

1.4.2.1.1 Configuración de los caracteres.

Las especificaciones de este lenguaje común han sido tomadas del estándar americano en el año de 1976 por el departamento de Automatización de **THE AMERICAN BANKERS ASSOCIATION** (Asociación de Banqueros Estadounidenses).

1.4.2.1.2 Designación.

Para codificar los datos que deben aparecer en la banda libre, se dispone de 14 caracteres que incluye los símbolos que representan tanto los dígitos de 0 al 9, así como 4 símbolos especiales, denominados "delimitadores", utilizados para reconocer en donde comienza y termina cada campo

1.4.2.1.3 Descripción.

A continuación se describen los caracteres E13-B:

10 Dígitos Caracteres de 0 a 9

4 Símbolos

Carácter No. 11

Símbolo de "número de tránsito", Indica al lector en dónde empieza y en dónde termina el número de tránsito.

Carácter No. 12

Símbolo de "importe". Indica al lector en dónde empieza y en dónde termina la codificación de importe del cheque.

Carácter No. 13

Símbolo "a nuestro cargo". Indica los límites de los campos destinados al uso interno de cada banco.

Carácter No. 14

Símbolo de "guión". Sirve para separar campos y así facilitar su lectura visual.

1.4.2.1.4 Dimensión de los caracteres.

La dimensión de los caracteres, es la forma y tamaño de cada uno de los símbolos E13-B. La dimensión nominal de cada carácter es un múltiplo de la medida de 0.013". En las dimensiones del carácter existe una tolerancia de ±0.0015" sobre la medida nominal del carácter.

1.4.2.1.5 Altura de los caracteres.

La altura de cada uno de los caracteres E13-B es de 0.117 de pulgada

1.4.2.1.6 Ancho de los caracteres.

Cada uno de los caracteres E13-B está formado por filas y columnas de 0.013" de ancho. A estas filas y columnas se les conoce también como barras. El ancho mínimo aceptable de cada barra es de 0.010" (ver figura 1.4.2.1.6.1). El ancho mínimo de una barra doble es de 0.023". (ver figura 1.4.2.1.6.2). Dado que el perfil típico de un carácter no es una línea recta, la medición es tomada sobre el perfil promedio del carácter. El perfil promedio es una línea imaginaria que divide las irregularidades del contorno lo más equilibradamente posible y es paralela al perfil nominal del carácter. (ver figura 1.4.2.1.6.3).

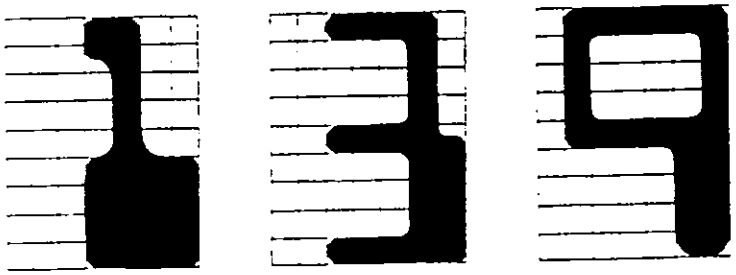


Figura 1.4.2.1.6.1 Filas y columnas

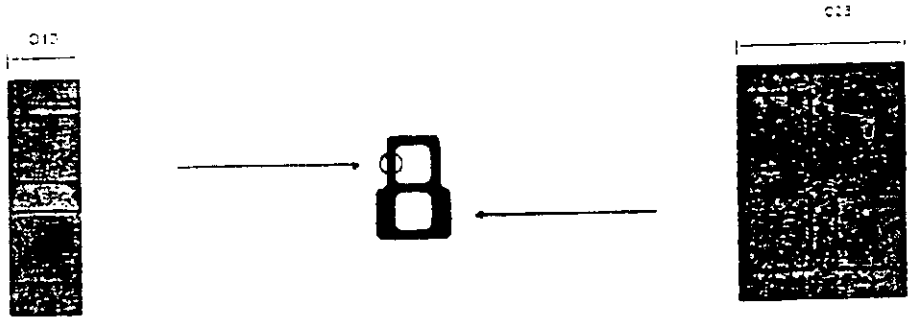


Figura 1.4.2.1.6.2 Tolerancias máximas y mínimas

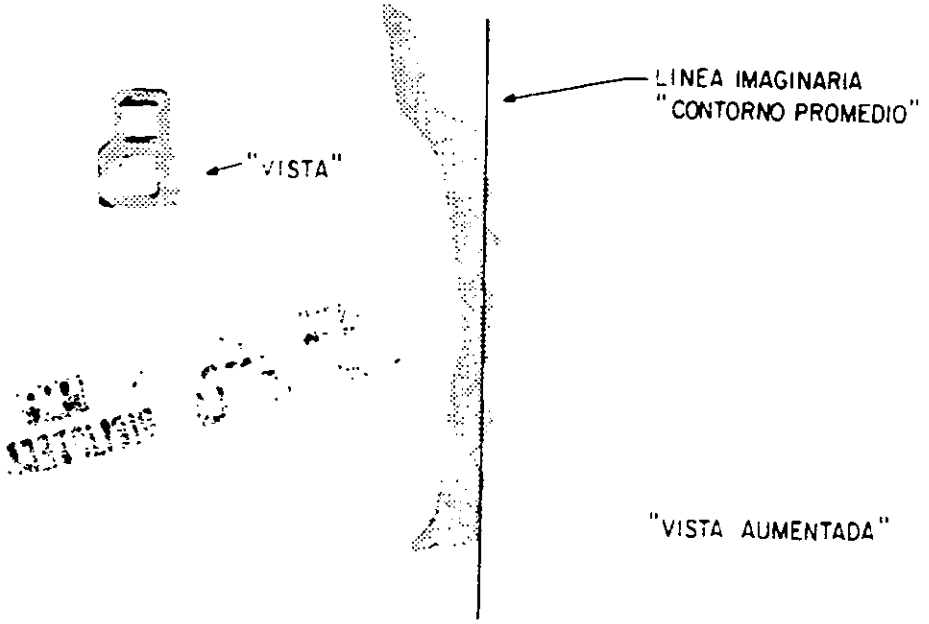


Figura 1.4.2.1.6.3 Perfil promedio

1.4.2.2 Descripción de los campos y uso de los símbolos.

La descripción de los campos se refiere a las consideraciones de uso y significado de los caracteres magnetizables en el cheque.

1.4.2.2.1 Campo de importe.

Dentro de este campo aparece codificado el importe por el cual fue expedido el cheque, considerando los centavos. El importe utiliza 10 dígitos (rellenando de ceros las posiciones no utilizadas a la izquierda). El campo del importe quedará delimitado por dos símbolos de importe.

"IMPORTE" (⑈ I I I I I I I I I I ⑈)

1.4.2.2.2 Campo de número de tránsito.

Este campo define los límites dentro de los que debe aparecer el número de tránsito, en ocasiones llamado también número de ruta. El número de tránsito consta de 11 dígitos y esta formado de la siguiente manera:

- Código de transacción (TT)
  Identifica el tipo de documento
- Plaza de compensación (PPP)
  Identifica la plaza a la que corresponde la cuenta
- Número de banco (BBB)
  Identifica al banco emisor del documento.
- Dígito verificador de intercambio (Di)
  Este dígito se calcula sobre los datos de número de banco y cuenta con un algoritmo especial.

El número de tránsito debe estar siempre delimitado por dos símbolos de tránsito.

ESTA TESIS NO DEBE SALIR DE LA BIBLIOTECA

"TRANSITO" (⑆ TTPPPBBBDi ⑆)

1.4.2.2.3 Campo de "a nuestro cargo"

Este campo contiene información relativa al número de cuenta y al número de cheque del documento, tiene capacidad para 19 caracteres. Debe contener un símbolo de a nuestro cargo.

(CCCCCCCCCCC ⁞ NNNNNNN)

1.4.2.2.4 Campo de clave de transacción.

Este campo también llamado certificación de autenticidad utiliza 4 posiciones para codificar la información y esta formado de la siguiente manera:

- Código de seguridad
  Es usado por los bancos para certificar la autenticidad del documento; o bien para imprimir algún otro dato que sea de su interés.
- Dígito verificador de premarcado (Dp)
  Este dígito se calcula aplicando un algoritmo con todos los datos premarcados en el cheque.

(SSSD$_P$)

Tres de los símbolos E13-B tienen la función de ser delimitadores de campo. El delimitador de campo indica al equipo lector donde inicia y donde termina un determinado campo. El guión no es delimitador de campo y únicamente puede ser utilizado como separador en el campo a nuestro cargo.

1.4.2.3 Plantilla común.

Se refiere a la localización de los caracteres magnetizables impresos en el cheque.

1.4.2.3.1 Márgenes de referencia.

- Referencia horizontal
  Todas las distancias que se indiquen en sentido horizontal, están medidas a partir del margen derecho del cheque.

- Referencia vertical
  Todas las alturas que se indiquen, están medidas tomando como base el margen inferior del cheque.

1.4.2.3.2 Banda libre de los caracteres magnetizables.

Las dimensiones mínimas de esta banda libre son 5/8" de ancho, medidas desde el margen inferior del cheque y 6 7/16", medidas desde el margen derecho del cheque.

1.4.2.3.3 Localización horizontal y límites de los campos.

Cada uno de los campos impresos en el cheque, tienen un área reservada para su codificación dentro de la banda de impresión, la cual se define a continuación especificando la distancia que deben tener los límites izquierdo y derecho de cada campo a partir del margen derecho del cheque.

- Campo de importe
  Los límites de este campo están a ¼" y 1 7/8" del margen derecho del documento.
  El lado derecho del primer símbolo que aparezca en este campo debe quedar impreso a 5/16", con una tolerancia de más o menos 1/16".

- Campo a nuestro cargo
  Los límites de este campo están a 1 15/16" y 4 5/16" del margen derecho del cheque.

- Campo de número de tránsito
  Los límites de este campo están a 4 5/16" y 5 11/16" del margen derecho del cheque.
- Campo de certificación de autenticidad
  Los límites de este campo están a 5 11/16" y 6 1/16" del margen derecho.

(ver figura 1.4.2.3.3.1)

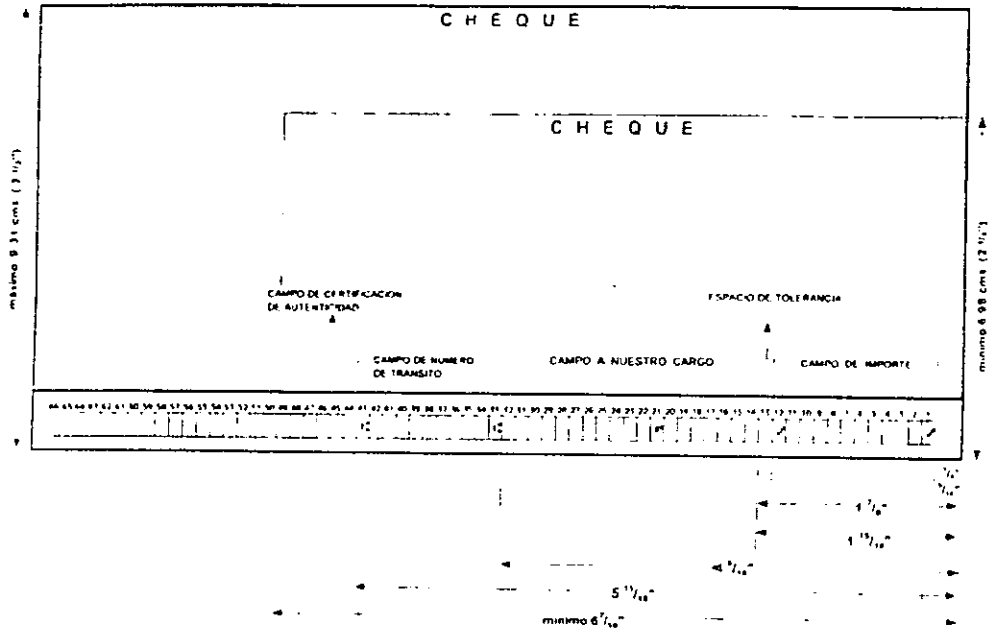


Figura 1.4.2.3.3.1 Plantilla común y limites de los campos

1.4.2.3.4 Localización vertical de los caracteres.

- Banda de impresión

  Todos los caracteres magnéticos deben quedar impresos dentro de una banda de ¼" de ancho, la cual está centrada dentro de la banda libre de 5/8" (16 mm) localizada en la parte inferior del cheque. (ver figuras 1.4.2.3.3.1 y 1.4.2.3.4.1)

- Tolerancia vertical

  La orilla inferior de los caracteres magnetizables debe quedar impresa o codificada a 3/16" como mínimo del margen inferior del cheque. La orilla superior de los caracteres debe quedar a una altura no mayor a 7/16 de pulgada del margen inferior del cheque. (ver figura 1.4.2.3.4.1)

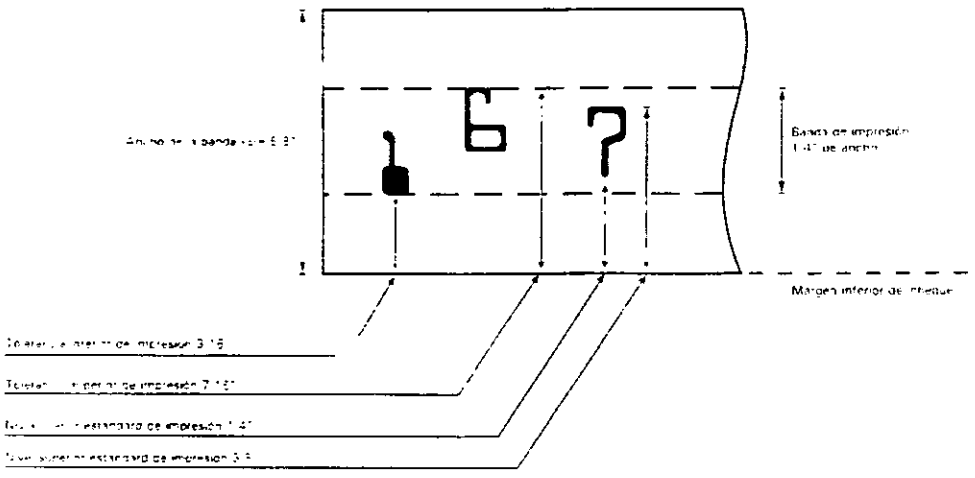


Figura 1.4.2.3.4.1 Tolerancia vertical.

1.4.2.3.5 Inclinación.

Es la rotación del carácter sobre su eje vertical, eje que debe ser perpendicular al extremo inferior del documento. La máxima inclinación permisible en la impresión de un carácter magnetizable es de ± 1.5 grados. (ver figura 1.4.2.3.5.1)

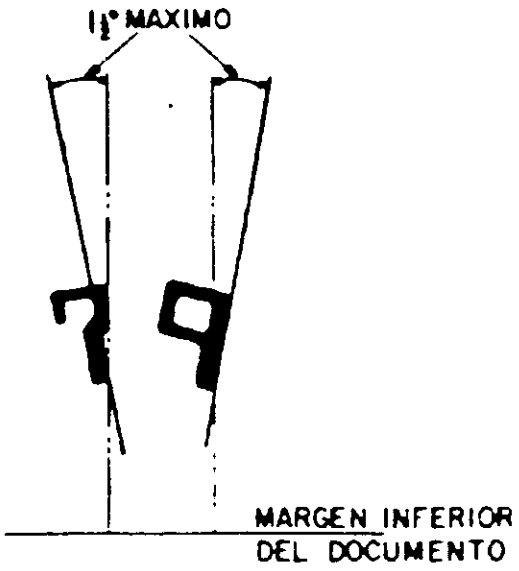


Figura 1.4.2.3.5.1 Inclinación de los caracteres.

1.4.2.3.6 Alineación.

Es la alineación relativa vertical de caracteres adyacentes localizados dentro de un mismo campo, esta alineación no deberá variar en sentido vertical más de 0.007" (ver figura 1.4.2.3.6.1)

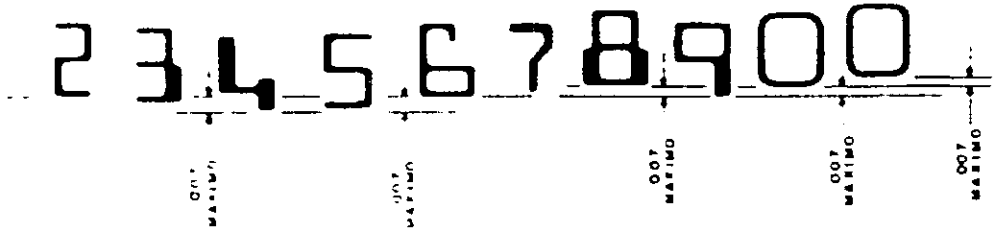


Figura 1.4.2.3.6.1 Alineación de los caracteres.

1.4.2.3.7 Espaciado.

Es la distancia entre el perfil derecho de caracteres adyacentes, la distancia entre los perfiles derechos de caracteres adyacentes debe ser 0.125" ±0.10". (ver figura 1.4.2.3.7.1)

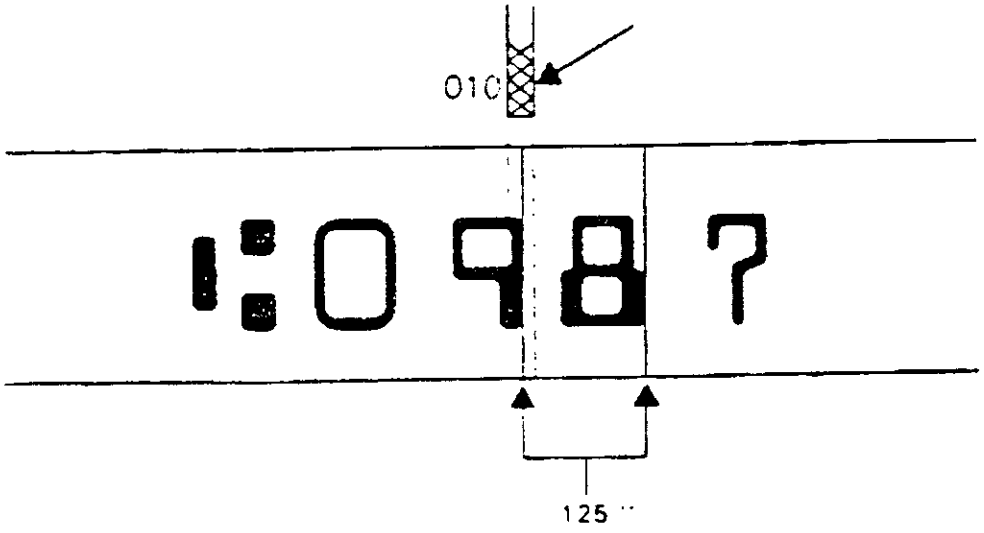


Figura 1.4.2.3.7.1 Espaciado.

1.4.3 Señales.

Las señales, son la representación gráfica del impulso eléctrico que produce un carácter magnetizado al ser leído electrónicamente.

1.4.3.1 Nivel de la señal.

Por nivel de la señal se debe entender la amplitud del impulso eléctrico que genera un carácter magnetizable en los dispositivos sensores electromagnéticos. La amplitud de la onda es el parámetro que se utiliza para medir el nivel de señal, cuando el nivel de la señal aumenta o disminuye, la amplitud de la onda aumenta o disminuye. (ver figura 1.4.3.1.1)

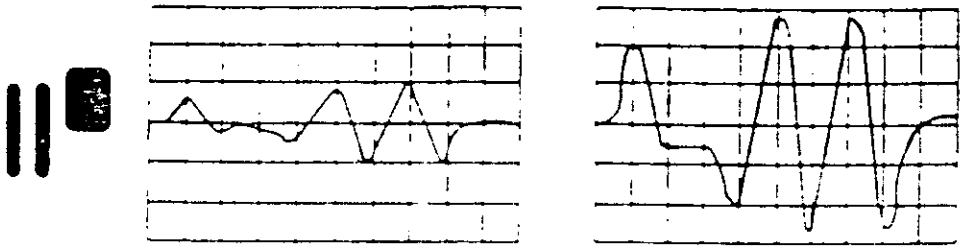


Figura 1.4.3.1.1 Amplitud de la onda.

Se ha establecido lo que se conoce como nivel de señal "nominal". Este nivel de señal es el que cada carácter producirá bajo el supuesto de que estuviese impreso bajo las mismas condiciones de impresión. Cada carácter produce un nivel de señal diferente debido a que su configuración es diferente. (ver figura 1.4.3.1.2)

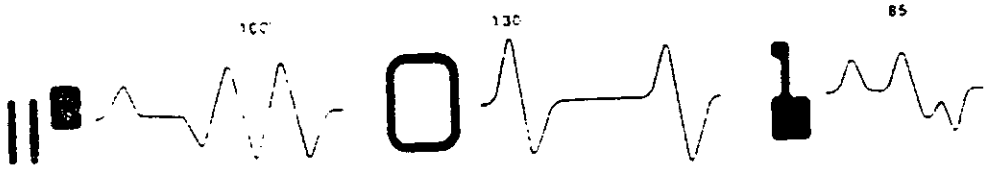


Figura 1.4.3.1.2 Nivel de la señal nominal.

Las especificaciones para el nivel de la señal permiten una desviación desde el 50% hasta el 200% del valor nominal. Como algunas desviaciones pueden ocurrir por variaciones de tinta o absorción del papel, se debe procurar que las señales generadas por los caracteres magnetizables estén los más cerca posible del nivel de señal de 100%.

Los valores nominales para cada uno de los caracteres E13-B se muestran en la siguiente tabla:

Rangos del nivel de señal

Carácter	Mínimo	Nominal	Máximo
0	65.0	130	260
1	42.5	85	170
2	52.5	105	210
3	42.5	85	170
4	52.5	105	210
5	52.5	105	210
6	52.5	105	210
7	37.5	75	150
8	52.5	105	210
9	82.5	165	330
Tránsito	52.5	105	210
Importe	35.0	70	140
Nuestro cargo	50.0	100	200
Guión	33.5	67	134

Para calcular el porcentaje de desviación del nivel de la señal con respecto al valor nominal se debe usar la siguiente formula:

% del valor nominal = (nivel de la señal / nivel de señal nominal) X 100

1.4.3.2 Formas de onda de las señales.

Cada uno de los caracteres E13-B produce un impulso eléctrico con perfil diferente a los demás, lo que permite el reconocimiento de cada uno de ellos cuando es leído. La banda magnetizable es leída por los equipos de lectura de derecha a izquierda. (ver figura 1.4.3.2.1)

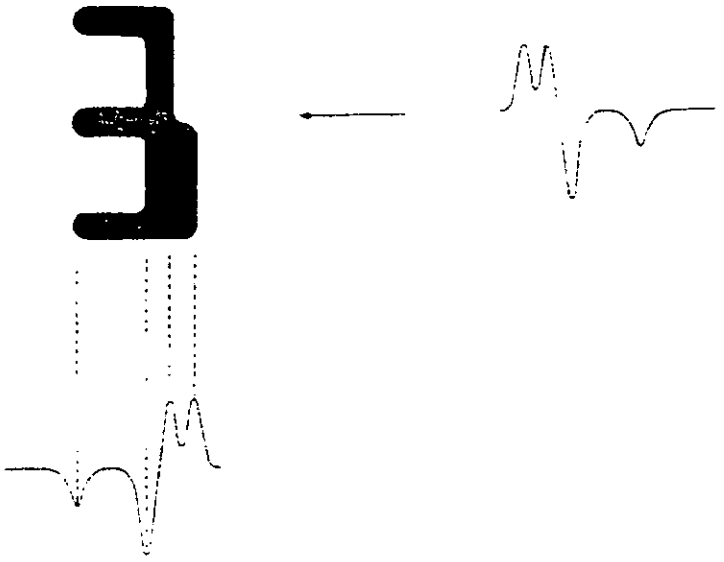


Figura 1.4.3.2.1 Forma de onda de un carácter vista en un osciloscopio.

La forma y amplitud de la onda está determinada por el largo y la pendiente del perfil vertical del carácter, el ancho de las columnas verticales, de los espacios vacíos, de la cantidad y uniformidad del óxido de hierro en la tinta. Cuando las partículas metálicas en la tinta se magnetizan, se crea un campo eléctrico positivo en la parte derecha de cada partícula y un campo negativo en el lado izquierdo. (ver figura 1.4.3.2.2)

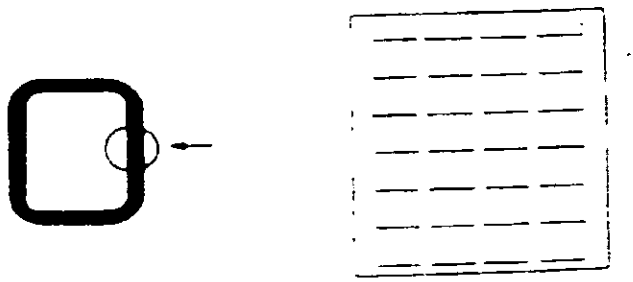
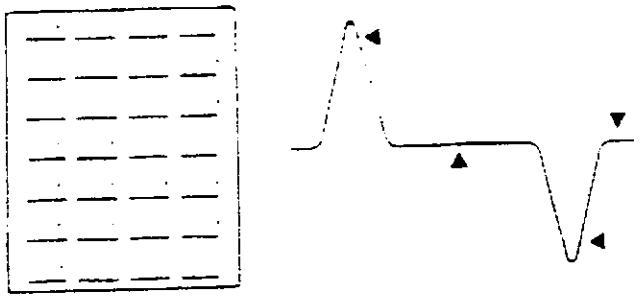


Figura 1.4.3.2.2 Carga de las partículas.

Esto ocasiona que el perfil derecho del carácter tenga una carga positiva y el perfil izquierdo una negativa. La parte central del carácter tiene prácticamente cantidades iguales de cargas positivas y negativas, que se cancelan mutuamente, resultando en una carga acumulada neutral. Conforme el sensor de lectura recorre el carácter, las columnas que tienen una suma positiva (los perfiles derechos) generan una curva ascendente o positiva. Las columnas que tienen una suma negativa (perfiles izquierdos) generan una curva descendente o negativa. (ver figura 1.4.3.2.3)



Figuras 1.4.3.2.3 Curvas ascendentes y descendentes.

Si la suma de las cargas positivas y negativas en la columna son iguales, las cargas se cancelan unas a otras y el perfil de onda tiende a regresar o permanecer en la línea

base. Los picos y los valles son directamente proporcionales a la carga positiva o negativa del perfil del carácter. Dos perfiles verticales de la misma altura, contenido y la misma concentración de óxido de hierro, producen la misma altura en el perfil de onda. (ver figura 1.4.3.2.4)

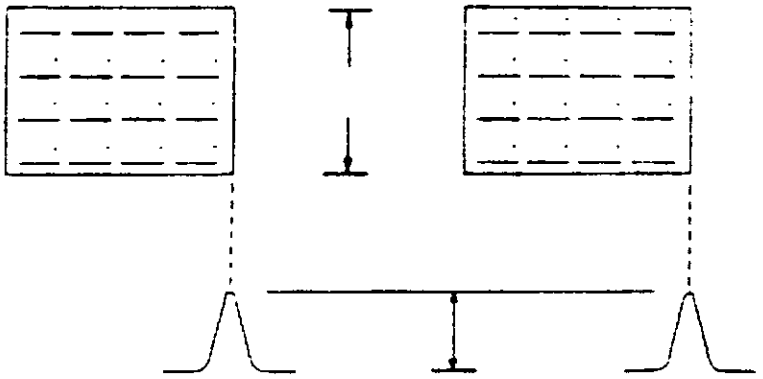


Figura 1.4.3.2.4 Misma altura en el perfil de onda.

Las cargas positivas y negativas a lo largo del perfil vertical aumentan conforme la altura de la columna se incrementa debido a la concentración del óxido de hierro presente en la columna.

Por lo tanto, al aumentar la altura del perfil vertical del carácter se incrementa la altura del perfil de onda. (ver figura 1.4.3.2.5). La altura de la onda también se incrementa en función del contenido de óxido de hierro presente en la columna del carácter

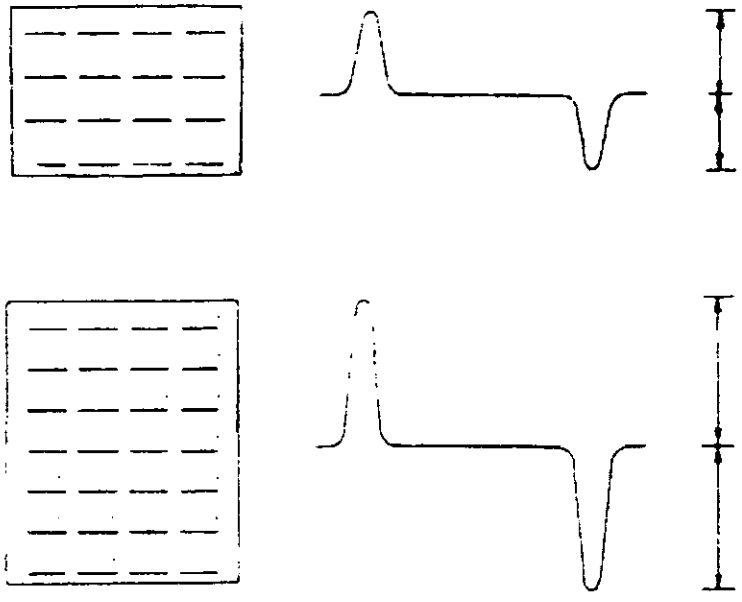


Figura 1.4.3.2.5 Altura del perfil de onda.

Al aumentar el ancho de la columna se incrementa el área neutral de los perfiles verticales de entrada y salida. Esto se debe al mayor tiempo que tarda la cabeza lectora para recorrer la distancia entre ambos perfiles. El incremento en el tiempo ocasiona que los picos generados por los perfiles verticales aparezcan con mayor separación y produce en la onda, un tramo neutral más largo. (ver figura 1.4.3.2.6)

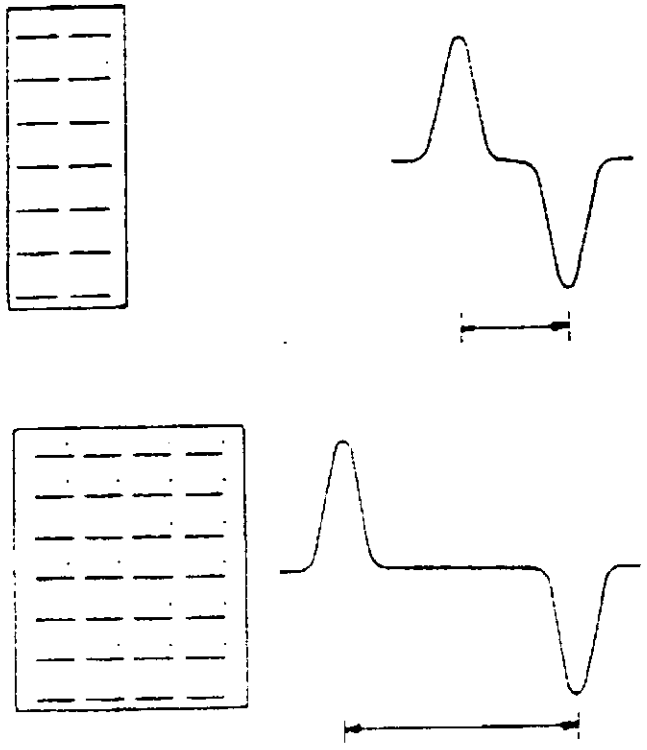


Figura 1.4.3.2.6 Tramos neutrales.

A continuación se muestran las formas de onda de todos los caracteres E13-B en la figura 1.4.3.2.7

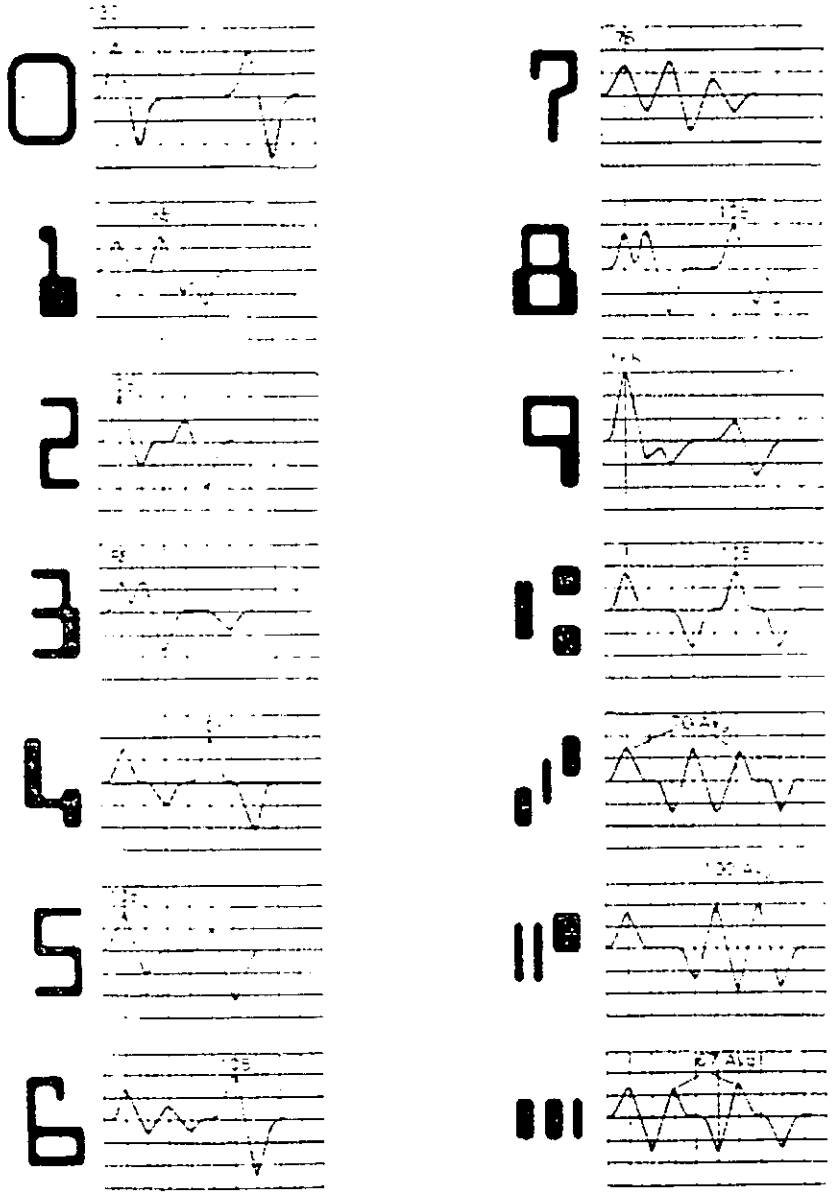


Figura 1.4.3.2.7 Formas de onda de los caracteres E13-B.

1.4.3.3 Tinta magnetizable.

Tinta para la impresión de caracteres E13-B la cual contiene un alto contenido de óxido de hierro lo que permite que dichos caracteres puedan ser magnetizados y consecuentemente reconocidos electrónicamente por los equipos lectores.

1.4.3.4 Perfil irregular.

El perfil irregular es la parte del contorno del carácter que está hecha de picos y valles. Se permite que los picos y valles del contorno del carácter invadan la zona de tolerancia de 0.0015" a 0.0035", siempre y cuando esa invasión se limite al 25% del contorno del carácter. (ver figura 1.4.3.4.1)

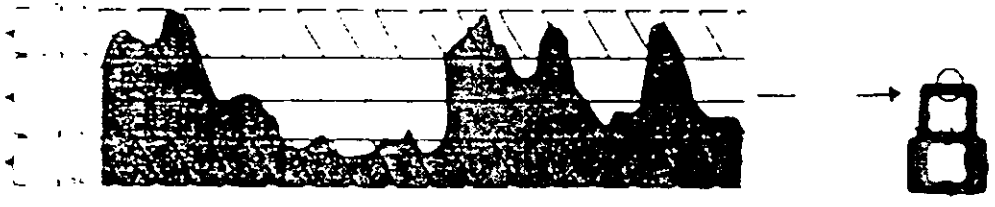


Figura 1.4.3.4.1 Perfil irregular

1.4.3.5 Falta de tinta

Falta de tinta es la ausencia de tinta dentro del contorno especificado del carácter. Se permite un solo faltante de tinta en cualquier lugar del carácter si este puede estar contenido en un recuadro de 0.008". (ver figura 1.4.3.5.1)

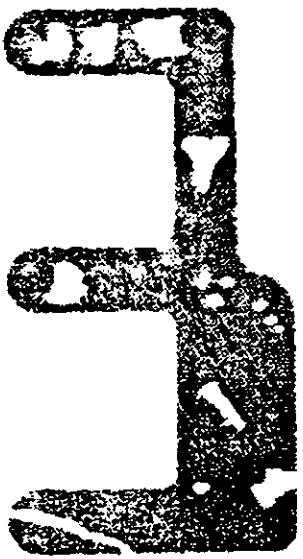


Figura 1.4.3.5.1 Falta de tinta en el carácter.

1.4.3.6 Uniformidad de la tinta.

La uniformidad de la tinta es la regularidad en la distribución de la película de tinta dentro de los contornos del carácter. La aplicación de la película de tinta debe hacerse de tal manera que. dentro del contorno del carácter, ninguna zona contenga una cantidad mayor de tinta que otra. (ver figura 1.4.3.6.1)

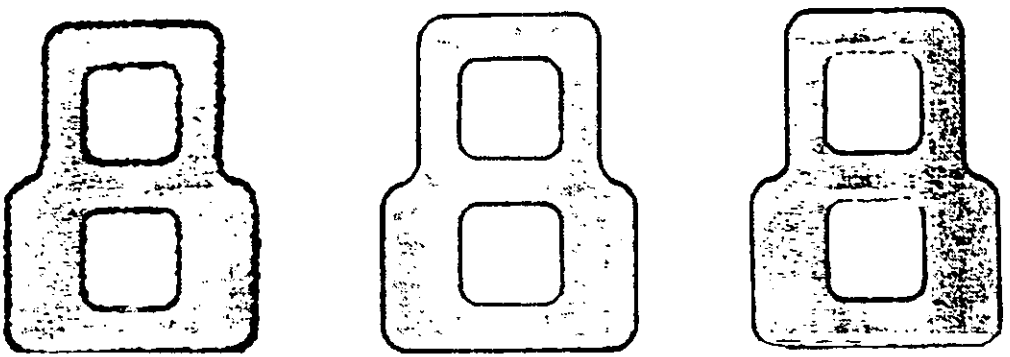


Figura 1.4.3.6.1 Uniformidad de la tinta en el carácter.

1.4.3.7 Manchas de tinta.

Cualquier tinta magnetizable que aparece dentro de la banda libre y fuera del contorno de los caracteres magnetizables. Las manchas de tinta magnetizable en el anverso del documento están dentro de especificaciones cuando pueden estar contenidas en un recuadro de 0.003". (ver figura 1.4.3.7.1)

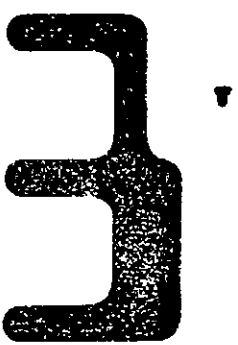


Figura 1.4.3.7.1 Manchas de tinta en los caracteres

1.4.3.8 Bajorrelieve.

Bajorrelieve es la variación en la superficie del papel producida por el impacto en el momento de impresión del carácter. El bajorrelieve en el frente del documento no debe exceder de 0.001" (diferencia entre la superficie del papel y la superficie del carácter impreso). (ver figura 1.4.3.8.1)

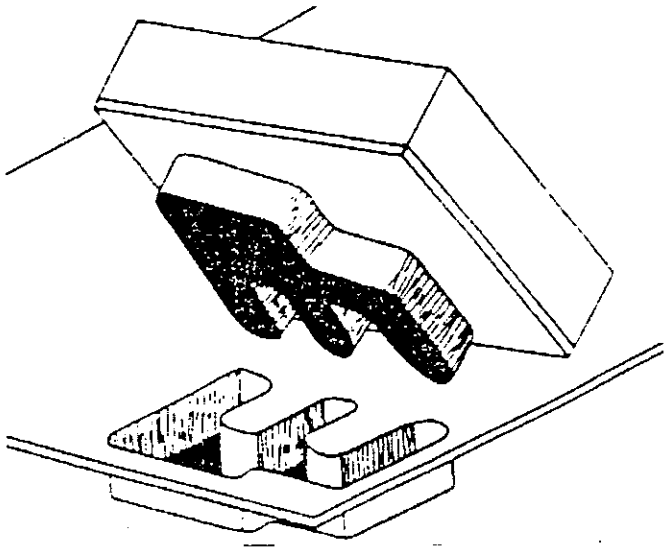


Figura 1.4.3.8.1 Bajorrelieve

El bajorrelieve ocasiona que al leerse un carácter, éste pase más lejos de la cabeza lectora. ello puede ocasionar una perdida en la señal magnética. La tolerancia permitida es de 0.001".

1.4.4 Validación.

El proceso de validación requiere de una previa identificación de los campos que componen el cheque, la cual consiste en obtener cada uno de estos, a través de los delimitadores impresos, la validación se efectúa en dos fases

- La primera fase permite registrar un carácter valido
- La segunda fase se efectúa por medio de programas de computadora, a través de los cuales podemos saber si la información contenida en un campo es valida. Por ejemplo, que la clave de un banco exista.

1.4.4.1 Imagen lógica.

Es la información lógica que se maneja una banda magnética después de haber sido leída por un equipo lector. Esta imagen puede aparecer en pantalla o registrarse en un archivo de computadora.

1.4.4.2 Peso, módulo y dígito verificador.

Dentro del campo tránsito la novena posición representa el dígito verificador de intercambio, y dentro del campo de certificación de autenticidad la cuarta posición representa el dígito verificador de premarcado. Estos dígitos son utilizados para validar la información leída en los demás campos del cheque y de esta manera verificar la correcta información impresa en el documento.

El peso y módulo con los que se calculan los dígitos verificadores han sido asignados por la Asociación Mexicana de Banqueros.

1.4.5 Formato único por banco.

El formato único por banco es un formato de banda magnética de características comunes para todas las zonas geográficas donde un determinado banco tiene sucursales. Para una institución financiera la adopción de un formato único significa fundamentalmente, uniformar la utilización del campo a nuestro cargo, es decir, normalizar el peso, módulo, la longitud de la cuenta y la localización de los componentes del campo a nuestro cargo.

Entre las ventajas que se pueden encontrar con un formato único por banco están:

- El manejo de la información dentro de la institución sea estandarizada
- Se disminuyen los errores de captura
- Se obtiene mayor seguridad en la información que se valida
- Se puede validar la totalidad de la banda magnetizable.

1.4.6 Normatividad en la emisión de cheques bancarios.

1.4.6.1 Autoridades en la materia.

La Asociación Mexicana de Bancos (AMB) junto con el Banco de México y la Cámara de Compensación son los encargados de la estandarización de los cheques y están supervisados por la Comisión Nacional Bancaria y de Valores

1.4.6.2 Reglamentación de las operaciones a través de cheques.

Todas las operaciones que se realicen a través de cheques se rigen por la "Ley General de Títulos y Operaciones de Crédito", contenidas en el capitulo IV secciones primera y segunda.

1.5 CONCEPTOS BÁSICOS DE CONTABILIDAD FINANCIERA.

1.5.1.1 Origen.

Desde siempre el individuo se ha esforzado por tener información. Los procedimientos para ello, han evolucionado en función de las caracteristicas y necesidades del medio. Las primeras etapas de la contabilidad como medio para controlar y proporcionar información financiera, se aprecia en el siglo XV, cuando en Italia, el monje Fray Luca Paciolo crea libros para registrar la obtención y aplicación de recursos consecuentes a las operaciones realizadas por las entidades, definiendo a su vez reglas para su manejo. Así, esos libros satisfacen las necesidades de un periodo de la historia. Una expresión más del proceso evolutivo de la contabilidad, se tiene a fines del siglo XVIII, cuando Edmond Le Granje, en Francia, implanta el libro mayor tabular, cuya importancia es superior, por contar con las caracteristicas básicas de los registros tabulares posteriores.

En la actualidad, la tecnología está a las ordenes de la contabilidad para implantar, reajustar o sofisticar sistemas completos, que proporcionen información financiera con claridad, veracidad y oportunidad deseada, lo cual permiten descansar en ella, la confianza para tomar decisiones de diferente índole.

1.5.1.2 Definición.

El aprovechamiento óptimo de los recursos con los cuales satisface el hombre sus necesidades, requiere de un adecuado control de los mismos, que asegure su mas conveniente utilización y le de la oportunidad de evitar que se desperdicien. Este control implica tomar nota de cada una de las transacciones que con los bienes económicos se realicen, hacer grupos que contengan operaciones del mismo género y determinar el volumen con que llega a realizarse cada tipo de operación.
La contabilidad constituye, precisamente, el medio de controlar dichas transacciones cuyas caracteristicas son substancialmente económicas, lo cual ha dado lugar a que se

considere que la información que proporciona es económica. La contabilidad capta únicamente el efecto financiero de las transacciones efectuadas con bienes económicos. (El término financiero es aplicable a operaciones, documentos, instituciones, personas, etc., relacionados con la obtención y el uso del dinero).

Congruentes con lo hasta aquí comentado, la contabilidad se define como:

Contabilidad es la técnica constituida por los métodos, procedimientos e instrumentos, aplicables para llevar a cabo el registro, clasificación y resumen de los efectos financieros que provocan las operaciones que realiza la empresa, con el objeto de obtener la información necesaria para elaborar estados financieros, figura 1.5.1.2.1.

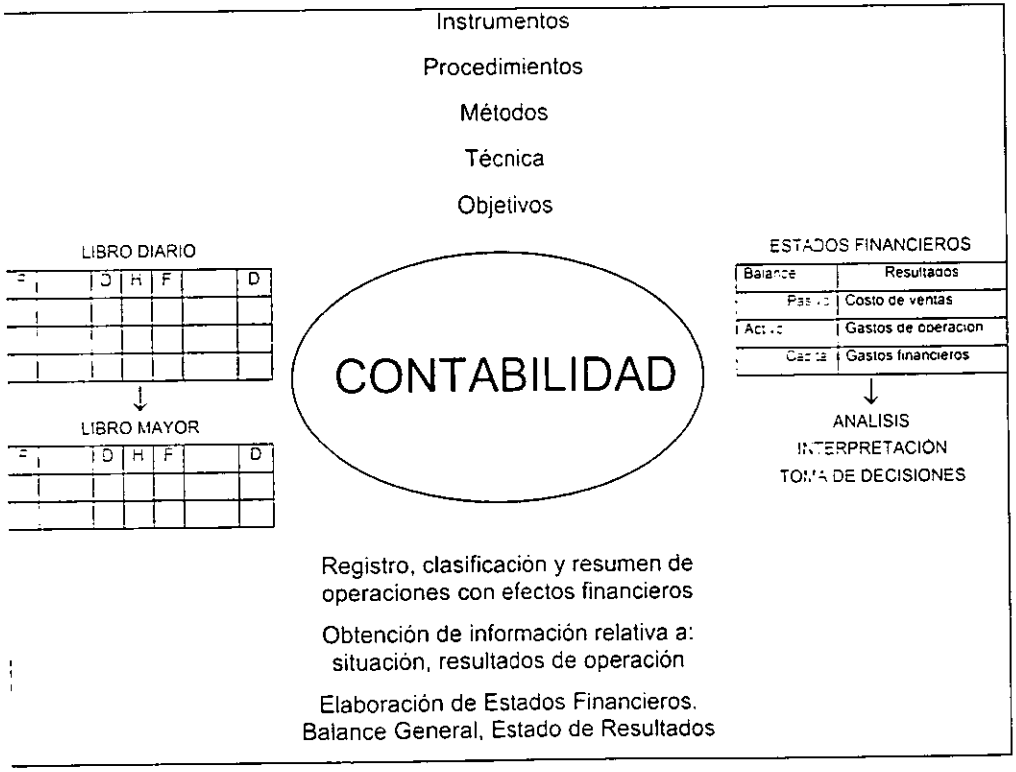


Figura 1.5.1.2.1 Contabilidad

1.5.2 Importancia y utilidad.

La sociedad de nuestros días es una expresión de toda una corriente de comunidades con características particulares, siendo algunas de ellas tan específicas que han constituido etapas fundamentales en el curso de la historia.

Todo este devenir histórico, hasta hoy en día, así como experimenta condiciones de diferencias, también presenta características y necesidades comunes, como lo es, la necesidad de información financiera y la participación imprescindible de la contabilidad como medio para satisfacer directamente tales necesidades.

Se ha confirmado que cada sociedad ha tenido, tiene y tendrá necesidad de contar con información financiera. Pero la sociedad en sí, como un todo, se encuentra constituida por instituciones, gobiernos. empresas. organizaciones profesionales y demás que a mayores recursos materiales y humanos que poseen para la consecución de fines determinados, van requiriendo de amplia información financiera para tomar decisiones que le permitan alcanzar sus metas.

Tiempo atrás, por las características propias de la sociedad, la necesidad de controlar los recursos de las entidades no tenía las atenciones que hoy exige. En la actualidad. al vivir en una comunidad con características múltiples y complejas impulsa al ser humano a ser más analítico e investigar con amplitud para tomar decisiones que con mayor o menor repercusión afectará el curso de sus actividades. Por las mismas razones, los recursos de las entidades de nuestros días guardan una **importancia** de tal magnitud que crea en ellas la necesidad imprescindible de contar con información que exprese la forma en que se ha obtenido.

Parece ser, que los matices de la sociedad actual dejan vislumbrar un futuro en el cual la necesidad de controlar los recursos y obtener información financiera será de exigencia mayor.

El avance tecnológico no es que refleje un estancamiento ni mucho menos es en sí la evolución de la sociedad quien brinda retos a cada instante para medir la capacidad creativa del mismo ser humano quien resulta ser el creador de necesidades y satisfactor directo de las mismas.

Existen entidades comerciales, industriales, agrícolas, ganaderas, pesqueras, etc., que implantan ciertos sistemas para controlar sus recursos y como resultado de ellos expresan tener utilidades en tanto que pagan sus deudas y conservan un remanente.

Hoy en día los problemas son múltiples y las decisiones deben tomarse con mayor rapidez y superior índice de eficiencia. Por tanto se requiere de una adecuada información financiera.

La contabilidad, es el medio que, por sus métodos y técnicas, permite controlar y visualizar, a través de estados financieros, información clara, veraz y oportuna de todos los recursos de la entidad y solo así poder tomar decisiones, conscientes de sus consecuencias.

Fines fundamentales de la contabilidad son:

- Establecer un control riguroso sobre cada uno de los recursos y las obligaciones de la empresa.

- Registrar, en forma clara y precisa, todas las operaciones efectuadas por la empresa durante el ejercicio fiscal.

- Proporcionar en cualquier momento, una imagen clara y verídica de la situación financiera que guarda el negocio.

- Prever con bastante anticipación el futuro de la empresa.

- Servir como comprobante y fuente de información, ante terceras personas, de todos aquellos actos de carácter jurídico en que la contabilidad puede tener fuerza probatoria conforme lo establecido por la ley.

1.5.3 Principios de contabilidad generalmente aceptados.

Las reglas y convenios de la contabilidad financiera son comúnmente llamados "***principios***". Los principios contables no detallan exactamente cómo debe registrarse cada acontecimiento o negocio. En consecuencia, existen infinidad de aspectos en los cuales la práctica contable difiere de una empresa a otra. Algunas de estas diferencias son inevitables, ya que un grupo de reglas detalladas no podría aplicarse a todas las empresas. Algunas otras diferencias reflejan el hecho de que el contador tiene considerable margen dentro de ***los principios de contabilidad generalmente aceptados*** para expresar sus propias ideas en cuanto a la mejor forma de registrar un acontecimiento específico o informar acerca del mismo.

Por lo anterior, es imposible conocer el significado preciso de muchos de los datos contenidos en un informe contable, a menos que conozca cuál de las diversas interpretaciones igualmente aceptables ha sido seleccionada por la persona que formuló el informe.

A continuación describimos siete principios o conceptos que son básicos para la comprensión de la contabilidad:

1. **La moneda, común denominador.** En contabilidad financiera se registran sólo aquellos hechos que pueden ser expresados en términos monetarios. Por tanto, la contabilidad no proporciona un detalle completo de lo que sucede en el negocio, ni nos ofrece un cuadro exacto de las condiciones del mismo. Consecuentemente, la persona que lea un informe contable no debe esperar encontrar en él todos los hechos relacionados con el negocio.

La ventaja de expresar hechos en términos monetarios consiste en que el dinero nos provee de un común denominador mediante el cual los hechos relacionados con una negociación pueden reducirse a términos numéricos que pueden sumarse o restarse. Aun cuando el principio del común denominador es esencial y la moneda es, probablemente, el único denominador práctico, el uso de la unidad monetaria implica algo homogéneo, o sea, la similitud básica entre un peso y otro.

2. **La entidad mercantil.** La contabilidad se lleva para las negociaciones como entidades y no para las personas asociadas a las mismas. Al registrar los hechos en las cuentas la pregunta que importa es: ¿cómo afectan estas operaciones al negocio?, y no la forma en que puedan afectar a las personas que manejan o que son propietarias del mismo. Cuando el propietario toma efectivo de su negocio, por ejemplo, los registros contables muestran que el negocio tiene menos efectivo que antes, ya que el propietario tomó efectivo del negocio y lo llevó a su bolsa personal, con la idea de que ese dinero sigue siendo de su propiedad.

   Algunas veces es difícil definir la entidad para la cual se lleva la contabilidad. En el caso de una sociedad es fácil hacer la distinción. Una sociedad es una entidad legal separada de las personas dueñas de la misma, y la contabilidad de muchas sociedades corresponde exactamente al alcance de las operaciones de la entidad legal relativa.

   De la distinción entre la entidad mercantil y el resto del mundo se sigue la idea de que uno de los más importantes propósitos de la contabilidad financiera es proveer de bases para información sobre administración de recursos. A los directivos de un negocio se les han confiado fondos suministrados por los propietarios, por los bancos y por otros. La administración es responsable por el uso inteligente de esos fondos, y los informes financiero-contables, en parte, se han diseñado para mostrar cómo se ha desempeñado esa responsabilidad. Este es un aspecto parcial del propósito de la contabilidad financiera, pero explica los razonamientos que apoyan muchos principios de contabilidad.

3. **El concepto de continuidad de la empresa.** En contabilidad se asume que el negocio continuará operando por largo e indefinido tiempo. Un negocio se mira como un mecanismo que agrega valor a los recursos que utiliza y su éxito se mide por la diferencia entre el valor de lo que vende o del servicio que presta y el costo de los recursos que se usan para obtener esos ingresos. Los recursos adquiridos pero no consumidos totalmente en la obtención de los ingresos se muestran en los registros contables, no a su valor actual en el mercado, sino a su valor de costo.

4. **El costo como base de valuación.** Las cosas de valor que son propiedad de un negocio, se llaman activos. Es un concepto fundamental de la contabilidad el que los activos se registren al precio que se pagó por adquirirlos, cuyo concepto se relaciona con el de continuidad de la empresa.

   Como por diversas razones el valor real de un artículo puede variar con el transcurso del tiempo, la valorización contable de los activos no refleja necesariamente el valor actual de los activos, excepto a la fecha en que fueron adquiridos.

   El concepto contable del valor no significa que todos los activos se conserven a su precio original de compra en los registros contables, durante todo el tiempo que la empresa los posea. El valor contable de aquellos activos que tienen una larga vida, pero no obstante limitada, se reduce progresivamente hasta llegar prácticamente a cero, mediante el proceso llamado ***depreciación***. El objeto de la depreciación consiste en la eliminación gradual del ***costo*** del activo, de manera que vaya formando parte del costo de operaciones. La depreciación no necesariamente guarda relación con los cambios en el valor del mercado o con el precio real del activo.

   No obstante que algunos podrían argüir que la contabilidad debiera mostrar en todo tiempo el valor actual del negocio, hay razones poderosas para aceptar la base del costo como principio de valuación.

Una razón para valorizar los activos a su precio de compra consiste en que ésta es una base más segura y definida que el intento de estimar valores actuales del mercado.

Finalmente, si se basara en el valor del mercado para llevar sus cuentas, el contador estaría obligado a llevar también razón de los movimientos de los precios del mercado, lo que agregaría más complicaciones a su tarea.

5. **El principio de partida doble.** Las cosas de valor propiedad de una negociación son llamadas "activos". Los derechos de varios individuos o grupos sobre estos activos se llaman "participaciones". Como todos los derechos de una negociación están sujetos a los derechos de alguien y ya que el total de estos derechos no pueden exceder el monto de los activos, se deduce que:

   Activos = Participaciones

   Los sistemas contables están estructurados en tal forma que se registran los dos aspectos de cada acontecimiento que afecta a dicha igualdad y, estos aspectos son cambios en activos y cambios en participaciones.

   Consecuentemente, toda operación que registra la contabilidad afecta por lo menos a dos partes; no existe forma concebible de efectuar un cambio único en las cuentas, sobre la base del principio del doble aspecto o partida doble.

6. **El concepto de acumulación.** La esencia del concepto de acumulaciones que la utilidad neta se deriva de las operaciones que modifican la participación de los propietarios en un tiempo dado, y que esas modificaciones no son necesariamente los cambios en la situación de caja del negocio.

7. **El principio de realización.** Generalmente, el producto o ingreso se reconoce en el período contable en el cual se realiza. La realización ocurre cuando las mercancías o

servicios se suministran a los clientes a cambio de efectivo o de algún otro valor. Para los servicios el ingreso se reconoce en el periodo en el que se presta. Tratándose de productos tangibles, el ingreso no se reconoce cuando se recibe el pedido del cliente, ni cuando se firma un contrato, ni cuando las mercancías se facturan, sino cuando los artículos se embarcan o se expiden al cliente.

1.5.4 Registro de operaciones.

El elevado número de operaciones que realizan las empresas, provoca un también elevado número de aumentos y disminuciones en los diferentes conceptos que forman el balance.

Los aumentos y disminuciones de cada concepto deben ser anotados, de manera que se pueda conocer fácilmente la operación realizada, los conceptos que se afectan y la medida en que esos conceptos sufren aumentos o disminuciones.

De acuerdo a lo anterior tenemos que en contabilidad, ***cuenta*** es el registro en el que, ordenada y sistemáticamente, se lleva a cabo el relato y computo de los aumentos y disminuciones que sufre cada concepto afectado por las operaciones que realiza la empresa. Por lo tanto, deben establecerse tantas cuentas como conceptos de activo, de pasivo y capital contable integren el balance.

Para registrar correctamente en las cuentas las variaciones de los valores que representan es necesario considerar tanto la ***causa*** como el ***efecto*** que produce cada operación, ya que por sencilla que ésta sea afectará cuando menos a dos cuentas.

Las variaciones, o sea, los aumentos y disminuciones que sufran los valores de activo, pasivo y capital, por las operaciones que se efectúan en el negocio, se deben registrar en las cuentas correspondientes por medio de ***cargos*** y ***abonos***. Para esto se puede usar el concepto de mayor el cual tiene como finalidad registrar en un diagrama todas

las operaciones realizadas y dependiendo del concepto del que se trate, se registrarán en el "Debe" (lado izquierdo del diagrama) o el "Haber" (lado derecho).

Esto lo podemos ver en el siguiente esquema:

Esquema de mayor de cualquier concepto de activo

Debe	Haber
Aumenta	Disminuye

De forma inversa, tenemos:

Esquema de mayor de cualquier concepto de pasivo y capital

Debe	Haber
Disminuye	Aumenta

Al procesar o registrar cualquier tipo de operación realizada por alguna entidad, sin interesar su naturaleza, la misma cantidad colocada en el debe se anotará en el haber, lo cual constituye la ***teoría de la partida doble***.

Asi es posible clasificar cada uno de los conceptos que se presenten en cualquier balance general y estado de resultados con cuentas por lo que los esquemas de mayor controlan los aumentos y disminuciones de la cuenta que está representando. A estos movimientos realizados en el "***debe***" de cualquier esquema de mayor se denomina ***cargo*** y su contraparte, todo registro en el haber se conoce como ***abono*** y la suma de cargos de cualquier cuenta recibe el nombre de **movimiento deudor** y la suma de abonos de cualquier cuenta se denomina ***movimiento acreedor***. Según la naturaleza de las cuentas, el movimiento deudor y acreedor puede representar aumento o disminución; en virtud de ello, la diferencia entre los movimientos (deudor menos acreedor) se denomina ***saldo***.

Por lo tanto, en los estados financieros siempre se presenta el saldo de las cuentas.

1.5.5 Ecuación básica de la contabilidad.

La distribución de los elementos que integran al balance indicando la igualdad entre el activo y la suma del pasivo mas el capital contable en forma horizontal se le llama: ***balance general en forma de cuenta***.

El balance general en forma de cuenta se representa mediante la siguiente ecuación:

$$\text{Activo} = \text{Pasivo} + \text{Capital Contable}$$

de esta fórmula puede deducirse que:

$$\text{Pasivo} = \text{Activo} - \text{Capital Contable}$$

y que:

$$\text{Capital Contable} = \text{Activo} - \text{Pasivo}$$

Los elementos que integran el balance deben presentarse debidamente clasificados para facilitar su lectura y comprensión, y consecuentemente, hacer mas accesible la captación y evaluación de la situación financiera de la empresa de la que corresponde.

1.5.5.1 Definición y clasificación del activo.

En contabilidad se denomina ***activo*** al total de recursos de que dispone la empresa para llevar a cabo sus operaciones: total que se forma con las aportaciones de sus propietarios y con los recursos obtenidos en préstamo de personas ajenas a la empresa.

Los bienes y derechos de que dispone la empresa para el desarrollo de sus actividades, deben ser agrupados tomando en cuenta la finalidad inmediata para la cual fueron adquiridos o por la cual se requieren. De acuerdo con este punto de vista, identificamos tres grandes grupos dentro del activo:

Activo circulante. Grupo formado por todos aquellos recursos con los cuales la empresa lleva a cabo directamente sus operaciones principales, es decir, la compra y venta de mercancías.

El activo circulante se encuentra generalmente integrado con dinero existente en caja, dinero depositado en bancos, existencia de mercancías, cuentas por cobrar a clientes, documentos por cobrar y cuentas por cobrar a deudores diversos.

Activo fijo. Lo integran los bienes que se han adquirido no con el objeto de ser vendidos, sino para ser usados, es decir, con el fin de que presten un servicio a la empresa.

Los conceptos que forman este grupo son: los edificios propiedad de la empresa, el mobiliario y equipo de oficina, el equipo de reparto, el equipo de transporte, el importe de los depósitos de garantía, etc.

Activo diferido. Son los pagos que hace la empresa por anticipado a la percepción de un servicio, o por la adquisición de un bien material que no se consume de inmediato sino que será utilizado durante un lapso posterior.

Generalmente forman este grupo: los gastos de organización, los gastos de instalación, las rentas pagadas por anticipado, papelería y artículos de escritorio.

La característica de estos conceptos de activo es que, ya sea por el transcurso del tiempo, o por su uso o consumo, se convierten en gasto; razón por la cual se ha generalizado la expresión cargos diferidos para identificarlos.

1.5.5.2 Definición y clasificación del pasivo.

Se denomina ***pasivo*** al total de deudas contraídas por las empresas, es decir, representa las obligaciones que tiene la empresa de pagar los importes monetarios correspondientes a recursos obtenidos de personas ajenas.

Deben distinguirse dos tipos de obligaciones para la empresa:

1. Las correspondientes al pago de deudas contraídas con motivo de la percepción de un bien o un servicio a crédito. A corto plazo o a largo plazo.

2. Las derivadas de cobros efectuados por anticipado a la presentación de servicios o entrega de mercancías, que se llaman cobros anticipados.

Por lo anterior, identificamos tres grandes grupos dentro del pasivo:

Pasivo circulante. Grupo formado por las deudas que tiene que pagar la empresa a corto plazo. Generalmente se trata de deudas que contrae por compras de bienes, principalmente mercancías, o derivadas de prestamos recibidos.

Integran este grupo las deudas contraídas con los proveedores, las deudas que garantiza la empresa con letras de cambio o pagarés y que se presentan bajo el título de documentos por pagar y los acreedores diversos.

Pasivo fijo. Lo integran deudas que deben cubrirse a largo plazo. Generalmente se originan estas deudas por la adquisición de bienes inmuebles y en algunos casos por la adquisición de otros bienes del activo fijo, como maquinaria u otros equipos.

Pasivo diferido. Bajo este rubro se presentan los ingresos que recibe la empresa anticipadamente a la presentación de sus servicios; generalmente se trata de intereses y rentas cobradas por anticipado.

La característica de estos conceptos, que se presentan como pasivo por significar la obligación de prestar un servicio o de devolver la cantidad recibida por no prestarlo, es que conforme se realiza la prestación del servicio, tales cantidades se convierten en productos; por lo que se ha generalizado la expresión créditos diferidos para identificarlos.

1.5.5.3 Definición de capital contable.

Esta expresión es empleada en contabilidad para referir la suma de las aportaciones de los propietarios modificada por los resultados de operación de la empresa, es decir, es el capital social mas las utilidades o menos las pérdidas.

1.5.6 Estados financieros.

La contabilidad por una parte controla la multiplicidad de recursos de las entidades. Pero, como dicho control no tiene sentido por sí solo, la contabilidad proporciona información resultante de él, a través de documentos denominados técnicamente ***estados financieros***.

Por tanto, los ***estados financieros*** son los documentos básica y esencialmente numéricos, elaborados mediante la aplicación de la técnica contable, en los que se muestra, ya sea la situación financiera de la empresa, los resultados de su operación, u otros aspectos también de carácter financiero.

Existen diversos estados financieros, en función al tipo de información que proporcionan.

1.5.6.1 Objetivo y utilidad.

Todas las entidades por una u otra situación ven modificados sus recursos. Si compran mercancías pagándolas con efectivo, aumentan ciertos recursos, que son las

propias mercancías, pero a su vez disminuye otro recurso, el efectivo. Estas situaciones que afectan los recursos se denominan operaciones.

Todas las entidades, cualquiera que sea su actividad o giro: hospitales, centros deportivo. fábricas, bancos, sindicatos, tiendas comerciales, etc., cuentan con recursos y efectúan operaciones que en una o en otra forma los afectan.

Estos recursos como: edificios, máquinas de escribir, camionetas para repartos de mercancías, aparadores, mercancías, dinero en efectivo, etc., y todos los demás propios de cada entidad, tienen diversos orígenes, esto es, están a disposición de las entidades pero no necesariamente se encuentran pagados en su totalidad, quedando, en algunos casos, una cantidad pendiente de pagar, esto es, una ***deuda u obligación***.

Las entidades realizan así, una intensa variedad de operaciones pero todas ellas relacionadas directa o indirectamente con los recursos.

Por lo tanto. si la contabilidad es el medio para satisfacer necesidades de control e información. presentada esta, a través de ***estados financieros***, servirán para controlar las operaciones realizadas por las entidades e informar a través de un estado financiero denominado ***balance general***. Los recursos con lo que cuenta, obligaciones contraídas y la parte de los recursos que han sido adquiridos, valga la redundancia, con recursos propios. Toda esta información referida a una fecha determinada.

Por tanto. el balance general no es un estado financiero exclusivo de entidades de gran magnitud como: BANCOMER, la UNAM, etc.. sino de toda la entidad, sin importar su magnitud. actividad, o régimen legal.

De esta manera, de cualquier entidad considerando a esta como el conjunto de recursos materiales y humanos para la consecución de fines determinados se podrá formular un balance general.

El balance general es un estado financiero que informa de los recursos, obligaciones y recursos propios o patrimonio correspondiente a una entidad y a una fecha determinada.

Con esto se llega así a obtener una definición elemental del balance general que se formalizará en el siguiente punto.

1.5.6.2 Balance general.

El estado en el que se encuentra la empresa en un momento determinado respecto de la obtención de dinero y su empleo en relación con el logro del objetivo para el cual fue creada, esto es, su ***situación financiera***, puede presentarse mediante la descripción del total de recursos de que dispone y de las fuentes de que se han obtenido, precisamente, en la fecha a que se pretende mostrar esa situación financiera.

El documento contable que generalmente contiene esta información se denomina ***balance general***. Su nombre se originó en la palabra "balanza" y se debe precisamente a que muestra el equilibrio o igualdad que existe entre el total de recursos de la empresa y la suma de las deudas mas las aportaciones de sus propietarios. Figura 1.5.6.2.1

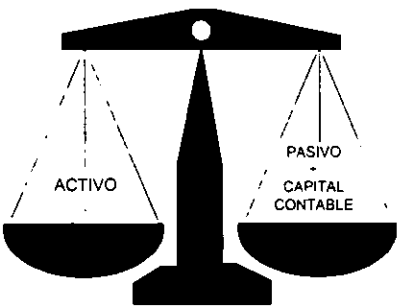


Figura 1.5.6.2.1 Balanza

Con base a lo anterior. este documento puede ser definido como:

Balance general es el estado financiero en el que se muestra la situación financiera de la empresa. mediante la descripción de su activo, su pasivo y su capital contable cuantificados a una fecha determinada.

La información que proporciona corresponde exclusivamente a una fecha determinada. ya que. la ininterrumpida realización de las operaciones de la empresa modifica constantemente la estructura y magnitud de su activo, su pasivo y su capital contable; además de que. aún cuando no realice ninguna operación, la pérdida de valor que sufre el edificio. mobiliario, y otros activos, por el simple transcurso del tiempo o por su utilización en las actividades de la empresa. también modifica su situación financiera.

1.6 SISTEMAS OPERATIVOS.

1.6.1 UNIX.

1.6.1.1 Introducción.

El sistema operativo UNIX ha evolucionado durante los últimos treinta años desde su invención como experimento informático hasta llegar a convertirse en uno de los entornos informáticos más populares e influyentes del mundo. Hoy en día más de un millón de computadoras cuyo tamaño se extiende desde las microcomputadoras con recursos limitados hasta las mini y supercomputadoras utilizan el sistema UNIX. Este crecimiento se está acelerando cada vez más conforme más y más usuarios sucumben a la sorprendente flexibilidad, potencia y elegancia del sistema UNIX.

Las siguientes son las características únicas del sistema UNIX que ha conducido a este crecimiento:

- *Herramienta de software.* El sistema operativo UNIX introdujo una nueva idea en la computación: los problemas pueden ser resueltos y las aplicaciones creadas mediante interconexión de unas cuantas piezas simples. Estas piezas son generalmente componentes completos diseñados para realizar una única tarea, y hacerla bien. Pueden construirse grandes aplicaciones a partir de secuencias de órdenes simples. Esta filosofía también se extiende al dominio del desarrollo, en donde subrutinas empaquetadas se combinan para formar nuevos programas ejecutables. Este concepto básico de reutilización del software es una de las principales razones por las que el sistema UNIX resulta ser un entorno muy productivo en el que se puede trabajar.

- *Portabilidad.* El sistema UNIX ha sido adecuado a casi cualquier computadora construida de tamaño moderado o grande. El sistema UNIX es una evolución natural para la maquinas baratas pero potentes, como es el caso de las computadoras

personales. Sólo unos cuantos cambios y adaptaciones mínimas han sido necesarios para hacer el sistema UNIX utilizable sobre las computadoras personales. El valor de la portabilidad no puede ser sobrestimado, porque el desarrollo de software es caro y tedioso. Aplicaciones importantes, tales como los procesadores de texto, las bases de datos y los sistemas gráficos, pueden llevar muchos años y esfuerzos en su desarrollo. Estas aplicaciones se diseñan generalmente para un entorno operativo especifico; si ese entorno no sobrevive a los avances de la tecnología del hardware, la inversión se pierde y los usuarios quedan abandonados. Generalmente se reconoce que el sistema UNIX proporciona el entorno adecuado para permitir el fácil traslado de aplicaciones desde microcomputadoras hasta **mainframes** (supercomputadoras), entre arquitecturas de maquinas antiguas a recientes, y especialmente entre diferentes versiones del sistema UNIX.

- *Flexibilidad.* Un atractivo importante del sistema UNIX para los creadores de hardware y software es su flexibilidad. UNIX ha sido adaptado a aplicaciones tan divergentes como la automatización de fábricas, los sistemas de conmutación telefónica y los juegos personales. Se le han ido añadiendo nuevas funciones y órdenes a paso rápido y la mayoría de los creadores manifiestan su preferencia por el sistema UNIX como "banco de trabajo" para sus aplicaciones. De hecho es tan fácil extender el sistema UNIX básico que algunos paquetes de aplicaciones son difícilmente reconocibles como que están basados en él.

- *Potencia.* El sistema UNIX es uno de los sistemas operativos más potentes disponibles sobre cualquier computadora. Su sintaxis de órdenes clara y concisa permiten a los usuarios realizar muchas cosas rápida y sencillamente, cosas que ni siquiera son posible en otros sistemas operativos. Los usuarios pueden aprovechar servicios y órdenes internos del sistema UNIX que serían herramientas adicionales caras (sí es que existen) en otros sistemas. Ningún sistema operativo es más rico en capacidades que el sistema UNIX.

- *Multiusuario y multitarea.* Debido a que el sistema UNIX es un entorno multitarea de tiempo compartido, se puede hacer más de una cosa a la vez fácilmente. En un sistema UNIX personal un usuario puede estar editando un archivo, imprimiendo otro archivo, enviando un correo a otra maquina y utilizando una hoja de cálculo electrónica simultáneamente. El sistema UNIX esta diseñado para manejar sin esfuerzo las necesidades múltiples y simultaneas de un usuario. También es un entorno multiusuario que soporta las actividades de más de una persona a la vez. Es decir que puede soportar varios usuarios a la vez y todos estos usuarios tienen la misma visión "privada" del sistema que tiene un solo usuario sobre una microcomputadora.

- *Elegancia.* El sistema UNIX esta ampliamente considerado como uno de los sistemas operativos más elegantes. Una vez que los usuarios algunos de los conceptos básicos del sistema UNIX, pueden realizar muchas y grandes tareas de un modo sencillo y hermoso. Los usuarios del sistema UNIX que se pasan a otros sistemas operativos se preguntan a menudo por qué las cosas no son ni siquiera posibles en otros entornos. Los creadores de otros sistemas operativos y otras aplicaciones toman prestadas con frecuencia ideas y temas del sistema UNIX para enriquecer los propios sistemas.

- *Orientación a red.* Las versiones modernas del sistema UNIX están organizadas para un uso de red fácil y funcional. Las herramientas de comunicación interna del sistema, la fácil aceptación de rutinas de dispositivos adicionales de bajo nivel y la organización flexible del sistema de archivos son naturales para el entorno de red, en el cual los usuarios de estaciones de trabajo personales comparten algunos recursos centralizados tales como datos o dispositivos de comunicación. Al ser cada vez más común el uso de computadoras en grupos de trabajo, las capacidades de conexión en red del sistema UNIX se vuelven más y más importantes. Las redes de hoy en día se extienden desde pequeñas redes de área local (LAN, Local Area Network) hasta enlaces a nivel mundial que permiten transferencia de datos a alta velocidad de enormes bases de datos. El sistema UNIX con su capacidad de

multitarea y su enorme base de software de comunicaciones, hace que la computación por red sea simple y fácil.

Las mejoras que ha sufrido el sistema operativo UNIX, radican principalmente en las siguientes áreas:

- *Robustez.* El sistema UNIX ha sido fortalecido de modo que se requiere muy poco mantenimiento del software para mantener el sistema ajustado y operando a su máximo rendimiento.

- *Consistencia.* Casi todas las ordenes han evolucionado a lo largo de los años hasta adoptar una sintaxis más consistente reduciendo significativamente la utilización confusa e inconsistente de diferentes órdenes.

- *Agentes de usuarios.* La mayoría de las implementaciones proporcionan ahora herramientas simplificadas (incluso sistemas expertos en algunos casos) para ayudar a la configuración y administración del sistema UNIX.

- *Nuevas características.* Muchas capacidades y órdenes nuevas se han introducido en el sistema UNIX en las últimas versiones. La mayoría de ellas han estado destinadas a incorporar algunas de las ideas de BSD (Berkeley Software Development) y XENIX a los productos AT&T del sistema V. Entre otros avances, existen nuevas características y una nueva organización del sistema para soportar conexión en red; el entorno de desarrollo ha sido mejorado; y está creciendo el soporte para procesamiento compartido en múltiples maquinas.

- *Interorperabilidad.* La gran flexibilidad del sistema UNIX se ha formalizado ahora en las interfaces binarias de aplicaciones (ABI, Applicactions Binary Interfaces) y la interfaz de programación de aplicaciones (API, Applications Programming Interface). Estas definiciones de entornos de desarrollo estándar dentro del sistema UNIX aseguran que el sistema evolucionará según vías predecibles. Por lo tanto los

programadores pueden confiar en que sus inversiones en productos de software, mantendrán su valor mientras el software del sistema UNIX (API) y las maquinas que utilizan el sistema UNIX (ABI) evolucionan en el futuro cercano. A una mayor escala, el sistema UNIX se está desplazando hacia un mayor compromiso con el estándar internacional POSIX de sistemas operativos. El estándar POSIX proporciona un conjunto mínimo de funciones que (eventualmente) serán soportadas por todos los sistemas de tipo UNIX en el mundo. Naturalmente, continuarán apareciendo innovaciones importantes. pero el estándar POSIX, el API y el ABI ayudan a proporcionar una vía segura para los usuarios y creadores de software.

- *Compartición de sistemas operativos.* Las modernas micromputadoras permiten que la maquina y los archivos sean compartidos entre el sistema UNIX y el sistema MS-DOS, o entre el entorno Apple Macintosh y el sistema UNIX. La compartición de sistemas operativos significa que las capacidades multitarea del sistema UNIX pueden ser utilizadas para ejecutar el sistema MS-DOS, como un proceso del sistema UNIX. permitiendo asi que todas las funciones "subordinadas" del sistema UNIX se ejecuten mientras están ejecutándose programas en el otro sistema operativo.

Los sistemas UNIX estándar proporcionan un sistema de archivos jerárquico con protección total, volúmenes desmontables. independencia de dispositivos y caracteristicas que facilitan la sencillez de la programación. Cualquier programa o grupo de programas puede ser ejecutado asincrónicamente de forma interactiva o subordinada sin cambio alguno.

Los sistemas UNIX no distinguen entre programas de usuario y programas del sistema, ni en capacidad ni uso excepto por las restricciones impuestas por la protección del archivo. El buffer de entrada/salida, las asignaciones de almacenamiento principal y de almacenamiento en disco son manejados automáticamente por el sistema y son invisibles al usuario.

Los sistemas operativos UNIX estándar se distribuyen con una serie de programas empaquetados que incluyen un editor de textos, un interprete de lenguaje de comandos programable, varios compiladores para lenguajes populares, un ensamblador, un editor encadenador, formateadores de documentos (con posibilidades matemáticas), programas de procesamiento de textos, un dispositivo de clasificación, una capacidad de investigación de estado, capacidad de comunicación entre usuarios, programas administrativos y de mantenimiento, bibliotecas normales del sistema y rutinas del usuario.

1.6.1.2 Objetivos del sistema operativo UNIX.

1. Conservar la sencillez del mismo y apoyarse en tan solo una cantidad mínima de funciones. A los programas de usuario se les deja la tarea de prever el uso de más procedimientos o funciones.

2. Generalidad, es decir, un solo método debe servir a diversos propósitos, éste se observa en los sistemas UNIX, en varias áreas, por ejemplo:

- El sistema usa las mismas llamadas para leer o escribir archivos, dispositivos y buffers de mensajes entre procesos.

- Se aplican los mismos mecanismos de nomenclatura, formación de seudónimos, y protección de acceso a los archivos de datos, directorios y dispositivos.

3. Crear un ambiente en el cual las grandes tareas pueden ser cumplidas combinando pequeños programas existentes. en vez de desarrollar nuevos programas.

1.6.1.3 Componentes y Funciones del Sistema Operativo UNIX (Arquitectura).

En la figura 1.6.1.3.1 se pueden ver los distintos niveles dentro de la arquitectura UNIX

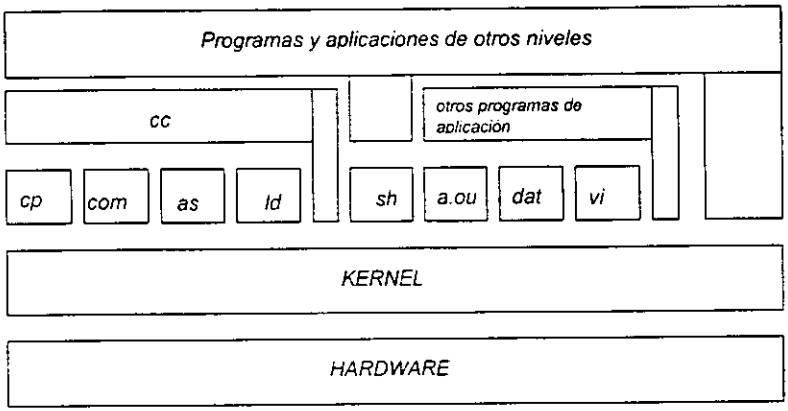


Figura 1.6.1.3.1 Arquitectura del sistema UNIX

1.6.1.4 Núcleo (kernel).

El sistema operativo UNIX se basa en un núcleo conocido como kernel que reside permanentemente en la memoria y atiende todas las llamadas del sistema, administra el acceso a los archivos y el inicio o suspensión de las tareas de los usuarios. El núcleo es un programa de aproximadamente 10.000 lineas, escrito casi en su totalidad en lenguaje C, solo la parte del manejo de interrupciones está escrito en lenguaje ensamblador del procesador en que éste opera.

Funciones del núcleo (kernel):

- Permitir la existencia de un ambiente en el que sea posible atender a varios usuarios y múltiples tareas de forma concurrente, repartiendo el procesador entre todos ellos, intentando mantener el grado óptimo de la atención individual.

- Operar como asignador de recursos para cualquier proceso que necesite hacer uso de las facilidades de los mismos.

- Creación de procesos, asignación de tiempos de atención y sincronización.

- Asignación de la atención del procesador a los procesos que los requieren.

- Administración de espacio en el sistema de archivos, que incluye:
  - Acceso. protección y administración de usuarios.
  - Comunicación entre usuarios y otros procesos.
  - Manipulación de E/S y administración de periféricos.

- Supervisión de la transmisión de datos entre la memoria central y los dispositivos periféricos.

El núcleo reside en la memoria central y tiene el control de todo sistema, por lo que ningún otro proceso lo puede interrumpir. solo lo pueden llamar para que proporcione algún servicio de los ya mencionados, conocidos como "llamadas al sistema".

El kernel de divide en:

- Despachador. Este asigna recursos. programa procesos y atiende los requerimientos de servicio.

- Manejador de Interrupciones. Supervisa la transferencia de datos entre la memoria principal y los dispositivos periféricos.

- Rutinas de apertura y cierre (sincronización). Se logra por un mecanismo llamado evento es decir, los procesos esperan a que ocurran los eventos.

Cuando se inicia la operación en la computadora se carga en la memoria una copia del núcleo que reside en el disco magnético (esta operación recibe en nombre de bootstrap). Para esto UNIX inicializa las interfaces básicas de hardware que incluyen un reloj que proporciona interrupciones periódicas. El núcleo prepara algunas estructuras de datos, que incluyen una sección de almacenamiento temporal para transferencia de información entre terminales y procesos, una sección para almacenamiento de descriptivos de archivos y una variable que indica la cantidad de memoria principal.

1.6.1.5 Manejo de memoria.

Dependiendo de la computadora en que se ejecute. UNIX utiliza dos técnicas de manejo de memoria: swapping y memoria virtual. Lo estándar en sistemas UNIX es un sistema de intercambio de segmentos de un proceso entre memoria primaria y memoria secundaria. llamado swapping, lo que significa que se debe mover una imagen de un proceso al disco si éste excede la capacidad de la memoria principal y copiar el proceso completo a memoria secundaria conforme sea necesario. Si un proceso necesita crecer, pide más memoria al sistema operativo y se le da una nueva sección. lo suficientemente grande para acomodarlo. Entonces se copia el contenido de la sección usada al área nueva. se libera la sección antigua y se actualizan las tablas de descriptores de procesos. Si no hay suficiente memoria en el momento de la expansión, el proceso se bloquea temporalmente y se le asigna espacio en la memoria secundaria. Se copia a disco y posteriormente, se devuelve a memoria principal cuando se tenga el espacio adecuado. lo cual sucede al cabo de algunos segundos.

El proceso que se encarga de los intercambios entre memoria y disco es llamado swapper. éste jamas podrá perder su posición privilegiada en la memoria principal. El

núcleo se encarga de que nadie intente interrumpir a este proceso, del cual dependen todos los demás procesos.

Cuando se decide traer a memoria principal un proceso en estado de "listo para ejecutar", se le asigna memoria y se copian allí sus segmentos. Entonces el proceso cargado compite por el procesador con todos los procesos cargados. Si no hay suficiente memoria, el proceso de intercambio examina la tabla de procesos para determinar cual puede ser interrumpido y llevado a disco.

Por otro lado cuando UNIX opera en maquinas más grandes, entonces utiliza el manejo de memoria de paginación por demanda. El tamaño de la página en UNIX es de 512 bytes en algunos sistemas y de 1.024 en otros. Para reemplazo se emplea un algoritmo que mantiene en memoria las páginas usadas más recientemente.

1.6.1.6 Manejo de Entradas y Salidas.

El manejo de entradas y salidas se divide en dos sistemas complementarios:

- Sistema de E/S estructurado en bloques. Este se emplea para el manejo de discos y cintas magnéticas. emplea bloques de tamaño fijo de 512 o 1024 bytes para leer o escribir. Su propiedad esencial consiste en que es posible leer o escribir cada bloque en forma independiente de los demás; en otras palabras el programa. en cualquier momento. puede leer o escribir en cualesquiera de los bloques. Los discos son dispositivos de bloque.

- Sistema de E/S por caracteres. Este se emplea para la atención de terminales, líneas de comunicaciones e impresoras, funciona byte por byte. Un sistema de E/S por caracteres entrega o acepta flujo de caracteres de bloque de que se trate; este no es direccionable y no tiene ninguna operación de localización.

UNIX emplea programas especiales conocidos como **drivers** (controladores) para atender a cada familia de dispositivos de E/S. Los procesos se comunican con los dispositivos mediante llamadas a su manejador.

Para los procesos, los manejadores aparecen como si fueran archivos en los que se lee y escribe, logrando con esto homogeneidad y elegancia en el diseño.

Cada dispositivo se estructura internamente mediante descriptores llamados número mayor, número menor y clase (de bloque o caracteres). Para cada clase hay un conjunto de entradas, en una tabla, que apuntan a los manejadores de los dispositivos. El número mayor se usa para asignar el manejador correspondiente a una familia de dispositivos. El número menor del dispositivo pasa al manejador como argumento y este lo emplea para tener acceso a uno de varios dispositivos físicos semejantes.

Las rutinas que el sistema emplea para ejecutar operaciones de E/S están diseñadas para eliminar las diferencias entre los dispositivos y los tipos de acceso. No existe distinción entre acceso aleatorio y secuencial, ni hay un tamaño de registro lógico impuesto por el sistema. El tamaño de un archivo ordinario está determinado por el número de bytes escritos en él: no es necesario predeterminar el tamaño de un archivo. El sistema mantiene una lista de área de almacenamiento temporal (buffers), asignados a los dispositivos de bloques. El kernel usa estos buffers con el objeto de reducir el trafico de E/S. Cuando un programa solicita una transferencia, se busca primero en los buffers internos para ver si el bloque que se requiere ya se encuentra en la memoria principal (como resultado de una operación de lectura anterior). Si es así, entonces no será necesario realizar la operación física de entrada o salida. La entrada / salida en un sistema UNIX se maneja principalmente en cinco llamadas al sistema: open, close, read, write y seek (para obtener y colocar información entre archivos y terminales, se usan otras tres llamadas gtty, stty y stat).

1.6.1.7 Manejo de archivos de información.

La estructura básica del sistema de archivos en UNIX es jerárquica, lo que significa que los archivos no están almacenados en un nivel sino en varios. Se puede tener acceso a cualquier archivo mediante su trayectoria, que especifica su posición absoluta en la jerarquía y los usuarios pueden cambiar su directorio actual a cualquier posición. Existe un mecanismo de protección para evitar los accesos no autorizados.

La distinción entre un directorio y un archivo ordinario, es que el sistema se reserva el derecho de alterar el contenido de los primeros y que el usuario sólo los puede manipular mediante los comandos mkdir y rmdir.

Los directorios contienen información para cada archivo, que consiste en su nombre y en un número que el kernel utiliza para manejar la estructura interna del sistema de archivos conocido como el nodo-i. Hay un nodo-i para cada archivo, que contiene información de su dirección en el disco, su longitud, los modos de acceso, las fechas de acceso, el autor, etc. Existe además una tabla de descriptores de archivos, que es una estructura de datos residente en el disco magnético a la que se tiene acceso mediante el sistema de E/S por bloques.

El UNIX pueden existir varios sistemas de archivos independientes y una misma unidad de disco magnético puede contener varios de ellos. Cada uno de los sistemas de archivos esta dividido internamente en cuatro secciones o particiones lógicas:

- Bloque 0. Se reserva para los procedimientos de bootstrap.

- Identificador 1. Contiene lo que se conoce como el "superbloque", que almacena un descriptor de la estructura de todos los sistemas de archivos.

- Identificador 2. Es una lista de definiciones de archivos llamada lista-i (esta es la tabla de archivos), en la que cada definición de archivo es una estructura de 64 bytes del nodo-i. El desplazamiento de un nodo-i particular dentro de la lista-i es el número-i de un archivo y actúa como su índice. La combinación del nombre del dispositivo y su número en esta lista sirven para identificar en forma única todo archivo.

- El final de cada sistema de archivos se usa para almacenar el contenido de los archivos y es la sección de mayor tamaño.

El control del espacio libre en el disco se mantiene mediante una lista ligada de bloques disponibles. Cada bloque contiene la dirección en disco del siguiente bloque en cadena. El espacio restante contiene las direcciones de grupos de bloques del disco que se encuentren libres. De esta forma, con una operación de E/S el sistema obtiene un conjunto de bloques libres y un apuntador para obtener más.

Un nodo-i contiene 13 espacios para direcciones (de 4 bytes de longitud cada uno), en los cuales se encuentra la localización de un archivo. Las primeras 10 direcciones apuntan directamente a los primeros 10 bloques del archivo. Esto es suficiente para describir un archivo de hasta 10 x 512 bytes de longitud. Si el archivo es más grande, entonces se utiliza la dirección 11, que apunta a un bloque que contiene hasta 128 direcciones de bloques adicionales.

Las operaciones de E/S en archivos se llevan a cabo con la ayuda de la correspondiente entrada del nodo-i en la tabla de archivos del sistema. El usuario normalmente desconoce los nodos-i y los números-i porque las referencias se hacen por el nombre simbólico de la trayectoria. Los procesos emplean internamente funciones primitivas (llamadas al sistema) para tener acceso a los archivos; las instrucciones más comunes son: open, creat, read, write, seek, close y unlink.

Toda estructura física se maneja "desde afuera" mediante la filosofía jerárquica de archivos y directorios, en forma totalmente transparente para el usuario. Desde el punto de vista del usuario hay tres clases de archivos:

- Ordinarios
- Directorios
- Especiales

Archivos ordinarios. Se usan para almacenar información, pueden contener un programa, el texto de un documento, los registros de una compañía, o cualquier tipo de información que desee procesar en una computadora. El sistema no presupone una estructura particular en un archivo, sino que deja a los programas de los usuarios la tarea de manejar y controlar su estructura.

Archivos directorios. Estos proporcionan la liga entre los nombres de los archivos y los archivos mismos, es decir, determinan una estructura en el sistema de archivos. En UNIX es responsabilidad de los usuarios la formación de su estructura arborecente en particular, pero el sistema es el único que puede alterar internamente el contenido de un directorio.

Archivos Especiales. Reside el medio de control sobre dispositivos de E/S. Cada dispositivo esta asociado con al menos uno de esos archivos, que se leen y se escriben como si fueran un archivo ordinario de disco, pero en realidad son solicitudes de lectura y escritura que activan el dispositivo asociado. Los archivos especiales no contienen información, sino que son utilizados para proporcionar un canal conveniente para los mecanismos de E/S. Existe un directorio dedicado a contener los archivos especiales (/dev). Para grabar información sobre una cinta magnética, por ejemplo, se escribe en el archivo /dev/mt. Existen archivos especiales para cada línea de comunicación, cada disco, cada unidad de cinta y para la memoria principal física.

Las ventajas de tratar a los dispositivos de E/S de esta forma son múltiples: un archivo y un dispositivo de E/S se vuelven similares; los nombres de los archivos y de dispositivos tienen la misma sintaxis y significado, así que a un programa que espera un nombre de archivo como parámetro puede dársele un nombre de dispositivo con lo que se logra interacción rápida y fácil entre los procesos de alto nivel. Por último, los archivos especiales están sujetos al mismo mecanismo de protección de los archivos regulares.

1.6.1.8 Modo de protección.

El modo de protección consiste en asignar a cada archivo el número único de identificación de su dueño, junto con 9 bits de protección que especifican permisos de lectura, escritura y ejecución para el propietario y para otros usuarios. Antes de cualquier acceso se verifica su validez consultando estos bits, que residen en el nodo-i de todo archivo. Además de lo anterior existen otros tres bits que se emplean para manejos especiales relacionados con la clave de supervisor (superusuario).

1.6.2 Windows 9x/Windows NT.

1.6.2.1 Windows 95.

1.6.2.1.1 Introducción.

Windows 95 esta construido sobre una base de sistema operativo que añade nuevas capacidades al sistema. Algunas características nuevas, como el nuevo sistema de archivos, ya han aparecido en otros productos de sistemas operativos de Microsoft principalmente en Windows NT y en Windows para trabajo en grupo. Windows 95 integra estas características nuevas y otras para proporcionar un entorno de modo protegido de 32 bits completo para aplicaciones Windows; y aunque se mantiene la compatibilidad con MS-DOS, no hay realmente una colección de archivos en Windows 95 a la que se puede apuntar y etiquetar como MS-DOS. Realmente, Windows 95 es, por primera vez en la historia de ésta línea de producto, un sistema operativo completo.

En el transcurso de los lanzamientos de las versiones sucesivas, Windows ha evolucionado desde su papel original como extensión gráfica de MS-DOS hasta abarcar muchas de las funciones de carga de programas. Con Windows 95, la transformación es total. Windows 95 es ahora un sistema operativo completo que incorpora compatibilidad con MS-DOS. El "modelo de aplicación única" de Windows 95 permite ejecutar MS-DOS como sistema operativo de segunda línea, para el caso en que se quiera ejecutar una aplicación que no funcione en Windows.

1.6.2.1.2 Arquitectura de Windows 95.

En la figura 1.6.2.1.2.1 se muestra un diagrama de bloques de los principales componentes de Windows 95. Veamos estos componentes con un poco más de detalle:

La máquina virtual del sistema (o sencillamente VM del sistema) es el nombre dado en Windows 95 al entorno que soporta todas las aplicaciones y los componentes del subsistema Windows, como por ejemplo, la interfaz de Dispositivo Gráfico (GDI).

Las aplicaciones Windows de 32 bits son las nuevas aplicaciones Windows que se usan en modelo de memoria de 32 bits del procesador 80386 y un subconjunto de la interfaz de programación de aplicaciones (API) Win32 de Microsoft. En Windows 95, cada una de las denominadas aplicaciones Win32 tiene un espacio de direcciones privado que es inaccesible a otras aplicaciones. Windows 95 puede planificar con derecho preferente las aplicaciones de 32 bits.

La interfaz de órdenes es una aplicación Windows de 32 bits que proporciona al sistema la interfaz de usuario esencial. La interfaz de órdenes en Windows 95 concentra las funciones de las utilidades del Administrador de Programas, Administrador de Archivos y Administrador de Tareas de Windows 3.1 en una única aplicación.

Las aplicaciones Windows de 16 bits son las aplicaciones Windows "antiguas". Estas aplicaciones usan un modelo de memoria segmentada de la familia de procesadores Intel (en realidad, el modelo de memoria de un 80286). Como en Windows 3.1, las aplicaciones de 16 bits que se ejecutan en Windows 95 comparten un espacio de direcciones único y no pueden ser planificadas con derecho preferente. Microsoft a éstas aplicaciones las denomina "aplicaciones Win16"

La capa de interfaz de programación de aplicaciones proporciona en Windows 95 una compatibilidad completa con la API de Windows 3.1, así como el soporte para la nueva API de 32 bits accesible sólo para aplicaciones Windows de 32 bits. La API de 32 bits es un subconjunto de la completa API Win32 de Microsoft vista por primera vez en Windows NT y en la Win32 añadida a Windows 3.1

El Núcleo de Windows proporciona soporte a los servicios del nivel más bajo que requieren las aplicaciones Windows, tales como la asignación dinámica de memoria.

Para Windows 95, el Núcleo proporciona estos servicios a las aplicaciones de 16 y 32 bits.

La GDI es el corazón de las capacidades gráficas de Windows, gestionando los tipos de letras, las primitivas de dibujo y el color tanto para los dispositivos de visualización como para los de impresión. Aunque la GDI en Windows continua dando soporte a las aplicaciones de 16 bits existentes. incluye nuevas utilidades importantes disponibles sólo para los programas de 32 bits.

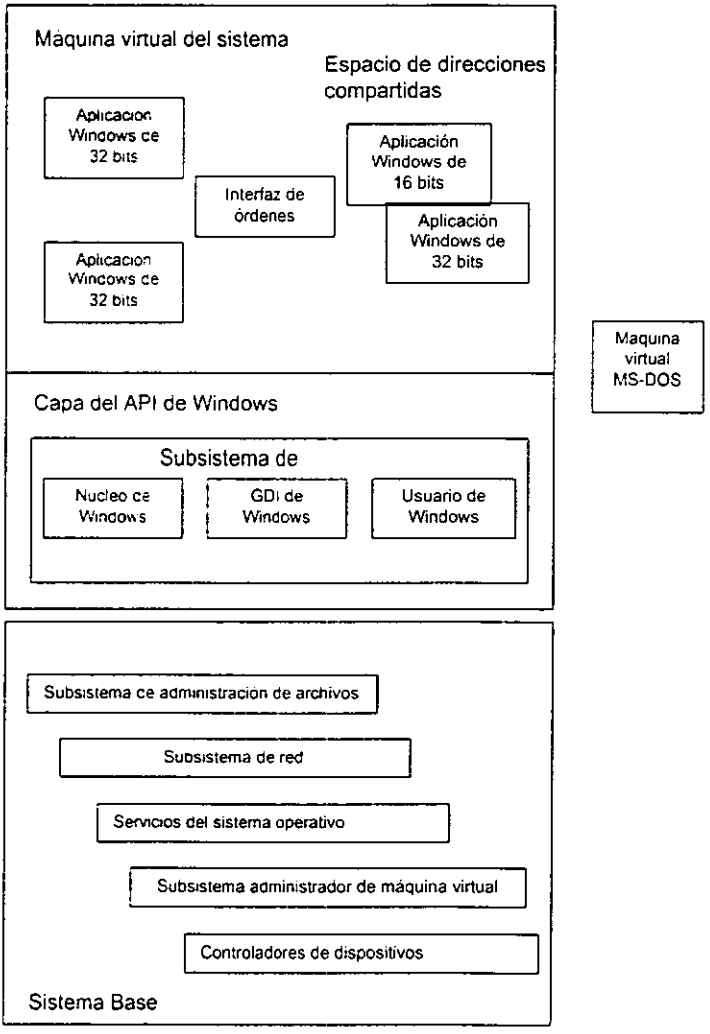


Figura 1.6.2.1.2.1 Componentes del sistema Windows

Usuario es como se denomina al administrador de ventanas, el componente de Windows 95 que dirige la creación y manipulación de ventanas, diálogos, botones y otros elementos de la interfaz de Windows.

Las maquinas virtuales MS-DOS dan soporte a la ejecución de aplicaciones MS-DOS en Windows. Como en Windows 3.1 el usuario puede ejecutar múltiples VM MS-DOS concurrentemente. Windows 95 incluye varias utilidades nuevas diseñadas para mejorar la gestión por parte del usuario de esas VM, pero el diseño básico de la VM MS-DOS no ha cambiado mucho.

1.6.2.1.3 El sistema base.

Los módulos restantes implementan diversos aspectos del sistema operativo subyacente en Windows 95. Normalmente, al grupo de estos componentes se le denomina "sistema base".

La administración de archivos ha cambiado de forma marcada en Windows 95. En Windows 3.1 es MS-DOS quien controla el sistema de archivos local del disco duro. Este control perjudicaba las prestaciones de Windows y la oportunidad de mejorar el soporte de archivos resultaba imposible mientras MS-DOS siguiera con el control. Bajo Windows 95 la situación es totalmente diferente, sobre todo ya no se utiliza MS-DOS para la administración de archivos en los discos locales. El nuevo sistema de gestión de archivos proporciona varias interfaces que admiten la coexistencia de los sistemas de archivos de todos los discos locales (incluido el sistema de archivos del CD ROM) y los múltiples sistemas de archivo de red.

El subsistema de red es la encarnación más reciente de la red igualitaria de Microsoft, vista por primera vez en 1992 en el producto Windows para trabajo en grupo y después en Windows NT. El subsistema de red utiliza un nuevo subsistema de administración de archivos para coordinar su acceso a los archivos remotos. Otros distribuidores de red

también pueden conectar sus productos a los nuevos servicios de administración de archivos, permitiendo que el usuario acceda simultáneamente a más de un tipo de red. Windows incorpora soporte para los protocolos SMB, Novell y TCP/IP.

Los servicios del sistema operativo incluyen en Windows 95 componentes importantes como el subsistema de configuración de hardware "plug and play" (conectar y listo), además de un grupo de funciones diversas entre las que se incluyen aquellas que satisfacen solicitudes de fecha y hora.

El administrador de maquina virtual es el corazón del sistema operativo Windows 95. Incluye el software que implementa todas las primitivas básicas del sistema para la planificación de tareas. operaciones de memoria virtual, carga y finalización de programas y comunicación entre tareas.

Los controladores de dispositivos pueden tener formas diferentes en Windows 95, entre otros controladores de modo real y los denominados controladores virtuales o VxD. Algunos sistemas aún pueden necesitar el uso de los antiguos controladores de dispositivos de MS-DOS en modo real para dar soporte a dispositivos de hardware concretos, pero una de las metas del desarrollo de Windows 95 ha sido desarrollar controladores de modo protegido para tantos dispositivos populares como fuera posible, incluyendo nuevos controladores de modo protegido para el mouse (ratón). los dispositivos CD ROM y muchos dispositivos de disco duro.

Los controladores de dispositivos virtuales o VxD, asumen el papel de compartición de un único dispositivo de hardware entre diversas aplicaciones. Por ejemplo, la ejecución de dos aplicaciones MS-DOS en distintas ventanas de la pantalla necesita que el sistema cree dos VM MS-DOS. cada una de las cuales requiere el acceso a una única pantalla física. El VxD del controlador de la pantalla tiene que aceptar estos requisitos de compartición. "VxD" también se usa como descriptor general para otros módulos del sistema operativo de 32 bits.

1.6.2.1.4 Beneficios.

- Para todos los intentos y propósitos, se elimino el modo real de MS-DOS. Finalmente Windows es un sistema operativo completo que no depende de MS-DOS y de su arquitectura en modo real ni de sus limitaciones.

- Una arquitectura nueva del sistema de archivos y la implementación en modo protegido de 32 bits del sistema de archivos FAT elimina la principal dependencia que tenía Windows con MS-DOS. Además el nuevo sistema de archivos proporciona mejoras significativas en las presentaciones del sistema.

- Windows proporciona soporte completo para aplicaciones de 32 bits, entre las que incluyen la API de 32 bits de Windows y los espacios de direcciones protegidos y privados.

- Windows 95 proporciona planificación con derecho preferente de todas las aplicaciones Windows.

- Windows 95 proporciona una arquitectura que puede soportar simultáneamente múltiples conexiones a red.

- Windows 95 soporta el manejo de nombre largos para los archivos de usuario, rompiendo con la limitante del manejo de nombres de archivo de no mayor a 8 caracteres.

- Incorporación de la facilidad "plug and play" (conectar y listo), lo que soluciono el problema de configuración del sistema, cuando se le incorporaba un nuevo dispositivo o tarjeta a la computadora.

1.6.2.2 Windows 98.

1.6.2.2.1 Algunas de sus características.

Windows 98 es la versión más reciente del sistema operativo de Microsoft. Windows 98 está diseñado para satisfacer las necesidades de los clientes y permite que las PCs trabajen mejor y diviertan más. Construido sobre las herramientas de vanguardia de Windows 95. Windows 98 mejora su desempeño, la confiabilidad, el acceso a Internet y es más fácil de usar. además, abre paso a una excitante gama de *hardware* y de juegos nuevos.

Windows 98 de Microsoft mejora diferentes aspectos para evitar que los usuarios pierdan tiempo en sus PCs, incluyendo las aplicaciones para abrir programas (las cuales son 36 por ciento más rápidas), la transmisión de páginas de Internet (que es 25 por ciento más rápida) y el apagado de la PC (que es de dos a cinco veces más rápido que Windows 95). Además, Windows 98 puede brindar a los usuarios 28 por ciento más de memoria (en promedio), gracias a un sistema más eficiente de almacenamiento de datos en el disco duro.

Las características claves y de confiabilidad incluyen:

- Reducción del número de pasos necesarios para la instalación del equipo, proporcionando una instalación más rápida y sencilla.

- Las aplicaciones se cargan más rápido, al identificar las que son más frecuentes y colocando en el mismo sitio los archivos necesarios para su inicio. para que el disco duro pueda cargarlos rápidamente.

- La salida del sistema es mucho más rápida, reduce dramáticamente el tiempo que se necesita para apagar la computadora.

- Tiene más espacio en el disco, gracias a la facilidad que tiene de almacenar información más eficientemente con la nueva herramienta FAT32.

- El asistente de mantenimiento calendariza y realiza automáticamente revisiones internas para que su PC siempre esté funcionando en las mejores condiciones.

Acceso y uso más fácil de Internet.

Windows 98 se desarrollo gracias a la retroalimentación recibida de los usuarios, con el objetivo de proporcionarles un sistema operativo más simple y fácil de utilizar. Este enfoque, combinado con el crecimiento tan rápido del uso de Internet en los últimos tres años, ha dado como resultado un sistema operativo diseñado para el usuario y su uso de Internet. Windows 98 integra tanto los paradigmas de exploración del web, como la naturaleza gráfica de HTML, para brindar a los usuarios una experiencia más enriquecedora y sencilla. Windows 98 simplifica la exploración por Internet para que los usuarios puedan aprovecharlo al máximo; haciendo que encuentren la información más fácilmente –sin importar dónde esté ubicada-, obtengan ayuda, mantengan los sistemas actualizados e instalen nuevo *hardware*.

Los beneficios clave para integrar la optimización del acceso y uso de Internet incluyen los siguientes:

- Paradigma de exploración consistente, el cual brinda a los usuarios una manera muy fácil de explorar y de encontrar información, sin importar dónde radique, ya sea dentro del disco duro o en Internet.

- La información más enriquecedora se envía a través del HTML y de HTML dinámico de Windows 98. Por ejemplo, los usuarios, al ir a la opción Mi Computadora y hacer clic en la carpeta de Panel de control, podrán tomar algún texto que describe lo que contiene dicha carpeta.

- Las funciones mejoradas de Internet Explorer incluyen el manejo más fácil y la personalización de innovaciones, tales como la barras de explorador y el énfasis en la seguridad.

- Mejorías en la herramienta para conexión telefónica a redes. la cual brinda una conexión a Internet más rápida, así como una conexión automatizada con los servicios en línea.

- La ayuda directa de HTML, junto con los asistentes para resolución de problemas y la ayuda del web, soluciona los problemas con mayor facilidad y simplifica dramáticamente la búsqueda en línea.

Las características del nuevo *hardware* y de entretenimiento incluyen lo siguiente:

- Apoyo de *hardware* superior, el cual permite a los usuarios aprovechar las innovaciones más recientes y estándares del *hardware* de las PCs. como las herramientas de USB.

- El soporte de monitores múltiples puede ser de gran utilidad en varias áreas para los usuarios, incluyendo la publicación, el desarrollo de sitios web, o la edición de videos y juegos.

- DirectX 5.0, incluido en el sistema operativo, brinda una mejor experiencia de audio y vídeo. la cual enriquece los juegos y los contenidos multimedia.

- La tecnología de apoyo de Intel MMX brinda mayor rapidez al audio y al vídeo.

- WebTV para Windows brinda la capacidad de ver programas de televisión en la PC, normales o mejorados, con una tarjeta de canales adicional. También proporciona una guía de búsqueda de la programación para saber qué programas televisivos están pasando y cuáles pasarán próximamente.

1.6.2.3 Windows NT.

1.6.2.3.1 Introducción.

Es la base de un sistema operativo de multiproceso simétrico que soporta múltiples entornos de sistemas operativos. Posee una interface gráfica de usuario (GUI) de Windows y ejecuta programas basados en Win32, Windows de 16 bits, MS-DOS (Microsoft Disk Operation System), POSIX y OS/2. Emplea principios avanzados de sistemas operativos, como son el manejo de la memoria virtual, la multitarea real, el tratamiento estructurado de las excepciones, y la incorporación del concepto de objetos del sistema operativo. Es seguro, potente, fiable y flexible. Posee un tipo de posibilidades que solo se encuentran en sistemas operativos de **mainframes** (supercomputadoras) y minicomputadoras.

El diseño de Windows NT se guió por una combinación de varios modelos. Windows NT emplea un modelo "cliente/servidor" para proporcionar múltiples entornos de sistemas operativos (inicialmente Windows, MS-DOS, OS/2 y POSIX) a sus usuarios. Emplea un modelo de "objetos" para manejar uniformemente los recursos del sistema operativo y concedérselos después a los usuarios. Un tercer modelo, el "multiproceso simétrico" (SMP. symmetric multiprocessing). permite que Windows NT consiga el máximo rendimiento en entornos con multiprocesadores

1.6.2.3.2 Estructura de Windows NT.

La estructura de Windows NT se divide en dos partes: la porción de modo usuario del sistema (los subsistemas protegidos de Windows NT o servidores) y la porción del modo kernel (el ejecutor de NT). En la figura 1.6.2.3.2.1 puede verse detallada la estructura de Windows NT.

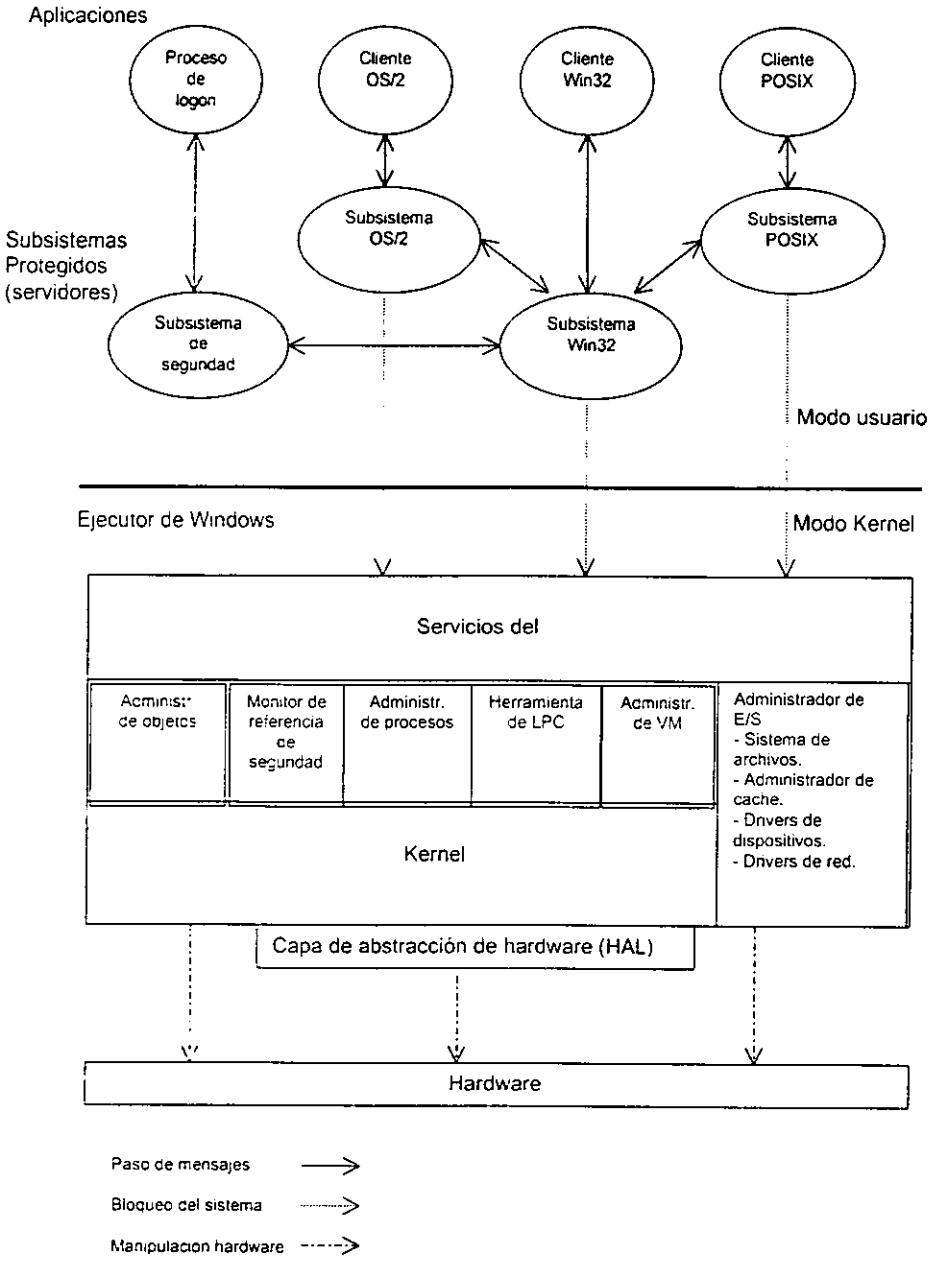


Figura 1.6.2.3.2.1 Estructura de Windows NT.

Los servidores de Windows NT se denominan subsistemas protegidos porque cada uno reside en un proceso independiente cuya memoria está protegida de otros procesos por el sistema de memoria virtual del kernel de NT. Puesto que los subsistemas no comparten memoria automáticamente, se comunican por medio del pase de mensajes. Las líneas continuas de la figura 1.6.2.3.2.1 representan los caminos que pueden tomar los mensajes que van entre clientes y servidores o entre dos servidores. Todos los mensajes pasan a través del kernel, pero para simplificar, esos caminos no se han simplificado en la figura.

El kernel de NT es como una maquina del sistema operativo capaz de soportar cualquier número de procesos de servidor. Los servidores dan al kernel de NT sus interfaces de usuario y de programación; y proporcionan entornos de ejecución para diversos tipos de aplicaciones.

1.6.2.3.3 Ejecutor.

El kernel de Windows NT consiste en una serie de componentes, cada uno de los cuales implementa dos conjuntos de funciones: servicios del sistema, que pueden ser llamados por subsistemas de entorno y otros componentes del kernel y funciones o rutinas internas, disponibles solamente para componentes del ejecutor. Las interfaces se muestran en la figura 1.6.2.3.3.1.

Los componentes del kernel mantienen independencia entre ellos, cada uno crea y manipula las estructuras de datos del sistema que necesita. Al estar las interfaces entre componentes cuidadosamente controlados. es posible eliminar completamente un componente del sistema operativo y reemplazarlo por otro que funcione de manera diferente. Mientras la nueva versión implemente todos los servicios del sistema e interfaces internas correctamente, el sistema operativo se ejecutará como antes. El mantenimiento del sistema operativo en consecuencia es una tarea sencilla, porque los componentes del kernel de NT interaccionan de una forma predecible.

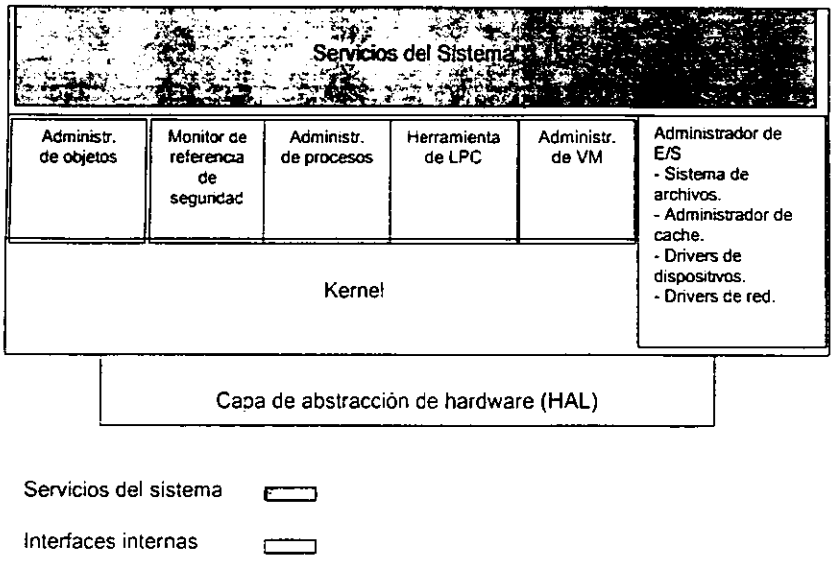


Figura 1.6.2.3.3.1 Interfaces del sistema.

Las responsabilidades de los componentes del kernel son:

- Administrador de objetos. Crea, administra y elimina objetos del kernel de Windows NT, y realiza el sumario de los tipos de datos que son utilizados para representar los recursos del sistema operativo.

- Monitor de referencia de seguridad. Refuerza las normas de seguridad en la computadora local. Resguarda los recursos del sistema operativo, ejecutando la protección y auditoría de los objetos en tiempo de ejecución.

- Administración de procesos. Crea y finaliza procesos y tareas (hilos). También suspende y continúa la ejecución de tareas; y almacena y recupera información acerca de los procesos y tareas de Windows NT.

- Mecanismos de llamadas de procedimiento local (LPC). Pasa mensajes entre un proceso cliente y un proceso servidor dentro de la misma computadora. LPC es una

versión flexible y optimizada de la llamada de procedimiento remoto (RPC, remote procedure call). una utilidad estándar de comunicaciones para procesos clientes y servidores a través de una red.

- Administrador de memoria virtual (VM, memory virtual). Implementa la memoria virtual. que es un esquema de gestión de memoria que proporciona un espacio de direcciones de un proceso. Cuando la utilización de la memoria es demasiado grande. el administrador de VM transfiere los contenidos de memoria seleccionada al disco y los recarga cuando se vuelven a utilizar; una practica conocida como paginación.

- Kernel. Responde a las interrupciones y excepciones. planifica la ejecución de tareas. sincroniza las actividades de los procesadores múltiples y proporciona un conjunto de objetos e interfaces de bajo nivel que los emplea el resto del kernel de NT para implementar objetos de más alto nivel.

- Sistema E/S. Lo forma un grupo de componentes responsables de procesar las entradas y ejecutar las salidas a la gran variedad de dispositivos. El sistema de E/S incluye los siguientes subcomponentes:

    - Administrador de E/S. Implementa los mecanismos entrada/salida independientes de dispositivos y establece un modelo para la E/S del kernel de NT.

    - Sistema de archivos. Controladores (drivers) de Windows NT que aceptan peticiones de E/S orientadas a archivos y las traducen en peticiones de E/S vinculadas a un dispositivo determinado.

    - Redirector de red y servidor de red. El redirector de red está formado por un conjunto de controladores (drivers) de sistemas de archivos que transmiten

peticiones de E/S a una máquina de la red, mientras que el servidor de red es el que recibe dichas peticiones.

- Controladores de dispositivos del kernel de Windows NT. Drivers de bajo nivel que manipulan directamente el hardware para escribir la salida o recuperar la entrada de un dispositivo físico o de la red.

- Administrador de cache. Mejora el funcionamiento de la E/S basada en archivos, almacenando la información más reciente leída del disco en la memoria del sistema. Emplea la utilidad de paginado del administrador de VM para escribir automáticamente en disco las modificaciones (trabaja en segundo plano o background).

- Capa de abstracción del hardware (HAL). Sitúa una capa de código entre el ejecutor y la plataforma hardware en la cual se está ejecutando Windows NT. Oculta detalles dependientes del hardware como las interfaces de E/S, controladores de interrupciones y mecanismos de multiprocesadores. En lugar de acceder al hardware directamente, los componentes del ejecutor de Windows NT mantienen el máximo de portabilidad llamando a las rutinas del HAL cuando necesitan información dependiente de la plataforma.

Windows NT es un sistema operativo transportable, diseñado para limitar al máximo la cantidad de código que dependa de una arquitectura de hardware determinada. Se necesita alguna cantidad de código dependiente del procesador (por ejemplo el Intel 80486 o el MIPS R40000), que ésta situada en las capas más bajas del kernel y en algunas secciones pequeñas del administrador de VM. Estos componentes, particularmente el kernel de Windows NT, ocultan las diferencias entre procesadores al resto del sistema operativo.

El código dependiente de la plataforma (por ejemplo el que necesita una implementación particular de una computadora basado en un MIPS R4000) está

situado en el HAL, y lo suministran los distintos fabricantes. Los controladores de dispositivos también contienen un código especifico que depende de la maquina, desde luego, pero evitan el código dependiente de la plataforma y del procesador, invocando a rutinas del kernel y del HAL.

1.7 COMUNICACIONES.

Introducción.

A finales de los años 60 y principios de los 70, las redes no estaban diseñadas de forma que fuera posible compartir recursos entre redes diferentes. Por su parte, los administradores de las redes eran reacios a permitir que los usuarios invadieran sus recursos por motivos de seguridad. Además, experimentaban una utilización excesiva de los recursos de las redes. Como resultado, era difícil que los usuarios extendieran el uso de sus sistemas de información a otros usuarios en redes diferentes. O bien las redes eran incompatibles entre sí, o no se podían comunicar debido a problemas de administración.

Desde entonces se ha hecho cada vez más patente la necesidad de que las aplicaciones de usuario compartan recursos. Pero para que puedan hacerlo, los administradores de las redes deben acordar primero un conjunto de tecnologías y normas comunes para que las redes puedan comunicarse. De ahí se sigue que las aplicaciones como la transferencia de archivos y el correo electrónico se deberían extandarizar también para permitir la interconexión entre aplicaciones de usuario. El Protocolo de Control de Transmisión y el Protocolo de Internet (TCP/IP) se desarrollaron con esos objetivos.

1.7.1 Redes de computadoras.

El término red por lo general significa un conjunto de computadoras y periféricos, que están conectados por algún medio, esta conexión puede ser directa (por medio de un cable) o indirecta (a través de un módem). Los diferentes dispositivos de la red se comunican entre sí por medio de una serie de reglas predefinidas (protocolos). La estructura de la red, es decir los dispositivos reales y la forma en como están conectados se llama topología de la red.

Normalmente las microcomputadoras necesitan distintos recursos periféricos como son: impresoras, discos duros, programas de aplicación, paquetería, etc., que se tienen que adquirir a costos adicionales.

En una red estos recursos en una sola micro se van a compartir con las demás, mediante un canal de comunicación. Las micros se conectan a este canal por medio de una interface, que es una tarjeta electrónica que se coloca en una de las ranuras de expansión de cada micro.

La microcomputadora que cuenta con los recursos periféricos recibe el nombre de servidor (SERVER) de la RED, que auxiliado por el sistema operativo de la RED viene a ser virtualmente el "cerebro" dedicado a administrar los recursos y las comunicaciones entre las demás micros, mismas que trabajando así, reciben el nombre de ***estaciones de trabajo***.

Si los dispositivos de una red están en una sola ubicación, como un edificio o grupo de habitaciones, se les llama red de área local LAN (***Local Area Network***). Las LAN por lo general tienen todos los dispositivos en la red conectados por un solo tipo de cable de red. Si los dispositivos están dispersos en forma amplia, como en edificios diferentes o ciudades diferentes, se establecen en varias LAN, unidas por una estructura más grande llamada red de área amplia o WAN (***Wide Area Network***). Una WAN está compuesta por dos o más LAN. Cada LAN tiene su propio cable de red que conecta todos los dispositivos de esta. Las LAN se unen por otro método de conexión, a menudo las líneas telefónicas de alta velocidad o cables de red muy rápidos, llamados columnas vertebrales.

1.7.2 Modelos de interconexión.

Redes de área local.

TCP/IP funciona a través de las LAN y las WAN. Aunque existen muchas topologías para las LAN, tres son las dominantes: bus, anillo y eje.

La red bus.

Esta red es la más sencilla; comprende una sola ruta de comunicaciones principal en la que cada dispositivo está conectado al cable principal (bus) por medio de un dispositivo llamado traceptor o caja de empalme, al bus también se le llama columna vertebral. Desde cada traceptor en el bus, otro cable, corre hasta el adaptador de red del dispositivo. En la figura 1.7.2.1 se muestra un ejemplo de una red de bus.

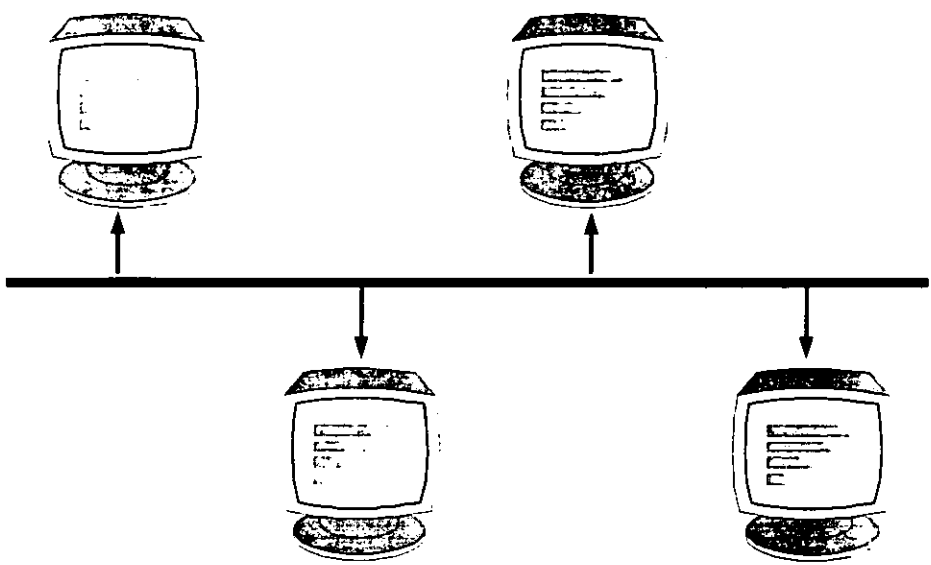


Figura 1.7.2.1 Topología de bus

La ventaja principal de bus es que permite un bus de alta velocidad además de que es inmune a problemas con cualquier tarjeta de red sola, dentro de un dispositivo en la red. Esto se debe a que el traceptor permite tráfico a través de la columna vertebral, ya sea que un dispositivo está conectado a la caja de empalme o no. Cada extremo del bus termina en un bloque de resistores u otro dispositivo eléctrico semejante que marque el final del cable desde el punto de vista eléctrico. Cada dispositivo en la ruta tiene un número de identificación especial, o dirección, que le permite al dispositivo saber cuál información recibida es para él.

Una red de bus por lo general se alambra siguiendo los contornos de las paredes y edificios conforme se necesita. La mayor parte de los dispositivos en la red de bus pueden enviar o recibir datos a lo largo del bus, empaquetando un mensaje con la dirección del receptor proyectado.

Una variación de la topología de la red de bus que se encuentra en muchas LAN pequeñas es que utilizan un cable Thin Ethernet o cable de par trenzado. A diferencia de la red de bus no hay transceptores en el bus, en lugar de ellos cada dispositivo está conectado de manera directa en el bus, usando un conector en T en la tarjeta de interfaz de la red, a menudo con un conductor llamado BNC. El BNC conecta a la máquina con los dos vecinos, en cada extremo de la red se añade un resistor sencillo del lado desocupado del lado del conector para terminar la red desde el punto de vista eléctrico.

La red de anillo.

En esta conexión, la información viaja ordenadamente en un solo sentido a través de un solo cable, describiendo una trayectoria de 360° en cuyo anillo imaginario, están conectadas en serie las estaciones de trabajo y el SERVER.

Una señal llamada TOKEN (receptáculo, a modo de estafeta) va circulando por la red y pasando por cada estación, si la primera resultó ser la solicitante, previa identificación

entrega la información, de lo contrario la deposita en "sobre cerrado" para que ésta a su vez la envíe a la siguiente, llevando la consigna de entregarla hasta identificar a la solicitante.

Cada estación de paso, cuando más colecta información adicional enviándola a la siguiente y así se le pasa la señal cerrando ciclos "circulares" por ello el protocolo apropiado para este caso se conoce como TOKEN PASSING.

En la figura 1.7.2.2 observamos una configuración de anillo, en donde las computadoras se conectan entre sí, como su nombre lo indica. Los mensajes que mandan alrededor del anillo deben estar específicamente dirigidos al nodo destino, debido a que las computadoras conectadas tienen acceso al mensaje.

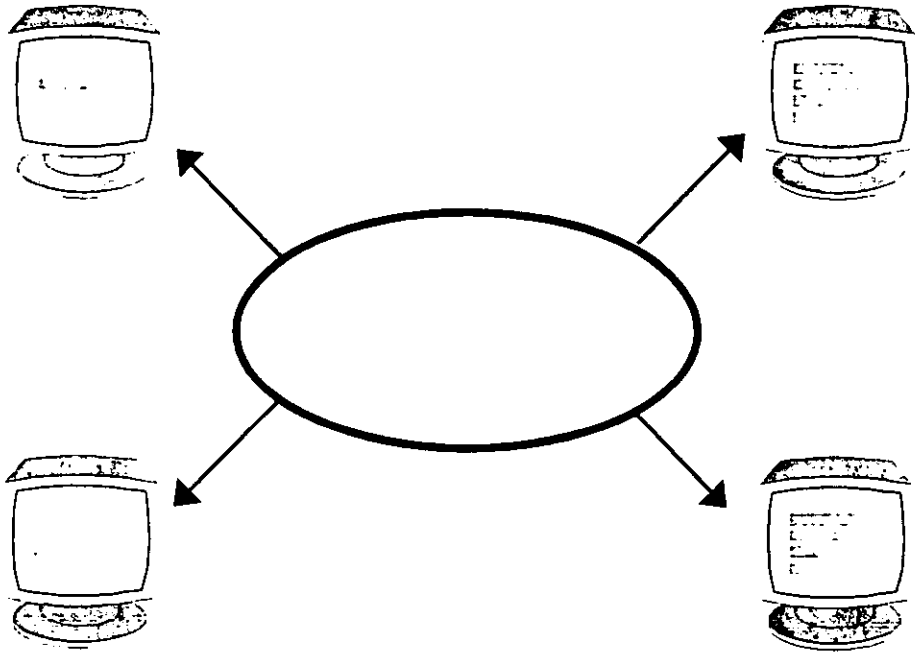


Figura 1.7.2.2 Topología de anillo

Red de estrella.

En este tipo de conexión, el elemento central es el Servidor con sus periféricos, como se puede apreciar en la figura 1.7.2.3. se mantiene preguntando constantemente a cada estación de trabajo mediante comunicación exclusiva y por turno si desea transmitir información, y en caso afirmativo, la atiende y al terminar prosigue con su interrogatorio permanente.

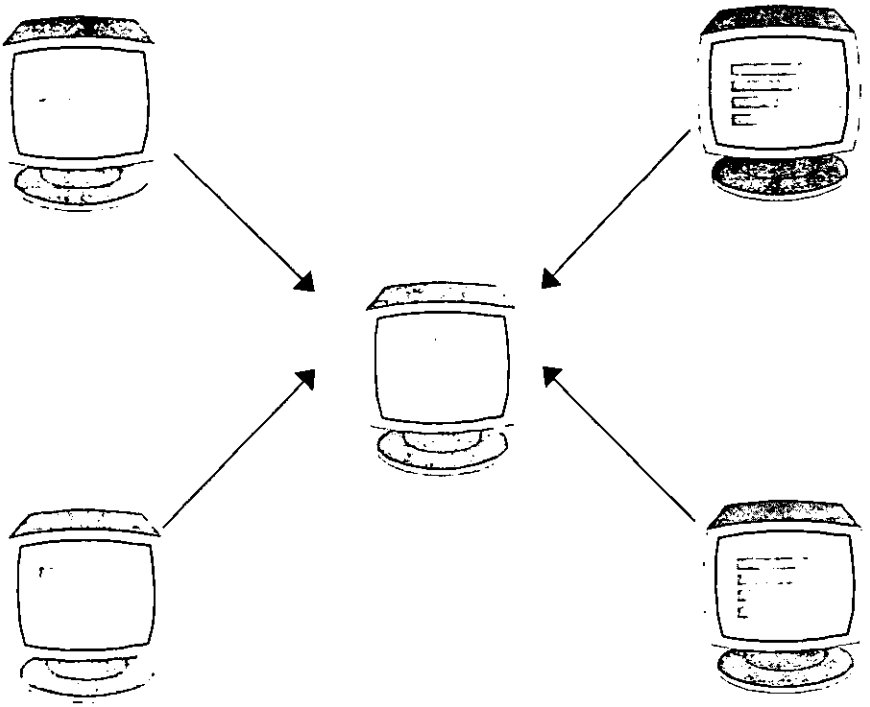


Figura 1.7.2.3 Topología de estrella

Esta topología es similar a la red de teléfonos, en donde existe un conmutador y cada llamada que se hace tiene que pasar por él para poder llegar a su destino.

No existe un número máximo de conexiones debido a que los servidores son cada vez más poderosos y soportan mayor número de dispositivos con un nivel de servicio muy alto. En general, el número máximo de estaciones que se pueden conectar al servidor depende del tráfico que se genere entre ellas, y cuando este es excesivo la red se divide mediante un dispositivo adicional cuya función es aislar el tráfico de un segmento a otro.

La gran desventaja de ésta topología es depender en su totalidad del nodo central por lo que al fallar éste, falla toda la red.

Red de malla.

Las redes tipo malla tienen la siguiente característica: todos sus dispositivos están conectados entre sí.

El número de conexiones punto a punto en una red de malla depende del número de dispositivos que integren la red. Tres dispositivos que integren una red necesitan tres conexiones punto a punto, cuatro dispositivos que integren una red de malla necesitan seis conexiones punto a punto.

Cada dispositivo está comunicado con todos los demás dispositivos. Desde un dispositivo en una red de malla existen múltiples rutas de acceso hacia otros dispositivos.

Si la ruta de acceso directo entre dos dispositivos no está disponible, el mensaje puede ser ruteado a través de los otros dispositivos.

Algunas redes híbridas utilizan red de malla entre su estructura. La figura 1.7.2.4 muestra un esquema típico de una red de malla.

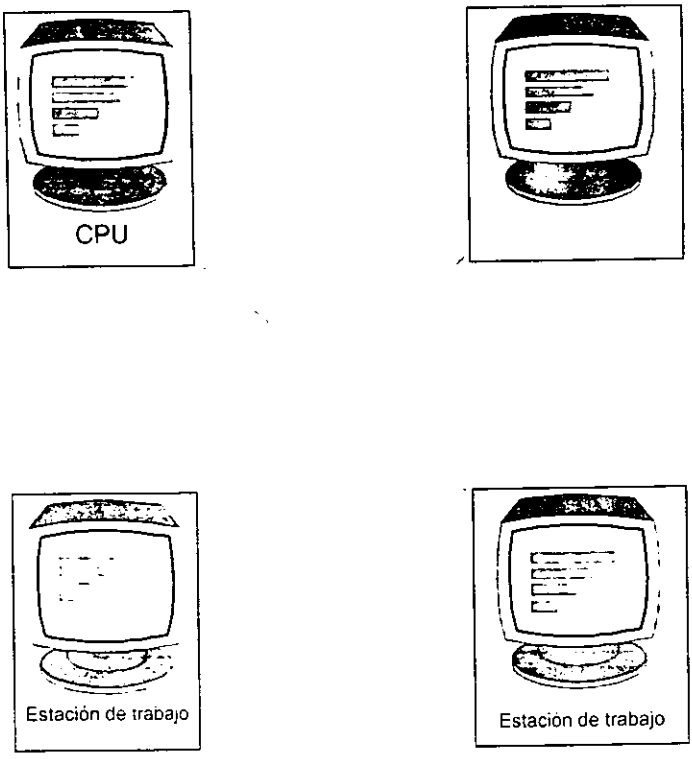


Figura 1.7.2.4 Topología de una red de malla

Redes de área amplia.

Como se mencionó antes. las LAN pueden combinarse en una entidad grande llamada WAN. Las WAN por lo general están compuestas por LAN unidas por una conexión de alta velocidad. En la entrada de cada LAN. una o más máquinas actúan como enlace entre LAN y una WAN: estos se llaman gateways (equipos de puerta de enlace). Un gateway es la interfaz entre una LAN y una WAN.

Las LAN pueden estar unidas a una WAN por medio de un gateway que maneja el paso de datos entre la columna vertebral de la LAN y la WAN. Otro dispositivo gateway. llamado puente, se usa para conectar las LAN usando el mismo protocolo de red. Los puentes se usan sólo cuando el mismo protocolo de red está en ambas LAN.

1.7.3 Modelo OSI.

El modelo OSI se muestra en la figura 1.7.3.1. Este modelo se basa en una propuesta que desarrolló la Organización Internacional de Normas (ISO, por sus siglas en inglés) como primer paso hacia la estandarización internacional de los protocolos que se usan en las diferentes capas.

El modelo se llama modelo de referencia OSI (***Open Systems Interconnetion,* Interconexión de Sistemas Abiertos**) de la ISO puesto que se ocupa de la conexión de sistemas abiertos. esto es, sistemas que están abiertos a la comunicación con otros sistemas.

El modelo OSI tiene siete capas. Los principios que se aplicaron para llegar a las siete capas son los siguientes:

1. Se debe crear una capa siempre que se necesite un nivel diferente de abstracción.

2. Cada capa debe realizar una función bien definida.

3. La función de cada capa se debe elegir pensando en la definición de protocolos estandarizados internacionalmente.

4. Los límites de las capas deben elegirse a modo de minimizar el flujo de información a través de las interfaces.

5. La cantidad de capas debe ser suficiente para no tener que agrupar funciones distintas en la misma capa y lo bastante pequeña para que la arquitectura no se vuelva inmanejable.

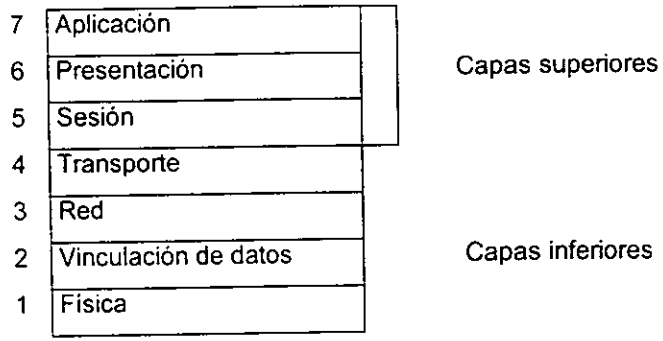


Figura 1.7.3.1 El modelo de referencia OSI

Las capas de aplicación, presentación y sesión están orientadas hacia la aplicación, puesto que son responsables de presentar la interfaz de la aplicación al usuario. Las tres son independientes de las capas de abajo de ellas y olvidan los medios por los cuales llegan los datos a la aplicación. Estas tres capas se llaman capas superiores. Las cuatro capas inferiores tienen que ver con la transmisión de los datos, que abarca el empaquetamiento, enrutamiento, verificación y transmisión de cada grupo de datos. Las capas inferiores no se preocupan por el tipo de datos que reciben o envían a la aplicación, sino que sólo tienen que ver con la tarea de enviarlos.

Capa de aplicación.

La capa de aplicación es la interfaz de usuario final para el sistema OSI. Es donde residen las aplicaciones como el correo electrónico, los lectores de noticias de Usenet o los módulos de desplegado de bases de datos. La tarea de la capa de aplicación es desplegar la información recibida y enviar los datos nuevos del usuario a las capas inferiores.

Capa de presentación.

La tarea de la capa de presentación es aislar las capas inferiores del formato de los datos de la aplicación. Convierte los datos de la aplicación a un formato común, en

ocasiones llamado la representación canónica. La capa de presentación procesa datos dependientes de la máquina, de la capa de aplicación, en un formato independiente de la máquina para las capas inferiores.

La capa de presentación es donde se pierden los formatos de archivo e incluso los formatos de carácter. La conversión del formato de los datos de la aplicación tiene lugar a través de un lenguaje de programación de red común que tiene un formato estructurado. La capa de presentación hace lo inverso para los datos que llegan, éstos se convierten del formato común a los formatos específicos de las aplicaciones, con base en el tipo de aplicación para que tenga instrucciones la máquina. Si los datos llegan sin instrucciones de reformateo, la información podría no ensamblarse en la manera correcta para la aplicación del usuario.

Capa de sesión.

La capa de sesión organiza y sincroniza el intercambio de datos entre procesos de aplicación. Funciona con la capa de aplicación para proporcionar conjuntos de datos sencillos llamados puntos de sincronización, que le permiten a una aplicación conocer cómo está progresando la transmisión y recepción de datos. En términos sencillos, la capa de sesión puede pensarse como una capa cronometradora y controladora del flujo.

La capa de sesión interviene en la coordinación de las comunicaciones entre aplicaciones, permitiendo a cada una conocer el estado de la otra. Un error en una aplicación lo maneja la capa de sesión para permitirle a la aplicación receptora saber que ha ocurrido un error. La capa de sesión puede resincronizar aplicaciones que están conectadas entre sí en ese momento. Esto puede ser necesario cuando las comunicaciones se interrumpen en forma temporal o cuando ha ocurrido un error que resulta en pérdida de datos.

Capa de transporte.

La capa de transporte, como su nombre lo indica, está diseñada para proporcionar la "transferencia transparente de datos de un extremo fuente de un sistema abierto, a un extremo de destino de un sistema abierto", según el modelo de referencia OSI. La capa de transporte establece, mantiene y termina comunicaciones entre dos máquinas.

La capa de transporte es responsable de asegurar que los datos enviados correspondan con los datos recibidos. Este papel de verificación es importante para asegurar que los datos se envien en forma correcta, con un reenvío se detecta un error. La capa de transporte maneja el envío de datos, determinando su orden y su prioridad.

Capa de red.

La capa de red proporciona el enrutamiento físico de los datos, determinando la ruta entre las máquinas. La capa de red maneja todas estas cuestiones de enrutamiento, liberando las capas superiores de este asunto.

La capa de red examina la topología de red para determinar la mejor ruta para enviar un mensaje, así como descifrar los sistemas de retransmisión. Es la única capa de red que envía un mensaje de una máquina fuente a una de destino, manejando otros trozos de datos que pasan a través del sistema en su camino hacia otra máquina.

Capa de vinculación de datos.

De acuerdo con el documento de referencia OSI, la capa de vinculación de datos "proporciona el control de la capa física y detecta y posiblemente corrige errores que pueden ocurrir". En la práctica, la capa de vinculación de datos es responsable de corregir errores de transmisión inducidos durante ésta (en oposición a los errores en los datos de aplicación en sí, los cuales se manejan en la capa de transporte).

Por lo general la capa de vinculación de datos está interesada en la interferencia de señales en los medios de transmisión físicos, sean a través de alambre de cobre, cable de fibra óptica o microondas. La interferencia es común, resultado de muchas fuentes, incluyendo rayos cósmicos e interferencia magnética dispersa proveniente de otras fuentes.

Capa física.

La capa física es la capa inferior del modelo OSI y tiene que ver con los "medios mecánicos, eléctricos, funcionales y de procedimiento" requeridos para la transmisión de datos, de acuerdo a la definición de OSI. Esto en realidad es el cableado u otra forma de transmisión.

1.7.4 Arquitectura TCP/IP.

Aunque el nombre TCP/IP implica que el alcance completo del producto es una combinación de dos protocolos - Protocolo de Control de Transmisión (Transmission Control Protocol) y Protocolo Internet (Internet Protocol) -, el término TCP/IP no se refiere a una entidad única que combina dos protocolos, sino a un conjunto más grande de programas de software que proporciona servicios de red como registros remotos, transferencias de archivos remotas y correo electrónico. TCP/IP proporciona un método para transferir información de una máquina a otra. Un protocolo de comunicaciones debe manejar los errores en la transmisión, administrar el enrutamiento y envío de datos y controlar la transmisión real por medio del uso de señales de estado predeterminadas. TCP/IP logra todo esto.

TCP/IP no es solo un producto, sino un nombre global para una familia de protocolos que usan un comportamiento similar. El término TCP/IP por lo general se refiere a uno o más protocolos dentro de la familia, no sólo al TCP e IP.

La figura 1.7.4.1 muestra los elementos básicos de la familia de protocolos TCP/IP. Como se ve TCP/IP no esta comprendido en las dos capas inferiores del modelo OSI (vinculación de datos y física), sino que comienza en la capa de red, donde reside el Protocolo Internet (IP). En la capa de transporte intervienen el Transmission Control Protocol (TCP) y el User Datagram Protocol (UDP, Protocolo de Datagrama de Usuario). Las utilerías y protocolos que forman el resto del conjunto TCP/IP se crean encima de éstos al usar las capas TCP o UDP e IP para su sistema de comunicaciones.

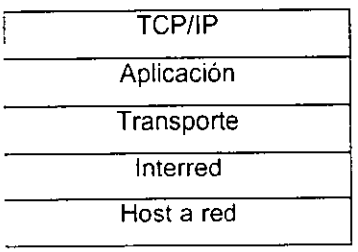


Figura 1.7.4.1. Modelo de referencia TCP/IP

La figura 1.7.4.1 muestra que algunos de los protocolos de la capa superior dependen del TCP (como Telnet y FTP), mientras que algunos dependen del UDP (como TFTP y RPC). La mayor parte de los protocolos TCP/IP de las capas superiores usan sólo uno de los dos protocolos de transporte (TCP o UDP), aunque unos cuantos, incluyendo DNS (Domain Name Service), pueden usar ambos.

TCP/IP es dependiente del concepto de clientes y servidores. Esto no tiene nada que ver con un servidor de archivos al que se tiene acceso mediante una estación de trabajo o PC sin disco. El término ***cliente/servidor*** tiene un significado sencillo en TCP/IP: cualquier dispositivo que inicie comunicaciones es el cliente y el dispositivo que responde es el servidor. El servidor esta respondiendo (sirviendo) a las solicitudes del cliente.

OSI Y TCP/IP.

La adopción del TCP/IP no esta en conflicto con las normas OSI, debido a que los dos se produjeron de manera concurrente. De alguna manera, el TCP/IP contribuyo al OSI y viceversa.

Sin embargo, existen varias diferencias importantes, las cuales surgen de los requerimientos básicos del TCP/IP que son:

- Un conjunto común de aplicaciones
- Enrutamiento dinámico
- Protocolos sin conexión en el nivel de red
- Conectividad universal
- Intercambio de paquetes

Las diferencias entre la arquitectura OSI y la del TCP/IP se relacionan con las capas encima del nivel de transporte y aquellas del nivel de red. OSI tiene una capa de sesión y una de presentación, en tanto que TCP/IP combina ambas en una capa de aplicación. El requerimiento de un protocolo sin conexión, también requirió que el TCP/IP incluyera además, las capas de sesión y presentación del modelo OSI en la capa de aplicación del TCP/IP. En la figura 1.7.4.2 se muestra una vista esquemática de la estructura en capas del TCP/IP, comparada con el modelo de siete capas del OSI.

El enfoque en capas dio origen al nombre TCP/IP. La capa de transporte usa el Transmission Control Protocol (TCP) o una de las diversas variantes, como el User Datagram Protocol (UDP). Sin embargo, sólo existe un protocolo para el nivel de red: el

Internet Protocol (IP). Esto es lo que asegura la conectividad universal del sistema, uno de los objetivos primarios del diseño.

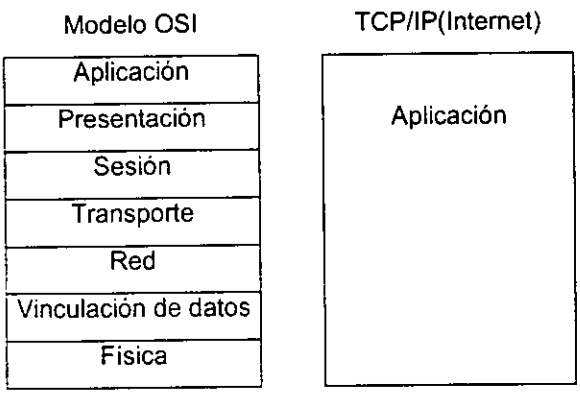


Figura 1.7.4.2. Las estructuras en capas del OSI y del TCP/IP

1.7.5 Pasado y presente.

La arquitectura TCP/IP a menudo se llama la arquitectura de Internet, debido a que el TCP/IP e Internet están entrelazados de manera muy próxima. Las normas de Internet fueron desarrolladas por la Defence Advanced Research Projects Agency (DARPA); y finalmente pasada a la Internet Society.

Internet fue propuesta originalmente por la precursora de la DARPA, llamada Advanced Research Projects Agency (ARPA), como un método para probar la vialidad de las redes de intercambio de paquetes. (Cuando el interés de la ARPA se volvió de naturaleza militar, se cambió de nombre). Durante su ocupación en el proyecto. ARPA previó una red de líneas arrendadas, conectadas por nodos interruptores. La red se llamó ARPANET y los nodos interruptores se llamaron Internet Message Processors (Procesadores de Mensajes Entre Redes).

Una necesidad reconocida en forma común fue la capacidad para transferir archivos desde una máquina a otra, así como la capacidad para soportar registros remotos. Los

registros remotos permitirían a un usuario en Santa Bárbara conectarse con una máquina en Los Ángeles a través de la red y funcionar como si estuviera frente a la máquina en la UCLA. El protocolo usado entonces en la red no era capaz de manejar estos requerimientos de funcionalidad nuevos, de modo que en forma continua fueron desarrollados, perfeccionados y probados nuevos protocolos.

El registro remoto y la transferencia de archivos remota por fin fueron puestos en práctica en un protocolo llamado Network Control Program (NCP). Junto con los registros remotos y transferencia de archivos del NCP, conformaron los servicios básicos para ARPANET.

Para 1973. era claro que NCP era incapaz para manejar el volumen de tráfico y la funcionalidad nueva propuesta. Se comenzó un proyecto para desarrollar un protocolo nuevo. Las arquitecturas TCP/IP y gateway fueron propuestas por primera vez en 1974.

El artículo publicado por Cerf y Kahn describía un sistema que proporcionaba un protocolo de aplicación estandarizada que además, usaba reconocimientos de extremo a extremo.

En realidad ninguno de estos conceptos era novedoso en ese tiempo pero de mayor importancia. Cerf y Kahn sugirieron que el nuevo protocolo fuera independiente de la red y el hardware de computadoras subyacentes. Además propusieron una conectividad universal a través de la red. Estas dos ideas fueron radicales en un mundo de Hardware y Software patentados, debido a que permitirían participar en la red a cualquier clase de plataforma. El protocolo se elaboró y se conoció como TCP/IP.

En 1981 se emitieron una serie de RFC (solicitudes de comentarios, parte del proceso para adoptar normas en Internet nuevas). estandarizando el TCP/IP versión 4 en ARPANET. En 1982. TCP/IP sustituyó al NCP como protocolo dominante de la red creciente. la cual ahora estaba conectando máquinas a lo largo del continente.

El crecimiento de ARPANET y su desaparición subsecuente ocurrieron con la aprobación de la Office of Advanced Scientific Computing para desarrollar un acceso amplio a las supercomputadoras. Crearon la NSFNET para conectar seis supercomputadoras diseminadas a lo largo del país por medio de líneas T-1 (los cuales operaban a 1.544 Mbps). Por fin en 1990 el Departamento de Defensa declaro obsoleta a ARPANET y fue desmantelada en forma oficial.

Realizaciones Berkeley y UNIX y TCP/IP.

TCP/IP se volvió importante cuando el Departamento de Defensa comenzó a incluir los protocolos como normas militares, los cuales fueron requeridos para muchos contratos. TCP/IP se volvió popular sobre todo debido al trabajo hecho en la UCB (Berkeley). La UCB había sido un centro de desarrollo UNIX durante años, pero en 1983 publicaron una versión nueva que incorporaba al TCP/IP como un elemento integral. Esta versión se puso a disposición del mundo como software de dominio público. Prácticamente todas las versiones del TCP/IP disponibles en la actualidad tienen sus raíces en las versiones Berkeley.

1.7.5.1 Servicio en modo conexión y en modo sin conexión.

Las capas pueden ofrecer dos tipos diferentes de servicio a las capas que se encuentran sobre ellas, los orientados a la conexión y los que carecen de conexión.

El servicio orientado a la conexión encuentra su modelo en el sistema telefónico. Para conversar con alguien, descolgamos el teléfono, marcamos el número, hablamos y después colgamos. De manera similar, para usar un servicio de red orientado a la conexión, el usuario del servicio establece primero una conexión y después la libera. El aspecto esencial de una conexión es que actúa como un tubo: el emisor empuja objetos (bits) por un extremo y el receptor los saca en el mismo orden por el otro extremo.

En contraste, el servicio sin conexión toma su modelo del sistema postal. Cada mensaje (carta) lleva la dirección completa de destino, y cada uno se encamina a través del sistema de forma independiente de todos los demás. Normalmente, cuando se envían dos mensajes al mismo destino, el primero que se envío era el primero en llegar . Sin embargo, es posible que el primero que se envío se retrase tanto que el segundo llegue primero. Con un servicio orientado a la conexión, esto es imposible.

Cada servicio se puede caracterizar por una ***calidad de servicio***. Algunos servicios son confiables en el sentido de que nunca pierden datos. Usualmente, un servicio confiable se implementa haciendo que el receptor acuse el recibo de cada mensaje, de modo que el emisor este seguro de que llegó. El proceso de acuse de recibo introduce una sobrecarga y retardos que con frecuencia valen la pena pero que algunas veces son intolerables.

Una situación típica en la que un servicio confiable orientado a la conexión es apropiado es la transferencia de archivos. El propietario del archivo quiere asegurarse de que todos los bits lleguen correctamente y en el mismo orden en que se enviaron.

No todas las aplicaciones requieren conexiones. Por ejemplo, a medida que el correo electrónico se vuelve más común, el que envió el correo electrónico de propaganda probablemente no quiera tomarse la molestia de establecer y después liberar una conexión sólo para enviar un artículo. Todo lo que se necesita es una forma de enviar un solo mensaje que tenga la probabilidad elevada de llegar, aunque no este garantizada. El servicio sin conexión no confiable con frecuencia recibe el nombre de ***servicio de datagramas***.

En otras situaciones, es deseable la comodidad de no tener que establecer una conexión para enviar un mensaje corto, pero la confiabilidad es esencial. Para estas aplicaciones se puede proporcionar el ***servicio de datagrama con acuse***. Esto es como mandar una carta certificada y pedir recibo. Cuando el recibo regresa, el

remitente tiene la seguridad absoluta de que la carta se entregó a la parte interesada y de que no se perdió en el camino.

Un servicio más es el ***servicio de petición y respuesta***. En este servicio el remitente transmite un datagrama sencillo que contiene una petición; la respuesta contiene la contestación. La petición/respuesta se usa mucho para instrumentar la comunicación en el modelo cliente-servidor, el cliente emite una petición y el servidor le responde. La figura 1.7.5.1.1 resume los tipos de servicio antes descritos.

	Servicio	Ejemplo
Orientado	Flujo de mensaje confiable	Secuencia de páginas
a la conexión	Flujo de bytes confiable	Ingreso remoto
	Conexión no confiable	Voz digitalizada
Sin	Datagrama no confiable	Correo electrónico chatarra
conexión	Datagrama con acuse de recibo	Correo registrado
	Petición/respuesta	Consulta de base de datos

Figura 1.7.5.1.1. Seis tipos diferentes de servicio.

1.7.5.2 Direccionamiento.

Protocolo de definición de direcciones.

Determinar direcciones puede ser difícil debido a que cada máquina en la red podría no tener una lista de todas las direcciones de las otras máquinas o dispositivos. Si no se conoce la dirección física de la máquina receptora, enviar datos de una máquina a otra, puede causar un problema si no existe un sistema de definición para determinar las direcciones. Para la administración de la red sería una pesadilla actualizar en forma constante una tabla de direcciones en cada máquina. El problema no se restringe a las direcciones de máquina dentro de una red pequeña, debido a que si se desconocen las direcciones de la red de destino remota, también ocurrirían problemas de enrutamiento y envío.

El protocolo de definición de direcciones (ARP) ayuda a solucionar estos problemas. El trabajo del ARP es convertir las direcciones IP a direcciones físicas (de red y locales) y, al hacerlo , elimina las necesidades de que las aplicaciones conozcan las direcciones físicas . En esencia, el ARP es una tabla con una lista de las direcciones IP y sus correspondientes direcciones físicas . La tabla se llama ***caché ARP***. La disposición de un caché ARP se muestra en la figura 1.7.5.2.1. Cada fila corresponde a un dispositivo, con cuatro piezas de información para cada dispositivo.

	INDICE IF	DIRECCIÓN FÍSICA	DIRECCIÓN IP	TIPO
Registro 1				
Registro 2				
Registro 3				
Registro 4				
Registro 5				

Figura 1.7.5.2.1. La disposición de la tabla de traducción de direcciones caché ARP.

- Índice IF: el puerto físico (interfaz)
- Dirección física: la dirección física del dispositivo
- Dirección IP: la dirección IP correspondiente a la dirección física
- Tipo: el tipo de registro en el caché ARP

1.7.5.3 Ruteo.

Para enviar mensajes a través de redes, el software de la capa IP de una máquina compara la dirección de destino del mensaje con la dirección de la máquina local. Si el mensaje no es para la máquina local, el mensaje se pasa a la siguiente máquina. Mover mensajes por una red pequeña es bastante fácil, pero las redes grandes y las

interredes aumentan su complejidad, requiriendo de gateways, puentes y enrutadores, los cuales tratan de establecer el mejor método de mover un mensaje hasta su destino. La definición de estos términos es:

- Un gateway es un dispositivo que ejecuta funciones de enrutamiento, por lo general como un dispositivo autónomo, que además puede traducir del protocolo de una red al de otra.

- Un puente es un dispositivo de red que conecta dos o más redes que usan el mismo protocolo.

- Un enrutador es un nodo de red que envía datagramas por la red.

La capacidad de conversión de protocolo del gateway es importante. La conversión de protocolo tiene lugar, por lo general, en las capas inferiores, en ocasiones incluyendo a la capa de transporte. Esto les proporciona más flexibilidad, al poder interpretar y traducir direcciones entre redes distintas. La conversión puede ocurrir de varias formas, como cuando se pasa de un formato de red de área local a Ethernet. Los gateways son usados principalmente en redes de área ancha (WANs), donde no se espera que nadie utilice más de 10,000 paquetes por segundo, uno de los requerimientos importantes para un puente en una red local.

En Estados Unidos, Internet tiene la NFSNET como vinculo de conexión, como se muestra en la figura 1.7.5.3.1. Entre las redes primarias conectadas a NFSNET están la Space Physics Analysis Network (SPAN) de la NASA, la Computer Science Network (CSNET) y otras diversas redes como la WESTNET. También hay otras redes más pequeñas orientadas al usuario como la Because It's Time Network (BITNET) y la UUNET, las cuales proporcionan conectividad a través de gateways para sitios más pequeños que no pueden establecer un gateway directo hacia Internet.

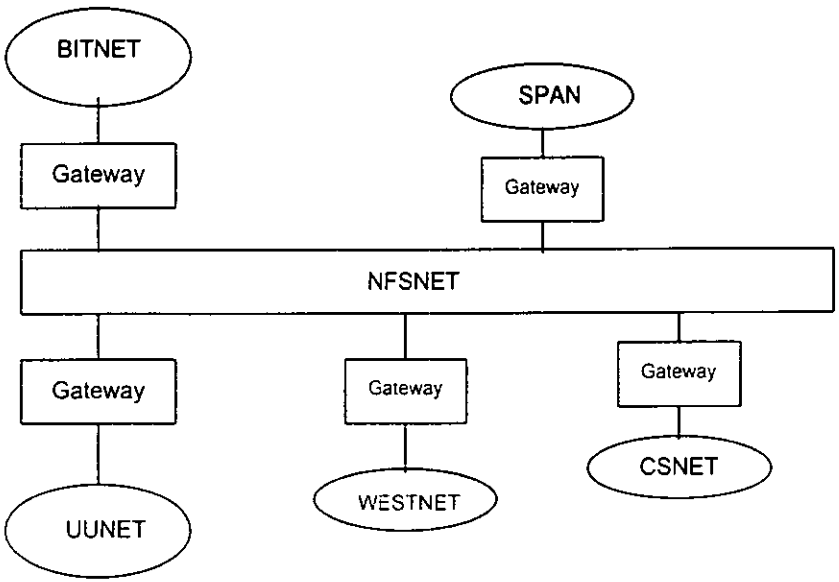


Figura 1.7.5.3.1 La red Internet de Estados Unidos.

Los puentes actúan como enlaces entre redes, las cuales con frecuencia tienen un puente en cada extremo de una línea de comunicaciones dedicada o a través de un sistema de paquetes como Internet. Podría aplicarse una conversión entre puentes para aumentar la velocidad de transmisión.

En general, los puentes son específicos del hardware: Ethernet a Ethernet, Token Ring a Token Ring, etc. es decir, ambas redes han de usar el mismo protocolo de comunicaciones. Por ejemplo, un puente Ethernet permitirá a dos o más redes Ethernet ser conectadas e interoperar juntas, sin depender de los protocolos o sistema operativo de red que estén siendo utilizados. Figura 1.7.5.3.2.

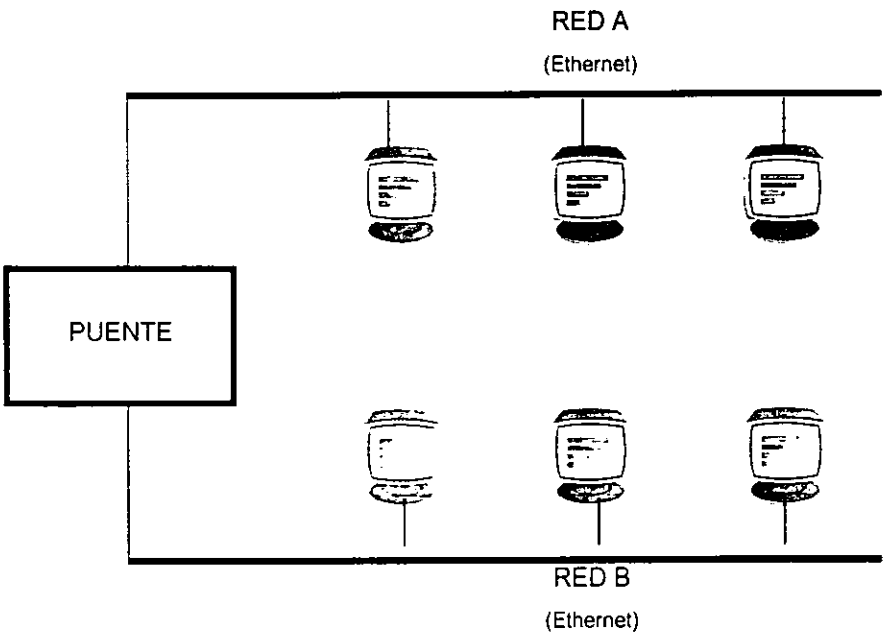


Figura 1.7.5.3.2. Puente entre dos redes Ethernet.

Los enrutadores operan al nivel de red. enviando paquetes a su destino. En ocasiones un cambio de protocolo puede realizarse con un enrutador que tiene disponibles varias opciones de envío, como Ethernet o líneas en serie.

Enrutamiento se refiere a la transmisión de un paquete de información de una máquina a otra. Cada máquina en la que entra el paquete analiza el contenido del encabezado del paquete y decide su acción con base en la información dentro del encabezado. Si la dirección de destino del paquete concuerda con la dirección de la máquina, el paquete deberá retenerse y procesarse con protocolos de nivel superior. Si la dirección de destino no concuerda con la de la máquina, el paquete se envía adelante por la red. El

envío puede ser a la máquina de destino misma o a un gateway o a un puente si el paquete tiene que dejar la red local.

Los ruteadores son específicos de los protocolos: un ruteador debe saber los protocolos usados por los datos que están siendo enviados. Por ejemplo, Netware de Novell usa protocolos llamados IPX (Internetwork Packet Exchange) y SPX (Sequenced Packet Exchange), mientras que el sistema operativo de 3COM, el 3+SHARE usa el protocolo XNS (de Xerox Network System). Ver figura 1.7.5.3.3.

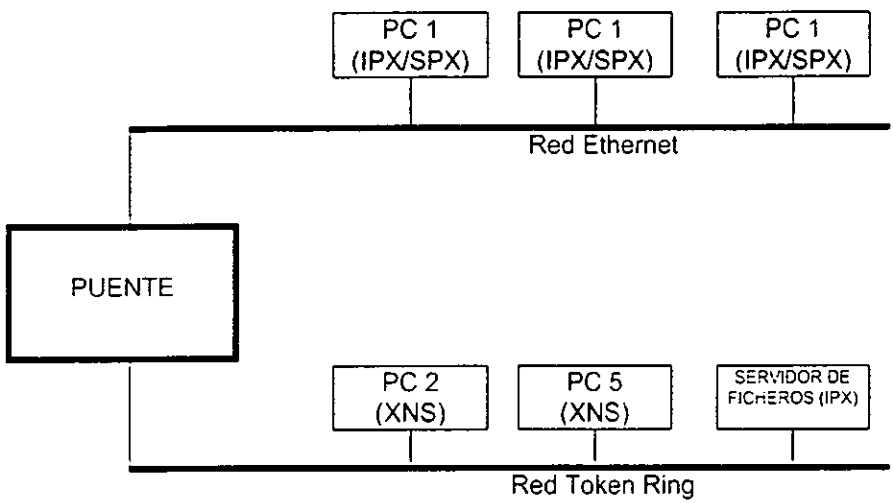


Figura 1.7.5.3.3. Puente entre dos redes Ethernet

Es necesario contar con una ruta óptima de la máquina fuente a la de destino, así como manejar problemas como una carga pesada en una máquina mediadora o la pérdida de una conexión. Los detalles de la ruta están contenidos en una tabla de enrutamiento y varios algoritmos complicados funcionan con la tabla de enrutamiento para desarrollar una ruta óptima para un paquete.

Crear una tabla de enrutamiento y mantenerla con registros válidos son aspectos importantes de un protocolo. Aquí hay unos cuantos métodos para crear una tabla de enrutamiento:

- Se crea una tabla fija con un mapa de la red, la cual debe modificarse y releerse cada vez que haya un cambio físico en cualquier parte de la red.

- Se usa una tabla dinámica que evalúa la carga y los mensajes del tráfico de otros nodos para afinar una tabla interna.

- Se usa una tabla de enrutamiento central fija que se carga desde el depósito central, mediante los nodos de red a intervalos regulares o cuando es necesario.

Cada método tiene ventajas y desventajas. El enfoque de la tabla fija, ya sea que se localice en cada nodo de red o, se transfiera a intervalos regulares desde una tabla fija mantenida en forma centralizada, es inflexible y no puede reaccionar con rapidez ante los cambios en la topología de la red. La tabla central es mejor que la primera opción, tan sólo porque es posible para un administrador mantener la tabla única con mucha mayor facilidad que una tabla en cada nodo.

La tabla dinámica es la mejor para reaccionar ante los cambios, aunque requiere mejor control, software más complejo y más tráfico de red. Sin embargo, las ventajas por lo general sobrepasan a las desventajas y una tabla dinámica es el método usado con mayor frecuencia en Internet.

1.7.5.4 Servicio de nombres.

En lugar de usar la dirección IP de 32 bits completa, muchos sistemas adoptan nombres más significativos para sus dispositivos y redes. Los nombres de redes por lo general reflejan el nombre de la organización (como tpci.com y bobs_cement). Los nombres de los dispositivos individuales dentro de una red pueden variar desde nombres

descriptivos en redes pequeñas (como tims_machine y laser_1) hasta convenciones de denominación más complejas en redes más grandes (como hpws_23 y tcpi704). Traducir entre estos nombres y las direcciones IP sería prácticamente imposible en una escala tan amplia como Internet.

Para solucionar el problema de los nombres de las redes, el Centro de Información de Redes (NIC) mantiene una lista de nombres de redes y las correspondientes direcciones gateway de la red. Este sistema creció de una lista de archivos planos hasta un sistema más complicado llamado Servicio de Nombre de Dominio (Domain Name Service, DNS) cuando las redes se volvieron demasiado numerosas para que funcionara con eficiencia el sistema de archivos planos.

DNS usa una arquitectura jerárquica muy parecida al sistema de archivos UNIX. El primer nivel de denominación divide a las redes en la categoría de subredes, como ***com*** para comercial, ***mil*** para militar, ***edu*** para educación y así sucesivamente. Debajo de cada una de éstas hay otra división que identifica a la subred individual, por lo general una para cada organización. Esto se llama ***nombre de dominio*** y es único. El administrador del sistema de la organización puede dividir más las subredes de la compañía conforme se desee, con cada red llamada ***subdominio***. Por ejemplo, el sistema **merlin.abc_corp.com** tiene el nombre de dominio **abc_corp.com**, mientras que la red **merlin.abc_corp** es un subdominio de **merlin.abc_corp.com**. Una red puede identificarse con parte del nombre del dominio completo.

DNS usa dos sistemas para establecer y rastrear los nombres de dominio. Un ***solucionador de nombres*** en cada red examina la información en su nombre de dominio. Si no puede encontrar la dirección IP completa, interroga a un ***servidor de nombres***, el cual tiene disponible la información completa del NIC. El solucionador de nombres trata de completar la información de direccionamiento usando su propia base de datos. la cual se actualiza de manera muy parecida a la del sistema ARP cuando va a interrogar a un servidor de nombres. Si un servidor de nombres interrogado no puede resolver la dirección, interrroga a otro servidor de nombres, y así sucesivamente, a través de toda la interred.

Existe una cantidad considerable de información almacenada en el solucionador de nombres y en el servidor de nombres, así como un conjunto completo de protocolos para interrogatorio entre los dos. El concepto general de la resolución de direcciones es importante, cuando se entiende cómo traduce Internet entre nombres de dominio y direcciones IP.

1.7.5.5 Suite del protocolo TCP/IP.

Los protocolos de computadora definen la manera como tienen lugar las comunicaciones. Si una computadora está enviando información a otra y ambas siguen el protocolo de manera apropiada, el mensaje llega, sin importar qué tipo de máquinas sean y qué sistemas operativos ejecuten. En esencia, un protocolo de computadora es un conjunto de reglas que coordina el intercambio de información.

Un protocolo único para cubrir todos los aspectos de la transferencia sería demasiado grande, poco manejable y excesivamente especializado. Por consiguiente, se han desarrollado varios protocolos, cada uno manejando una tarea específica.

Los protocolos Internet se han convertido en una de las familias de protocolos más ampliamente utilizada en el mundo. Aunque se conocen con el nombre genérico de TCP/IP, los protocolos Internet constan de muchos protocolos diseñados para dar soporte a las operaciones de comunicación entre redes.

Ahora veremos los elementos básicos de la familia de protocolos TCP/IP:

- Telnet. Acceso remoto de terminales.

- FTP (File Transfer Protocol, Protocolo de Transferencia de Archivos). Transferencia de archivos

- SMTP (Simple Mail Transfer Protocol, Protocolo de Transferencia de Correo Simple). Transferencia de correo electrónico.

- Kerberos. Protocolo de seguridad.

- DNS (Domain Name Service, Servicio de Nombre de Dominio). Permite a una computadora con un nombre común convertirse en una dirección de red especial.

- SNMP (Simple Network Management Protocol, Protocolo simple de administración de la red). Proporciona mensajes de estado y reporta problemas a través de una red hacia un administrador.

- NFS (Network File System, Sistema de Archivos en Red). Permite a múltiples máquinas tener acceso a los directorios de cada una de las otras, de manera transparente.

- RPC (Remote Procedure Call, Llamada Remota a Procedimiento). Es un conjunto de funciones que permite a una aplicación comunicarse con otra máquina.

- TFTP (Trivial File Transfer Protocol, Protocolo de Transferencia de Archivos Simple). Ejecuta las mismas tareas que FTP, pero usa un protocolo de transporte diferente.

- TCP (Transmission Control Protocol, Protocolo de Control de Transmisión). Es un protocolo de comunicaciones que proporciona una transferencia confiable de datos.

- UDP (User Datagram Protocol, Protocolo de Datagrama de Usuario). Es un protocolo orientado hacia la ausencia de conexión.

- IP (Internet Protocol. Protocolo de Internet). Es responsable de mover los paquetes de datos ensamblados. ya sea por el TCP o el UDP a través de las redes.

1.7.5.5.1 Aplicaciones y utilidades.

El término TCP/IP no se refiere a una entidad única que combina protocolos, sino a un conjunto más grande de programas de software que proporciona servicios de red como registros remotos, transferencia de archivos remotas y correo electrónico. Un protocolo de comunicaciones debe manejar los errores de transmisión, administrar el enrutamiento y envío de datos y controlar la transmisión real por medio del uso de señales de estado predeterminadas. TCP/IP logra todo esto.

1.7.5.5.2 Protocolo de control de transferencia.

El protocolo de control de transmisión proporciona una cantidad considerable de servicios a la capa IP (Protocolo de Internet) y a las capas superiores. De mayor importancia, proporciona un protocolo orientado hacia la conexión para las capas superiores, que permite a una aplicación estar segura de que un datagrama enviado por la red fue recibido íntegro. En este papel, el TCP (Protocolo de Control de Transferencia) actúa como un protocolo de validación del mensaje que proporciona comunicaciones confiables. Si un datagrama se altera o pierde, por lo general el TCP maneja la retransmisión, en lugar de las aplicaciones de las capas superiores.

El TCP maneja el flujo de datagramas desde las capas superiores hasta la capa IP, así como los datagramas que llegan desde la capa IP hacia los protocolos de niveles superiores. El TCP tiene que asegurar que las prioridades y la seguridad se manejen de manera apropiada. El TCP debe ser capaz de manejar la terminación de una aplicación superior que estaba esperando la llegada de datagramas, al igual que las fallas en las capas inferiores. El TCP también debe mantener una tabla de estado de todas las series de datos que entran y salen de la capa TCP. El aislamiento de todos estos servicios en una capa separada permite que las aplicaciones se diseñen sin importar el control del flujo o la confiabilidad del mensaje. Sin la capa TCP, cada aplicación tendría que aplicar los servicios por si misma, lo cual es un desperdicio de recursos.

El TCP reside en la capa de transporte, ubicada encima del IP pero debajo de las capas superiores y sus aplicaciones, como se muestra en la figura 1.7.5.5.2.1. El TCP reside solo en dispositivos que en realidad procesan datagramas, como asegurando que el datagrama ha ido desde la máquina fuente hasta la máquina neta. No reside en un dispositivo que tan solo enruta datagramas, de ahí que por lo general no haya una capa TCP en un gateway. Esto tiene sentido, debido a que en un gateway el datagrama no tiene necesidad de ir más arriba de la capa IP en el modelo en capas.

Debido a que el TCP es un protocolo orientado hacia la conexión responsable de asegurar la transferencia de un datagrama, desde la máquina fuente hasta la máquina de destino (comunicaciones de extremo a extremo), el TCP debe recibir mensajes de comunicaciones desde la máquina de destino para acusar recibo del datagrama. El término ***circuito virtual*** por lo general se usa para referirse a las comunicaciones entre las dos máquinas en los extremos, la mayor parte de las cuales son solo mensajes de acuse de recibo (ya sea de confirmación del recibo o una falla en el código) y número de secuencia de datagrama.

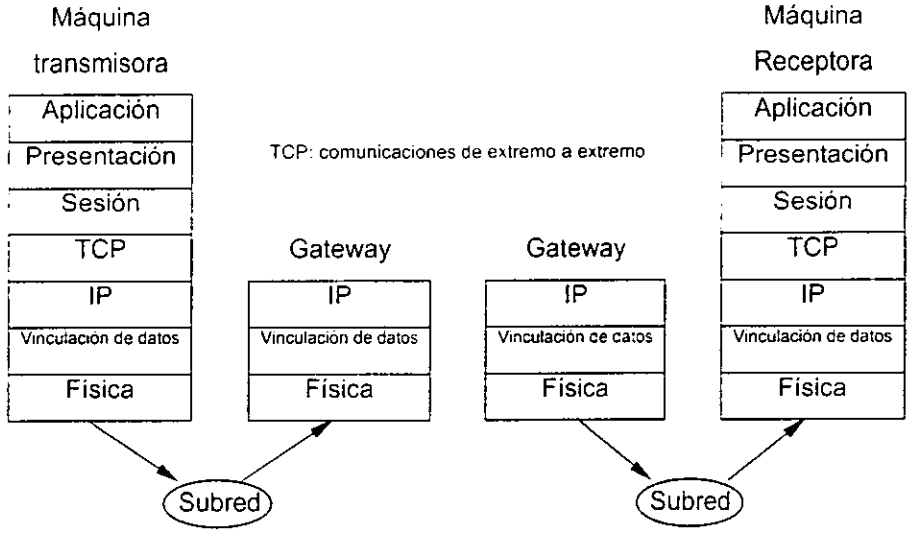


Figura 1.7.5.5.2.1. El TCP proporciona comunicación de extremo a extremo.

1.7.5.5.3 Protocolo de transferencia de archivos simple.

El Trivial File Transfer Protocol (TFTP) es uno de los protocolos de transferencia de archivos en uso más sencillos. Difiere del FTP de dos maneras principales: no se registra en la máquina remota y usa un protocolo de transporte sin conexión User Datagram Protocol (UDP; Protocolo de Datagrama de usuario) en lugar del TCP. Al usar el UDP, el TFTP no supervisa el proceso de la transferencia de archivos, aunque tiene que emplear algoritmos más complejos para asegurar la integridad apropiada de los datos. Al evitar el registro remoto se evitan los problemas de autorización de acceso y de archivo. El TFTP usa el identificador de puerto TCP número 69, aún cuando el TCP no participa en el protocolo.

El TFTP tiene pocas ventajas sobre el FTP. Por lo general, no se usa para transferencias de archivos entre máquinas donde se podría usar el FTP en su lugar, aunque el TFTP es útil cuando comprende una terminal o estación de trabajo sin disco. De manera común el TFTP se usa para cargar aplicaciones y fuentes en estas máquinas, así como para el arranque. El TFTP es necesario en estos casos debido a que las máquinas sin disco no pueden ejecutar el FTP hasta que están cargadas por completo por un sistema operativo.

Los requisitos de tamaño de ejecución y memoria pequeños del TFTP lo hacen ideal para incluirlo en el arranque, donde el sistema solo requiere el TFTP, el UDP y un controlador de red. todo lo cual puede proporcionarse en un EPROM pequeño.

El FTP maneja las autorizaciones de acceso y archivo imponiendo restricciones por si mismo.

El TFTP soporta algunos mensajes de error básicos, pero no puede manejar errores sencillos como recursos insuficientes para una transferencia de archivos o incluso una falla para localizar un archivo solicitado.

1.7.5.5.4 Protocolo TCP/IP.

Como IP es una red no orientada a conexión, es TCP quien se debe encargar de las tareas de fiabilidad, control de flujo, secuenciamiento, aperturas y cierres. Aunque TCP e IP estén tan relacionados que incluso se les denomine "TCP/IP", TCP puede soportar otros protocolos. Por ejemplo, otro protocolo no orientado a conexión como ISO 8473 (Protocolo de redes no orientadas a conexión o CNLP) podría funcionar con TCP (si se realizan algunos ajustes de los interfaces entre módulos). Además, los protocolos de aplicación, como el protocolo de transferencia de archivos (FTP) y el protocolo de transferencia de correo simple (SMTP) se apoyan en muchos servicios que proporciona TCP.

1.7.5.5.5 Protocolo de acceso a red.

Sistema de archivos de red (NFS).

El procesamiento distribuido y las arquitecturas cliente/servidor han significado que muchos usuarios puedan tener en su escritorio, máquinas pequeñas y eficaces que se comunican con un servidor más grande que se encuentra en alguna parte de la red. Las aplicaciones que el usuario necesita a veces están colocadas en lugares que no forman parte de su escritorio, por lo que se requiere de algún método para tener acceso a archivos remotos (aplicaciones y datos). Aunque tanto Telnet como rlogin permiten que el usuario utilice una máquina remota, ninguno de estos sistemas aprovecha la máquina del sistema del usuario. La compartición de periféricos también se ha vuelto importante conforme crecen las redes de área local. Para ayudar a integrar las estaciones de trabajo dentro de las redes de área local, así como para simplificar el acceso a los archivos remotos y a la compartición de periféricos. Sun Microsystem presentó el **Sistema de Archivos de Red (NFS)**.

NFS permite que una aplicación lea y escriba los archivos que residen en los servidores NFS, con el acceso para el servidor NFS completamente transparente para la aplicación y para el usuario. Para los desarrolladores, NFS no requiere de código extra ni de un manejo especial, lo cual lo hace en especial atractivo. Este acceso transparente a la estructura de archivos de otras máquinas se logra mediante el enlace lógico del servidor NFS con el cliente, un proceso que se llama ***sobreposición***.

Protocolos NFS.

El protocolo NFS comprende varios protocolos, de los cuales solo uno se llama protocolo NFS. Los protocolos del producto NFS están diseñados como un conjunto de capas, similares al modelo OSI. Las capas del producto NFS pueden compararse con las capas OSI, en la figura 1.7.5.5.5.1; cada protocolo del producto NFS tiene una RFC Internet dedicada a su especificación.

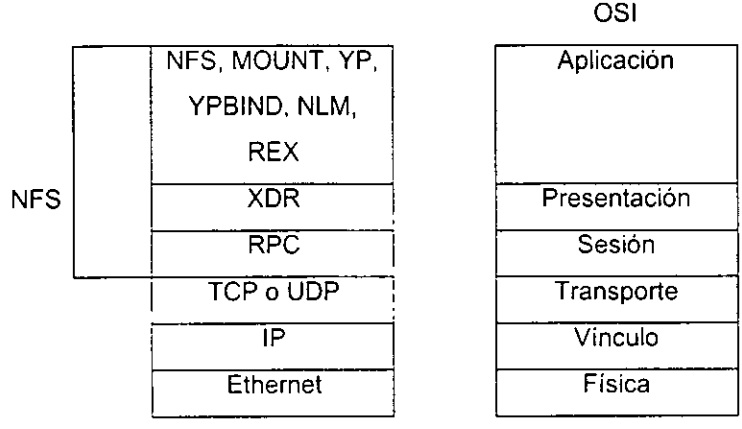


Figura 1.7.5.5.5.1. Las capas de protocolos de NFS

El producto NFS se basa en el modelo de capas de OSI, dando como resultado protocolos que son independientes entre sí, y protocolos que se encuentran en diferentes capas. La filosofía del diseño es que cualquier protocolo de una sola capa pueda reemplazarse con cualquier otro, suponiendo que la funcionalidad de los protocolos fuera la misma. A la fecha, no existen alternativas comunes para los

productos de dos capas bajas, RPC y XDR, aunque existen varias para los de capas altas.

Llamada de procedimiento remoto (RPC).

El protocolo de llamada de procedimiento remoto (RPC) actúa como la capa de sesión y como el intercambiador de mensajes para todas las aplicaciones basadas en NFS. RPC está compuesto por un conjunto de procedimientos que pueden incorporarse dentro de las aplicaciones de alto nivel para manejar cualquier tipo de acceso a la red requerido.

Los desarrolladores de aplicaciones pueden crear sus propios procedimientos RPC entre un cliente (*el que produce la solicitud*) y un servidor (*el que procesa la solicitud*). Un grupo de procedimientos se llama ***servicio***. Cada servidor puede utilizar un solo servicio, por lo que cada servicio se le asigna un ***número de programa*** para identificarse a sí mismo, tanto ante el cliente como ante el servidor.

RPC funciona sobre la red que se encuentra entre un cliente y un servidor. El proceso seguido por RPC se muestra en la figura 1.7.5.5.5.2. Éste comienza con la activación del procedimiento por parte del cliente, desde el cual se envía un mensaje de solicitud hacia el servidor. Después de recibir el mensaje y de extraer la solicitud, el servidor ejecuta el procedimiento solicitado y sobrepone un mensaje de respuesta con los resultados. Al recibir la respuesta, el cliente quita el mensaje y continúa con la ejecución normal de la aplicación. Cada paso del procedimiento lo controlan las rutinas que se encuentran en el interior de la biblioteca RPC.

Los mensajes de RPC pueden enviarse utilizando TCP o UDP. Comúnmente, RPC se utiliza con UDP, debido a que el protocolo basado en la conexión no es necesario y a que UDP por lo general es más rápido. Sin embargo, UDP impone un tamaño para el paquete, lo cual puede ocasionar algunos problemas con los procedimientos. Asimismo, UDP no garantiza la entrega, por lo que una aplicación que utiliza UDP debe manejar aspectos de confiabilidad.

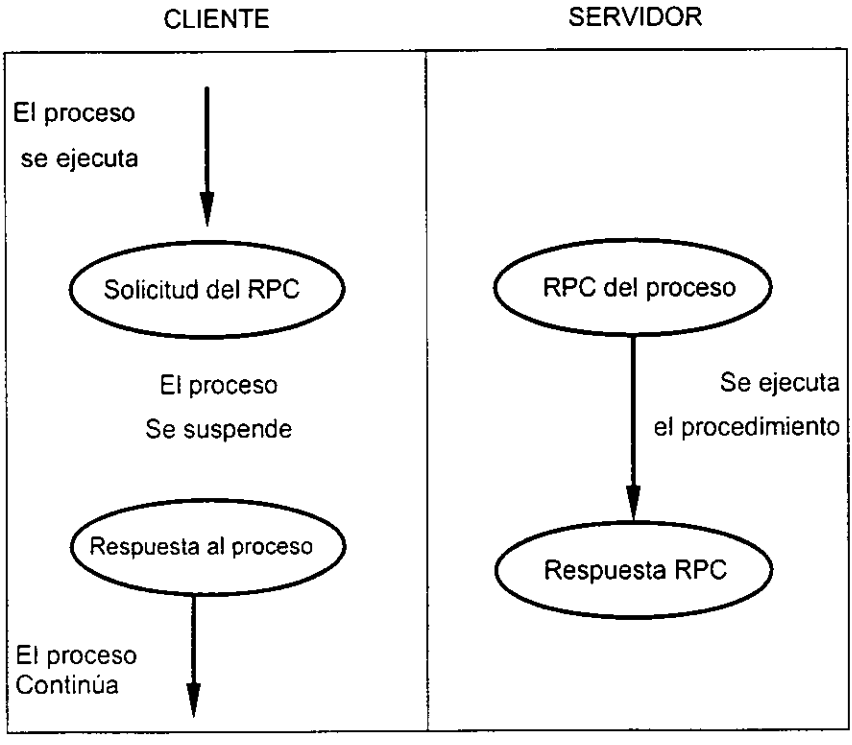


Figura 1.7.5.5.5.2 La ejecución de una RPC

Representación de datos externos (XDR).

La representación de datos externos (XDR) es el método mediante el cual los datos se codifican dentro de un mensaje RPC.

XDR se utiliza para asegurar que los datos de un sistema sean compatibles con otros. Podría parecer que no se requiere de una definición formal, pero considere la posibilidad de una máquina basada en EBCDIC que se está comunicando con una máquina basada en ASCII.XDR permite que ambas terminales hagan una conversión

de su representación de datos local a un formato común, eliminando cualquier duda acerca del significado de los datos. (El mayor problema de conversion no es de EBCDIC a ASCII. Algunos sistemas utilizan bits altos como significativos, y otros utilizan bits bajos. Asimismo, los formatos para definir los tipos de números difieren considerablemente.)

Se ha producido un lenguaje especial, parecido a C, que se llama XDR, con el fin de simplificar el manejo de los datos que están en el formato XDR. Éste puede utilizarse dentro de otros lenguajes de programación.

Protocolo Mount.

Como se mencionó anteriormente, NFS funciona sobreponiéndose en un sistema de archivos para servidor NFS en un sistema de archivos para el cliente. Sin embargo, el protocolo NFS en realidad trata con el acceso y la información de los archivos, y no está conectado con el sistema de archivos. Este procedimiento de sobreposición del sistema de archivos lo trata el producto NFS como un aspecto independiente, que utiliza el protocolo Mount (***Sobreposición***). Mount utiliza UDP.

El protocolo Mount interviene para devolver un manipulador de archivo de un servidor al cliente. permitiendo que el cliente tenga acceso a un área del sistema de archivos del servidor. El protocolo regresa no sólo el manipulador de archivo, sino también el nombre del sistema de archivos en el cual reside el archivo solicitado. La sobreposición consiste en un número de procedimientos que facilitan las comunicaciones entre el cliente y el servidor, y está diseñado específicamente para tratar con archivos.

El protocolo Mount interviene solamente durante la conexión original entre un cliente y un servidor. El proceso Mount hace un seguimiento de las conexiones. pero una vez que la conexión se establece, el protocolo Mount deja todo el control a NFS. Todo lo que NFS necesita para tener acceso a un sistema de archivos es un manipulador de archivo válido (que proporciona Mount cuando hace la conexión).

1.8 DISEÑO ESTRUCTURADO DE SISTEMAS.

1.8.1 Conceptos Básicos de Diseño Estructurado.

La Programación Estructurada, fue la primera metodología que surgió como respuesta para solucionar problemas de programación en los sistemas de cómputo.

Es una metodología de organización y programación de sistemas que mantiene una lógica ordenada, simple y directa, para facilitar su entendimiento y su modificación. Con la cual el programador puede elaborar el algoritmo que se va a implementar, de una manera más lógica, clara y comprensible, lo cual hará que programar sea un proceso coherente y disciplinado, disminuyendo el número de errores y el tiempo invertido. Es deseable que los programas cumplan con las siguientes características:

- Altamente confiables.
- Fáciles de entender.
- Fáciles de corregir.
- Fáciles de mantener.
- Eficientes en cuanto a memoria, tiempo y usuario.

Las normas de la programación estructurada van desde el uso de lenguajes para el diseño de programas. hasta efectuar las últimas pruebas y correcciones, así como el tipo de estructuras lógicas. diagramas de bloques que la programación estructurada permite. También propone una serie de estándares de programación:

- Estilo
- Propiedad
- Estructura
- Claridad
- Sangrado
- Unidades lógicas permitidas
- Documentación de programas, etc.

Mediante los cuales aseguramos que los programas tendrán como características principales claridad, facilidad de comprensión y eficiencia en cuanto a memoria utilizada y tiempo de ejecución, lo que permitirá que satisfagan las necesidades del usuario. Además de ser entendibles para otros programadores y sencillos de ser modificados para su mantenimiento.

1.8.1.1 Diagrama de Flujo Estructurado.

El diagrama de flujo estructurado es una representación gráfica del sistema que permite visualizar la instrumentación, documentación y mantenimiento de los sistemas. Este modelo se genera independientemente del tiempo y de las relaciones jerárquicas entre módulos de un programa o sistema.

El diagrama de flujo estructurado esta formado por los siguientes elementos:

- Módulos
- Conexiones (Flechas)
- Interfaces (Nombres de los datos que entran y salen de cada módulo)

El diagrama Estructurado muestra lo siguiente:

- La división del sistema en módulos
- Jerarquía y organización de los módulos
- Interfaces de comunicación entre módulos
- Nombres (funciones de los módulos)

La lógica del programa es el orden en el que la computadora ejecuta las instrucciones del mismo. Un diagrama de flujo estructurado es un gráfico que describe el orden de operaciones en el algoritmo. Puede por lo tanto, utilizarse para diseñar la lógica de un problema, antes de escribir el programa. Si se va actualizando el diagrama de flujo con todos los cambios en el programa, sirve también como documentación.

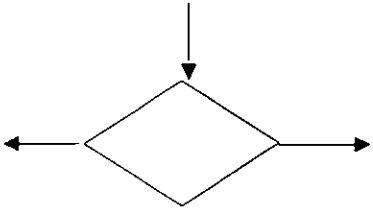
En general, un diagrama de flujo estructurado contará con las siguientes fases:

- Inicio
- Entrada de datos
- Proceso
- Salida de datos
- Fin

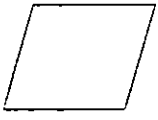
Los símbolos que suelen utilizarse se muestran a continuación:

Símbolo terminal, identifica el comienzo y final del proceso.

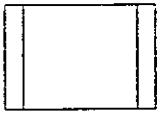
Símbolo de proceso de Información.



Símbolo Decisión, indica elegir entre dos alternativas.



Símbolo de Entrada/Salida.



Proceso predefinido, con Diagrama por separado.

Las flechas, se utilizan para conectar símbolos.

1.8.1.2 Pseudocódigo.

Paralelamente a la programación estructurada, surge una nueva forma de descripción de algoritmos llamada Pseudocódigo. El Pseudocódigo es una herramienta utilizada en el diseño de algoritmos para resolver problemas que empleando los principios de la programación estructurada, permite expresar el flujo de ejecución de las instrucciones de una forma clara, sin ambigüedad alguna.

El pseudocódigo no es un lenguaje de programación, es tan sólo una forma de diseñar la solución a un problema de manera que su traducción posterior a un lenguaje de programación de alto nivel sea sencilla.

1.8.1.2.1 Reglas Generales del Pseudocódigo.

No hay forma estándar de diseñar un pseudocódigo, depende en gran medida del criterio del programador. A pesar de ello, es aconsejable respetar lo más rigurosamente posible las siguientes normas de carácter general:

- Todo pseudocódigo comienza con la palabra *"Inicio"* y termina con la palabra *"fin"*.
- Cada instrucción se debe escribir en una línea.
- Para su descripción, se utilizan una serie de palabras reservadas, que inicialmente se escribían en inglés, como: *start; end: if, then, else, end_if: do_while;* etc. En la actualidad, estas palabras se utilizan en el idioma del país respectivo, así en español estas palabras reservadas son: Inicio; fin: si, entonces, si no, fin_si; mientras; etc.
- Debe escribirse indentado para mostrar claramente las dependencias de control dentro de los módulos.
- Cada estructura utilizada tendrá un solo punto de comienzo y un solo punto de fin de estructura. Algunos autores suelen utilizar un corchete para unir el principio y fin de cada estructura.

- Se escribirá en minúscula, excepto aquellos nombres que elige el programador como son los nombres de variables, de ficheros, de módulos, etc., que se escribirán con mayúsculas.
- Para referenciar un módulo se especifica simplemente su nombre entre los símbolos "<" y ">".

1.8.1.3 Estructuras Lógicas de Programación.

La Programación Estructurada es un criterio de programación basado en el Teorema de la Estructura de Bohn y Jacopini que dice lo siguiente: "Todo programa propio, es decir, con un solo punto de entrada y un solo punto de salida, puede ser escrito utilizando únicamente tres tipos de estructura de control: estructura secuencial, estructura condicional y estructura repetitiva".

A la programación estructurada se le denomina así porque en ella se hace uso de Estructuras Lógicas de Programación, las cuales son simplemente, un conjunto de instrucciones iterativas y de decisión, que indican la manera lógica en que se comporta el programa. Para aumentar la eficacia de la programación se necesita que los programas tengan una estructura fácil de interpretar, de modo que los haga más comprensibles, manejables, modificables y fiables.

Las estructuras lógicas de programación sirven para controlar las acciones que realiza un programa.

1.8.1.3.1 Estructura Secuencial.

Una estructura secuencial es la indicación de un proceso, el cual puede involucrar una o más acciones, y estas se ejecutan una a continuación de otra, sin posibilidad de omitir ninguna. Por lo que una secuencia (bloque) puede ser muy simple e implicar a una sola acción o ser muy compleja (ver figura 1.8.1.3.1.1).

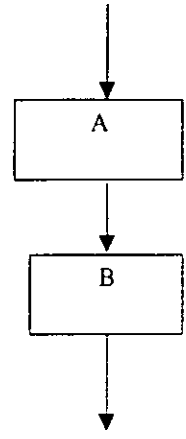


Figura 1.8.1.3.1.1 Estructura Secuencial

1.8.1.3.2 Estructura Condicional.

Una estructura condicional es la que realiza un conjunto u otro de instrucciones dependiendo del cumplimiento o no de una determinada condición. Existen tres tipos de estructuras condicionales:

Estructura Condicional Simple.

Se evalúa la condición y, sólo si es cierta, se ejecuta el conjunto de acciones correspondiente (ver figura 1.8.1.3.2.1).

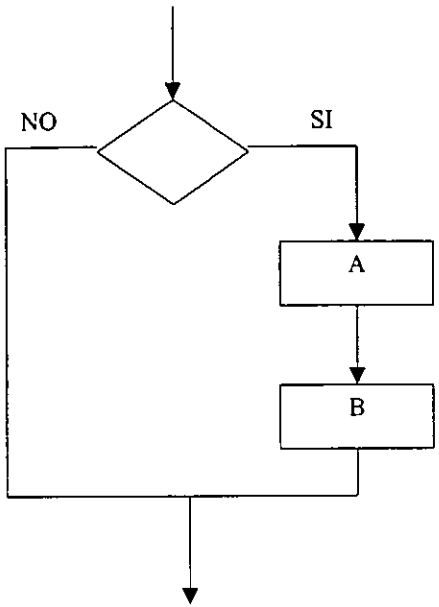


Figura 1.8.1.3.2.1 Estructura Condicional Simple

Estructura Condicional Doble.

En esta estructura. dependiendo de la certeza o falsedad de una condición se ejecutará un bloque de instrucciones u otro (ver figura 1.8.1.3.2.2).

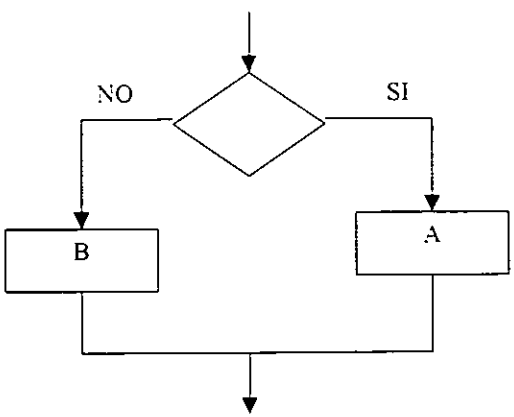


Figura 1.8.1.3.2.2 Estructura Condicional Doble

Estructura Condicional Múltiple.

Dependiendo del valor natural que tome la variable numérica implicada en la condición que vaya a evaluarse, se ejecutará una de las *n* acciones correspondientes (ver figura 1.8.1.3.2.3).

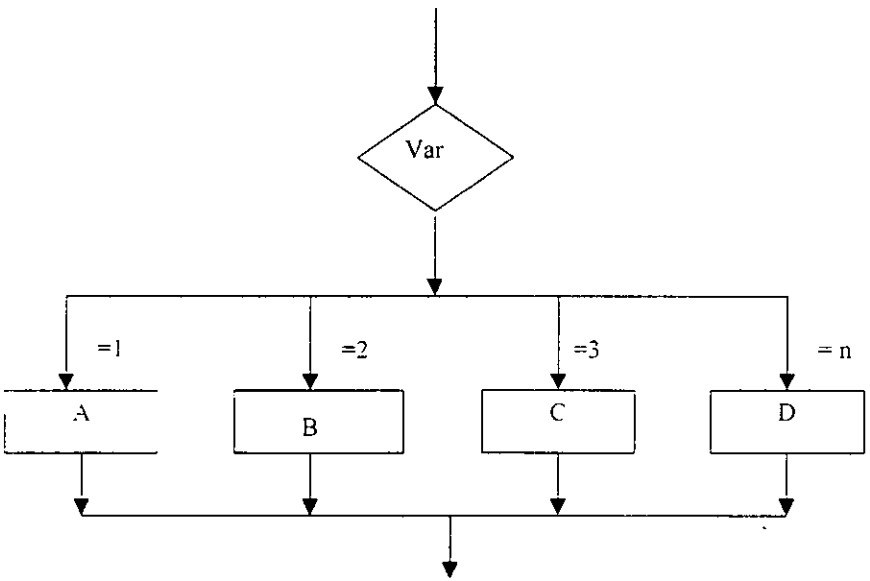


Figura 1.8.1.3.2.3 Estructura Condicional Múltiple.

1.8.1.3.3 Estructura Repetitiva

Permite la repetición de un bloque de instrucciones dependiendo de la veracidad o falsedad de una condición. Existen tres tipos fundamentales de estructuras repetitivas:

Estructura Tipo Mientras.

El bloque de acciones se repetirá mientras que la condición sea cierta.

La condición se evalúa al comienzo de la estructura. Esto implica que el bloque de instrucciones puede no ejecutarse ninguna vez si la condición de salida es inicialmente falsa (ver figura 1.8.1.3.3.1).

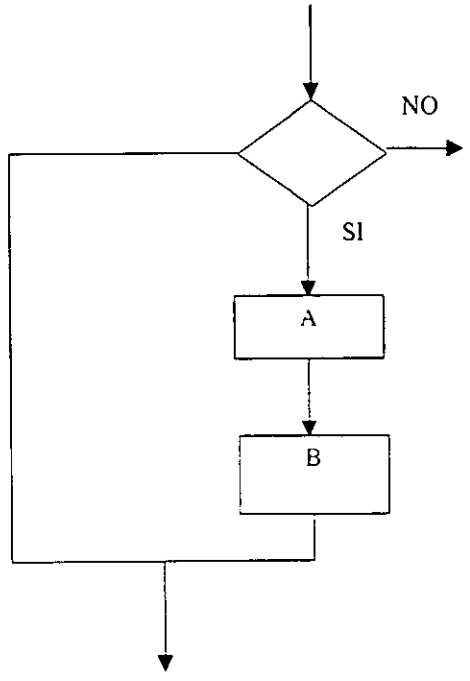


Figura 1.8.1.3.3.1 Estructura Tipo Mientras.

Estructura Tipo Hasta.

El bloque de acciones se repetirá hasta que la condición sea cierta.

Dicha condición se evalúa al final de la estructura. Esto implica que el bloque de instrucciones se ejecutará al menos una vez, aunque la condición de salida ya sea cierta al entrar en dicha estructura (ver figura 1.8.1.3.3.2).

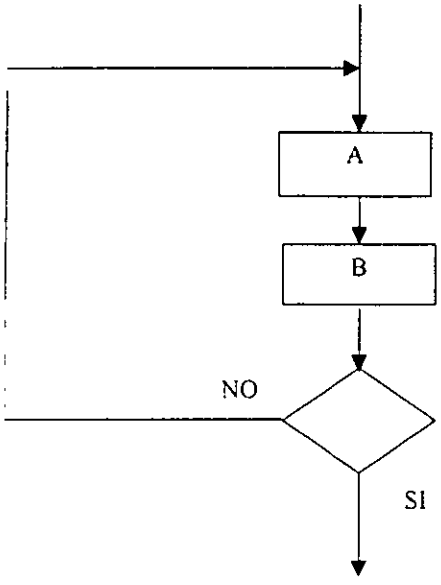


Figura 1.8.1.3.3.2 Estructura Tipo Hasta.

Estructura Tipo Para.

Si el número de iteraciones (repeticiones) es fijo o se conoce de antemano, se puede utilizar una estructura tipo Para o Desde en lugar de utilizar una estructura tipo Mientras. La estructura Para ejecuta las acciones del ciclo un número específico de veces y controla automáticamente el número de repeticiones.

Para dicho control, hay que definir dentro de la estructura el nombre de una variable numérica, su valor inicial, su valor final, y un incremento fijo. Es la propia estructura la que se encarga de inicializar la variable con el valor indicado, incrementarla en cada iteración y, cuando sobrepasa el valor final especificado, termina la repetición del ciclo. La evaluación de la condición de salida se hace al comienzo de cada iteración (ver figura 1.8.1.3.3.3).

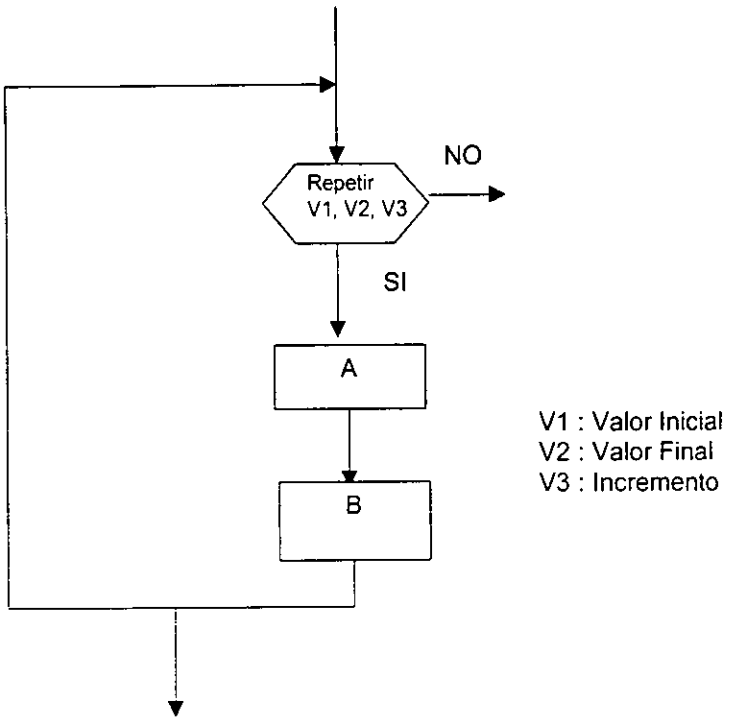


Figura 1.8.1.3.3.3 Estructura Tipo Para.

Toda estructura Para puede también realizarse con una estructura Mientras. La diferencia fundamental entre los dos tipos de estructura, radica en que en la estructura Mientras hay que realizar, mediante instrucciones, la inicialización de la variable que controla el ciclo, así como su incremento; y en la estructura Para esto se hace automáticamente. Por lo que se puede deducir que toda estructura Para se puede escribir como una estructura Mientras. Pero no todas las estructuras Mientras se pueden escribir como estructura Para; tan sólo aquéllas cuya condición de salida venga determinada por una variable numérica con incremento fijo.

Estructura Tipo Iterar.

La salida, en este tipo de estructuras, se realiza en cualquier punto del ciclo. Esta estructura sería la más general, ya que se puede transformar en una estructura tipo Mientras o tipo Hasta, con sólo situar la condición de salida al comienzo o al final del ciclo, respectivamente. La mayoría de los lenguajes de programación no admiten, este tipo de estructura, por lo que solo se menciona que existe.

1.8.2 La Estructura de Programas de Computadora.

Con el avance Tecnológico y la fabricación de computadoras muy rápidas y con precio asequible, resultó que lo que encarecía en gran medida la realización de un programa era el tiempo del programador. Lo que indujo a crear un modelo de programación que aportase las siguientes características:

- Adaptarse fácilmente a modificaciones.
- Propiciar una puesta a punto más rápida.
- Posibilitar una cómoda interpretación de los programas por diferentes programadores.

Así pues, la tendencia actual es tratar de conseguir, ante todo, programas simples y claros, que puedan ser mantenidos y actualizados fácilmente. Esta técnica de programación consiste en dividir la estructura del programa en partes bien diferenciadas lógicamente, llamadas módulos, que puedan ser analizadas y programadas por separado.

1.8.2.1 Diseño Modular.

Un módulo, está constituido por una o varias instrucciones físicamente contiguas y lógicamente encadenadas, las cuales se pueden referenciar mediante un nombre y pueden ser llamadas desde diferentes partes del programa. Cuando en un punto de un

programa se referencia un módulo, el programa le cede el control para que se ejecuten todas sus instrucciones. Finalizado el mismo, el control se devuelve al punto del programa desde donde se llamó al módulo, y se continua con la ejecución de la instrucción siguiente a la que realizó la llamada. Un módulo puede ser un programa, una función, un subprograma, etc. Si los módulos que componen un programa son parte integrante del mismo, se les llama rutinas, subrutinas, procedimientos o funciones. Si por el contrario, son programas independientes, reclamados desde otros programas, se les denominan subprogramas (ver figura 1.8.2.1.1).

Los sistemas más fáciles de modificar son aquellos que están formados por pequeños módulos manejables, cada uno de los cuales es hasta donde sea posible, independiente de los demás, de manera que se les pueda retirar, modificar y devolver sin afectar el resto del sistema.

No hay una norma fija para dividir un programa en módulos. Se hace a criterio del programador. Sin embargo, se deben seguir unas normas más o menos generalizadas:

- El sistema está compuesto de una jerarquía de módulos (cajas negras), cada módulo sólo puede tener un punto de entrada y otro de salida.
- Cada módulo es suficientemente pequeño para ser manejable, el módulo principal debe ser conciso, mostrando claramente los módulos que lo componen y su relación. Es decir, debe indicar la solución completa del problema.
- Los módulos deben tener la máxima independencia entre ellos, es decir que cada módulo puede ser modificado sin producir efectos de propagación de cambio.
- Un módulo debe representar por sí mismo una estructura lógica coherente y resolver una parte bien definida del problema, es decir debe desempeñar una sola función.

La programación modular aporta una serie de ventajas frente a la programación convencional:

- Los programas son más sencillos de escribir y depurar, pues se pueden hacer pruebas parciales con cada uno de sus módulos.
- La corrección o modificación de un módulo se hace más cómoda y, en general, no tiene por qué afectar al resto de los módulos.
- Un programa se puede ampliar fácilmente con sólo diseñar los nuevos módulos necesarios.
- Un mismo módulo escrito una sola vez puede ser referenciado desde varios puntos del programa, evitando la repetición de instrucciones ya escritas.

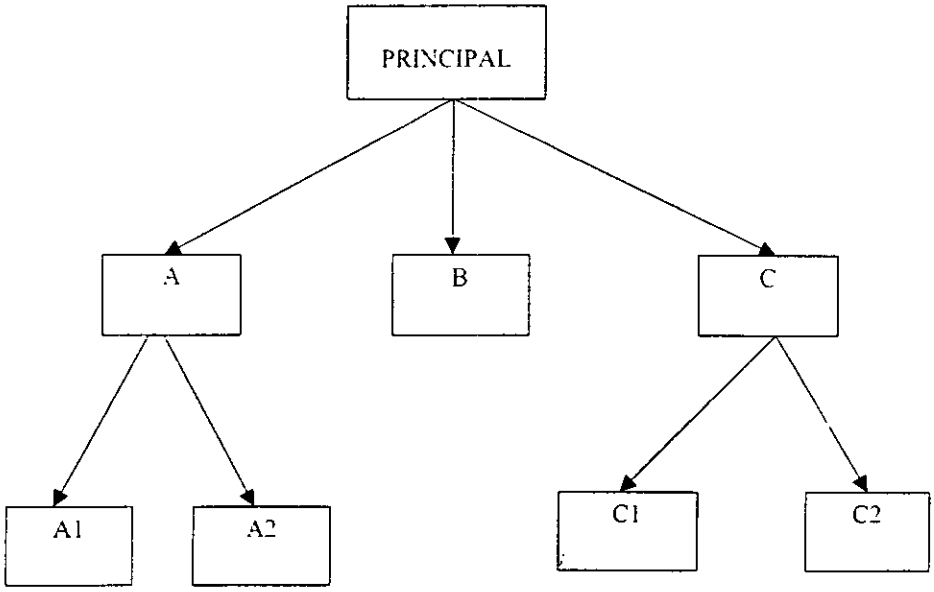


Figura 1.8.2.1.1 Diseño Modular.

El modelo anterior es semejante a una jerarquía militar u organizacional, el diseño estructurado proporciona cuatro conceptos muy útiles para la evaluación del diseño:

1.8.2.2 Acoplamiento entre Módulos.

El acoplamiento es una forma de medir la independencia entre módulos, debemos procurar que éstos sean lo más independientes posible, esto es, que la función de un módulo no dependa de lo que se haya hecho en otro, se desea minimizar el acoplamiento entre módulos. No es posible anularlo, ya que los módulos forman parte de un todo que es el sistema.

Tipos de Acoplamientos:

- A través de datos: es el más deseable (Parámetros). Se representa con una flecha con cola vacía.
- A través de control: No es lo ideal, pero en ocasiones resulta conveniente (Banderas como parámetros) se representa con una flecha con cola llena.
- Externo o Patológico: Es la relación de un módulo con otro en la cual uno de ellos hace referencia al anterior del otro. Nunca es aceptable (Goto).

1.8.2.3 Cohesión de un Módulo.

Cohesión es una medida de solidez interna del módulo. Se dice que el módulo es sólido cuando tiene una función que realizar y todas sus variables y actividades están encaminadas a dicho fin. Se desea Maximizar la cohesión de cada módulo. (Maximizar el grado de pertenencia de todas las partes al módulo) un módulo cohesivo automáticamente produce poco acoplamiento, se cumple que a mayor cohesión menor acoplamiento y a menor cohesión mayor acoplamiento.

Tipos de Cohesión:

- Accidental (la peor). Ejem. Módulos de 60 líneas

- Lógica (Cercana a la peor). Ejem. Rutina general de lectura.
- Temporal (mala). Ejem. Módulo de inicialización.
- De Procedimiento (Regular). Ejem. Módulo de todo el then de un if.
- De Comunicación (Regular a Buena). Subfunciones Agrupadas. Ejem: Calcula e imprime.
- Funcional (Ideal). Lleva a cabo solamente una función.

1.8.2.4 Alcance de Efecto.

Cuando en un módulo se toma una decisión, y dependiendo del resultado de esa decisión se invoca a uno o más módulos, al conjunto de módulos sobre los cuales el efecto de invocarlos por otro módulo se conoce con el nombre de alcance de efecto.

1.8.2.5 Alcance de Control.

Es el conjunto de módulos a los cuáles puede llamar directamente, más los módulos que a su vez estos módulos pueden llamar, etc.

Un módulo no debe tomar una decisión que afecte a módulos que no estén dentro de su alcance de control.

1.8.2.6 Criterios de Diseño.

1.8.2.6.1 Diseño TOP-DOWN (General-Particular).

Es un criterio de diseño que consiste en ir programando de lo general a lo particular, es decir, de los módulos principales a los módulos que son llamados por ellos. Gráficamente sería programar de los módulos de arriba a los módulos de abajo, recorriendo una por una las ramas del diagrama modular.

1.8.2.6.2 Diseño BOTTOM-UP (Particular-General).

A diferencia del anterior, la metodología de diseño bottom up, implica comenzar de lo particular a lo general. Se deben ir diseñando y programando los módulos por separado y posteriormente, ir agrupándolos hasta conseguir la integración total del sistema.

El uso de una filosofía de diseño y programación es muy recomendable, ya que permite una disciplina en la elaboración del sistema, de tal manera que no se programará ningún módulo de otra rama del Diagrama Modular, hasta que los módulos de la rama en cuestión estén concluidos y trabajando perfectamente.

1.8.3 Estructura y Procedimiento.

1.8.3.1 Estructura.

La estructura general de un programa puede considerarse como un procedimiento principal dividido en dos partes perfectamente diferenciadas:

- Un *"encabezamiento"*.
- Un *"bloque"*.

El encabezamiento se compone del nombre del programa y una lista de parámetros encerrados entre paréntesis. Donde uno es un parámetro de entrada y el otro es de salida de información.

El bloque consta de seis secciones donde todas, excepto la última pueden ser vacías.

- Declaración de Etiquetas.
- Definición de Constantes.
- Definición de tipos.
- Declaración de variables.

- Declaración de procedimientos y funciones.
- Cuerpo del Programa encerrado entre un *Begin* y *End*.

1.8.3.2 Procedimiento.

El procedimiento para elaborar un programa, debe contar con una metodología bien definida que considere el ciclo de vida del software, es decir una secuencia de fases a desarrollar en el proceso de programación. Las fases típicas son:

Planeación del software y Definición de Requisitos (Análisis).

Se establecen y especifican los requerimientos del software. Se refiere al período durante el cual se especifican las características funcionales y detalles operacionales. La entrada a esta fase son las necesidades declaradas para el software y la salida de la misma es un documento de requerimientos. Precede al diseño y especifica lo que debe hacer sin esperar cómo hacerlo.

Diseño.

El proceso de diseño comienza tan pronto como se especifican los requisitos. El diseño es una fase predominantemente creativa. Una parte importante del proceso de diseño del software es la descomposición del sistema de software completo en un conjunto de subsistemas. Cada subsistema se descompone posteriormente en una serie de programas y procedimientos más pequeños. Otra parte del proceso de diseño es determinar la representación interna de los datos. La entrada a esta fase es un documento de requerimientos (depurado y validado); la salida es un diseño expresado en alguna herramienta adecuada (por ejemplo pseudocódigo).

Implementación.

Una vez que el diseño se ha terminado, este se codifica, es decir, se implementa como un programa en un lenguaje de programación específico. En esta fase pueden intervenir los mismos programadores de fases anteriores o nuevos. Por esta causa, es importante que el diseño del software sea cuidadosamente documentado en un conjunto de diagramas de estructuras y pseudocódigo esencialmente. Una vez que el programa ha sido codificado debe ejecutarse y depurarse. Cuando no quedan errores se habrá terminado la fase de depuración y codificación, y es momento de probar el producto desarrollado para el sistema.

Prueba.

La prueba de un programa debe ser un proceso riguroso, que normalmente se ejecutará por un grupo distinto de los programadores originales; en esta fase deben estar implicados los usuarios del software. Es importante identificar todos los errores temporalmente, ya que el software que controla un proceso delicado debe estar libre de errores antes de su primera utilización.

Funcionamiento y Mantenimiento.

El software debe poder correrse eficazmente durante el periodo de vida útil del mismo, incluso en entornos cambiantes. El proceso de actualizar o mantener un programa consiste en corregir los nuevos errores que se produzcan o incorporar los cambios necesarios.

El costo total de un proyecto es una función del tiempo implicado y del número de personas que trabajan en el proyecto a lo largo de su ciclo de vida completo. La división del ciclo de vida de los sistemas en fases constituye una primera estimación o análisis de costo.

Se ha observado que no todas las fases contribuyen de igual modo al costo total del proyecto. Así, por ejemplo, la fase de mantenimiento puede contribuir tanto al costo como todas las fases de desarrollo combinadas. El ingeniero de software debe tratar de reducir en la medida de lo posible, el costo total; para ello debe distribuir juiciosamente el tiempo empleado entre todas las fases.

Los Equipos de Programación.

En la actualidad es difícil y raro que un gran proyecto de software sea implementado por un solo programador. Normalmente, un proyecto grande se asigna a un equipo de programadores, que por anticipado deben coordinar toda la organización global del proyecto.

Cada miembro del equipo es responsable de un conjunto de procedimientos, algunos de los cuales pueden ser utilizados por otros miembros del equipo. Cada uno de estos miembros deberá proporcionar a los otros las especificaciones de cada procedimiento, condiciones y su lista de parámetros formales; es decir, la información que un potencial usuario del procedimiento necesita conocer para poder ser llamado.

Es misión del equipo de programadores crear librerías de procedimientos, que posteriormente pueden ser utilizadas en otras aplicaciones. Una condición importante debe cumplir los procedimientos: estar comprobados y ahorro de tiempo/memoria.

1.9 CARACTERÍSTICAS, VENTAJAS Y DESVENTAJAS DE BORLAND PASCAL WINDOWS 7.0.

1.9.1 Antecedentes.

PASCAL fue concebido por su creador Niklaus Wirth a fines de la década de los setentas, como un lenguaje para la enseñanza de técnicas de programación a estudiantes universitarios, pero con el correr de los tiempos se ha convertido en un estándar del mundo de la programación.

PASCAL es un lenguaje de propósito general para la resolución de aplicaciones de todo tipo: gestión, científicas, de ingeniería, etc. Entre sus características más sobresalientes se encuentran las siguientes:

- Lenguaje excelente para el aprendizaje de la programación.
- Lenguaje de propósito general.
- Lenguaje procedural.
- Lenguaje estructurado.
- Lenguaje recursivo.
- Tiene gran riqueza de tipos (tipos de datos simples y estructurados, así como datos definidos por el usuario).
- Gracias a su compilador produce programas ejecutables rápidos y eficientes.
- Facilidad para realizar programación modular debido a la posibilidad de diseñar subprogramas o módulos del tipo procedimiento y función.

Unos años después de la aparición del PASCAL. Borland Internacional Inc. Creó un entorno con el que se podía programar en Pascal estándar y que además, añadía por su cuenta instrucciones nuevas y potentes al Pascal conocido hasta entonces. Incluso se podía escribir el programa y compilarlo sin necesidad de salir del entorno, puesto que constaba de un editor para escribir los programas y de un compilador muy rápido, en el que aparecían los posibles errores cometidos por el programador durante la

escritura de su programa al tiempo de compilarlo. A este nuevo entorno, denominado así por contener, como ya se ha dicho el editor y el compilador integrados en el mismo programa, se le dio el nombre de TURBO PASCAL. Este ha llegado a ser considerado un lenguaje de programación casi independiente del Pascal Estándar aún siendo totalmente compatible, los programas en Pascal Estándar funcionan perfectamente y más rápido aún que con un compilador y enlazador (Linker).

Además del entorno de programación (editor/compilador), nuevas utilerías. Entre las características con que cuenta TURBO PASCAL pueden citarse las siguientes:

- Un compilador eficiente.
- Biblioteca de utilerías.
- Depurador.
- Utiliza poco espacio de memoria.
- Compilación independiente (se pueden compilar por separado varias unidades de un mismo programa).
- Modularidad.
- Etc.

Con el tiempo nacieron nuevas técnicas y estructuras de programación, por lo cual, han ido apareciendo en el mercado nuevas versiones cada vez más completas de Turbo Pascal. En la versión 5.5 se añadió una nueva técnica para programación denominada orientada a objetos, más complicada pero también más potente y gráfica, que apareció como un nuevo método de programar según la actual ingeniería de Software.

En la reciente versión 6 se ha añadido una nueva reordenación de los Menús Pull-Down y de las ventanas interactivas del entorno Turbo Pascal que se utilizaban hasta ahora, y un paquete de librerías, denominadas Turbo Vision, que permiten al usuario crear y utilizar en sus programas menús Pull-Down y/o de otra clase, ventanas de texto interactivas, otras herramientas gráficas que son utilizadas por el mismo entorno de Turbo Pascal, y la posible utilización del ratón en general.

En la Versión de Turbo Pascal para Windows 7.0 ha desaparecido Turbo Vision y es sustituido por otra librería denominada ObjectWindows.

Turbo Pascal Para Windows.

Como se mencionó anteriormente, en la versión para Windows, desaparece Turbo Vision, que no podía funcionar con Windows, siendo sustituido por otra librería denominada ObjectWindows, que se encarga de la creación en Windows de la tarea que fue destinada a Turbo Vision. Con Turbo Pascal para Windows podemos crear aplicaciones que funcionarán en Windows con eficiencia, si bien no es un trabajo tan sencillo como realizar aplicaciones para DOS.

El entorno para Windows fue creado en principio para aquellos usuarios que no conocían bien el funcionamiento de un sistema informático, pues es más fácil captar las posibles acciones a realizar por la computadora si éstas están presentadas en forma gráfica. Windows ha terminado por ser un potente entorno capaz de utilizar incluso multitarea.

1.9.2 CARACTERISTICAS DE TURBO PASCAL.

A continuación daremos una descripción general de las sintaxis más simples de Pascal, así como sus variantes en Turbo Pascal, como sus distintas estructuras que va de la programación más sencilla hasta la avanzada, siguiendo las normas de la programación estructurada y modular.

1.9.2.1 CABECERAS.

Todo programa en Turbo Pascal, ya sea principal o un simple módulo de un programa, una unidad, etc., ha de tener una cabecera; un título, por decirlo de otra forma.

Las cabeceras en un programa principal no son obligatorias, sin embargo en las unidades, sí lo son, por lo que es bueno acostumbrarse a utilizar cabeceras en cualquier tipo de programa. Las cabeceras de los programas en Turbo Pascal llevan el siguiente formato:

Program <Nombre_del_Programa> (Entrada,Salida);

Program: Este identificador es el indicativo de que la línea actual es una cabecera.
<Nombre_del_Programa>: Nombre que deseamos darle a nuestro programa, compuesto por letras, números y símbolo de subraya.
(Entrada,: Indica cuál será el archivo de datos que se utilizarán como entrada para el programa. (No obligatoria para Turbo Pascal).
Salida):: Indica cuál será el archivo de datos que se utilizarán como salida del programa.

Si no se va a emplear ningún archivo, la entrada ha de ser *Input,* que indica que dicha entrada será el teclado de la computadora, y la salida será *Output*, que indica que la salida será la pantalla de la computadora.

1.9.2.2 Puntos y Comas.

Se escribe punto y coma al final de la mayoría de las líneas de un programa. Se puede exceptuar el punto y coma en las instrucciones que precedan una sentencia *END* o una sentencia *UNTIL*. Tampoco se escribe punto y coma al final de las siguientes instrucciones. Las marcadas con un asterisco pertenecen exclusivamente a Turbo Pascal.

- *Begin,Case, Const, For. If, Implementation* , Interface*, Repeat. Type, Var, While, With.*

1.9.2.3 Begin y End.

Estas instrucciones marcan el principio y el fin de un bloque de sentencias. O bien principio y fin de un programa, si éste es el último End del programa se agregara un punto.

1.9.2.4 Comentarios (Documentación Interna).

Sirven para escribir comentarios que faciliten la comprensión de los módulos del programa o modificaciones posteriores. Los comentarios en Turbo Pascal se escriben entre los símbolos *(** y ***)*.

1.9.2.5 Operadores Aritméticos.

Los operadores aritméticos son: Suma +, resta -. multiplicación *. división entera *Div*, división real /. módulo o residuo *Mod*.

Prioridades y Paréntesis:

El sistema evalúa la fórmula de izquierda a derecha y considerando una serie de prioridades. Las prioridades de los operadores aritméticos son: los paréntesis son los que tienen la mayor prioridad, le sigue la división y multiplicación y por último la suma y la resta que tienen la misma prioridad.

1.9.2.6 Constantes y Variables.

Los datos Constantes tienen un valor único en todo el programa, se declaran entre la cabecera *(Program)* y la declaración de variables *(Var)*, después del identificador *Const*.

Las variables son el otro tipo de datos que podemos utilizar en un programa, cuyo valor puede variar como su nombre lo indica. Las variables se declaran justo después de las

constantes, abajo del identificador *Var.* Las hay de varios tipos: Tipo lógico (Valor true y false), tipos numéricos (Valor entero y real).

Los tipos para Turbo Pascal son:

- Tipo Byte: Valor oscila ente 0 y 255.
- Integer: Su rango está entre -MAXINT y MAXINT (Dependiendo de la máquina).
- LongInt: Su rango es mucho mayor desde el número –2.147.483.648 al +2.147.483.647.
- ShortInt: Son enteros cortos, rango entre –127 y 128.
- Real: Depende de la máquina pero aproximadamente oscila entre 2.9E-39 y 1.7E+38. Separando la parte decimal con un punto.
- Word: Tipo Byte largo, oscila entre 0 y 65.535 son enteros.
- Double: Reales que oscilan entre 5.0E-324 y 1.7E+308. Es necesario tener coprocesador matemático y desactivar el modo emulación.
- Extended: Su enorme rango oscila entre 3.4E-4932 y 1.1E+4932. Con coprocesador matemático.
- Single: Rango oscila entre 1.5E-45 y 3.4E+38.
- Comp: Oscila entre –9.2E18 a 9.2E18.
- Char: Dato que sólo admite un carácter.
- String: Dato que admite más de un carácter, y máximo 255. Sus datos van entre comillas.
- Datos tipo Enumerados y Subrango.

1.9.2.7 Asignación e Identificadores.

La asignación de datos en Pascal se lleva a cabo cuando se le da valor a una variable, mediante el símbolo ":=". Un identificador es el nombre que demos a una variable, a una constante, a un procedimiento, función o programa. Estos sólo pueden contener las letras del alfabeto, puede contener números, sin empezar con ellos. El único símbolo permitido es la subraya.

1.9.2.8 Escritura de Datos (Write y Writeln).

Palabras reservadas para escritura de variables y letreros, y salto de línea.

Gotoxy: Sirve para situar los datos de *Write* o *Writeln* en un espacio determinado de la ventana de trabajo, se emplea en compañía de la librería *WinCrt* (Unidades de Turbo Pascal) y detrás la palabra Uses seguida de las unidades a utilizar en el programa.
Formato: *Gotoxy(<columna>,<fila>);*

1.9.2.9 Lectura de Datos (Read y Readln).

La sentencia *Read* es básicamente otra instrucción para asignar datos o variables, pero la asignación de valor se lleva acabo durante la ejecución del programa, no durante su escritura.

Formato: *Read(<Identificador>);*
Readln(<Identificadro>);

La diferencia entre *Read* y *Readln* es que *Readln* produce un salto de línea una vez que hemos leído su dato.

1.9.2.10 Estructuras Selectivas.

Las estructuras selectivas permiten ejecutar una serie de instrucciones u otras, dependiendo de una condición que previamente ha sido escrita por el programador.

If Then [Else]

Estructura selectiva simple, Si ocurre tal cosa, hacer tal otra. Su formato es:

If <condición> then
<sentencia o bloque de sentencias>

[Else

<sentencia o bloque de sentencias>]

Case of end.

Esta estructura selectiva nos permite tener opción de realizar una de varias cosas si se cumple la condición.

1.9.2.11 Estructuras Repetitivas.

Las estructuras que permiten repetir varias veces un bloque de instrucciones se denominan estructuras repetitivas.

For to/Down to Do

Permite repetir instrucciones un determinado número de veces, ya que esta estructura delimita de forma precisa el número de iteraciones que realizará el bucle.

For <variable>:= <dato inicial> to/down to <dato final> Do

<sentencia o bloque de sentencias>;

Repeat Until

Repite las instrucciones contenidas en su interior un número indeterminado de veces, dependiendo de una condición, de forma que el ciclo se repite hasta que una de las instrucciones de su interior modifica la condición que permite finalizar el ciclo.

Formato: *Repeat*

<Sentencia o bloque de sentencias>

Until <condición>;

While Do

Este ciclo repite las instrucciones contenidas en él mientras se cumple una condición lógica al principio, su formato:

While <condición> Do
<sentencia o bloque de sentencias>

Tipos de Datos definidos por el Usuario.

Para utilizar este tipo de datos definidos por el usuario usamos la instrucción *TYPE*

1.9.2.12 Arreglos.

Turbo Pascal cuenta con estructuras de datos dinámicas y estáticas. Las estructuras de datos estáticas son aquellas en las que se asigna una cantidad fija de memoria cuando se declaran.

Las estructuras de datos dinámicas no necesitan especificar su tamaño y pueden crecer o disminuir en tiempo de ejecución. Entre las estructuras estáticas una de las más conocidas y utilizadas es el *"array"* (Arreglos). Existen *arrays* de una, de dos y de más dimensiones. Todos los elementos de un array deben ser exactamente del mismo tipo. así que no pueden mezclarse dentro de un array enteros y caracteres, ni ninguna otra combinación. Su formato sería:

<Identificador> : Array [<Rango>] of <Tipo>;

1.9.2.13 Registros.

En muchas ocasiones se necesita procesar elementos que están relacionados entre sí, pero que no son del mismo tipo, para ello Turbo Pascal cuenta con una estructura denominada *"registro"*.

Un registro es un tipo de datos estructurado que consta de un conjunto de elementos que pueden ser del mismo tipo o de tipos diferentes, los componentes de un registro se denominan *"campos"*. Se puede acceder a cada campo de un registro directamente utilizando un selector de campo. En general los registros se utilizan agrupados en conjuntos conocidos como *"arreglo de registros"*.

1.9.2.14 Apuntadores.

Los apuntadores forman estructuras dinámicas. Como su nombre lo indica apuntan a un dato que están almacenados en celdas con cierto número o dirección. Para indicar que una variable va a ser un apuntador utilizamos una flechita "^" que se antepone al tipo de datos que vamos a apuntar. Después en el programa hay que crear otra variable, asignarle el lugar donde vamos a apuntar.

Existen tres operaciones básicas a realizar con un apuntador. La primera es definir el apuntador con la instrucción *New*.

New(<Apuntador>);

<Apuntador> es la variable que nos va a servir de apuntador.

La segunda operación consiste en asignarle un dato al apuntador para que le apunte o inicializar a cero, o bien que apunte a *NADA*.

Apuntador:=NIL;

Para que apunte a un valor: *Apuntador^:= dato:*

Para liberar la memoria ocupada por un apuntador mediante *Dispose:*

Dispose(<apuntador>);

1.9.2.15 Procedimientos y Funciones.

Un procedimiento o una función es un trozo de programa que resuelve una tarea más o menos grande dentro el problema general. Y se puede acceder más de una vez sin necesidad de escribir tantas veces como se necesite.

Su formato es similar al de los programas. Para manejar los datos del programa principal al procedimiento se crearon los denominados Parámetros, y así poder transmitir los datos de un subprograma a otro.

Una Función trabaja de la misma forma que un procedimiento, con la diferencia de que devuelven como mínimo un dato. o resultado de cierto cálculo que se asigna al nombre de la función en cuestión.

1.9.2.16 Archivos

Los archivos son la única estructura de datos que puede almacenarse en memoria secundaria o externa. Los archivos son estructuras de datos dinámicas que pueden ser almacenados en disco para su uso posterior mediante la ejecución de programas adecuados. Existen archivos de programas, autoejecutables, de recursos. Los tradicionales son los que se clasifican por su tipo de acceso.

Archivos de acceso secuencial, en los que para acceder a un dato hay que pasar previamente por todos los que están almacenados antes que él. Los de acceso directo o aleatorio, en los cuales se accede a un dato directamente. Existen otros como los archivos de texto. Un archivo esta compuesto de registros y estos a su vez están divididos en campos.

Ejemplo:
```
Var
    Archivo : File of tipo;
```

Donde Tipo, es la clase de datos que contiene puede ser Integer, real, String, etc.

Y si deseamos que contenga registros definidos declaramos una variable del tipo archivo:

Ejemplo:

```
Type
    Ficha = Record
        Nombre : String[20];
        Sueldo  :  Real
    End;
Var
    Archivo: File of Ficha;
```

La forma de asignarle nombre a un archivo en Turbo Pascal es mediante la instrucción Assign: *Assign (<Variable de tipo File>. <Nombre>):*

Dada la estructura de los archivos secuenciales algunas instrucciones importantes son:

Reset(<Variable de tipo File>): Abre un archivo para lectura.

Rewrite(<Variable de tipo File<): Abre un archivo para escritura.

Write(<Variable de tipo File>.<registro>): Escribe un registro en el archivo.

Read(<Variable de tipo File>.<registro>): Lee un registro del archivo.

Para emplear un archivo que no tiene tipo simplemente declaramos:

Var Fichero : File;

Para el manejo de archivos aleatorios utilizamos la instrucción Seek que nos permite situarnos en un registro en particular:

Seek(<variable de tipo File>.<no. de registro>).

Por último, Turbo Pascal considera a los dispositivos externos como si fueran archivos de Entrada/Salida, y sirve para elaborar programas que utilizan estos dispositivos.

1.9.2.17 Unidades de Turbo Pascal (TPU).

TPU son las iniciales de Turbo Pascal Unit, por lo que a las librerías les denominaremos Unidades o TPU, que son funciones o procedimientos para llevar acabo alguna instrucción determinada:

Uses: Para poder cargar las librerías se utiliza la cláusula Uses, se escribe antes de la cabecera del programa, indicando la lista de librerías a utilizar.
Formato: *Uses <librería>,<librería>,...,<librería>;*

Para Turbo Pascal para Windows las librerías son: Strings, System, WinCrt, WinDOS, WinProcs, WinTypes, Wobjects.

1.9.2.18 Interface e Implementación.

La cabecera de TPU es UNIT, y se divide en dos partes, en la primera se escriben los nombres de los procedimientos y funciones, en la otra los contenidos de esos procedimientos. Las cláusulas son las que separan las dos partes de Unidad.

La interface es la conexión que hay entre la Unidad y cualquier programa que la reclame. La sección Implementación contiene el desarrollo de la unidad. La escritura de todas y cada una de las subrutinas que componen la TPU.

1.9.2.19 Librerías de enlace Dinámico (DLL).

Las Librerías de Enlace Dinámico (DLL) se compilan y se utilizan sólo cuando se necesitan, lo que permite compartir datos de archivos, objetos, registros, código y recursos (iconos, mapas de bits, menús, etc.), con otras aplicaciones cuando sea necesario.

1.9.2.20 Directivas del Compilador.

Las Directivas del compilador guían al compilador para que ejecute partes del programa o el programa entero de cierta forma. Nos permite observar cómo continuaría el programa si un error se produjese, y poder seguir ejecutando el programa.

La mayoría de las directivas funcionan con interruptores y se pueden conectar y desconectar a voluntad. Por ejemplo: *{$I} Entrada/Salida , {$B} Evaluación lógica, etc.*

1.9.2.21 Interrupciones del BIOS y del DOS.

ROM BIOS iniciales de Read Only Memory (Memoria de sólo lectura), Basic Input Output System (Sistema básico de Entrada/Salida). La memoria ROM contiene una serie de instrucciones que no son modificable por un programador, siendo BIOS una parte de la memoria ROM. La ROM tiene un pequeño programa IPL que sólo busca el sistema operativo en el disco cada que encendemos la computadora. El BIOS contiene una serie de rutinas que permiten llevar acabo alguna acción que en los lenguajes de programación no existen.

Por otro lado existen rutinas denominados Interrupciones del DOS, definidas por el fabricante del sistema operativo. Se les puso este nombre porque en determinado momento interrumpían la ejecución normal de un proceso.

Las interrupciones del BIOS no son diferentes. Detienen un proceso, comienzan a ejecutar su rutina correspondiente y cuando ha finalizado ordenan proseguir con la ejecución normal del programa.

Para DOS las interrupciones funcionan del mismo modo, pero como su número de interrupción es siempre el mismo, no es necesario escribirlo como parámetro.

1.9.3 Programación Orientada a Objetos.

La programación orientada a objetos (POO) es un gran paso en la programación. Permite realizar programas con datos inteligentes y es un paso más en los avances de la comunicación programador-computadora. Además, los datos quedan mucho más estructurados y presentables, es prácticamente impensable diseñar aplicaciones importantes sin utilizar los métodos de Turbo Pascal para Windows.

Además, con esta programación se ha creado el concepto de herencia de los datos, de forma que un dato hijo puede heredar de su ancestro basándose en su estructura.

Este tipo de programación nació a principio de los años sesenta. Sin embargo no se han empezado a encontrar aplicaciones hasta hace poco tiempo. La compañía Borland incluyó este tipo de programación en la versión 5.5 de Turbo Pascal. En la versión 6 desarrolló una librería de unidades (TPU) que, utilizando la programación orientada a objetos, permiten el diseño de menús y ventanas interactivas, incluso utilizando el ratón, llamadas Turbo Vision. En Turbo Pascal para Windows es necesario conocerla, ya que se utiliza prácticamente para cualquier aplicación potente para Windows y su programación orientada a objetos.

1.9.3.1 Objetos y Métodos.

Los objetos son la base de este tipo de programación. Siendo un objeto un registro (Record) creado con Type, pero que incluye, además de los campos, una serie de procedimientos y funciones que en esta técnica de programación son llamados métodos, y que se utilizan para el perfecto aprovechamiento de los campos de registro. Todo el trabajo lo llevan a cabo los objetos, que se encargan de pedir los datos, procesarlos y presentar los resultados.

El conjunto de campos y procedimientos incluidos en un objeto se denomina encapsulación, puesto que los datos y los procedimientos son como cápsulas que

contienen todo. Para crear la encapsulación, sustituimos el identificador Record por Object y además de los campos incluimos en el registro las cabeceras de los métodos (procedimientos y funciones) que va a contener el objeto: Ejemplo.

```
Type
    Objeto = Object
        Nombre : String;
        Apellidos: String;

        Procedure LeerDatos;
        Procedure Procesamiento;
        Procedure GrabarenArchivo;
        Procedure CerrayYSalir;
    End;
```

Una norma a seguir es que en el objeto los campos han de ser escritos antes de los procedimientos y funciones utilizadas por éste. El *type* no crea el objeto. hay que declarar una variable del tipo Objeto: *Var Var1 : Objeto;*

Para escribir los métodos o llamarlos desde el programa principal hemos de hacerlo en la misma forma que utilizábamos los registros, esto es, escribiendo el nombre de una variable declarada como el registro objeto, después el punto y por último el nombre concreto del procedimiento que vayamos a utilizar. Ejem. *Procedure Objeto.LeerDatos;*

1.9.3.2 Herencia.

La herencia de los objetos implica que unos objetos pueden utilizar la estructura de otros. Así, se dice que los nuevos objetos son descendientes de los que ya estaban creados, y que éstos últimos son ancestros de los nuevos. Se trata de que el objeto descendiente pueda utilizar los métodos del objeto ancestro.

1.9.3.3 Polimorfismo, Virtual y Constructor.

El Polimorfismo es un problema en el que tenemos más de una forma de responder a un mismo mensaje. Es decir, al llamar al método, cuyo nombre está repetido, existirían dos formas de responder a la llamada. Para evitar este problema, hemos de indicar el objeto dueño del método que deseemos utilizar en cada momento.

Otra forma de evitarlo es utilizar los llamados métodos virtuales, en los que se escriben los identificadores Virtual y Constructor en aquel método que tengan preferencia. Para su escritura se añade el identificador virtual en el método que tiene el nombre repetido y como cabecera del primer método, hemos de escribir el identificador constructor, en vez de procedure.

Este tipo de definición de métodos en objetos se denomina definición de métodos virtuales (aparentes), y se dice que son del tipo late binding (enlace tardío) puesto que no se traducen en tiempo de compilación, sino que se crean una Tablas de Métodos Virtuales (VMT) en memoria dinámica del sistema que contienen los datos sobre los métodos virtuales, que se traducen en tiempo de ejecución. Los métodos constructores son virtuales desde el punto de vista de que se utilizan como una especie de cabecera del programa.

1.9.3.4 Apuntadores con Objetos.

Al igual que en los registros definimos un objeto y le asignamos un tipo apuntador:

Formato:

```
Type
PunteroAObjeto = ^Objeto;
Objeto = Object
        Nombre : String;
        Teléfono: String[14];
        Procedure Inicio;
```

```
        Procedure Desarrollo;
        Procedure Fin;
        Procedure Proceso(Var Opción:Char);
End;
```

La diferencia con los apuntadores viene a la hora de inicializar el apuntador, añadiendo a *New* un nuevo parámetro, que es el método que inicializará los campos del registro, poniendo a cero todos los datos, incluyendo el apuntador, con una sola instrucción. Del mismo modo, la instrucción *Dispose* permite añadir un parámetro más que será otro método del objeto que cierre los campos del objeto.

1.9.4 Programación para Windows.

1.9.4.1 ObjectWindows.

La programación para Windows nos permite utilizar la potencia de ésta, el uso del ratón, de botones, de iconos, de mapas de bits, y una infinidad de elementos para poder incluso utilizar la tan traída y llevada multitarea gestionada para Windows.

ObjectWindows cambia radicalmente la filosofía de programación. Nos referimos a que programar con *Objectwindows*, varía la forma en que son utilizadas las estructuras selectivas o las estructuras repetitivas, así como las variables y constantes. En el Pascal tradicional la secuencia del programa era consecutiva, en Turbo Pascal para Windows un programa comienza en la primera línea y ya no se sabe si después será ejecutada la siguiente, porque depende del usuario que esté manejando el programa. Si usa un menú, habrá que ejecutar la parte del programa dedicada a manejar dicho menú. O bien puede dirigirse al primer icono o botón que encuentre y pulsarlo, y pondrá en funcionamiento la rutina que permite manejar correctamente el botón seleccionado. Puede decirse que lo que hace un programa en Windows siempre está en el aire en espera de que una opción de una ventana concreta sea utilizada. Por lo que es sencillo utilizar la programación orientada a objetos, ya que las instrucciones Virtual.

Constructor y Destructor permiten dejar una serie de rutinas esperando en la tabla de métodos virtuales (VMT) a que el usuario seleccione una opción, y de ese modo, las ejecute.

1.9.4.2 Estructura de un Programa con ObjectWindows.

1.9.4.2.1 Creación de una Ventana.

Para escribir una aplicación que siga la filosofía de *ObjectWindows* primero hay que definir una ventana sobre la que se ejecutará el programa principal. Hemos de utilizar la librería *Wobjects* que contiene los objetos y métodos que permiten abrir la ventana *(Tapplication)*. Un programa Object Windows tiene aproximadamente la siguiente estructura:

```
Program <Nombre del programa> [(Input,Output)];
[{SR <archivo de recursos>}]
Uses  Wobjects [.
      <Otras librería necesarias>];
Type
      Tipo Programa = Object(Tapplication)
                  Procedure InitMainWindow; Virtual;
            End;
      Procedure TipoPrograma.InitMainWindow;
      Begin
            ..
            ..
      End;
      Var
            Variable : TipoPrograma;
      Begin
            Variable.Init('<nombre para el programa>');
```

```
        Variable.Run;
        Variable.Done
End.
```

Primero indicamos si vamos a utilizar un archivo de recursos que contenga algún icono, botón, etc. Luego debemos declarar la librería *WinProcs*. Imprescindiblemente utilizaremos la sección *Type* para crear un objeto que será encargado de definir la ventana. Además declaramos una variable del tipo de dicho objeto para inicializarla con *Init*, ejecutarlo con *Run* y terminarla con *Done*.

La estructura de cualquier programa con Object Windows necesita el procedimiento definido por Borland llamado *InitMainWindow*, el programa principal constará de sólo tres instrucciones que se encargarán de inicializar la aplicación *(Init)*, de ejecutarla *(Run)* y de Cerrarla (*Done*). Para que funcione la ventana añadiremos el relleno del procedimiento *InitMainWindow:*

```
Procedure Aplicacion.InitMainWindow;
Begin
        MainWindow:=New(Pwindow.Init(NIL,'Titulo')
End;
```

La variable *MainWindow* se ha de inicializar con *NEW*, hacemos que la variable apunte a *Pwindow* seguida de *Init*, que a su vez apunta a Nada y lleva detrás el título que poseerá la ventana al ejecutar el programa. Usando Constructor podremos inicializar la ventana y manejar tanto el tamaño como el titulo.

Existe la posibilidad de utilizar ventanas de diálogo que previamente hemos diseñado con *Whitewater Resource Toolkit* (Herramienta para diseñar archivos de recursos). Si utilizamos una ventana ya definida, hemos de cargar el archivo de recursos que contiene las ventanas interactivas a utilizar incluyendolas con la directiva *{$R*.

A continuación podemos observar una ventana WRT (ver figura 1.9.4.2.1.1), la ventanita que observamos dentro de la ventana grande cuyo título marcado como *Untitled Dialog* es el prototipo de ventana que extenderemos o reduciremos, así como también nos será posible rellenarlo de múltiples objetos, iconos, texto,etc.

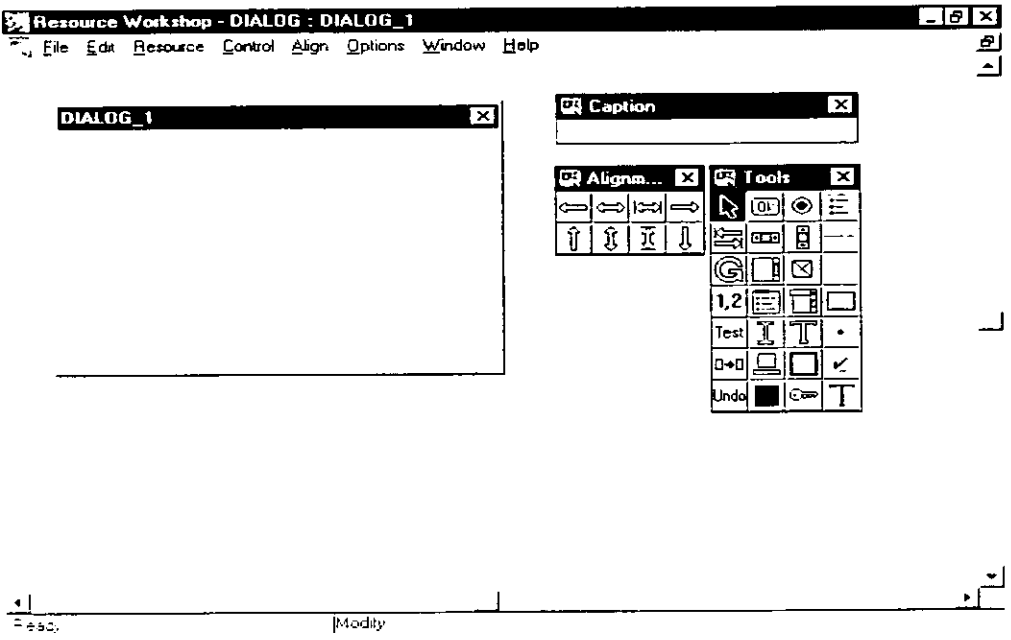


Figura 1.9.4.2.1.1 Creación de una ventana utilizando WRT.

1.9.4.2.2 Cerrar la Ventana.

Para cerrar la Ventana Principal hay que confirmar su salida, para ello se ha definido el método *CanClose*, que permite abrir una ventana de diálogo para confirmar la salida, antes de llevarla a cabo. *Canclose* es una función lógica, si es cierta, permitirá cerrar una ventana, de lo contrario la aplicación seguirá activa, es un método Virtual, que se define en el objeto descendiente de *TWIndow*.

1.9.4.2.3 Escritura de Datos.

La instrucción que ha de utilizarse para escribir texto en la pantalla es *TextOut*. *TextOut* necesita ciertos parámetros implícitos de *ObjectWindow:*

Formato: *TextOut(Controlador: HDC; X, U: Integer: Cadena: Pchar: Recuento:Integer);*

Donde:

- *Controlador* es una variable de tipo *HDC* (Tipo Thandle tipo Word), que contiene la ventana de la pantalla sobre la que vamos a dibujar el texto.
- *X* e *Y* son las coordenadas de la pantalla en las que comenzará a escribirse el texto.
- *Cadena* es la variable que contiene el texto a escribir.
- *Recuento* es el número de caracteres de la cadena.

1.9.4.2.4 Lectura de Datos con Ventanas de Diálogo.

Una forma de requerir datos del usuario para el manejo del programa. es ejecutar una ventana de diálogo que al estar ya definida hay que apuntarla irremediablemente. Esta ventana estándar recibe el nombre de *PinputDialog.* Son ventanas de Diálogo especiales estándar definidas por Turbo Pascal y orientadas a esta actividad. sólo pueden requerir datos de tipo *Pchar* o cadenas terminadas en *Nul.* Otra función de éste tipo es *StrCopy* que sirve para inicializar las variables de datos. de lo contrario aparecerían las ventanas con basura. Otro procedimiento *Paint* sirve también para realizar funciones de lectura, escritura o dibujos.

1.9.4.2.5 Visualización de Gráficos. Los Mapas de Bits.

Los gráficos y dibujos que podemos insertar en una ventana son mapas de bits. Un mapa de bits es un archivo que hemos grabado en un disco cuyo contenido es un gráfico o dibujo que tiene el formato de un mapa de bits.

Existen varios formatos estándar que usan diversas aplicaciones como *Deluxe Paint, Painbrush.* etc. El formato más estándar es el mapa de bits. *Whitewater Resource Toolkit y Resource Workshop* son aplicaciones que trabajan con este formato. Por lo que se puede dibujar un gráfico en cualquier aplicación y grabarlo en formato de mapa de bits y utilizarlo con Turbo Pascal para Windows (ver figura 1.9.4.2.5.1).

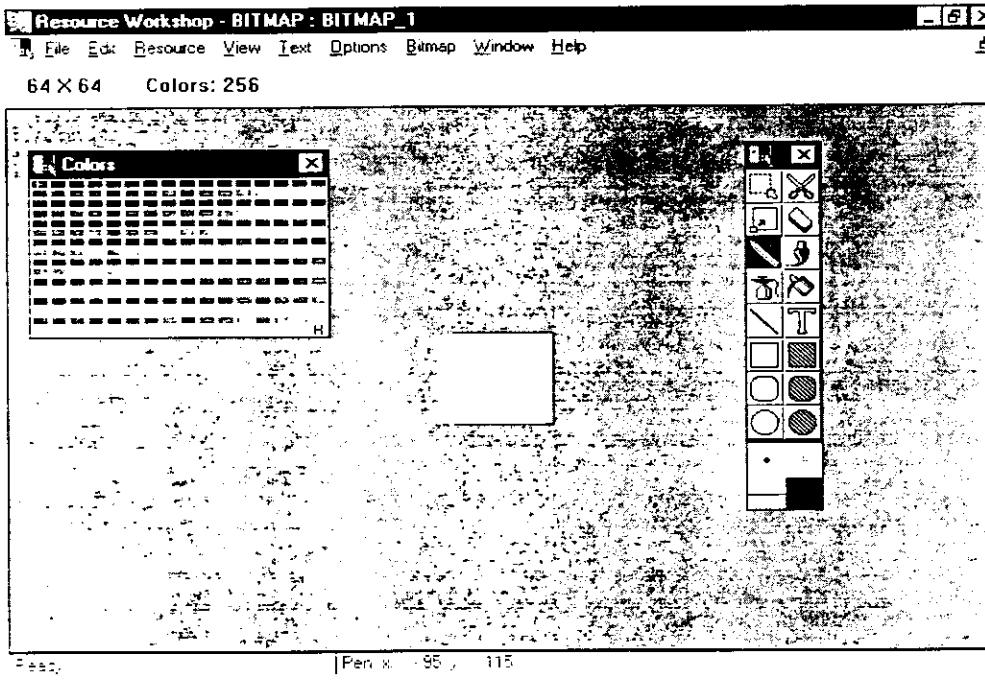


Figura 1.9.4.2.5.1 Visualización de Gráficos. Los Mapas de Bits.

En la pantalla anterior hemos presentando el editor con un dibujo sencillo para que se vea mejor cómo se diseñan los mapas de bits.

Para incluir un gráfico en Turbo Pascal, primero hay que indicar con el directivo *$R* el archivo de recursos que vamos a utilizar y que contiene los gráficos y dibujos a insertar en nuestras ventanas. Después, en el constructor de inicialización de la ventana hemos de añadir una nueva instrucción, que se encargará de recoger en una variable el

dibujo correspondiente. Esta instrucción es *LoadBitMap,* y para escribir realmente el gráfico en la ventana, se encarga la instrucción *BitBlt* (Abreviatura de *bitMapBuilt,* construir Mapa de Bits). Para el correcto funcionamiento del dibujo en la ventana, existen dos variables necesarias (*MemDC* y *Objeto*). Donde MemDC se encargará de preparar el terreno para el dibujo haciendo compatibles el controlador de la ventana y el gráfico a dibujar y Objeto indicará a *Bitblt* cuál de los gráficos ha de ser el que se dibuje, utilizando *SelectObject.*

1.9.4.2.6 Uso del Ratón y Cursores.

El uso del ratón en Windows no sólo es muy habitual, sino que sin él trabajar en Windows sería muy pesado. Algunos tipos de botones de algunas aplicaciones necesitan ratón para su funcionamiento.

En turbo Pascal para Windows, el ratón no es indispensable, pero sí muy útil. En cuanto el entorno IDE. Borland ha desarrollado procedimientos y funciones para la perfecta gestión del ratón en una aplicación Windows. Estos métodos son virtuales y. sólo funcionan cuando se detecta algún cambio en el estado del ratón, como es movimiento o la pulsación de uno de sus botones.

El método que se encarga de detectar si se ha pulsado el ratón es *WMLButtomDown (Left Buttom Down,* botón izquierdo pulsado). En realidad, este método se ejecuta cuando la constante *Wm_LbuttonDown* es activada, y ésta se activa cuando se pulsa el botón izquierdo del ratón y si ocurre en la ventana principal. se ejecutará el procedimiento *WMLButtonDown.*

Cursores

La flechita que utiliza el ratón para señalar en la pantalla. puede ser sustituida por otro dibujo llamado cursor. Con las herramientas WRT pueden diseñarse los cursores que uno desee.

La pantalla siguiente muestra el editor de cursores para el ratón, elaborando algún gráfico que usemos para ese fin (ver figura 1.9.4.2.6.1).

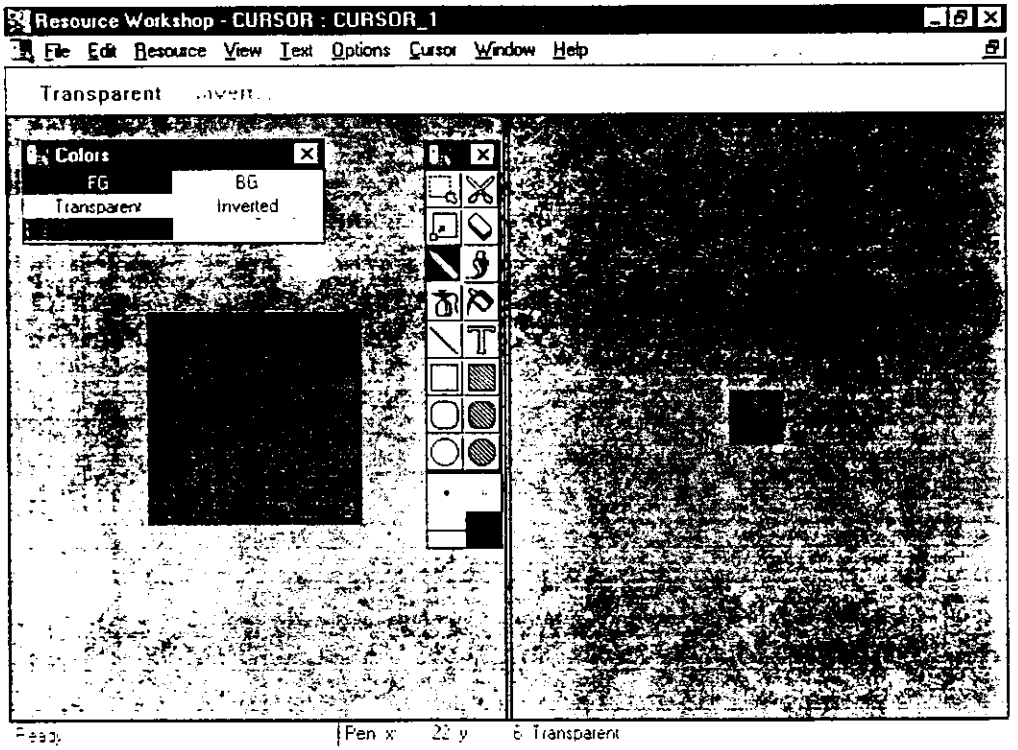


Figura 1.9.4.2.6.1 Uso del Ratón y Cursores.

1.9.4.2.7 Dibujo Programado, Líneas, Rectángulos, etc.

Al igual que en Turbo Pascal para DOS, existen varias instrucciones para dibujar en una ventana de la aplicación con Turbo Pascal para Windows. Existe un cursor gráfico en cada ventana creada por Turbo Pascal, que no es visible, aunque si es transladable. Para desplazar el cursor gráfico se emplea la instrucción:

MoveTo(DC,X,Y);

DC : Es el dispositivo controlador del contexto (ventana) en que se moverá el cursor.
X e Y: son coordenadas a las que el cursor será trasladado.

La siguiente instrucción *LineTo*, traza una línea desde las coordenadas actuales del cursor gráfico hasta una nueva posición indicada en la misma instrucción.

LineTo(DC,X,Y)

Para dibujar un rectángulo usamos la instrucción *Rectangle*, cuyo formato es:

Rectangle(DC,X1,Y1,X2,Y2):

1.9.4.2.8 Menús.

Una de las herramientas de *WRT (Whitewater Resource Toolkit)* más utilizada, es la creación de menús. Para incluir un menú en una ventana sólo hay que cargarlo. dentro de la estructura *With* incluyendo una línea más:

Menu:= LoadMenu(Hinstance,'Nombre');

Turbo Pascal para Windows genera unas pantallas donde se especifican las características que deseamos tenga el menú.

En la siguiente pantalla se representa las dos ventanas que permiten la creación y posterior modificación de menús para una aplicación (ver figura 1.9.4.2.8.1).

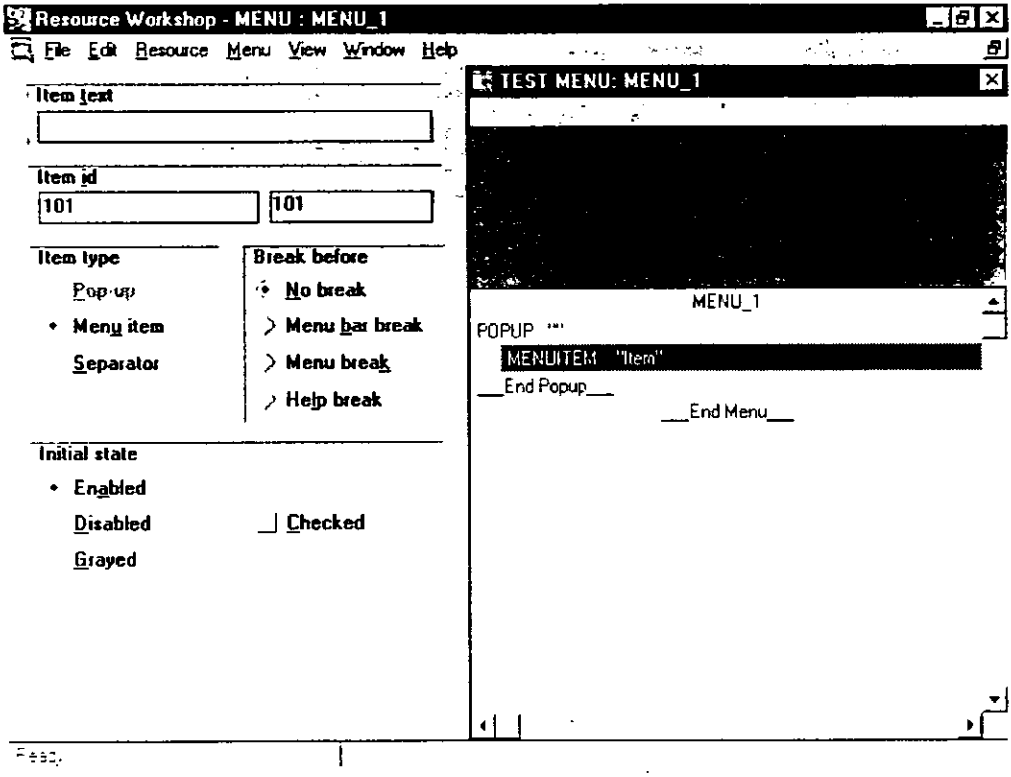


Figura 1.9.4.2.8.1 Menús.

La parte superior permite examinar los resultados de la creación o modificación del menú que se está diseñando en la inferior.

En dicha ventana podemos observar dos mensajes: *Item text* y *Value*. En la sección *Item text* podemos escribir los *items* que compondrán el menú en cuestión; es decir sus nombres correspondientes. En la sección *Value* se escriben los códigos numéricos de cada *item*, que empleará el programa Turbo Pascal para la selección correspondiente.

1.9.4.2.9 Iconos.

Al arrancar Windows nos encontramos con un sin número de estos iconos, cada uno da paso a una aplicación diferente (ver figura 1.9.4.2.9.1).

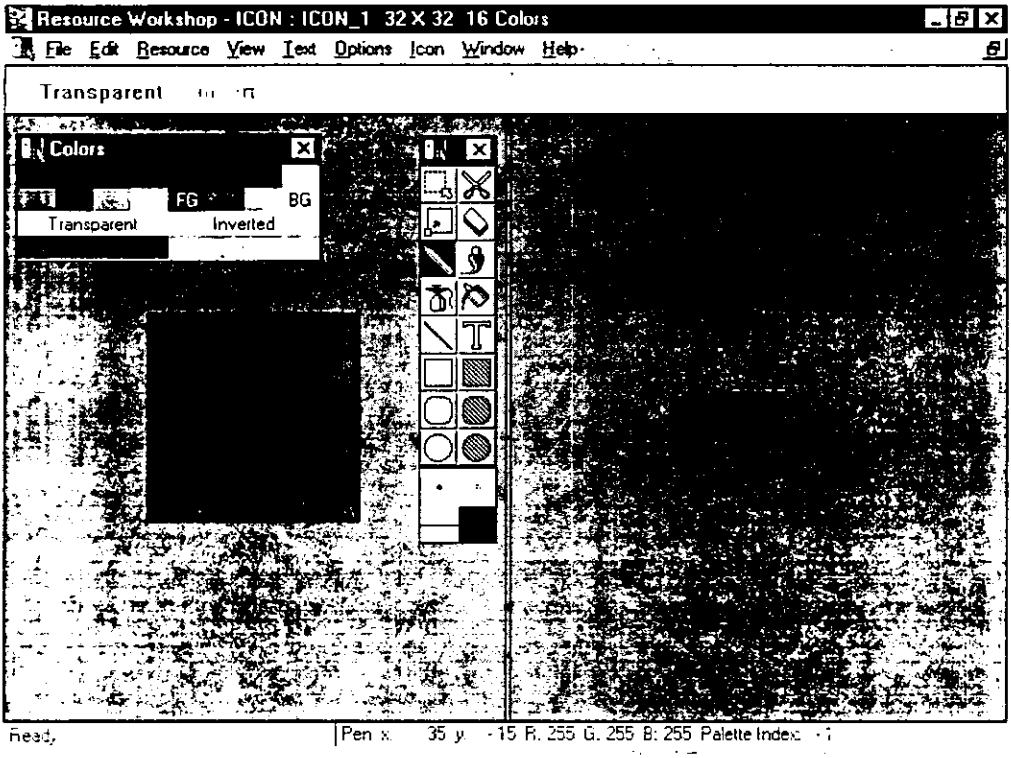


Figura 1.9.4.2.9.1 Iconos

Turbo Pascal nos proporciona una ventana para diseñar un icono. Una herramienta importante es la posibilidad de recortar ventanas de un icono para borrarlas, desplazarlas o copiarlas después con el botón marcado con una. Para borrarlas las teclas [Mayusculas]+[Supr], y para duplicarlas las teclas [Ctrl]+[Ins].

1.9.5 VENTAJAS Y DESVENTAJAS.

1.9.5.1 Ventajas.

Como se dijo anteriormente PASCAL en un lenguaje de propósito general y entre las ventajas a tomarse en consideración se encuentran las siguientes:

- Es un lenguaje procedural y estructurado, lo cual facilitaría realizar la programación de Sistemas de una forma modular.
- Es un lenguaje recursivo, lo cual podría ser de utilidad para la realización de búsquedas.
- Tiene gran riqueza de tipos.
- Produce programas ejecutables rápidos y eficientes, lo cual garantiza cierta autonomía e integridad a los sistemas.
- Cuenta con un ambiente editor fácil de manejar.
- Permite el manejo de estructuras de datos.
- Permite el manejo de cadenas de una forma moderadamente fácil.
- También permite el manejo de los dispositivos de entrada/salida.
- Con Turbo Pascal para Windows se emplea programación orientada a objetos con una gran ventaja ya que se compactan más los datos, quedan más presentables y se ahorra cierta cantidad de memoria y tiempo.
- Con el empleo de la programación orientada a objetos, el tamaño del programa principal disminuye considerablemente, dejando todo el trabajo a los objetos.
- Por estar empleando Windows, esto permite el uso de tareas compartidas, multitarea.

1.9.5.2 Desventajas.

- PASCAL es un lenguaje de propósito general y no está especialmente enfocado a la manipulación de las bases de datos.

- Si se usan estructuras de datos de tipo estático (arrays) se corre el riesgo de desperdiciar memoria o de que haga falta.
- En el empleo de estructuras de tipo dinámico (colas) la programación se vuelve bastante compleja.
- Se necesitan programar rutinas para manipular los dispositivos de Entrada/Salida.
- Es necesario dominar la teoría de apuntadores y la programación orientada a objetos para poder diseñar cómodamente con el nuevo sistema Turbo Pascal.
- Se necesitan conocimientos del ambiente Windows.
- Turbo Pascal para Windows usando la metodología orientada a objetos sólo es conveniente para programas grandes, para programas pequeños resultaría contraproducente.

1.10 CARACTERÍSTICAS, VENTAJAS Y DESVENTAJAS DE DELPHI 4.0

1.10.1 Introducción.

Dentro de la categoría de entornos de desarrollo visual para Windows existen actualmente diferentes productos, entre ellos Borland Delphi, de Inprise. La primera versión de Delphi apareció en el mercado hace aproximadamente cuatro años y supuso el lanzamiento de Borland en el área de este tipo de herramientas. Basándose en un compilador de indudable calidad, el de Borland Pascal, Delphi es capaz de generar aplicaciones de menor tamaño y mucho más rápidas que las que sea posible desarrollar con otros productos similares, en parte gracias a que el código que genera Delphi es directamente ejecutable, no tratándose de un pseudo-compilado o p-code que posteriormente tiene que ser interpretado en tiempos de ejecución.

La aparición de Delphi 2.0 incorporó muchas novedades al entorno, como la posibilidad de generar código de 32 bits, para Windows 95 y NT, nuevos componentes y herramientas para trabajo con bases de datos y unas posibilidades de conectividad importantes. A todo esto Delphi 3.0 añadió nuevas posibilidades, como la creación de controles ActiveX, servidores de Internet, etc.

Delphi 4.0, amplía aún más las capacidades de Delphi en el desarrollo para Windows 98 y NT. Además de ofrecer un entorno en el que la escritura de código es más fácil que nunca. Delphi cuenta con todas las características para crear aplicaciones con avanzadas interfaces de usuario, servicios locales y distribuidos y acceso de orígenes de datos.

1.10.1.1 Versiones de Delphi 4.0

Delphi es un producto único, basado en un entorno de desarrollo visual y un lenguaje orientado objetos. Sin embargo comercialmente existen tres versiones diferentes, que cuentan con diferentes elementos adicionales.

La versión básica, denominada *Standard*, incluye los elementos comunes a las tres versiones. Se compone del entorno y desarrollo visual Delphi 4.0, el Borland DataBase Engine de 32 de bits y todos los componentes del Delphi necesarios para la creación de programas en Windows 95/98 y NT.

El nivel intermedio de Delphi lo ocupa la versión *Proffesional*, que además de contar con los elementos ya mencionados también incorpora el servidor local de InterBase, el código fuente de los componentes Delphi, algunos controles ActiveX de ejemplo y algunos componentes adicionales para el trabajo con bases de datos y creación de aplicaciones Internet.

La versión superior de Delphi se le conoce como *Client/Server Suite*, además de incorporar todo lo descrito en los dos puntos anteriores, también se acompaña de una licencia de InterBase Server, un controlador de versiones de código fuente. controladores SQL para Oracle, Sybase, Informix, etc., y las herramientas SQL Explorer y SQL Monitor. También permite la creación de servidores Web y aplicaciones distribuidas usando DCOM (Distributed Component Object Model) , CORBA (Common Object Request Broker Architecture), etc. Como se puede observar, esta versión está especialmente enfocada al desarrollo de aplicaciones para trabajo en entornos cliente/servidor, facilitando incluso un servidor de bases de datos para Windows, lo cual puede facilitar mucho el trabajo.

1.10.1.2 Requerimientos de Delphi 4.0

El espacio en disco oscila entre los 40 megabytes, para la instalación mínima de la versión estándar, y casi 200 para una instalación completa de la versión Client/Server Suite. La instalación mínima, copiará al disco tan sólo lo necesario dejando el resto de los archivos en el CD, lo que tiene el inconveniente de que cada vez que necesitemos una de las herramientas no instaladas, el CD deberá encontrarse en la unidad lectora.

Los requerimientos de memoria de Delphi 4.0, son los propios de una herramienta de este tipo, siendo el mínimo de 16 megabytes. Este mínimo, sin embargo, es poco realista, ya que con esa cantidad de memoria los accesos a disco durante procesos como en la compilación serán continuos y el funcionamiento de Delphi resultará algo lento. En la práctica 32 megabytes de memoria es una configuración adecuada para el correcto funcionamiento de Delphi 4.0

El software necesario será Windows 95/98 o NT 4.0, en este último caso se debe tener instalado el Service Pack 3 o posterior. Si disponemos de Windows 3.1 también se puede optar por la versión 1.02 de Delphi, que tiene unos requerimientos muchos menores.

1.10.2 Características.

1.10.2.1 Elementos Iniciales.

Al ejecutar Delphi, en la pantalla inicial siempre aparecen los mismos elementos. Son precisamente lo que se utilizan con mayor frecuencia aunque, pueden existir varias ventanas más.

En la figura 1.10.2.1.1 puede verse el aspecto típico de Delphi cuando es cargado. Se pueden observar las siguientes secciones: Ventana principal, Paleta de componentes, Ficha o formulario, Inspector de objetos y Menú principal.

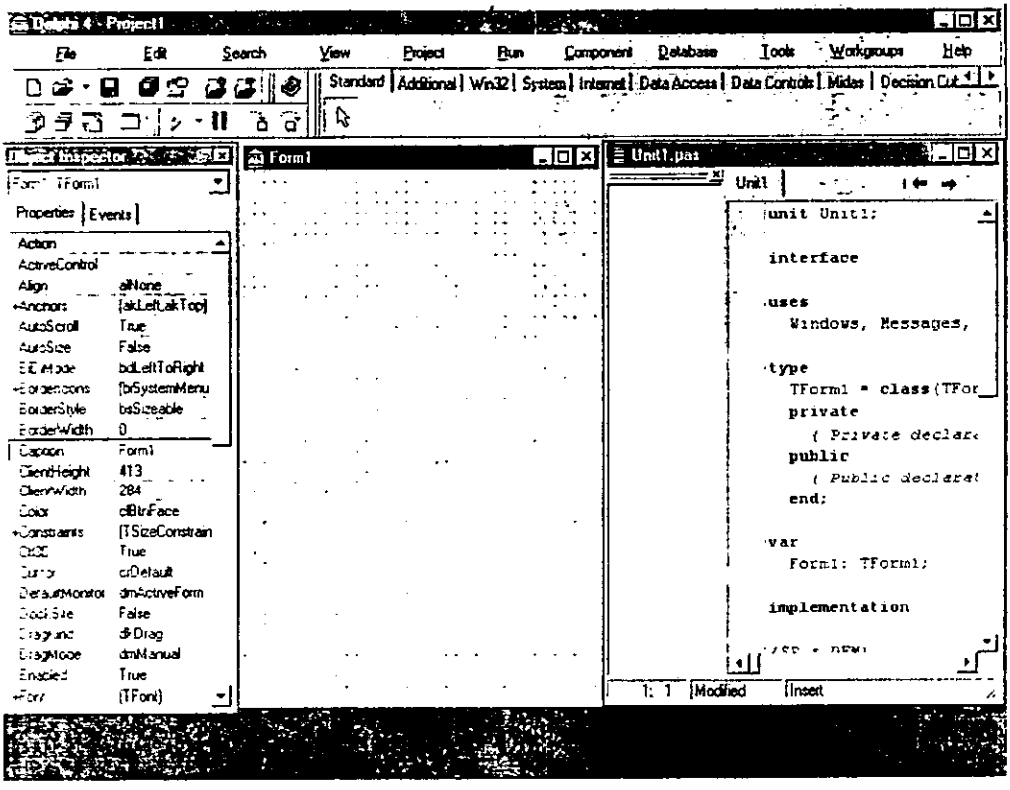


Figura1.10.2.1.1 Elementos iniciales en el entorno de Delphi 4.0

1.10.2.2 Creación de la interfaz de un programa.

La creación de una aplicación con Delphi se inicia, generalmente, con el diseño de su interfaz, para lo cual será necesario manipular la ficha o fichas con que cuenta el proyecto, insertando y manipulando componentes, propiedades y eventos, así como escribiendo el código que sea necesario.

1.10.2.3 La ventana del programa.

Toda aplicación Windows de tipo estándar, cuenta con una ventana, un espacio de pantalla delimitado mediante un borde, en el cual se visualiza o solicita información. En caso de que la aplicación cuente con varias ventanas siempre existirá una que sea la inicial o la ventana principal, desde la cual se accederá a la demás. Una ventana Windows cuenta con un borde, un fondo, un título unos botones y un menú del sistema. Todos estos elementos no tienen por qué existir siempre, se puede encontrar con una ventana que tiene el menú del sistema pero no botones de maximizar y minimizar.

En un programa desarrollado en Delphi las ventanas de la aplicación son las fichas, cuyas características se establecen mientras se construye el programa, es decir durante el diseño. La ventana principal será la ficha que se haya creado en primer lugar, aunque esto es configurable.

El aspecto de una ficha durante el diseño no tiene por que ser precisamente el aspecto que tendrá la ventana del programa cuando éste se ejecute y de hecho en la mayoría de las ocasiones no lo será. Por ejemplo, durante el diseño se muestra en el interior de la ficha una matriz, cuya finalidad es facilitar la alineación de los componentes en su interior. Esta matriz de puntos no será visible durante la ejecución, ya que en ese momento el programa ya está terminado y no tiene sentido la función para la cual está pensada dicha matriz.

Inicialmente las dimensiones de la ejecución serán las mismas que nosotros hayamos dado a la ficha durante el diseño, aunque este aspecto también depende del valor que asignemos a ciertas propiedades. Podemos modificar las dimensiones que por defecto tiene la ficha simplemente colocando el apuntador del mouse en uno de sus extremos y arrastrándolo. De igual forma también podemos modificar la posición en la que por defecto aparecerá la ventana de ejecución del programa.

Aunque durante el diseño en la ficha siempre existirán, en el extremo derecho, los botones de maximizado, minimizado y cierre, ello no implica que durante la ejecución estos botones aparezcan también, ya que es posible desactivarlos o hacerlos desaparecer, simplemente alterando el valor de la propiedad adecuada.

1.10.2.4 Matriz de puntos.

Durante el diseño, el fondo de la ficha cuenta por defecto con una matriz de puntos cuya finalidad es facilitar la alineación de múltiples componentes, de tal forma que éstos queden ajustados según nos interese.

Al insertar un componente en la ficha, así como el desplazarlo de un punto a otro o al modificar sus dimensiones, los puntos de la matriz serán por defecto la medida de incremento o decremento mínima.

Tanto la distancia entre los puntos de la matriz como el hecho de que todos los ajustes tengan como límite uno de esos puntos son aspectos configurables. Si se selecciona la opción *Environment options* del menú *Tools* se accederá a la ventana que se muestra en la figura 1.10.2.4.1, en cuya primera página existe un recuadro con las opciones que afectan a la matriz. Desactivando la opción de *Display grid*, que está activada por defecto, conseguiremos hacer desaparecer la matriz de puntos de la ficha, que aparecerá durante el diseño prácticamente con el mismo aspecto que tendrá la ejecución. En caso de que se opte por mantener la matriz de puntos, las opciones de *Grid size X* y *Grid size Y* nos permiten establecer la distancia, horizontal y vertical respectivamente, entre dos puntos consecutivos. Por último, con la opción *Snap to grid*, podemos indicar si se desea que al mover un componente, o modificar sus dimensiones, dicha distancia sea la unidad mínima de trabajo.

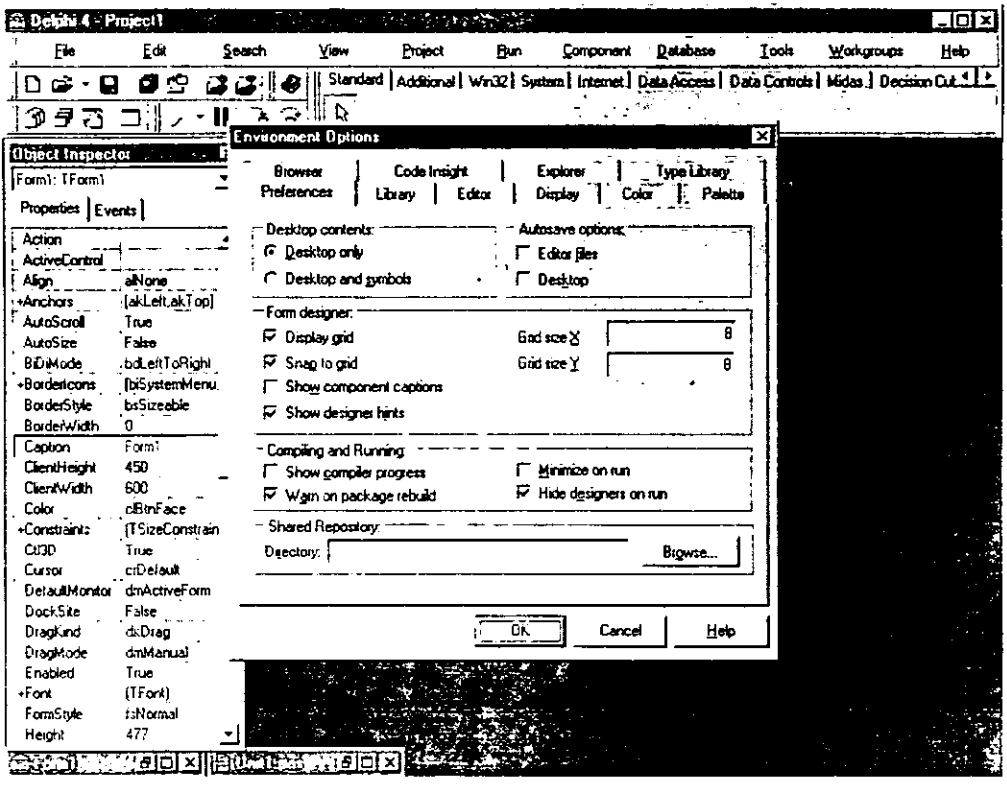


Figura 1.10.2.4.1 Ventana de preferencias con opciones de la matriz de puntos.

1.10.2.5 Inserción de componentes.

Una parte muy importante del desarrollo de una aplicación con Delphi consiste en insertar en la ficha los componentes apropiados, situándolos adecuadamente y estableciendo las propiedades que correspondan. Todos estos pasos, totalmente visuales, se ven completados con la escritura del código que se asociará a los eventos.

Para insertar un componente en la ficha, lo primero que tendremos que hacer será seleccionarlo en la *Paleta de componentes*, para lo cual deberemos activar la página en la que se encuentre dicho componente. La selección del componente se efectúa simplemente pulsando con el botón izquierdo del mouse sobre el icono que lo representa, tras lo cual bastará con desplazar el apuntador del mouse al lugar

aproximado de la ficha en que se desea insertar, pulsando de nuevo el mismo botón. El nuevo componente se ajustará al punto más cercano y tomará unas dimensiones por defecto.

El tamaño de un componente puede ser modificado en el mismo momento en que se inserta en la ficha, para lo cual, una vez seleccionado en la *Paleta de componentes*, habrá que desplazarse hasta el punto de la ficha en que se desee realizar la inserción, pulsando el botón izquierdo del mouse y manteniéndolo pulsado al tiempo que se desplaza a un punto opuesto, estableciendo las dimensiones deseadas. Al liberar el botón izquierdo del mouse veremos aparecer el componente con el tamaño que hayamos fijado.

En ocasiones es necesario insertar en una misma ficha múltiples componentes del mismo tipo, por ejemplo varios botones. En lugar de repetir el proceso anterior, seleccionando el componente, insertándolo y volviéndolo a seleccionar una y otra vez, podemos dejarlo seleccionado de forma fija. Para ello tendremos que pulsar la tecla <Mayús> antes de seleccionar el componente. tras lo cual no tendremos más que desplazar el apuntador hasta la ficha insertando tantos componentes como necesitemos. simplemente pulsando el botón izquierdo del mouse cada vez. Para eliminar la selección del componente y poder así trabajar con los componentes que se han insertado deberemos pulsar sobre el icono que representa el apuntador del mouse, que aparece en el extremo izquierdo de la *Paleta de componentes*.

1.10.2.6 Manipulación de Componentes.

El trabajo con un componente no termina con la inserción en la ficha, en muchos de los casos es necesario moverlo a otro punto, modificar su tamaño o realizar otras tareas que determinarán el comportamiento del componente durante la ejecución.

Para manipular un componente lo primero que debemos hacer es seleccionarlo, situando el apuntador del mouse sobre él y pulsando el botón izquierdo. Un

componente seleccionado se diferencia de los demás en que tiene un borde alrededor con cuadritos.

El cambio de posición de un componente es muy simple, una vez que esta seleccionado, basta con arrastrarlo a la nueva posición y listo. Para modificar su tamaño debemos mover el apuntador hasta uno de los cuadritos que hay en el borde, arrastrando éste en la dirección que nos interese. En lugar del mouse también se puede usar el teclado para conseguir las mismas acciones. La combinación <Control-Flecha> nos permite desplazar punto a punto el componente seleccionado en la dirección que nos interese, mientras que con <Mayús-Flecha> podemos incrementar o decrementar su altura y anchura. Pulsando <Control-Mayús-Flecha> podemos desplazar el componente no punto a punto, sino utilizando el incremento que se haya fijado en la matriz de puntos.

Aunque lo habitual es manipular un sólo componente cada vez, Delphi también permite seleccionar múltiples componentes de forma simultánea, de tal forma que podemos desplazarnos conjuntamente, así como modificar propiedades comunes. Para seleccionar varios componentes podemos usar dos métodos diferentes: mantener pulsada la tecla <Mayús> mientras con el botón izquierdo del mouse vamos marcándolos, o bien trazar con el apuntador del mouse, manteniendo el botón izquierdo pulsado, un recuadro que contenga total o parcialmente los componentes son los que han resultado seleccionados fácilmente, ya que cada uno de ellos tendrá un recuadro a su alrededor.

Cada vez que se inserta un nuevo componente, éste toma unas propiedades por defecto, como pueden ser las dimensiones, título, etc. En caso de que nos interese insertar en una ficha un componente con las mismas propiedades de otro ya existente, en lugar de usar el método descrito en el punto anterior podemos realizar una operación de copiar y pegar. Para ello seleccionaremos el componente original y pulsaremos <Control-C>, copiándolo al portapapeles. A continuación pulsaremos sobre el fondo de la ficha y pegaremos el componente mediante la combinación <Control-V>.

Una vez que todos los componentes de una ficha se encuentran en la posición deseada y con las dimensiones necesarias, para evitar una modificación accidental se puede utilizar la opción Lock Controls del menú Edit, que los bloqueará de tal forma que no sea posible modificar posición o tamaño ni con el mouse ni con el teclado, aunque si será posible hacerlo editando las propiedades correspondientes.

1.10.2.7 Modificación de propiedades.

Tras la inserción de un componente en una ficha la tarea más habitual suele ser el establecimiento de propiedades, mediante las cuales se definen diferentes características que de una forma u otra afectarán al aspecto del componente o a su funcionamiento. Ciertas propiedades se modifican en forma indirecta cuando manipulamos un control en la ficha, alterando, por ejemplo, su posición. Sin embargo, la mayor parte de las propiedades habrá que editarlas directamente sirviéndonos para ello del *Inspector de objetos*.

Cada vez que en la ficha se selecciona un componente, automáticamente el Inspector de objetos muestra su nombre, en la parte superior, y sus propiedades y eventos, en las dos páginas con que cuenta. Cada página está dividida en dos columnas, conteniendo la de la izquierda el nombre de la propiedad o evento, un parámetro que no podemos modificar. La columna de la derecha es la que mantiene el valor de la propiedad o bien el nombre del método asociado al evento. El ancho de cada una de las columnas puede ser modificado, situando el apuntador del mouse en la línea central que las separa y arrastrándolo según nos interese.

También se puede seleccionar un determinado componente desplegando la lista que hay en la parte superior de Inspector de objetos, en la que aparecerán los nombres de todos los componentes existentes en la ficha sobre la que estamos trabajando.

1.10.2.8 Uso de los eventos.

El código de un programa Windows no se ejecuta de forma secuencial, sino que lo hace según los mensajes que recibe del propio sistema. Estos mensajes se generan, por ejemplo, al pulsar un botón, introducir un carácter o desplazar el apuntador del mouse. En Delphi estos mensajes son gestionados automáticamente por cada componente y traducidos en eventos. Nosotros podemos asociar un método a un evento, de tal forma que cuando dicho evento se presente, se ejecute el código de ese método.

No todos los mensajes de Windows tienen un evento equivalente en los componentes de Delphi, pero en la mayoría de los casos, o en los más habituales, siempre encontraremos un evento apropiado para la finalidad que necesitemos.

Pulsando sobre la solapa *Events* del *Inspector de objetos* podremos abrir la página de eventos, en la que veremos los nombres de los eventos con los que cuenta el componente que tengamos seleccionado en ese momento en la ficha. A la derecha de cada uno de éstos puede existir el nombre de un método, que será el que se ejecute en el momento en que se genere el evento. Haciendo doble clic sobre la columna derecha de un evento abriremos automáticamente la ventana de código situando el cursor en el método adecuado. En caso de que dicho evento no contase hasta ese momento con un método asociado, Delphi automáticamente creará el cuerpo de un nuevo método e introducirá su nombre en el *Inspector de objetos*, a la derecha del nombre del evento.

Múltiples eventos pueden compartir un mismo método siempre y cuando los eventos sean del mismo tipo. El tipo de un evento determina, básicamente, los parámetros que el método recibirá cuando sea llamado. El evento *OnClick*, por ejemplo, recibe un sólo parámetro, mientras que el evento *OnMouseMove* recibe cuatro. Estos eventos no podrían, por lo tanto, compartir un mismo método.

1.10.2.9 Edición de código.

En un programa Delphi la interfaz se crea de una forma visual, sin embargo siempre existirán acciones que necesiten la escritura de algo de código, para poder realizarse. Por ejemplo, se requiere escribir un programa que muestre una ventana con un botón, de tal forma que al pulsarse éste aparezca un mensaje con un saludo. La parte visual consistiría en insertar el botón en la ficha, establecer el título, su posición y dimensiones, color, etc. Al pulsar el botón, momento en el cual se generaría un evento, se debería ejecutar un método en que previamente nosotros habríamos escrito el código necesario para mostrar el citado mensaje.

La mayor parte del código de un programa Delphi estará asociado a algún evento, al que se accederá mediante la página de eventos del *Inspector de objetos* o bien haciendo doble clic sobre el componente. En cualquier caso Delphi se encargará de establecer el nombre de los métodos, determinar los parámetros necesarios y sus tipos, de tal forma que sólo se tiene que introducir las sentencias que se desee ejecutar, sin preocuparnos por cómo se transferirá el control a dicho código, apartado del que se ocupará adecuadamente el componente afectado.

A pesar de lo mencionado, en Delphi también puede existir código que no esté directamente ligado a un determinado evento de un componente. Se puede, por ejemplo, escribir un código para definir un nuevo objeto o crear un determinado procedimiento que después será usado desde el programa.

El código de un programa Delphi se estructura en lo que se denomina módulos, que son archivos de texto conteniendo código. Cada una de las fichas de un proyecto tiene asociado un módulo, pudiendo además existir módulos independientes que se añadan para definir funciones, procedimiento u objetos.

La edición de código en Delphi se realiza en la ventana del *Editor de código*, una ventana que normalmente permanece bajo la ficha en la que se está trabajando.

Podemos alternar entre la ficha y el correspondiente módulo de código simplemente pulsando la tecla <F12>.

El *Editor de código* de Delphi es un editor multi-archivo, lo que quiere decir que es capaz de mantener abiertos múltiples módulos de código en forma simultanea. En la parte superior de la ventana, como puede verse en la figura 1.10.2.9.1, existirá una solapa por cada módulo abierto, lo que nos permitirá cambiar de forma rápida y simple entre el código de los distintos archivos. También disponemos de dos botones, en la parte superior derecha de la ventana, que facilitan la navegación adelante y atrás por los últimos puntos en los que se ha estado, de forma similar a los botones de un navegador de Web. Por defecto a la izquierda del *Editor de código* se muestra el *Explorador de código.*

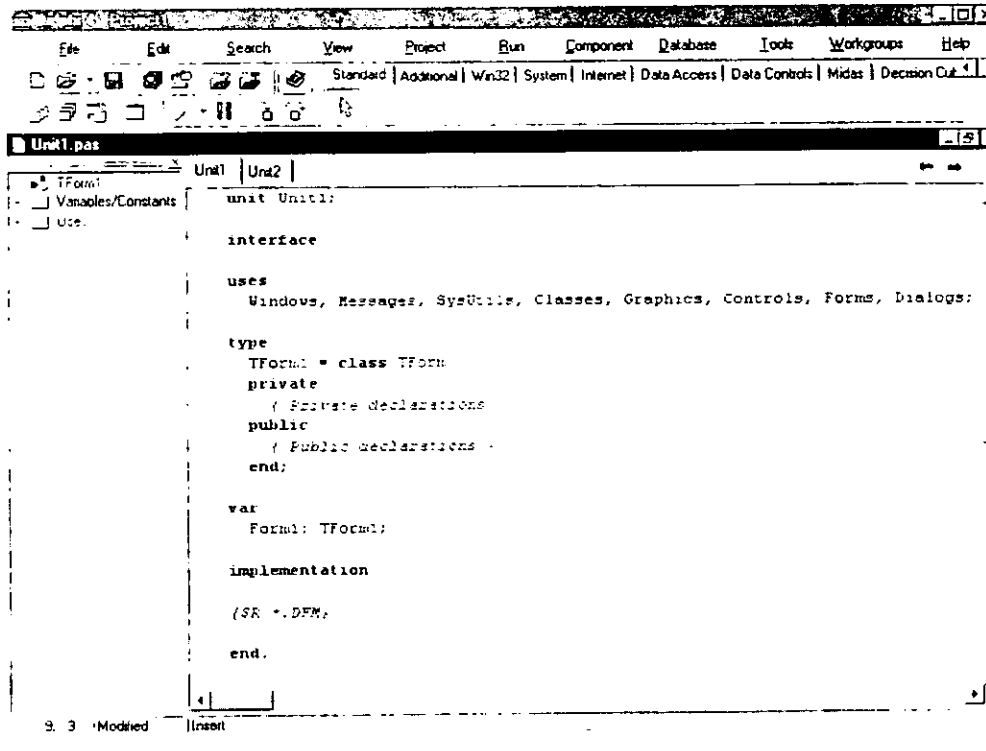


Figura 1.10.2.9.1 Ventana del editor de código de Delphi.

Para trabajar con el Editor de código de Delphi se puede utilizar las mismas combinaciones de tecla que estamos acostumbrados a usar en otras aplicaciones, aunque se puede seleccionar uno de entre varios modos de funcionamiento diversos. Este editor tiene la característica de diferenciar en el código, usando colores y estilos, los diferentes elementos de cada sentencia, como pueden ser palabras reservadas, variables o comentarios. Todas las características se pueden personalizar mediante la ventana de opciones del editor, a la que se puede acceder desplegando el menú emergente, pulsando el botón derecho del mouse sobre la ventana del editor, y seleccionando la opción *Properties*. En la ventana, que cuenta con cuatro páginas, se puede configurar desde los colores de cada parte del código hasta el aspecto del cursor, pasando por los puntos de los tabuladores o la columna en la que se mostrará el margen derecho.

El Explorador de código es una lista jerárquica en la que se muestran las clases, variables y en general, todos los elementos existentes en el módulo del código seleccionado. Se puede hacer doble clic sobre cualquier elemento para ir directamente a él, sin necesidad de buscar secuencialmente por el código. Podemos decir que el Explorador de código nos ofrece una imagen "a vista de pájaro" del módulo de código.

1.10.2.10 Ejecución de un programa.

El fin último de todo el proceso que se ha descrito, desde la inserción del componente hasta la escritura del código, es conseguir un programa que sea posible compilar y ejecutar. Se puede realizar este proceso, ejecutando el programa desde el principio del entorno de Delphi, simplemente pulsando la tecla <F9>.

En caso de que el proceso de compilación se detenga por haber encontrado algún error en el código, el Editor de código mostrará en la parte inferior una lista con los errores encontrados, lista que se puede usar para desplazarse a las diferentes líneas que contienen dichos errores.

Suponiendo que no exista error alguno el programa se compilará y será ejecutado. En el momento en que éste termine, regresará el control a Delphi.

Habitualmente el proceso de ejecución de un programa desde el propio entorno de Delphi se repite una y otra vez, con el fin de ir viendo si el funcionamiento del programa es correcto y se ajusta a lo que se desea conseguir. A pesar de que ello no debe ser motivo de preocupación, ya que la compilación del código por parte de Delphi es muy rápida.

1.10.3 Fundamentos de Object Pascal.

1.10.3.1 Introducción.

A pesar de que la mayor parte del desarrollo de una aplicación con Delphi se realiza de forma visual, insertando componentes y modificando sus propiedades, ningún programa tendrá una aplicación práctica si tan sólo cuenta con una interfaz, sin un código de programa que sea capaz de procesar las entradas de datos y generar unos resultados, o lo que es lo mismo, sin que el programa sea interactivo.

Consecuentemente, el conocimiento del lenguaje que utiliza Delphi es de suma importancia, ya que de lo contrario difícilmente se puede escribir nada de código. Este lenguaje llamado Object Pascal, es un heredero de Turbo Pascal de Borland y seguramente el Pascal más evolucionado que existe actualmente. Se trata de un lenguaje completamente estructurado, con orientación a objetos, claro y muy elegante, en cuanto respecta a la claridad y eficiencia del código.

1.10.3.2 Estructura General.

El código de un programa Object Pascal se almacena en lo que se denomina módulos o *units*. Un módulo no es más que un archivo de texto, generalmente con extensión PAS, que tiene una estructura general siempre idéntica, contando con un encabezado, en el

que se establece el nombre del módulo, y un bloque principal, delimitado entre las palabras Begin y End, en que se alojarán las sentencias o código a ejecutar. Las siguientes cuatro líneas de código, por ejemplo serían un programa completo en Object Pascal.

```
Program Prueba;
Begin
        ShowMessage('Hola');
End.
```

En este programa existen una serie de palabras reservadas que en el *Editor de código* de Delphi aparecerán en negrita, como son *Program, Begin* o *End*. Estas palabras son reconocidas directamente por el lenguaje Object Pascal, mientras que la línea que comienza con *ShowMessage*, por el contrario es una llamado a un procedimiento.

1.10.3.3 El punto, y el punto y coma.

En el programa anterior, se observa que al final de algunas líneas existe un punto y coma, que se usa como separador de sentencias. El punto y coma se dispondrá, por regla general, al final de cada sentencia que no éste compuesta estrictamente de una palabra reservada, como Begin, tras la cual no existe este elemento. También se puede omitir el punto y coma en la sentencia anterior a la línea que contiene la palabra End.

Ciertas construcciones, como el condicional If...Then...Else, exigen que el punto y coma sea omitido en la secuencia anterior a la palabra Else.

En el código de un programa pueden existir múltiples bloques, delimitados por la citadas palabras Begin y End. Al bloque más exterior, el que se inicia con Begin y termina con End, se le denomina bloque principal del módulo. Dentro de éste pueden existir otros bloques, por ejemplo definiendo métodos y funciones, tipos, etc. La palabra End del

bloque principal, que determina el final del código, irá siempre seguida de un punto en lugar de hacerlo con un punto y coma.

1.10.3.4 Módulos y la cláusula Uses.

Como se menciono anteriormente, una aplicación de Delphi suele estar compuesta de múltiples módulos, cada uno de los cuales almacena el código de una ficha o cualquier otro código necesario, como puede ser el de la definición de objetos o procedimiento y funciones. El encabezado de un módulo se inicia con la palabra Unit y no con la palabra Program. En realidad el módulo que contiene en su encabezado la palabra Program es un módulo más, en cuanto a su formato y contenido, pero se diferencia de los demás en que él es el módulo principal por el que se iniciará la ejecución del programa.

En un programa Delphi el módulo principal es el archivo de proyecto, un archivo con extensión DPR, el cual relaciona todos los elementos de un proyecto generando un módulo de código. El resto de los módulos de la aplicación, por lo tanto, contiene código que no será ejecutado directamente, hasta en tanto desde el módulo principal u otro módulo en ejecución se realice la llamada oportuna.

La división del código de una aplicación en múltiples módulos es algo muy práctico, ya que permite un fácil mantenimiento del código, en contraposición a la construcción monolítica de un solo archivo de texto que se utilizaba antes.

Además de que los módulos de código se añaden al proyecto, Delphi cuenta con una serie de módulos prefabricados, con código ya ejecutable, preparados para ser usados desde cualquier programa. El código de estos programas se almacena en archivos con extensión DCU y en ellos se pueden encontrar procedimientos, funciones, objetos y componentes a los que sólo tendremos que llamar para realizar una acción determinada.

Para poder usar el código contenido en un cierto módulo desde otro modulo es necesario incluir en el primero una cláusula llamada Uses, seguida del nombre del módulo o módulos a usar. Si se abre en el Editor de código de Delphi el archivo de un proyecto cualquiera, se podrá ver que tras la cláusula Uses se hace referencia al módulo Form. en el que se encuentra todo el código necesario para hacer funcionar a una ficha, y también a cada uno de los módulos que existan en el proyecto.

1.10.3.5 Comentarios.

Al escribir el código de un programa existe una muy buena costumbre que consiste en ir insertando comentarios explicativos sobre lo que se quiere hacer en cada punto, comentarios que facilitarán el posterior mantenimiento de ese código. así como su comprensión por una tercera persona que se viera implicada en su uso.

La implementación de Object Pascal que encontramos en Delphi 4.0 nos permite incluir comentarios en el código de múltiples formas. El estilo más clásico consiste en delimitar el comentario mediante las parejas de caracteres (* y *), situando la primera de ellas al inicio del comentario y la segunda al final. En la actualidad, sin embargo. se utilizan más los caracteres { y }, que dispuestos en los mismos puntos se consigue el mismo efecto.

Con cualquiera de los dos métodos anteriores se puede insertar comentarios que abarquen varias líneas, ya que el comentario en sí no acaba hasta que se encuentre la pareja *) o el carácter }.

Existe un tercer método. en este caso para insertar un comentario que ocupa una sola línea. El inicio de comentario se marca con los caracteres // y termina al final de la línea. Este método es más cómodo para insertar comentarios cortos, situados generalmente tras alguna sentencia.

A continuación se puede ver un pequeño programa con varios comentarios. en el que se usan los tres métodos que se acaban de comentar.

```
(* Este es un pequeño programa de prueba *)
Program Prueba;
{ Este programa simplemente mostrará un mensaje por pantalla}
Begin
        ShowMessage('Hola'); // Despliega  el mensaje
End.
```

1.10.3.6 Constantes y variables.

En el código de un programa habitualmente se utilizan diversos valores que pueden aparecer en forma de constantes o variables. Estos valores, que no son palabras reservadas. llamadas a funciones, definiciones o ningún otro elemento similar, suelen usarse como parámetros de control y almacenamiento de datos. En el programa anterior, por ejemplo, la cadena 'Hola' es un valor constante. que se usa como parámetro en la llamada a la función ShowMessage.

Una variable es un nombre que se asigna a un espacio de memoria y que sirve para almacenar un valor de un determinado tipo. Se puede utilizar una variable, por ejemplo, para almacenar un número entero, sin tener que saber de antemano cuál será ese número.

En Object Pascal, a diferencia de lo que ocurre en otros lenguajes. cualquier variable debe ser declarada antes de utilizarse. Esto, que en principio puede parecer algo estricto o incomodo. es una gran ventaja. ya que no existe la posibilidad de que el compilador al encontrar un identificador desconocido interprete que se trata de una variable que no se ha declarado, introduciendo así en el programa un error difícil de encontrar.

1.10.3.7 Identificadores.

A la hora de definir una variable, al igual que al escribir un procedimiento o función, definir un nuevo tipo o una nueva clase de objeto, tendremos que asignar un identificador que nos permita referirnos a ese elemento que vamos a definir de una forma inequívoca. Este identificador ha de ser siempre lo más descriptivo y claro posible, ajustándose en todo caso a una ciertas reglas de nomenclatura.

Un identificador deberá comenzar siempre con una letra o un carácter de subrayado, e irá seguido de cualquier número de caracteres. Estos caracteres habrán de ser necesariamente letras, dígitos numéricos o caracteres de subrayado y debe de tenerse en cuenta que no se consideran letras válidas ni las acentuadas ni la Ñ. Tampoco se permiten los espacios en blanco en el interior de un identificador.

Obviamente cuando creamos un identificador, para que una referencia a él sea invocada, el identificador deberá ser único, es decir, no podrán existir dos identificadores iguales, con la misma secuencia de caracteres, ya que de lo contrario el compilador no podría distinguir entre uno y otro y de hecho generará un error de duplicación de forma inmediata, al intentar compilar el programa.

A pesar de lo dicho, y como veremos posteriormente, es posible crear identificadores iguales siempre que éstos existan en ámbitos diferentes. De esta forma dos procedimientos diferentes podrían declarar en su interior dos variables con el mismo nombre sin problema alguno, ya que cada una de las variables sólo existe en el interior del procedimiento en que se ha declarado.

1.10.3.8 Tipos.

Tanto para declarar una variable como para utilizar un valor constante, es necesario que conozcamos los diferentes tipos de datos con los que puede trabajar Object Pascal. Estos tipos nos permitirán trabajar con números, en diferentes precisiones, cadenas de

caracteres, apuntadores, etc. Cada uno de los tipos está representado por una palabra clave, que de una forma más o menos clara representa al tipo de dato. En la tabla 1.10.3.8.1, se enumeran los tipos más habituales.

Tabla 1.10.3.8.1 Tipos de datos de Object Pascal.

Tipo	*Valores que puede contener*
Byte	Números enteros comprendidos entre 0 y 255
ShortInt	Números enteros comprendidos entre –128 y 127
SmallInt	Números enteros comprendidos entre –32768 y 32767
Word	Números enteros comprendidos entre 0 y 65535
Integer	Números enteros comprendidos entre –21147483648 y 2114783647
LongInt	Igual que el Integer
LongWord	Números enteros entre 0 y 4294967295
Cardinal	Números enteros entre 0 y 2147483647
Int64	Números enteros entre –2^63 y 2^63
Boolean	True o False ocupando un byte de memoria
ByteBool	Igual que Boolean
WordBool	Igual que Boolean, pero ocupando dos bytes de memoria
LongBool	Igual que Boolean, pero ocupando cuatro bytes de memoria
Single	Números en coma flotante con hasta 8 dígitos
Double	Números en coma flotante con hasta 16 dígitos
Real	Igual que el Double
Extended	Números en coma flotante con hasta 20 dígitos
Comp	Números enteros de hasta 20 dígitos
Char	Un carácter cualquiera
AnsiChar	Lo mismo que Char
WideChar	Carácter de 16 bits para soporte Unicode
String	Cadena de caracteres
ShortString	Cadena de un máximo de 255 caracteres
AnsiString	Cadena de caracteres

Pchar	Apuntador a secuencia de caracteres terminada con nulo
PansiChar	Lo mismo que char
PwideChar	Apuntador a secuencia de caracteres de 16 bits
Pointer	Apuntador genérico
Variant	Tipo de datos variable

Los tipos enumerados en esta tabla pueden ser divididos en dos grupos, según sean tipos genéricos o tipos fundamentales. Los tipos genéricos, como Integer o String, no son iguales en Delphi 1.0 y en Delphi 4.0, ya que mientras en el primer caso un entero ocupa 16 bits, en el segundo se almacena en 32 bits, por lo que el rango de valores es diferente. Otro tanto ocurre con el tipo String, que puede almacenar una cadena de 255 caracteres o sin límite, dependiendo de la versión del compilador. En contraposición se tiene a los tipos fundamentales, como ShortInt o AnsString, cuyo tamaño es siempre el mismo.

Además de los tipos que describen en la tabla 1.10.3.8.1, pueden existir muchos tipos más, ya que Object Pascal permite crear nuevos tipos de datos. Así podemos encontrar con un tipo llamado TdateTime que permite el almacenamiento de fechas y horas.

1.10.3.9 Declaración de variables.

La declaración de una variable puede tener lugar a diferentes niveles o ámbitos, como puede ser el ámbito del módulo, antes del bloque principal, o en el ámbito local, en el interior de algún bloque de definición de procedimiento o función.

En cualquier caso, el inicio de la sección de declaración de variables estará siempre marcado por la ventana Ver, que irá seguida de una o más líneas en cada una de las cuales es posible declarar una o más variables de un determinado tipo. Cada una de estas líneas estará compuesta del identificador de una variable y el tipo, separando ambos elementos por dos puntos. En caso de que se vaya a declarar varias variables

del mismo tipo, se puede disponer los identificadores uno tras otro separándolos por comas, especificando al final el tipo.

En caso de que las variables que se están declarando sean del tipo String, AnsiString o ShortString, de forma opcional, se puede especificar, entre corchetes, el número de caracteres que como máximo podrá contener dicha cadena. Si usamos el tipo AnsiString y no especificamos una longitud máxima, el espacio para la cadena se asignará dinámicamente según las necesidades.

En el siguiente fragmento de código puede verse cómo se han declarado cuatro variables, una del tipo Byte, dos del tipo Integer y una del tipo String con un límite máximo de 25 caracteres.

```
Var // Declaración de variables
    Contador : Byte;          // Una sola variable
    N1, N2 : Integer;         // Dos variables del mismo tipo
    Nombre : String[25];      // Una cadena de 25 caracteres máximo
```

1.10.3.10 Matrices.

A veces es necesario mantener múltiples valores del mismo tipo, para los cuales, obviamente, podríamos declarar variables individuales. Esta técnica, sin embargo, no es útil cuando el número de variables es muy grande. Por ejemplo, si se desea almacenar cien números para realizar cálculos, desde luego no sería muy cómodo tener que declarar cien identificadores distintos y gestionarlos de forma individual en posteriores asignaciones y expresiones.

En casos como el descrito es mucho más cómodo usar matrices, nombre con el que se le conoce a un tipo de variables que se caracteriza por ser capaz de contener múltiples valores del mismo tipo, haciendo referencia a cada uno de ellos mediante un índice.

Object Pascal permite crear matrices prácticamente de cualquier tipo, incluso es posible crear matrices de matrices. Esto quiere decir que se puede crear matrices no sólo de números, sino también de cadenas, de objetos, o de cualquier otro tipo que nosotros hayamos definido previamente.

La declaración de una matriz es muy similar a la declaración de cualquier otra variable, si exceptuamos que delante del tipo, tras los dos puntos, se tiene que colocar la palabra Array, indicando entre corchetes el rango de elementos que deseamos que tenga la matriz y a continuación la palabra Of seguida del tipo.

En Object Pascal un rango de valores se escribe especificando el primer valor seguido de dos puntos y el último valor del rango. Así, si deseamos declarar una matriz de, por ejemplo, veinticinco elementos de tipo Byte, se usará la siguiente sentencia:

Matriz : Array[1..25] Of Byte;

En este caso la variable Matriz en una matriz que contiene veinticinco elementos, con los índices que van desde el 1 hasta el 25. Cada uno de los elementos es una variable del tipo Byte, que se puede usar en cualquier ámbito y expresión en la que se admita un valor de este tipo.

La matriz anterior es de una sola dimensión, es decir, se puede representar como si fuese una lista de celdillas, en cada una de las cuales se almacena un valor individual. También se puede definir matrices de dos, tres o cualquier otro número de dimensiones.

1.10.3.11 Estructuras de Control.

El código de un programa por regla general no se ejecuta de forma secuencial, desde la primera sentencia hasta la última. Lo que es habitual es que en cada caso se ejecuten unas sentencias u otras, dependiendo de ciertas condiciones, y también que algunas

sentencias se ejecutan cíclicamente, creando procesos repetitivos. Estos dos elementos, los condicionales y los procesos repetitivos, son los que básicamente permiten controlar el flujo de ejecución de un programa.

Tradicionalmente en un entorno como DOS, el programa tenía el control absoluto del sistema y era responsable de todo el proceso por el cual se ejecutaban unas sentencias u otras dependiendo de las pulsaciones del teclado, o movimientos del mouse que se realicen. En Windows y entornos similares es el propio sistema el que se ocupa de gran parte del trabajo y el código de nuestro programa se ejecuta cuando Windows le transfiere el control.

Como se dijo anteriormente, la totalidad del código que se escriba estará asociado a algún evento, por lo que su ejecución tendrá lugar cuando dicho evento se produzca. Será en el interior de esos bloques de código, procedimientos y funciones en los que usaremos las siguientes estructura de control.

- *Condicionales*
- *Repeticiones o Ciclos*

1.10.3.12 Procedimientos y Funciones.

Un procedimiento o una función es un conjunto de sentencias, de líneas de código, al cual se asigna un identificador, de tal forma que desde cualquier otro punto del módulo, o incluso de la aplicación, es posible ejecutar esas sentencias tan sólo haciendo referencia del identificador.

Tanto un procedimiento como una función pueden recibir parámetros que le permitan al código trabajar con los datos apropiados en cada caso. Una función, además, devuelve un parámetro mediante el cual es posible facilitar un resultado.

Podemos decir que una función es un procedimiento capaz de devolver un parámetro, siendo esa la única diferencia entre ambos.

Para definir un procedimiento se tiene que usar la palabra reservada Procedure, tras la cual dispondremos el identificador por el que se va a conocer al procedimiento. Si lo que se desea es crear una función se utilizará la palabra Function en su lugar. La línea será terminada con un punto y coma, que seguirá al identificador.

Un procedimiento es un bloque en el cual es posible definir tipos, declarar constantes y variables y escribir código ejecutable. Los apartados de definiciones y declaraciones se iniciarán con las palabras Type, Const y Var que ya se conocen, mientras que el inicio del bloque de código se marcará con la palabra Begin, que irá emparejada con su correspondiente End dispuesto al final del bloque.

Todas las definiciones de tipos y declaraciones de variables que se efectúen en el interior del procedimiento son locales a dicho procedimiento, por lo que no estarán disponibles desde ningún otro punto del módulo ni del programa.

La finalidad principal de un procedimiento es evitar la repetición de un código que va hacer usado repetidas veces, aislándolo y asignándole un nombre que facilite su ejecución en cualquier punto en el que sea necesario. Por ejemplo, se tiene que escribir un programa en que en ocasiones tiene necesidad de provocar una espera de cinco segundos. En lugar de codificar las sentencias necesarias cada vez, sería mucho más eficiente la creación del procedimiento siguiente:

```
{Procedimiento que causa una espera de 5 seg. }
Procedure Espera:
Var
    Ahora : LongInt;
Begin
    {Se toma el número de milisegundos transcurridos}
```

```
    Ahora := GetTickCount;
    {Mientras no transcurran 5 segundos}
    While GetTickCount – Ahora < 5000 Do
        {Se esta en una ciclo sin hacer nada}
    Continue;
End;
```

Una vez creado, un procedimiento puede ser llamado desde cualquier punto del módulo, si fue definido en la parte de implementación, o desde cualquier parte de la aplicación, si se declaró en la parte de interfaz del módulo. La sentencia de llamada es tan simple como la siguiente:

```
Espera; {Esperando 5 segundos}
```

1.10.4 Fundamentos de orientación a objetos.

Object Pascal es un lenguaje totalmente orientado a objetos, una característica hasta ahora reservada a lenguajes como SmallTalk o C++. Esta característica hace de Object Pascal un lenguaje muy potente, ya que la orientación a objetos simplifica en gran medida muchos de los problemas que plantea el desarrollo de aplicaciones complejas usando un lenguaje procedural, como puede ser Pascal, Basic o C.

1.10.4.1 ¿Qué es un objeto?

En Delphi prácticamente todos los elementos son objetos y el ejemplo más claro son los componentes. Existe una serie de objetos predefinidos, como los citados componentes y otros que nos están accesibles en tiempo de diseño, sino sólo desde el código de nuestro programa. También se pueden definir nuestros propios tipos de objetos.

Un objeto siempre parte de un molde, al que habitualmente se conoce como clase. En la definición de esta clase se establecen las características del objeto y se implementa su funcionalidad. Una vez que el molde, que es la clase, está terminado, se puede crear objetos de esa clase de igual forma que podemos crear una variable.

La definición de un nuevo tipo de objeto es en cierta forma similar a la definición de un registro. La primera diferencia es que la palabra Record se sustituye por la palabra Class y la segunda en que la definición esta estructurada en varios apartados a los que se les denomina Private, Protected, Public, Published o Automated, según los casos. Otra diferencia es que en el interior de la definición de una clase no se declaran tan sólo variables, también existen procedimientos y funciones, a los que se les denomina métodos, y otros elementos, como pueden ser las propiedades.

1.10.4.2 Encapsulación.

Se conoce con este nombre a una de las tres principales características de la programación orientada a objetos. Un objeto contiene en su interior las variables necesarias para almacenar los datos sobre los que va a trabajar, así como los procedimientos y funciones precisos para manipular esos datos. Ningún código externo al propio objeto puede acceder a sus variables que contiene el propio objeto, estando seguro de que su contenido no ha sido alterado externamente y que por lo tanto será válido en todo momento.

Un objeto, visto desde esta perspectiva, es como una cápsula, en la cual están contenidas las herramientas y los materiales necesarios para desempeñar la función que le corresponda.

1.10.4.3 Herencia.

Es la segunda palabra mágica de la orientación a objetos y seguramente la característica más útil, ya que permite ahorrar la escritura de mucho código. Mediante la

herencia es posible definir una nueva clase de objeto a partir de otra que ya existe, heredando todos sus miembros, tanto de datos como métodos y propiedades. En la nueva definición es posible añadir nuevos miembros y redefinir los que se han heredado.

En Object Pascal se denomina ascendiente a la clase que actúa como base y descendiente a la que se deriva a partir de ella. La relación de múltiples ascendientes y descendientes traza lo que normalmente se denomina una jerarquía de clases.

1.10.4.4 Poliformismo..

Seguramente este aspecto es el más incomprendido de la programación orientada a objetos, pero también es de los más útiles. Un objeto polimórfico es aquel que en tiempo de compilación, cuando se esta escribiendo el código, no se conoce su tipo, la clase a que pertenece, información que obtiene posteriormente en tiempo de ejecución.

El poliformismo permite que un programa manipule de una forma homogénea objetos que son diferentes, ahorrando también escritura de mucho código. Como es lógico un objeto polimórfico debe disponer de una información mínima en tiempo de compilación, que obtiene de la clase base o ascendiente de la jerarquía a la que pertenece.

1.10.5 Beneficios.

"Al combinar diseño visual basado en componentes de alto rendimiento con el compilador más rápido del mundo y el conjunto más amplio de herramientas de desarrollo Cliente/Servidor, Delphi proporciona una ventaja competitiva para la construcción de aplicaciones muy diversas: soporte a toma de decisiones, de información ejecutiva en Intranet, automatización de ventas, gestión de inventarios, compras, pedidos, etc., un universo tan amplio como son las actuales necesidades de los usuarios corporativos".

Para las aplicaciones en que se requiera convertir los datos corporativos en información significativa para la toma de decisiones. esta nueva versión del Delphi incluye Business Insight que, a través del análisis multidimensional de los datos, facilita la creación de sofisticados sistemas de soporte de decisiones, elaboración de informes, construcción de gráficos y diagramas en el acto.

Para fusionar componentes empresariales Delphi cuenta con Active Insight, que ofrece la posibilidad de reutilizarlos al máximo: en esta versión del producto se pueden crear componentes ActiveX para ser usados con las herramientas de desarrollo existentes.

Y para que los desarrolladores creen objetos reutilizables, se integra compatibilidad COM (modelo de objeto común) de Microsoft que es fácilmente usada en un entorno multi-tier. Esta versión también ofrece la posibilidad de crear una AtiveForm en Delphi o bien, de convertir una base ActiveX en una ficha Delphi para incrementar la velocidad de las aplicaciones.

Otra característica importante es el desarrollo de ejecutables eficientes que pueden ser distribuidos rápidamente a través de una Intranet, mediante la compilación de paquetes y la reducción del tamaño de los ejecutables.

Para disminuir los errores de codificación y sintaxis, Delphi incluye asistentes de código que permiten desde crear templates hasta simplificar la creación de código. Los asistentes de Delphi, CodeInsight Wizards, permiten aumentar la productividad del programador al reducir los errores de codificación y sintaxis, porque ya no tendrá que recordar las propiedades y los eventos de un componente o las declaraciones de un procedimiento, Delphi le proporciona automáticamente esta información.

De la misma forma, Delphi optimiza la conectividad mediante el acceso a las más populares bases de datos de la industria. como son Access, FoxPro, Informix, Oracle. DB2. SQL Server, entre otras, con lo que se obtiene compatibilidad entre las bases de datos existentes en la empresa y las nuevas aplicaciones.

Por otra parte, para incrementar la velocidad en las aplicaciones empresariales para Web, así como la difusión de información de bases de datos por Internet, la versión Cliente/Servidor ofrece soporte para los principales estándares de Internet, y permite construir complejas aplicaciones de bases de datos con componentes enlazados a la información.

Gracias a su arquitectura abierta y a su compatibilidad con DLL, Delphi también facilita la creación de aplicaciones de datos que deben transmitirse por la gran red. Por otra parte, para simplificar el desarrollo en Internet, WebBridge permite a los desarrolladores programar con una sola API común para NSAPI e ISAPI. Esta flexibilidad protege el código base ante las evoluciones de los estándares de Internet.

Esta versión incluye WebModules, herramientas que centralizan la información que se publica en las aplicaciones en tiempo real de Internet, tramitan peticiones, definen acciones y crean páginas en HTML. Y con el HTTP Dispatcher que viene en esta versión, los desarrolladores pueden producir contenidos de Web usando los mismos procedimientos que en el desarrollo Cliente/Servidor.

En cuanto a la reducción de mantenimiento, Delphi integra la suite MIDAS que facilita la creación de aplicaciones multi-tier con arquitecturas de cliente delgado mediante Remote DataBroker, ConstraintBroker y Business ObjectBroker.

Remote DataBroker facilita la creación de datos distribuidos y aplicaciones de partición. Los controladores de base de datos se mantienen de manera centralizada, se configuran en el servidor y no en cientos de clientes.

Por su parte, el ConstraintBroker y el ActiveDictionary propagan automáticamente las modificaciones de las reglas de negocio cuando estas se cambian, con esto se conservan las reglas en el servidor y se mantiene un tráfico reducido en la red.

Con Business ObjectBroker, se equilibra la carga de las aplicaciones fundamentales para la empresa y se obtiene una completa seguridad ante cualquier tipo de fallas. Si un servidor se bloquea, el cliente se conecta automáticamente al siguiente servidor disponible, lo que asegura confiabilidad 24x7.

Por último, para aumentar el rendimiento de aplicaciones Cliente/Servidor, Delphi incluye componentes como el SQL Database Explorer que facilitan la visualización de metadatos específicos.

Finalmente hubo un aspecto interesante de Delphi, se trata de la tecnología de la suite MIDAS que se incluye en la versión Cliente/Servidor y que permite desarrollar objetos de reglas de negocio con alta disponibilidad del servidor con seguridad frente a fallas, equilibrio de cargas, conjuntos de datos distribuidos y proceso de transacciones, aplicaciones de cliente reducidas, propagación automática de reglas de integridad de servidores de bases de datos y conectividad de base de datos de alta velocidad, funciones que pueden trabajar individualmente o con otras y que soportan sistemas operativos y estándares del mercado.

CAPÍTULO 2.

PLANTEAMIENTO DEL PROBLEMA Y PROPUESTA DE SOLUCIÓN.

2.1 SITUACIÓN ACTUAL.

Actualmente el proceso de compensación electrónica juega un papel importante en el ámbito bancario, ya que es una forma de realizar la compensación de cheques únicamente con la información lógica de las bandas magnetizables grabadas en un archivo, esto permite que el proceso de compensación se realice con mayor rapidez puesto que se evita el manejo físico de los cheques. La banca mexicana opta por esta opción para realizar la compensación porque ofrece ventajas tales como:

- Reducción de la ventana de tiempo en el proceso de compensación con respecto al proceso automatizado.
- Se aumenta la ventana de tiempo en la recepción de archivos ya que no se leen ni capturan cheques como en el proceso automatizado.
- Se puede procesar más información.
- Se implementan estándares de validación más eficaces.

El proceso de compensación electrónica se realiza en un mainframe Honeywell Bull DPS7000 (Data Proccess System) el cual tiene las siguientes características de hardware y software:

- 6 unidades de cintas magnéticas.
- 1 controlador de discos.
- 1 controlador de cintas.
- 8 discos duros de 500 Mb c/u.
- 32 Mb de RAM.
- Impresora de impacto de 600 lpm.
- Programas en ANSI COBOL.
- Sistema operativo GCOS 7.

El proceso de compensación electrónica involucra directamente a cuatro áreas específicas dentro de la cámara de compensación y varios proveedores externos, sin estos elementos no sería posible llevar a cabo el proceso. A continuación se enuncian las áreas involucradas y las funciones que ellas realizan:

Oficina de recepción y entrega.

Es la encargada de atender directamente con los clientes (delegados de los bancos), es la oficina encargada de recibir los documentos y medios magnéticos que presentan los delegados de los bancos, además son los responsables de entregar los resultados de salida (archivos en medios magnéticos, cheques y reportes) a cada delegado de los bancos.

Sus funciones son:

- Sellar la copia de la "carta presentación " del banco, como acuse de recibo.
- Recibir los medios magnéticos y conforme van llegando los envía por un montacargas a la oficina de cómputo que se encuentra en otra área, además entrega las "cartas de presentación" a la oficina de control.
- Telefonear al área de cómputo para avisarle que le ha enviado los medios magnéticos.
- Recibe telefonema de la oficina de cómputo quien les indica si el archivo fue aceptado o rechazado.
- Avisa personalmente al delegado que su archivo fue aceptado o rechazado.
- Recibe telefonema de la Oficina de Cómputo quien les informa que ha enviado los resultados de salida por el montacargas (medios magnéticos y reportes).

- Genera en Excel una lista con los bancos participantes, para que los delegados acusen de recibido en el momento de recibir sus resultados de salida. (ver reporte en el apéndice B).

- Genera una matriz de distribución de manera manual, ésta es un arreglo de renglones y columnas donde se identifica que banco presenta y quienes son los que reciben (ver reporte en apéndice B).

Oficina de cómputo.

Es la oficina encargada de realizar las tareas necesarias para validar los archivos que presentan los bancos, así como los encargados de generar los resultados de salida y ejecutar los procedimientos internos que garanticen el respaldo total de la información y buen funcionamiento del equipo de cómputo.

Sus funciones son:

- Preparar los medios magnéticos (cintas y disquetes) que serán utilizados en la generación de resultados de salida.

- Encender el aire acondicionado y esperar a que la temperatura se encuentre dentro del rango de 68 a 75 grados Farenheit.

- Encender los componentes del equipo DPS7000 en secuencia mediante lineas de comando. Además durante este proceso debe ir verificando que los mensajes que manda el sistema sean los correctos ya que de no ser así debe iniciar desde el principio y en el peor de los casos pedir la ayuda de los ingenieros de software, el proceso normal de encendido dura media hora aproximadamente.

Secuencias de encendido:

- Unidades de disco del 1 al 8 en orden ascendente.
- Controladores de discos y cintas.
- Impresora.
- Unidades de cinta en orden ascendente.
- Consolas y sus impresoras.
- CPU.

Posteriormente empieza la carga automática del sistema operativo. Al terminar la carga se verifica que los dispositivos hayan levantado, en caso contrario, se levanta cada uno a través de lineas de comando.

- Verificar y activar mediante líneas de comando una cinta llamada "SYSOUT", ésta evita la saturación en disco duro de las salidas que no son enviadas a impresión, ya que ciertos trabajos son enviados directamente a la impresora y el resto se almacenan en la cinta SYSOUT permitiendo con esto, recuperarlos e imprimirlos posteriormente. En caso de no activar la cinta de "SYSOUT" el espacio en disco puede llegar al 0% y detener todos los trabajos que se encuentran corriendo en el sistema.

- Activar el MICROFIT, que es una aplicación que le permite la transferencia de archivos entre el DPS7000 y una PC, ésta aplicación le servirá para poder validar los archivos que sean presentados en disquete.

- Una vez encendido el equipo, el operador inicializa el sistema de compensación electrónica con lineas de comando, esto le permite borrar bitácoras y archivos del proceso anterior, además de inicializar banderas del sistema. éste proceso le lleva en condiciones normales aproximadamente 20 minutos.

- El operador ingresa la fecha del proceso al sistema, mediante lineas de comando, para indicarle con que fecha serán validados los archivos de entrada.

- Lanzar un proceso llamado "ORDENADOR", el cual permite ejecutar tareas durante la validación de los archivos de entrada, se hace mediante el siguiente comando:

S: RORD

- Ingresar y validar los archivos que presentan los bancos, la validación de cada archivo con aproximadamente 12,000 registros, tarda en promedio 8 minutos (1,500 registros en 1 min.), el tiempo es proporcional al número de operaciones; durante el proceso de validación se ejecutan 2 programas los cuales actualizan bitácoras, banderas y dispersan los registros correspondientes de cada banco.

- Avisar por teléfono a la oficina de recepción y entrega del resultado de la validación, estos a su vez comunican personalmente el resultado al delegado del banco, en el caso de existir error en el archivo, el delegado se comunica con su banco para que corrija el o los archivos.

- Monitorear constantemente mediante lineas de comando que el sistema se encuentre estable.

- Una vez que se han validado todos los archivos de entrada el operador procede a generar primeramente los reportes de control de salida mediante lineas de comando; por ejemplo para imprimir un reporte, se escribe:

S: EJ WRITER,, %SYSH VL=TEICI1.ED1.ENT.SAL.R000019970312

Los reportes de control obtenidos son los siguientes (ver apéndice b):

- Reporte de cifras de control.
- Hoja resumen de compensación.
- Reporte de operaciones erróneas.
- Reporte de archivos procesados.

El operador comunica por teléfono a recepción y entrega que ha enviado los reportes de control por el montacargas, la oficina de control revisa las cifras globales obtenidas. Posteriormente control da aviso por teléfono a cómputo para indicarle el resultado de la validación; si ésta es correcta, cómputo ejecuta el procedimiento para generar las cintas de salida, las hojas de compensación para cada banco (HCOM) y los reportes de operaciones erróneas (ROE), en caso de que exista error, la oficina de control avisa por teléfono a la oficina de sistemas, quien procede a detectar y corregir los errores.

Para generar los resultados de salida se necesita de un operador de consola y dos operadores que se encargan de montar, desmontar las cintas, además de cortar los listados de la impresora. Los resultados de salida (archivos en medios magnéticos y reportes) son enviados a recepción y entrega conforme van siendo generados para cada banco.

- Genera los respaldos de entradas y salidas, también genera la información de los reportes detalle de lo presentado y de lo recibido en cintas; posteriormente microfilma y envía las microfichas a los bancos.

- Al finalizar el proceso de compensación electrónica que tarda aproximadamente 5 horas con un volumen aproximado de 45,000 registros, procede a apagar el equipo en la secuencia inversa a como lo encendió.

Otra función de la oficina de cómputo es reportar las fallas de hardware al proveedor quien asigna a un ingeniero de servicio para reparar la falla, mientras que otro da el

mantenimiento preventivo diariamente, también se encarga de avisar a la oficina de sistemas las fallas que ocurren durante el proceso de compensación.

Oficina de control.

Es la encargada de verificar que las cifras globales sean las correctas y determinar cuando y en que momento se debe de entregar los resultados de salida a los delegados de los bancos.

Sus funciones son:

- Realizar la suma global de los importes que aparecen en las "cartas de presentación" de los bancos.

  Suma A = "carta de presentación 1" + "carta de presentación 2"+ ...
  "carta de presentación n"

- Sumar los importes globales de los reportes "hoja resumen de la compensación" y "reporte de operaciones erróneas", con esta suma determina el importe total por banco y Global de todos los bancos.

  Suma B = "Importes del reporte de resumen de compensación" + "importes del reporte de operaciones erróneas

- Comparar que las cifras obtenidas sean iguales: Suma A = Suma B

- Verificar que el reporte de "archivos procesados" contenga todos los archivos que presentaron los bancos para su proceso.

- En caso de falla, son los encargados de avisar por teléfono a la oficina de sistemas que ha ocurrido un error.

- Informar por teléfono a cómputo que las cifras han checado y que puede proceder a generar los resultados de salida restantes.

- Archiva las tabulaciones realizadas junto con las "cartas presentación" y los reportes de control.

Oficina de sistemas.

Es la encargada de realizar las actualizaciones del sistema operativo, desarrollar y dar mantenimiento a las aplicaciones que se encuentran en el sistema DPS7000.

Sus funciones son:

- Desarrollar programas en ANSI COBOL para satisfacer los requerimientos de los usuarios.
- Instalar las actualizaciones del sistema operativo.
- Dar mantenimiento a los programas ya existentes.
- Resolver problemas de software y operativos que surjan durante el proceso de compensación.
- Implementar mejoras que ayuden a mejorar el rendimiento de los sistemas.

Proveedores externos.

Son los encargados de mantener el hardware y aire acondicionado en condiciones óptimas para el buen funcionamiento de los equipos.

Sus funciones son:

- Responder con tiempos óptimos para solucionar los problemas en los equipos.
- Realizar el mantenimiento preventivo a los equipos.

Una vez revisadas las actividades de cada oficina y la de los proveedores, es importante señalar a manera de resumen las desventajas operativas que se tienen con el equipo y sistema de compensación actual:

- El sistema no puede validar más archivos de entrada, una vez que es ejecutado el proceso para obtener los archivos y reportes de salida.
- El sistema no permite borrar archivos de entrada ya registrados y validados, es necesario volver a inicializar el sistema de compensación electrónica y procesar nuevamente los archivos de entrada.
- El sistema graba los archivos de salida de los bancos en orden ascendente y en caso de error en la unidad de cinta o medio magnético, solamente se puede grabar al finalizar la generación del archivo del último banco.
- Los catálogos de bancos y códigos de transacción están en "hardcode", en consecuencia se deben compilar los programas cada vez que se da de alta un banco.
- Los respaldos de la información tanto de entrada y salida son tardados ya que se realizan en aproximadamente una hora. En el caso del respaldo de la información de los programas y del sistema operativo se tarda de 3 a 4 horas.
- El proceso de recuperación es igualmente tardado ya que para recuperar un archivo se debe llegar secuencialmente hasta la cinta en donde se encuentra ubicado el archivo o programa.
- Las actualizaciones del sistema operativo se realizan en aproximadamente 5 horas.
- No existe redundancia de hardware ni software (no hay sistema de respaldo).
- El mantenimiento es costoso.
- El día que falla el equipo, existe la inseguridad de procesar la información.
- Solamente existen 3 equipos con el mismo hardware y software, estos se encuentran ubicados en México, Guadalajara y Monterrey (ver apéndice B)
- Necesita de instalaciones amplias y adecuadas especialmente, es decir, se requiere de un centro de cómputo (ver apéndice B)
- Existe gran dependencia de los proveedores.

- El equipo tecnológicamente es obsoleto.
- Se programa en ANSI COBOL.
- La capacitación es costosa.

2.2 EL USUARIO Y SUS REQUERIMIENTOS.

Cuando hablamos de un requerimiento nos referimos a alguna característica que es deseable que se incluya en un sistema de software. Lo anterior puede consistir en la inclusión de un formato determinado para la captura de datos, la producción de un tipo específico de salida, entre otros. Es así que la determinación de los requerimientos, vincula el estudio de un sistema existente con la recopilación de detalles relacionados con el mismo.

Dado que los analistas de sistemas no trabajan en el área que finalmente usará el sistema, no tienen los mismos conocimientos, hechos y detalles que los usuarios y personal a cargo de esas áreas. Por consiguiente, el primer paso del analista es comprender la situación. Dar respuesta a un grupo específico de preguntas, será de gran ayuda para comprender los requerimientos básicos. Otros requerimientos dependen de si el sistema esta orientado a transacciones, toma de decisiones o se extiende por varios departamentos. Por ejemplo, la necesidad de informar al gerente de inventarios de un pedido inusualmente grande, que está por llegar, subraya la importancia de establecer una comunicación entre los departamentos de ventas, compras y almacén.

Esta etapa es la base en el planteamiento del diseño del sistema ya que la aceptación del sistema a desarrollar depende en un porcentaje muy alto de una buena identificación de los requerimientos del usuario.

A continuación mencionaremos y describiremos algunos de los requerimientos del usuario para el desarrollo del sistema propuesto en este trabajo.

En este caso en particular, tenemos la situación de que existen diferentes tipos de usuarios del sistema de compensación electrónica de cheques (véase la figura 2.2.1).

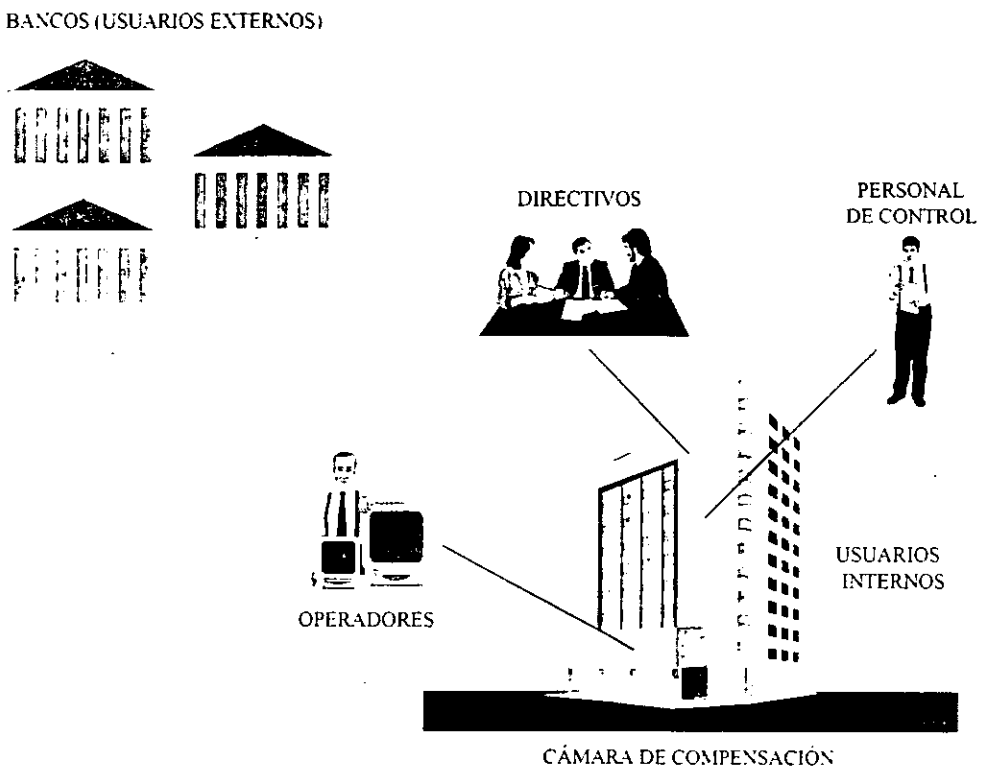


Figura 2.2.1 Usuarios internos y externos.

Estos tipos pueden clasificarse de la siguiente manera:

a) Usuarios Internos.

b) Usuarios Externos.

a) Dentro de los usuarios internos tenemos a los operadores del sistema que son los que interactuan directamente con el mismo. Tenemos personal que realiza funciones de control dentro de todo el proceso de la compensación electrónica y ellos también son usuarios del sistema, aunque no interactuan directamente con él. El personal de control, usa los reportes producidos por el sistema, para fines de validación del proceso

de compensación electrónica de cheques. Existe personal, a nivel gerencial, que también tiene que ver con la información producida por el sistema y ellos también están incluidos dentro de los usuarios internos.

b) Los usuarios externos son los bancos usuarios del servicio de compensación electrónica de cheques. Estos usuarios proporcionan la entrada al sistema, la cual consiste en uno o más archivos de computadora con la información correspondiente a los cheques leídos por el propio banco. A su vez, estos usuarios (bancos) reciben salidas del sistema que consisten en los resultados de la compensación, para ese banco en particular. Es importante mencionar aquí, que los bancos no realizaran modificaciones a sus propios sistemas a causa del cambio del sistema existente en la cámara de compensación bancaria, por lo que las entradas y salidas que ellos presentan y reciben respectivamente, serán las mismas utilizadas hasta la fecha.

Ahora, partiendo del hecho de que se cuenta con un sistema computarizado para llevar a cabo la compensación electrónica: nuestro primer requerimiento es, que el nuevo sistema lleve a cabo por lo menos las mismas funciones que el sistema existente. Este requerimiento global se puede detallar de la siguiente manera:

* Aceptación de los mismos medios de entrada y salida al sistema. Dentro de los medios de entrada y salida actualmente en uso, tenemos las cintas magnéticas y los disquetes. Este requerimiento tiene el fin de que los usuarios externos (bancos), no resientan el cambio del sistema.

* Validación de la información contenida en los medios de entrada. Cada medio de entrada contiene, en un archivo de computadora, la información correspondiente a una remesa de cheques. En este caso el nuevo sistema deberá llevar a cabo la validación de los datos de entrada, tarea que incluye la verificación de cada uno de los registros del archivo. La validación de los registros, a su vez, se lleva a cabo por medio de la validación de los distintos campos que conforman el registro. Para llevar a cabo la

validación de los distintos campos del registro, el nuevo sistema debe poder interpretar el formato especial de los archivos de entrada (véase el apéndice C Formato del archivo de compensación electrónica).

* Dispersión de las operaciones por banco girado. Una vez que la entrada de datos ha sido aceptada y validada, la información es clasificada por medio de un campo dentro de los registros, el cual nos indica el banco receptor de la información. Aquí tenemos el equivalente a separar los cheques físicamente por banco receptor, pero, en la compensación electrónica, no tenemos los cheques físicos sino un registro de computadora por cada uno de ellos.

* Obtención del reporte hoja de compensación (HCOM). Este reporte se describe e ilustra en la sección 1.2.5 (Salidas del proceso).

* Obtención del reporte resumen de compensación. Igual que en el requerimiento anterior, este reporte esta descrito en la sección 1.2.5.

* La producción del reporte de control. Véase la sección 1.2.5.

* Obtención del archivo de liquidación. El uso de este archivo se describe en la sección 1.2.5.

* Generación de los reportes de anomalías presentadas y de anomalías recibidas. Ver la sección 1.2.7 (Compensación Electrónica).

* Generación de archivos para microfichas. Ver sección 1.2.5 para más información.

* Generación de archivo para estadísticas. Descrito de igual manera en la sección 1.2.5.

* Producción de los archivos de salida para cada una de las instituciones que recibieron cheques a su cargo durante el proceso de compensación electrónica. Estos archivos deben estar en el formato establecido, de tal manera que el cambio del sistema sea transparente para los usuarios externos (véase el apéndice C Formato del archivo de compensación electrónica).

* El tiempo en que se debe llevar a cabo el proceso de la información y la generación de los resultados, deberá ser equiparable con el tiempo utilizado por el sistema actual, ya que se tienen horarios establecidos para la entrega de resultados a la banca.

Existe además, el deseo por parte de los usuarios internos, de que se incluyan nuevas características en el sistema propuesto, las cuales se detallan a continuación:

* Un sistema amigable con el usuario. Actualmente las tareas son realizadas a través de comandos de línea. En específico se desea que el nuevo sistema proporcione un ambiente gráfico para su operación.

* Una presentación más útil para el banco en lo que se refiere al reporte de anomalías recibidas.

* La capacidad dentro del sistema para poder recibir y enviar archivos de entrada y salida respectivamente, hacia un servidor de archivos, incorporándose de esta manera un nuevo medio de entrada al sistema. Este nuevo medio de entrada/salida haría el uso de telecomunicaciones, haciendo más rápida y cómoda la entrega y recepción de la información.

* Tener la opción de poder realizar el respaldo de los archivos de entrada y salida en tres medios de almacenamiento diferentes: disquetes, disco duro o cinta magnética. Actualmente sólo se puede hacer el respaldo en cinta magnética.

* Inclusión de un nuevo reporte llamado carta de presentación. En ocasiones, el banco prepara incorrectamente su carta de presentación. Debido a que la carta de presentación ampara lo que el banco dice estar presentando a compensación, no se le puede aceptar si contiene datos incorrectos. El delegado tendría que regresar a las instalaciones del banco que representa para generar una carta de presentación con la información correcta. Con este nuevo reporte. la carta de presentación se puede generar en la cámara de compensación, donde el delegado, una vez revisada la información de la misma, podría firmarla y sustituirla por la incorrecta.

* Inclusión de un nuevo reporte conocido como matriz de distribución de sobres. Este es un formato de control que se produce actualmente en forma manual. Partimos del hecho de que el banco ha leído por sí mismo (sin la intervención de la cámara de compensación), los cheques que presenta a la compensación electrónica. Asimismo, el banco ha realizado la separación física de los cheques por banco girado. Los cheques correspondientes a cada banco receptor, se introducen dentro de un sobre o bolsa de plástico, para su posterior entrega a los bancos girados (receptores). El personal de control tabula en una hoja. los sobres presentados y recibidos por cada banco. Este formato de control tiene el aspecto de una matriz de dos dimensiones que muestra la distribución de los sobres que cada banco presenta y/o recibe, y por esa razón recibe el nombre de matriz de distribución de sobres.

* Generación de un nuevo reporte que muestre el resumen de las transferencias (remesas) ingresadas al sistema. Actualmente. aunque existe un reporte que nos lista las transferencias alimentadas al sistema. dicho reporte no proporciona toda la información que se requiere de cada uno de los archivos de entrada.

* Producción de un nuevo reporte llamado acuse de recibo. Cuando se entregan los resultados de la compensación al delegado del banco, una vez que revisa que sus resultados están completos, pone en una hoja su nombre y firma de conformidad que esta recibiendo la información para el banco que representa. Actualmente esta hoja es

un formato producido en Excel. Se requiere que esta hoja sea producida desde el nuevo sistema y que incluya los datos correspondientes a los sobres recibidos, el total de documentos recibidos y el importe correspondiente a los mismos.

* Inclusión de utilerías. Para facilitar la labor de los usuarios, se requiere incluir opciones dentro del nuevo sistema, que permitan a los usuarios realizar la consulta en pantalla de las transferencias presentadas por un banco en particular; poder borrar una de las transferencias previamente introducidas al sistema y poder corregir ciertos datos de control de las transferencias, en caso de que exista alguna contingencia. Estas funcionalidades no existen en el sistema actual.

* Parametrización por parte del usuario. Actualmente el mantenimiento del sistema y de los datos de configuración del mismo, sólo pueden ser modificados por las personas que dan mantenimiento al software de compensación electrónica. Con el nuevo sistema se pretende reducir el soporte técnico y para esto se requiere que el mantenimiento a la configuración del sistema pueda ser realizada por el administrador del sistema directamente. Dentro de esta configuración esta el mantenimiento a distintos catálogos y otras opciones que dan una mayor funcionalidad al sistema.

Actualmente, existe la situación de que los operadores reciben una capacitación especial para operar el sistema, ya que deben tener un conocimiento de los comandos del sistema operativo para llevar a cabo la operación del software de compensación electrónica. Dicho lo anterior, el enunciado del requerimiento, puede resumirse de la siguiente manera:

* Que el nuevo sistema no requiera de personal especialmente capacitado para su operación.

En general tenemos el siguiente requerimiento:

* Que el sistema propuesto no sea costoso tanto en la parte de software como en la parte de hardware.

Tenemos además el siguiente requerimiento implícito:

* Que el nuevo sistema corra en una PC. Las PCs de uso común actualmente son computadoras con el procesador Pentium en sus distintas modalidades y velocidades de operación. El uso de cualquiera de estas máquinas y sus modelos posteriores, deberían permitir la instalación y la ejecución de la aplicación de manera exitosa. Con este último punto damos por concluida esta sección del usuario y sus requerimientos.

2.3 RECOPILACIÓN Y ANÁLISIS DE LA INFORMACIÓN.
(Metodologías de Recopilación y Análisis de Información).

2.3.1 Técnicas para la recopilación de información.

Existen diversos métodos a fin de recopilar datos sobre la situación existente, entre los más importantes están: la entrevista, el cuestionario, la observación, etc.

2.3.1.1 Entrevista.

Las entrevistas se utilizan para recabar información en forma verbal, a través de preguntas. El encargado de realizar la entrevista puede realizarla de forma individual o en grupos. En las investigaciones de sistemas, las formas cualitativas y cuantitativas de información son importantes. La información cualitativa está relacionada con opiniones, políticas y descripciones narrativas de actividades o problemas, mientras que las descripciones cuantitativas están relacionadas con números, frecuencias o cantidades. A menudo las entrevistas están relacionadas con la información cualitativa, de esta forma este método en muchas ocasiones es la mejor forma de conocer las actividades de algún proceso.

Mucha gente es incapaz de expresarse por escrito y por el contrario verbalmente puede expresar perfectamente sus ideas. Como resultado de esto, las entrevistas pueden descubrir rápidamente los malos entendidos, falsas expectativas o incluso resistencia potencial para las aplicaciones en desarrollo.

La estructura de las entrevistas varía. Si el objetivo de la entrevista radica en adquirir información general, es conveniente elaborar una serie de preguntas sin estructura, con una sesión de preguntas y respuestas libres. Y en caso de requerir datos más específicos sobre la aplicación o se desea asegurar una alta confiabilidad en las respuestas, las entrevistas estructuradas son las más convenientes.

Realizar entrevistas toma tiempo, por tanto no es posible utilizar este método para recopilar toda la información que se necesite en la investigación. Dado que un limitado número de personas será seleccionado para la entrevista, se debe tener cuidado de incluir a aquellas personas que tienen información que no se pueda encontrar en otra parte. Durante las primeras etapas de un estudio de sistemas, cuando los analistas determinan la factibilidad del proyecto, con frecuencia las entrevistas sólo se aplican en todos los niveles gerenciales y de empleados, y dependen de quién pueda proporcionar la mayor parte de la información útil para el estudio.

La habilidad del entrevistador es vital para el éxito en la búsqueda de hechos por medio de la entrevista. Las buenas entrevistas dependen del conocimiento del analista tanto de la preparación del objetivo de una entrevista específica como de las preguntas por realizar a una persona determinada. El tacto, la imparcialidad e incluso la vestimenta apropiada ayudan a asegurar una entrevista exitosa. La falta de estos factores puede reducir cualquier oportunidad de éxito.

Puede suceder también que quienes responden, ocasionen situaciones difíciles, lo que puede originar que se minimice el esfuerzo de la investigación con preguntas evasivas o incompletas. Así mismo los analistas deben observar cualquier intento de culpar a otras áreas o personas de los problemas existentes.

2.3.1.2 Cuestionario.

Los cuestionarios proporcionan una alternativa muy útil para las entrevistas; sin embargo existen ciertas características que pueden ser apropiadas en algunas situaciones e inapropiadas en otras. Así mismo, los cuestionarios pueden ser la única forma para que los analistas se relacionen con un gran número de personas para conocer varios aspectos del sistema. Por supuesto no es posible a través de este método observar las expresiones o reacciones de quienes responden los cuestionarios, lo que también puede ser una ventaja ya que se puede asegurar que el interpelado cuenta con mayor anonimato y pueden darse respuesta más honestas. Una de las

desventajas de los cuestionarios es que a pesar de poder aplicarse a un gran número de personas, es muy rara una respuesta total. Puede necesitarse de algún seguimiento a los cuestionarios para motivar al personal que corresponda.

El desarrollo y distribución de los cuestionarios es caro, por lo tanto, el tiempo invertido en esto debe utilizarse inteligentemente. También es importante el formato y el contenido de las preguntas en la recopilación de hechos significativos.

Existen dos formas de cuestionarios para recabar información: cuestionarios abiertos y cerrados, y se aplican dependiendo de sí se conocen todas las posibles respuesta, toda vez que los cuestionarios cerrados limitan las posibles respuestas de los interrogados.

2.3.1.3 Revisión de Registros.

En la revisión de registros se examinan los datos y descripciones que ya están escritos o registrados y en relación con el sistema. El término registro se refiere a los manuales escritos sobre políticas, regulaciones y procedimientos de operaciones estándar. Una desventaja es que en la mayor parte de las empresas los manuales y estándares sobre procedimientos de operación usualmente son obsoletos.

2.3.1.4 Observación.

La observación proporciona información de primera mano en relación con la forma en que se llevan las actividades. La observación en si es un arte, saber qué buscar y cómo guiar su resultado requiere de experiencia, sin embargo, el resultado de una buena observación es de gran utilidad al proporcionar de primera mano la forma en que se llevan los documentos, los procesos y si ocurren pasos específicos.

2.3.2 Recopilación de Información.

Un punto que no se debe perder de vista en todo análisis es el de informar a las personas que están ligadas con los procesos a automatizar, sobre la importancia del estudio que se efectuará para implantar el nuevo sistema, así como los beneficios que se obtendrán al término de éste y, por consiguiente, la necesidad de contar con información verídica proveniente de ellos para poder cumplir con los objetivos.

La recopilación y el análisis de datos deben ser realizados por un equipo integrado por los usuarios, el departamento de desarrollo de sistema y el grupo de administración de información. A continuación definiremos cuáles son las partes involucradas para el sistema de compensación electrónica de cheques, las cuales juegan un papel importante para la integración de dicha información.

Las áreas involucradas son (ver figura 2.3.2.1):

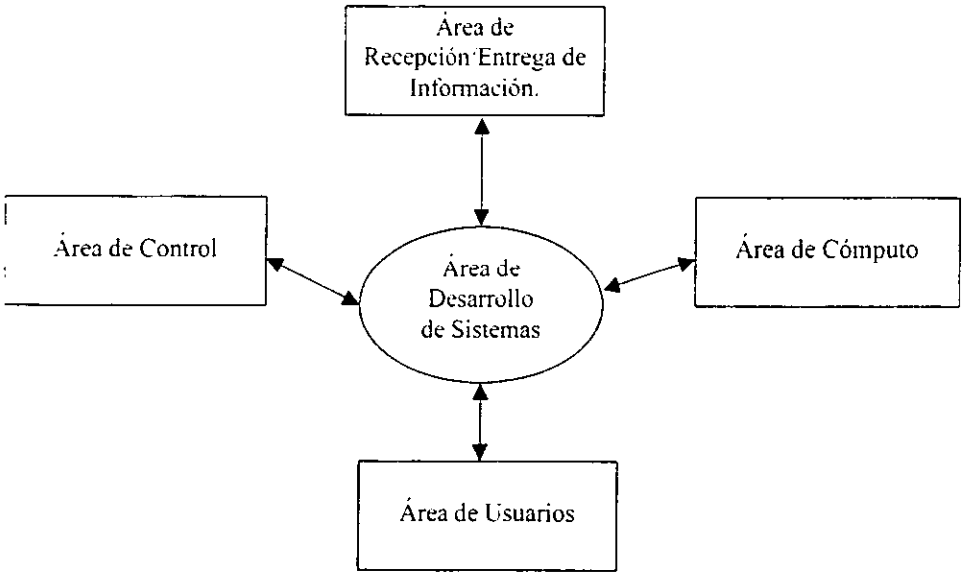


Figura 2.3.2.1 Áreas involucradas para la recopilación de información.

Siguiendo las metodologías para la recopilación de información, se llevaron acabo las siguientes actividades:

- Identificación de las áreas involucradas para ser entrevistadas.
- Se elaboraron listas del personal a entrevistar.
- Formulación de cuestionarios acerca del problema.
- Listado de requerimientos.

Con lo que se pretende obtener:

- Qué información llega, y que validación requiere.
- Cómo se procesa la información.
- A quién se le envía la información procesada y cómo se le envía.

2.3.3 Análisis de la Información.

El análisis de datos es de gran importancia ya que se puede responder a preguntas esenciales como son: ¿Qué procesos integran en sistema?, ¿Qué datos emplea cada proceso?, ¿Qué datos son almacenados? y ¿Qué datos ingresan y abandonan el sistema?.

El análisis de sistemas conoce el papel central que tiene la información en las organizaciones. Seguir el flujo de datos por todos los procesos, es la finalidad del análisis de la información, ya que dice mucho sobre cómo se alcanzan los objetivos de la organización; en el transcurso del manejo de transacciones y terminación de tareas los datos entran, son procesados, almacenados, recuperados, analizados, utilizados, cambiados y presentados como salidas. El análisis de la información estudia el empleo de los datos en cada actividad, documenta los hallazgos con diagramas que muestran en forma gráfica la relación entre procesos y datos, y en los diccionarios de datos que describen de manera formal los datos del sistema y los sitios donde son utilizados.

El objetivo primario en el análisis de la información es el de proporcionar las bases para el diseño de un sistema con un enfoque disciplinado, que nos permita clasificar la información existente en ciertos procesos, con la que podamos manipular las relaciones de datos necesarias para el usuario.

Independientemente del tamaño o complejidad del sistema, se deben realizar actividades coordinadas para llegar a la raíz del problema o necesidad y definir los requerimientos del sistema.

Hemos de recordar, que el proyecto que se pretende desarrollar tiene como finalidad reemplazar un sistema poco funcional, obteniendo un sistema mejorado y con un adecuado manejo de información para la compensación electrónica de cheques.

Por lo que se analizaron las siguientes tareas:

- Revisión de antecedentes. Se revisaron los procedimientos del proceso anterior, llegando a la conclusión de que no hay un sistema formal para el control de actividades para la recepción/entrega de información.

- Revisión del proceso actual y de las necesidades de información provenientes de los usuarios tanto internos como externos, basándonos en los resultados de las entrevistas y sesiones de trabajo. para determinar las necesidades de información para la generación de reportes tanto de entrada como de salida de información.

- Análisis de los procesos del sistema anterior a ser reemplazados, con la revisión de documentación técnica y problemas detectados tanto por el personal que da mantenimiento al equipo, como del área de cómputo.

Para el análisis de la información del presente proyecto, se puede clasificar los requerimientos del sistema en los siguientes módulos:

Requerimiento	Seguimiento
Recepción de la Información	Los respectivos delegados de los bancos, entregan "Carta Presentación", (Relación de la información que presenta el banco, con un total de documentos y total monetario global, datos del banco y fecha, sirviendo de acuse de recibo, con copia para CECOBAN), llevando además remesas de cheques y/o cintas o disquetes.
Validación de la Información de Entrada.	Se captura la información contenida en las cintas o disquetes, y se verifica cada uno de los registros y campos del archivo, donde cada registro corresponde a un cheque de un determinado banco.
Proceso de Dispersión de Operaciones por Banco Girado.	Clasificación de los registros por banco receptor.
Proceso de Generación de Reportes	Se obtienen diversos tipos de reportes, tanto de la compensación, como de anomalías, estadísticas, etc.
Producción de Archivos de Salidas para los Bancos, resultado del proceso de la compensación.	Por medio de este archivo los bancos reciben los cheques a su cargo originados por el proceso de compensación electrónica en un formato ya establecido.
Validación de la Información de Salida.	El área de Control revisa que los montos económicos resultado de la compensación electrónica cuadre con el monto de dinero de entrada.
Respaldo de Información	Se lleva un respaldo de la información de Entrada/Salida.

2.4 PLANTEAMIENTO DEL PROBLEMA.

Introducción.

Para encontrar la solución de un problema, primero debemos ser capaces de encontrar el problema y formularlo de manera que sea factible someterlo a una investigación.

Cualquier sistema implementado tiene un ciclo de vida que comienza con el planteamiento como una idea conceptual para convertirla en diseño. Si es factible el proyecto. se desarrolla e implementa para ponerse en operación, es decir, si el sistema cumple con todos los requerimientos del usuario y satisface sus necesidades.

El costo económico y la complejidad tecnológica de los sistemas que actualmente implementa el hombre son tan grandes, que es necesario planear estratégicamente los proyectos. analizando diversas opciones y tomando en cuenta todos los aspectos del proyecto.

Es asi como el éxito en el diseño de un sistema depende totalmente de lograr un completo entendimiento de las necesidades de la empresa y su organización, así como el ambiente en el cual habrá de operar.

Todo elemento desarrollado por el hombre primero es una idea en su mente. Los sistemas computacionales. como otros productos de la tecnología, se desarrollan en respuesta a los requerimientos detectados. Las fuentes que producen las ideas de productos de programación incluyen las necesidades del cliente generadas externamente. las necesidades internas de la organización, planes de mercadotecnia y los planes organizacionales.

El primer paso en la planeación de un proyecto de programación es preparar, en la terminología del cliente, un enunciado breve del problema que solucionará y de las

restricciones que existen en su resolución. El enunciado definitivo del problema incluye una descripción de la situación actual y de las metas que debe lograr el nuevo sistema.

La definición del problema requiere de un entendimiento cabal del dominio del problema y del entorno de éste. Las técnicas para obtener este conocimiento, por parte del planeador, son entrevistas con el cliente, observación de las tareas problemáticas, y desarrollo de las reales.

El segundo paso en la planeación de un proyecto de programación es determinar lo apropiado de una solución computacional. Para ser eficiente en costo, un nuevo producto de programación debe proporcionar los mismos servicios e información que el sistema antiguo, usando menos tiempo y personal, o proporcionar servicios e información que antes eran inaccesibles.

Habiendo determinado que es apropiada una solución computarizada para el problema, la atención se centra en las funciones de los principales subsistemas del sistema computacional. Un sistema computacional está formado por los subsistemas de personal, equipo y de productos de programación más las interconexiones entre ellos. El primer subsistema incluye operadores, personal de mantenimiento y usuarios finales. El segundo comprende el equipo de cómputo y los dispositivos periféricos, y puede tener otros dispositivos como sensores y accionadores para control de proceso. El tercer subsistema contiene programas que deben desarrollarse, más programas que ya existen y que pueden emplearse como están o modificándolos.

Las funciones que debe realizar cada subsistema principal se deben identificar, se deben establecer las interacciones entre subsistemas y determinar las restricciones en el desarrollo y operación para cada subsistema principal. Las limitaciones especifican número y tipo de equipos, cantidad y habilidades del personal, y características del producto de programación como funcionamiento, precisión y nivel de confiabilidad.

Dado el enunciado preciso del problema y la indicación de las restricciones que existen para su solución, se pueden formular metas y requisitos preliminares.
Por ejemplo, todo producto de programación debe ser útil, confiable, comprensible y eficiente en costos. Otras metas, como transportabilidad, entrega anticipada de subsistemas, y la facilidad de uso para los no programadores, dependerán de la situación particular.

Los requisitos especifican las capacidades que debe tener un sistema para la solución del problema. Éstos se establecen para la funcionalidad, el rendimiento, el equipo, la programación en el equipo, la programación y las interfaces con el usuario.

Definición del problema.

En la actualidad el uso de cheques se ha convertido en parte cotidiana de nuestras vidas, diariamente muchos de estos documentos son emitidos para su cobro. En nuestro país existen muchas instituciones bancarias y los clientes tienen la libertad de elegir en que banco desean abrir una cuenta de acuerdo a sus intereses y preferencias particulares. Seria muy inconveniente e impráctico que los clientes tuvieran que abrir una cuenta en cada banco para asi poder cobrar un cheque que llegaran a recibir algún día a cargo de algún banco en particular, por lo anterior, es que existen lo que se llama compensación de cheques en la cual las instituciones bancarias se reúnen en un centro de compensación, donde se lleva a cabo el intercambio de los documentos y el pago de los mismos entre las instituciones participantes.

El proceso de compensación de cheques, dado los volúmenes que se manejan diariamente, no podría realizarse de manera manual, por lo que se hace necesaria la existencia de sistemas de cómputo que permitan llevar a cabo esta tarea de manera confiable y rápida, debido a que este proceso de compensación no puede ser pospuesto para el día siguiente, por que los clientes de los bancos no podrían cobrar el importe del cheque que depositaron un día antes.

Para realizar el proceso de compensación de cheques se utiliza una unidad lectora de cheques la cual transforma la información contenida en los cheques físicos en registros lógicos y se procesa mediante software.

Actualmente la empresa cuenta con un sistema de compensación de cheques que le permite llevar a cabo el proceso de compensación electrónica. Sin embargo este sistema le genera grandes costos tanto en mantenimiento de equipo, instalaciones, personal, entre otros. Además los insumos y periféricos del equipo de cómputo son muy costosos. Asimismo el sistema actual no cumple con los requerimientos acordados para garantizar su funcionamiento adecuado durante y después del año 2000 tanto en lo que se refiere al hardware como al software; además de que la actualización del software es muy costosa y se requiere de personal especializado para manejarlo.

Otro factor que genera grandes costos a la empresa es la cantidad de personal que interviene en el proceso de compensación; ya que hay cuatro ingenieros y dos operadores que brindan el servicio de soporte técnico.

El sistema existente no permite a la empresa prestar este tipo de servicio en todas las sucursales que tiene en la República Mexicana; por que resulta una solución muy costosa que implica un alto costo del servicio para sus clientes.

En el sistema actual, cuando se esta realizando el proceso de compensación y existe un error en la transferencia de información de una institución bancaria, el sistema tiene que ser inicializado en un punto intermedio del proceso, de ésta manera, el proceso de compensación vuelve a iniciar con la información correcta. Esto provoca pérdida de tiempo en el proceso.

La información se respalda semanalmente en cintas magnéticas, este proceso es muy lento y consume muchos recursos del sistema. También la recuperación de la información es muy lenta, además, este proceso solo puede llevarse a cabo por el personal que opera el equipo.

Otros factores que inciden en el problema de lentitud son las características que presenta el equipo, ya que los modelos no son los últimos del mercado. El sistema actual utiliza consolas de operador, terminales tontas y usa comandos de línea para llevar a cabo todas las tareas que requieren los operadores del mismo, además de que es necesaria una capacitación especial para operar el sistema y dar mantenimiento a las aplicaciones.

El equipo instalado necesita un espacio amplio, ya que requiere de un sistema de aire acondicionado y un equipo de energía ininterrumpida.

En la empresa existe el área de cómputo (lugar donde se realiza el proceso de compensación) y el área de recepción y entrega (llamada también área de mostrador: en donde se reciben y entregan las entradas y salidas del sistema), las cuales en ocasiones no sincronizan su trabajo. Ahora se planea que las mismas personas que operan el sistema sean las que reciban la información, es decir, llevar el sistema al mostrador, debido a que son dos áreas que se encuentran físicamente distantes.

Además existe el área de control, la cual compara la información generada por el área de cómputo, con la información que cada banco presenta. Después de verificar que las cifras globales resultado de la compensación cuadren con la información de entrada, ésta unidad avisa al área de cómputo que la información es correcta, en este momento se generan las cintas y estados de cuenta que serán enviadas a los bancos correspondientes a través del área de recepción y entrega.

Otra de las áreas que intervienen es el área de sistemas, en caso de existir algún problema, apoyan para resolver alguna falla de software y operativos que surjan durante el proceso de compensación.

En la figura 2.4.1 muestra las áreas que se relacionan en el proceso de compensación, como se mencionó antes hay una relación muy estrecha entre éstas áreas.

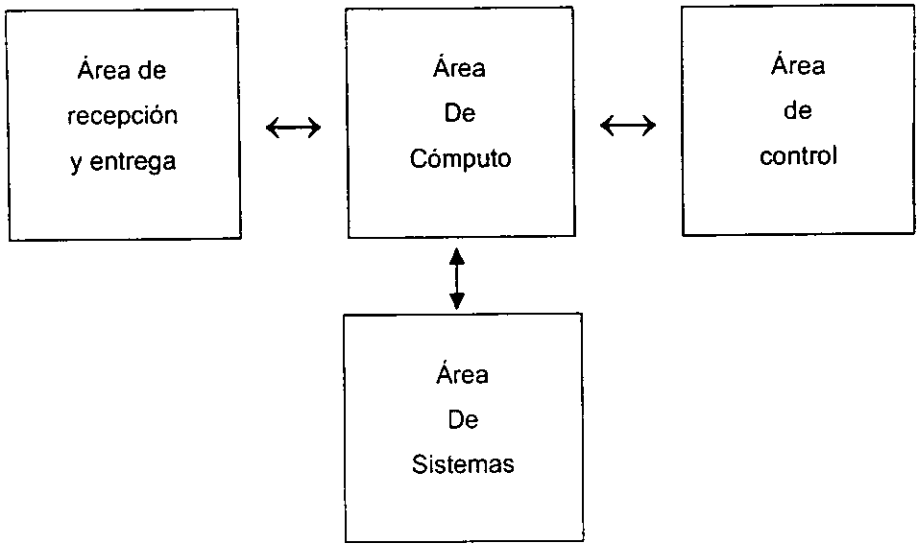


Figura 2.4.1 Áreas involucradas en el proceso de compensación.

Objetivo.

La identificación del objetivo del sistema es de suma importancia para el diseño del mismo, ya que se debe determinar lo que la empresa desea que realice el sistema a desarrollar.

Este proceso deberá determinar si el uso de los sistemas de información podrá lograr las metas de la empresa y encaminarlos a la construcción del mismo.

El objetivo general es diseñar e implementar un sistema de compensación electrónica de cheques en PC para una empresa de compensación bancaria, el cual debe aceptar y validar los archivos de entrada y producir al menos los mismos reportes y archivos de salida que el sistema existente, mismo que funciona en un equipo mainframe.

Con el sistema propuesto se pretende crear un sistema que funcione en una PC sin sacrificar la calidad, los resultados o los tiempos del proceso y además permitir que usando la infraestructura existente en las sucursales de la empresa (equipos PC) pueda

proporcionar el servicio de compensación electrónica de cheques a un costo competitivo. Con el uso de Pc's, el área de cómputo se reduce considerablemente.

Otro fin que se persigue es eliminar la necesidad de soporte técnico especializado tanto en la parte del hardware como en la del software ya que actualmente intervienen cuatro ingenieros de servicio y dos operadores para llevar a cabo esta tarea. Con el sistema propuesto el soporte se reduce a una persona, siendo posible realizar el soporte técnico a distancia, a través de una línea telefónica.

Además, el sistema propuesto contará con una interfaz gráfica que permita al usuario accesar al sistema de manera fácil y rápida, de esta manera el usuario se olvidará del uso de comandos de línea para manejar el sistema.

Para el desarrollo del sistema propuesto se utilizará un lenguaje visual para crear la interface del usuario (front end), y se hará uso del lenguaje de programación estructurada para programar las rutinas propias del sistema (back end).

También, el sistema propuesto tendrá una herramienta, en la cual se podrá corregir la información sin que sea necesario inicializar el sistema.

Otro de los propósitos del sistema es evitar la dependencia de la empresa con el proveedor de desarrollo, ya que el sistema actual fue comprado en el extranjero y el personal tiene que viajar para recibir la capacitación adecuada.

Ahora se planea que las mismas personas encargadas del área de recepción y entrega y el área de cómputo, sean las que operen el área de control, de esta manera habrá reducción de tiempo y costos. En la figura 2.4.2 se muestran las áreas antes descritas.

Con el sistema propuesto, se tiene como objetivo que la empresa funcione con mayor rentabilidad, eficiencia, flexibilidad y productividad al contar con un sistema que le permita proporcionar este servicio a nivel nacional con un mínimo de inversión.

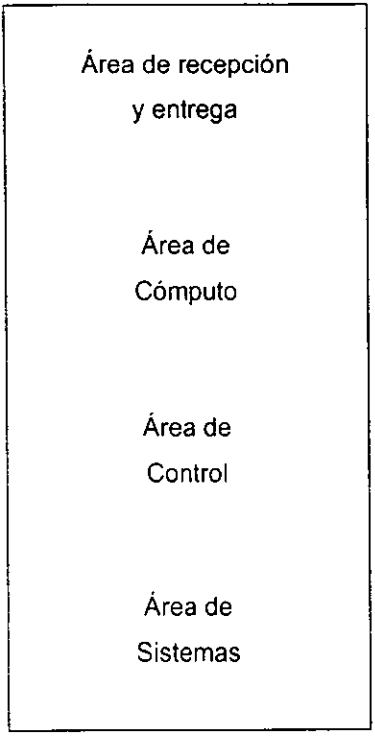


Figura 2.4.2. Áreas del proceso de compensación en el sistema propuesto.

2.5 DESCOMPOSICIÓN FUNCIONAL.

2.5.1 Introducción.

Un sistema o procedimiento es un conjunto de operaciones o de etapas que en forma cronológica se establecen para llevar a cabo un determinado tipo de trabajo.

Los diagramas de funcionalidad u operacionales son diagramas que presentan en forma gráfica la secuencia en que se realizan las operaciones de un determinado procedimiento y/o el recorrido de las formas o materiales. Con el fin de tener una visión de conjunto de un sistema o procedimiento se utilizan lo diagramas de flujo donde se muestran las unidades orgánicas o puestos que intervienen en el procedimiento, las operaciones que realizan, la secuencia de las mismas y el equipo utilizado en cada caso.

2.5.1.1 Técnicas de simplificación de operaciones.

El análisis del sistema o procedimiento implica descomponerlo en sus partes para conocer la naturaleza, los objetivos, los responsables de su ejecución, el lugar y tiempo en que se deber realizar.

Para analizar la información que permita después llevar a cabo un proceso de simplificación de operaciones se debe buscar las respuestas a las siguientes preguntas:

- ¿Qué trabajo se hace?. Se pretende determinar el propósito y objetivo de la naturaleza o tipo de labores que se realizan en la unidad orgánica y los resultados que se obtienen.

- ¿Quién lo hace?. Se intenta definir que unidades orgánicas o personas intervienen en el procedimiento y además las aptitudes del personal para la realización de una tarea especifica y la actitud que adopta en la relación de la misma.

- ¿Cómo se hace?. Se busca determinar los medios con los que se cuenta para realizar la tarea, que comprende los métodos y técnicas aplicados a la empresa y a los equipos e instrumentos utilizados.

- ¿Cuándo se hace?. Se determina el orden o secuencia en que se desarrollan las actividades de la unidad orgánica, es decir, la ubicación geográfica y domicilio de las oficinas, la funcionalidad de estás y la distribución del espacio de las mismas.

- ¿Porqué se hace?. Se busca la justificación de la existencia de ese trabajo o de su procedimiento. Se pretende conocer los objetivos de las acciones que integran el trabajo o procedimiento al igual que la primera pregunta.

El analista establece como resultado no sólo la descripción de las operaciones del sistema o procedimiento en forma más precisa y definida, sino también la posibilidad de mejorarlo y para ello se tiene la siguiente alternativa:

Eliminar el trabajo innecesario. Se parte del principio de que todo sistema o procedimiento es perfectible y por ello es necesario mejorarlo, eliminando esfuerzos y tareas innecesarias. Por ejemplo, el exceso en el llenado de formas, de firmas, registros, etc.

Modificar las operaciones o partes de ellas. Es conveniente subdividir el trabajo pero no tan detalladamente que haga ineficiente el procedimiento o sistema, al someterlo a repetidas revisiones que repercutan en pérdida de tiempo y energía. Por ejemplo se pueden cambiar dos operaciones en una sola o bien modificar el procedimiento al incorporar operaciones que lo hacen más lógico, completo y congruente.

Cambiar el orden de las operaciones. El simple cambio en el orden de las operaciones puede hacer más eficiente un procedimiento; por ejemplo eliminar el desplazamiento inútil de documentos.

Simplificar las operaciones necesarias. Aquí es necesario estudiar más detalladamente cada una de las operaciones del procedimiento para determinar posibilidades de simplificación. Pueden aplicarse los estudios de tiempo y movimientos, de esquemas del lugar de trabajo, etc.

Modificar las actividades de los puestos y equilibrar cargas de trabajo. Cuando se observa en algunos puestos que se consignan los asuntos por negligencia, falta de adiestramiento del personal o mala distribución de la carga de trabajo. Es necesario en los dos primeros casos controlar, motivar y capacitar al personal y en el último caso, replantear las actividades entre los empleados que participan en los procedimientos.

Seleccionar un trabajo importante. El seleccionar un trabajo requiere de una consideración detenida, ya que sólo un trabajo de gran volumen o técnica difícil ofrecerá la oportunidad para efectuar una simplificación de trabajo considerable.

Dividirlo. Es el proceso de clasificar los detalles o componentes de un trabajo, sistema o procedimiento, de tal manera que pueda examinarse en detalle. Para ello, es posible utilizar un diagrama.

Hacer preguntas sobre los detalles, con una mente abierta. Constituye un requisito absolutamente necesario para la preparación de los diagramas, ya que sólo un completo conocimiento puede producir una visión total de las operaciones para pasar luego al análisis que conduce a las mejoras.

Desarrollar las mejoras que se propondrán. Constituye el resultado final de la división, cuadros e interrogatorios y requiere una buena dosis de ingenio, imaginación y lógica.

Instalar las mejoras. Requiere la participación y cooperación de los empleados para que se pueda instalar la o las mejoras del sistema o procedimiento.

2.5.2 Procedimiento de la Compensación.

2.5.2.1 Antecedentes.

La Cámara de Compensación Bancaria es el órgano regulador de las instituciones bancarias en sus actividades de compensación.

Se regula por el Banco de México, teniendo como actividad principal, el intercambio de documentos negociables entre las diferentes instituciones de crédito, con el fin de saldar cuentas entre las mismas.

El funcionamiento de la Cámara de Compensación, es vigilado por: la Comisión Nacional Bancaria y de Valores, El Banco de México, la Secretaría de Hacienda y Crédito Público, y por las demás autoridades que a éste lo regulan.

2.5.2.2 Proceso de Compensación.

Para llevar a cabo el procedimiento de compensación se requieren los siguientes elementos:

1. Que las instituciones que operan en una plaza, región, o en todo el territorio nacional, tengan relaciones comerciales entre ellas y aceptan pagar sus deudas y créditos recíprocos mediante tal procedimiento.

2. Que se presenten formalmente todos los documentos para su relación y operación, y que los saldos se asienten en las cuentas que lleva el Instituto Central (Banco de México) con las instituciones bancarias. Sin que sea necesario pagar en efectivo los impuestos correspondientes.

3. El procedimiento puede abarcar una pieza, región o toda la República.

4. Que los créditos presentados aparezcan a favor de los bancos que utilicen el procedimiento compensatorio, y se deriven de títulos propios o que estos hayan sido presentados por el cliente para su cobro.

5. Que las operaciones de compensación se celebren en un local destinado para ello por el Instituto Central, en este caso en la Cámara de Compensación Bancaria (Cecoban).

La Ley del Banco de México establece en su articulo 24 fracc. XXVI "El Banco de México, prestará los servicios de compensación local. Los usuarios del servicio de compensación local serán las Instituciones de Crédito asociadas al Banco de México, que radiquen en las plazas en que tenga establecidas sucursales o agencias generales en lugares aledaños; y las no asociadas que sean autorizadas por el Banco de México debiendo tener una cuenta de cheques con el banco".

El reglamento del servicio de compensación por zona y nacional establece en su artículo 3°, "Que serán sujetos a cámara de compensación los cheques y giros bancarios a la vista con cargo de las instituciones de crédito en que tengan oficinas en la plaza en que se proporcione el servicio. Podrían también aceptarse otra clase de documentos a la vista que autorice el Banco de México".

El documento presentado a compensación de acuerdo a su reglamento tendrá que tener un sello especial de la institución respectiva que contenga, la fecha, el acuse de recibo y el número de la institución bancaria que envié ha compensación este título de crédito. Ningún documento será admitido a compensación si no lleva el sello de referencia. No es requisito indispensable para su pago, que los documentos estén suscritos por las personas que habitualmente estén autorizados para ello.

Para entender mejor como se realiza el procedimiento de compensación, es necesario dividir geográficamente las zonas en: local, por zona y nacional, esta división esta determinada por el Reglamento de Compensación Local y Nacional.

Es importante mencionar que el procedimiento que se efectúa en cualquiera de las tres divisiones realizadas a la compensación, es igual para cada una de ellas.

La compensación se divide en dos fases:

A) Compensación previa.

B) Compensación definitiva.

2.5.2.3 Compensación previa

La previa empieza a partir de las 16:30 horas todos los días hábiles del calendario que para tal efecto haya aprobado la Comisión Nacional Bancaria y de Valores, separando el delegado de la institución los documentos para ser enviados a compensar, asimismo se clasifican por las instituciones que libran los documentos que van a ser presentados para su cobro. Esta etapa es propiamente el intercambio o el canje de los documentos, cuya operación consiste, en que cada institución entrega a los bancos librados usuarios del servicio. los documentos que ha negociado durante el día.

Una vez que el delegado ha llegado al local de la cámara de compensación, se encarga de presentar los sobres a los delegados de las instituciones correspondientes y a su vez recibe de los delegados respectivos de las otras instituciones los sobres del banco que representa.

Luego de efectuado el intercambio de sobres, el delegado procederá a contar los documentos recibidos y las cantidades que amparen dichos documentos anotando seguidamente en la "hoja de compensación" y en el renglón correspondiente al banco que representa la cantidad que recibe de títulos de crédito y su valor. Hecho lo anterior el delegado efectúa la suma total de los documentos que fueron presentados. debiendo coincidir exactamente el total de los documentos presentados con los recibidos.

En resumen la "compensación previa", consiste en el intercambio de documentos entre los bancos y en las anotaciones correspondientes en la "hoja de compensación", para que el encargado de la cámara en base a esta hoja realice la liquidación previa de los documentos que son presentados.

2.5.2.4 Compensación definitiva.

Una vez terminada la primera fase de la compensación, los delegados se retiran a sus oficinas respectivas en donde proceden a examinar y comprobar los documentos que les han sido entregados; al día siguiente, se reúnen nuevamente los delegados en el local de la Cámara de Compensación, para efectuar la compensación definitiva; en ésta etapa los delegados bancarios se devuelven los documentos que se entregaron en la "previa", se hace la revisión correspondiente a cada documento y si por alguna causa es rechazado, se deberá en todos los casos anexar a los documentos un volante que se especifique la causa de devolución "... si el documento cumple con los requisitos exigidos para su compensación, cada delegado formulará un recibo en el que manifieste la conformidad de la institución que representa para que le sea cargado o acreditado el saldo resultante. Después se entrega al encargado del servicio su respectiva hoja de compensación, misma que lleva adjunta la ficha o recibo de conformidad con el saldo".

El encargado procede a su vez a formular la "hoja de compensación final" en donde efectúa la suma de los saldos deudores y de los saldos acreedores, debiendo corresponder a cada caso ambas cantidades.

Se entiende por acreedor. "Las instituciones que presento documentos con un valor mayor a los recibidos a su cargo".

Se entiende por deudor. "Las instituciones que recibieron documentos a su cargo con un valor mayor a los que presento a las demás instituciones".

Finalmente, con base en las sumas iguales obtenidas en la "hoja de compensación final" el encargado formulará dos fichas globales de cargo y abono respectivamente, las cuales junto con la "hoja de compensación final" y con la copia de los recibos de conformidad elaborados por los delegados se remite al departamento de contabilidad del Banco de México.

Cabe mencionar que si algún delegado "No se presenta dentro del término señalado, éste deberá a indicación del Banco de México, recibir los documentos a su cargo, sin que por esa recepción tenga derecho a presentar los que tenga a su cargo de las otras instituciones", deberá pagar una multa que establece en el reglamento de la cámara de compensación por zona. es la gestión de cobro que cada una de las oficinas del Banco de México realiza por conducto de sus corresponsales que operan en su jurisdicción. para crédito a sus cuentas.

Las oficinas del Banco de México al recibir los documentos, acusarán de recibo correspondiente a los bancos remitentes por la vía más rápida, revisarán las cartas remesas, las hojas resumen y se presentarán a compensación de la cámara local, con el fin de abonar sus importes a los bancos un día después de su recibo.

Los documentos sobre plazas distintas a las de la ubicación de la oficina del Banco de México, serán revisados por dichas plaza, les estamparán un sello al dorso del documento que hace constar que fueron tramitados por conducto del Banco de México y los enviarán de inmediato a los bancos girados o a sus bancos corresponsales al cobro, estando obligados unos y otros a proceder de inmediato a la liquidación de los documentos a fin de hacer eficaz el sistema de compensación.

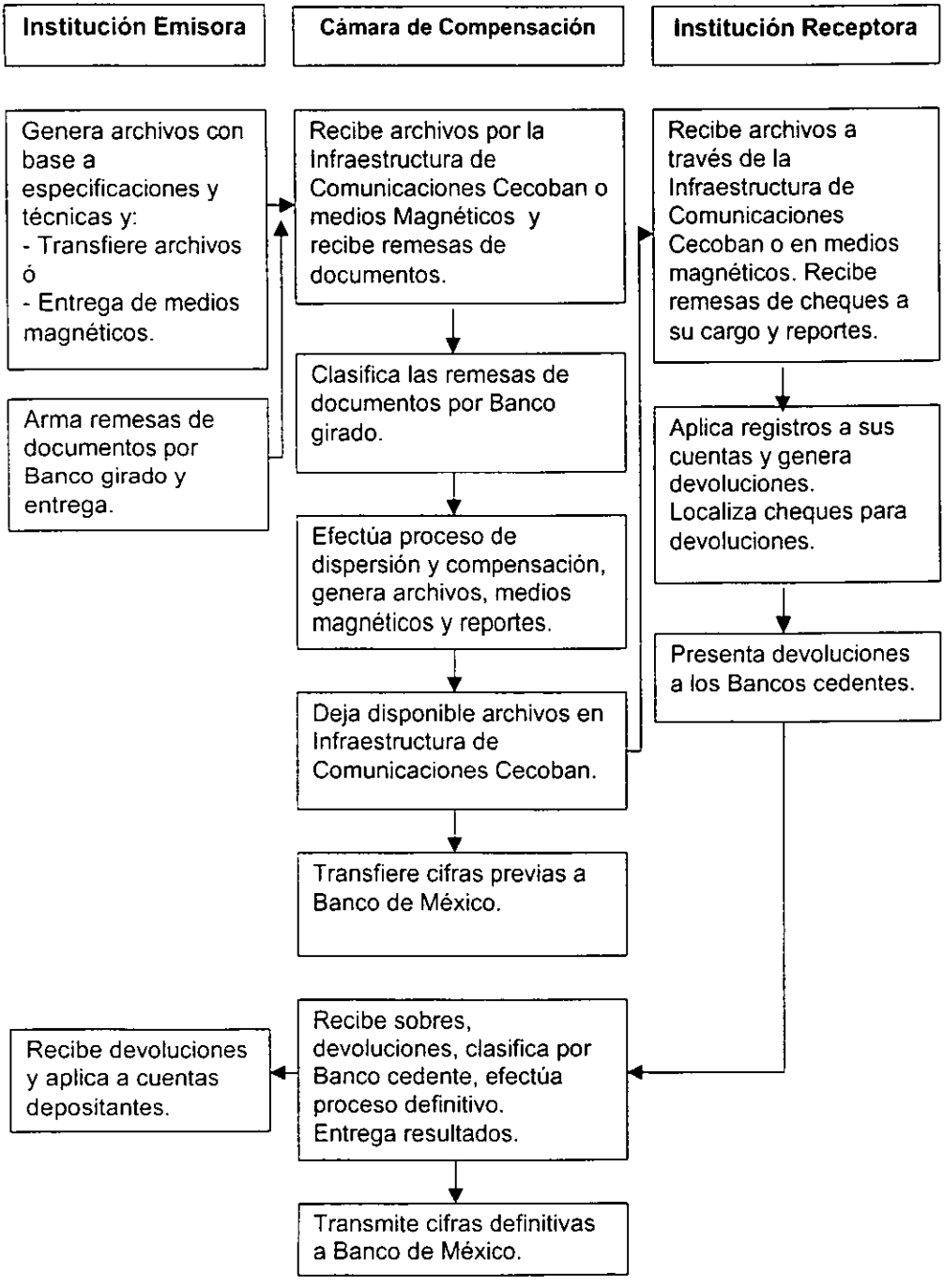
El Banco de México realizará la compensación respectiva tres días después de haber recibido los documentos. cargando sus importes a los bancos girados o a sus bancos corresponsales, según sea el caso.

La compensación nacional, se efectúa de la siguiente forma: "Los bancos que tengan documentos sobre zonas distintas a las oficinas del Banco de México a cuya jurisdicción estén adscritos, los enviarán directamente a aquella a la cual le corresponda la plaza del girado".

2.5.2.5 Diagrama Funcional de la Compensación Bancaria Electrónica de Cheques.

En la figura 2.5.2.5.1 se muestra el diagrama funcional del proceso de Compensación Electrónica de Cheques, en el que se observan las tres partes que intervienen en el mismo: Institución Emisora, Cámara de Compensación e Institución Receptora.

Figura 2.5.2.5.1, Proceso de Compensación Electrónica de Cheques



2.5.2.6 Diagrama Funcional del Proceso de Compensación Electrónica de Cheques.

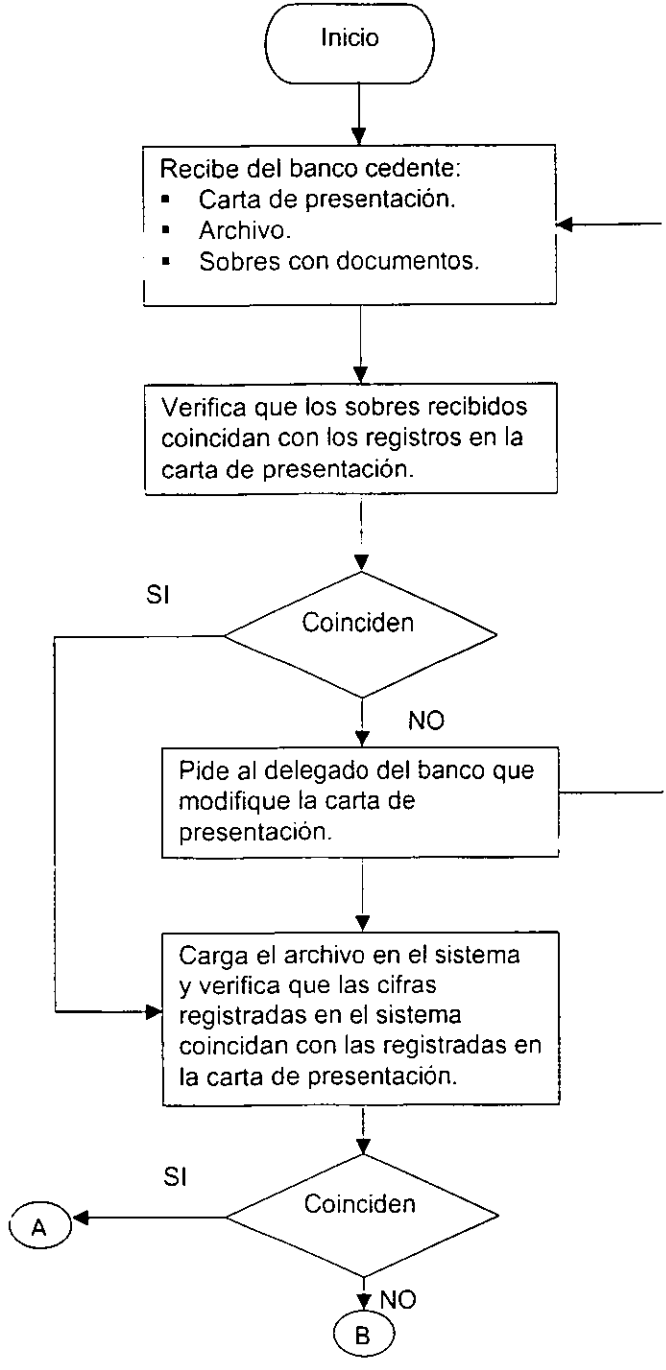
En la figura 2.5.2.6.1 se muestran las diferentes etapas del proceso de compensación electrónica de cheques, proceso que lleva a cabo la Cecoban todos los días hábiles del año.

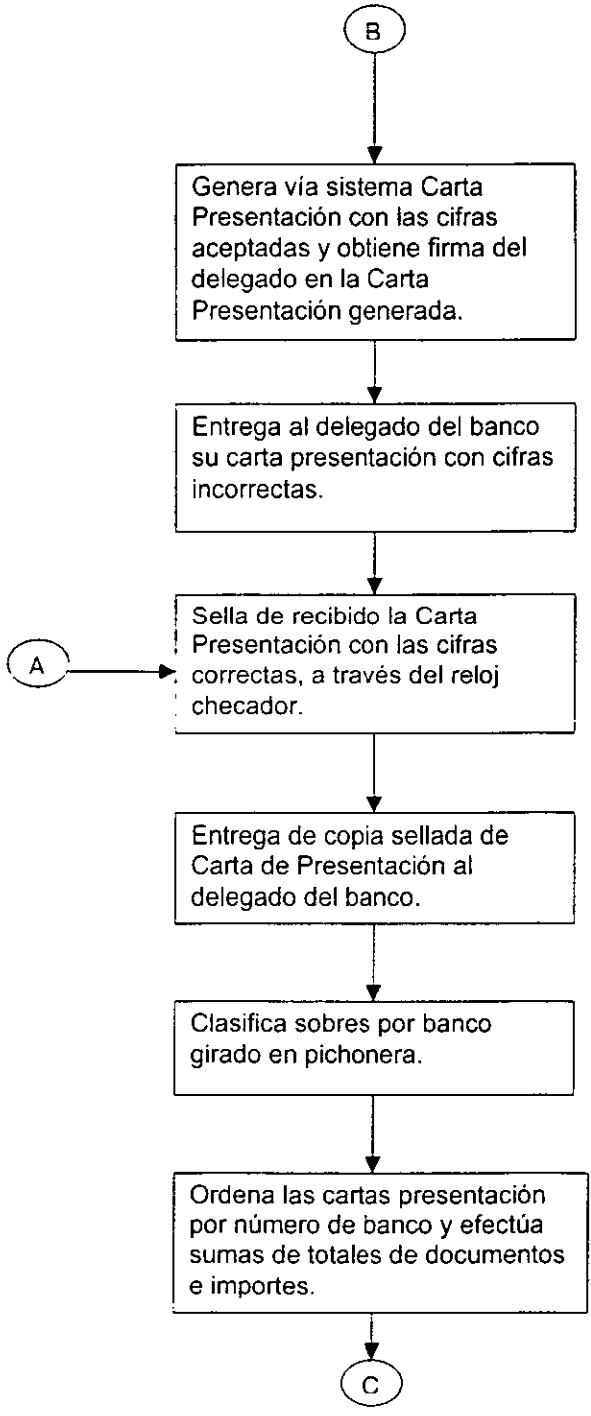
Destacan en el diagrama las condiciones que deben de cumplirse para que se lleve a cabo el proceso de compensación, tales como son que la carta presentación cuente con la información requerida (número de documentos e importe total de los mismos), así como que el archivo generado en el banco correspondiente sea presentado bajo los lineamientos establecidos para tal fin.

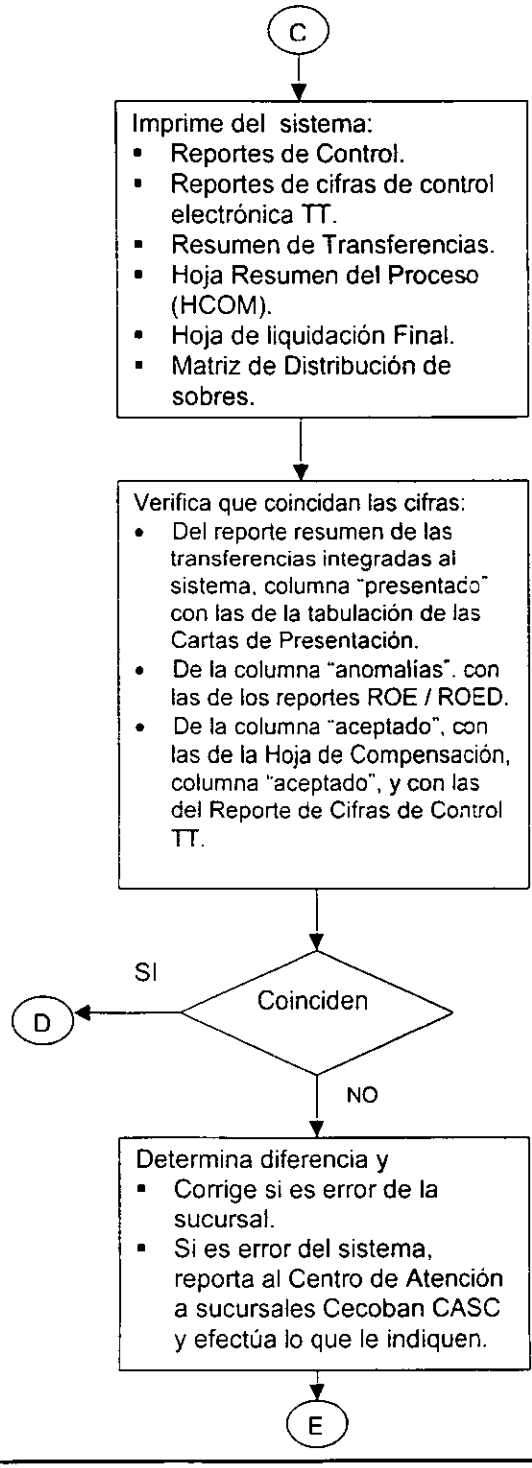
Como resultado del proceso se generan una serie de reportes que sirven para tanto para afectar las cuentas corrientes que los bancos tienen con el Banco de México, para que éstos hagan las afectaciones a las cuentas de sus clientes. Así mismo algunos reportes son utilizados por parte de la Cecoban, para dar validez al proceso y en caso de alguna reclamación contar con los soportes documentales correspondientes.

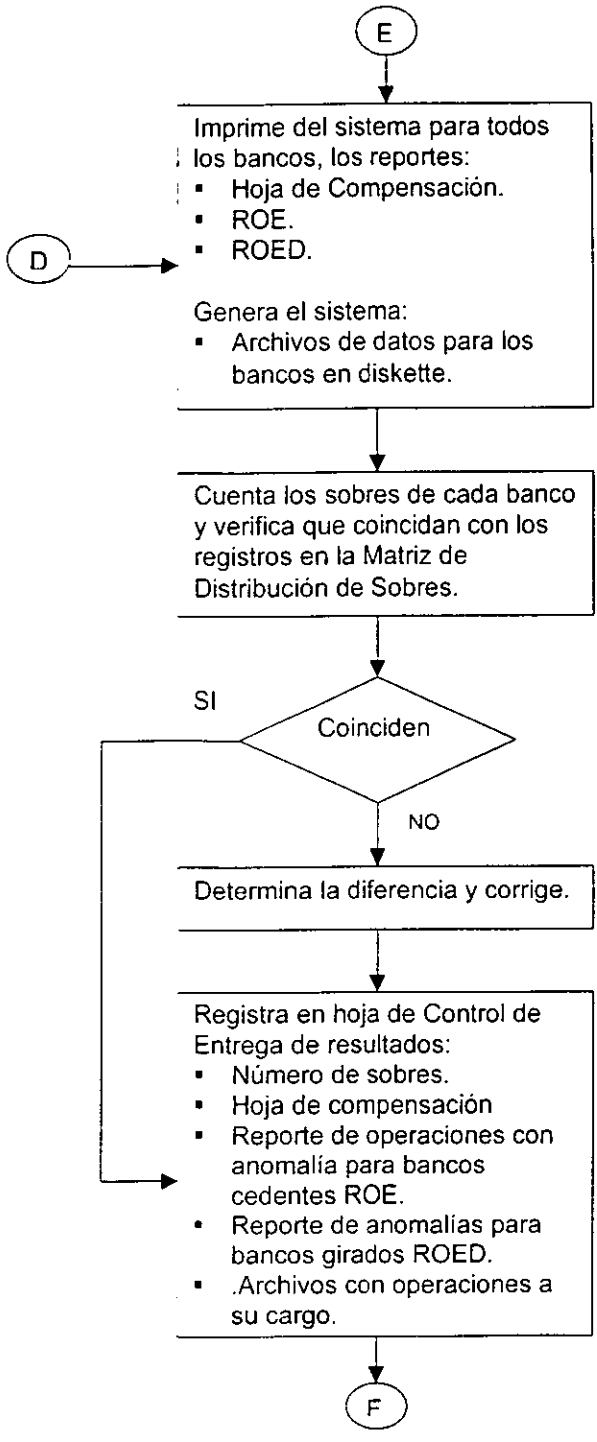
Otro resultado que se genera del proceso es el archivo que se envia al Banco de México, así como el correspondiente a cada banco, para que éste último haga de manera automatizada los abonos y cargos respectivos a las cuentas de sus clientes.

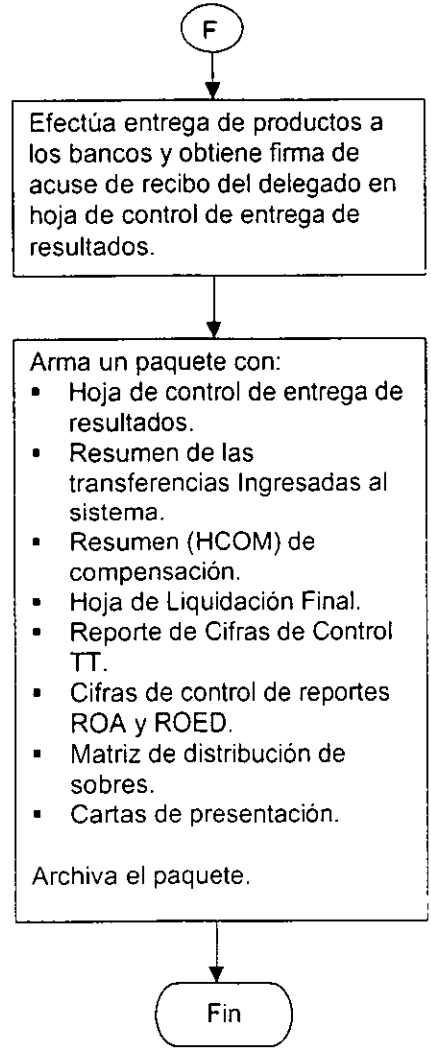
Figura 2.5.2.6.1 Diagrama Funcional del Proceso de Compensación Electrónica de Cheques




B
Genera vía sistema Carta Presentación con las cifras aceptadas y obtiene firma del delegado en la Carta Presentación generada.
Entrega al delegado del banco su carta presentación con cifras incorrectas.
A
Sella de recibido la Carta Presentación con las cifras correctas, a través del reloj checador.
Entrega de copia sellada de Carta de Presentación al delegado del banco.
Clasifica sobres por banco girado en pichonera.
Ordena las cartas presentación por número de banco y efectúa sumas de totales de documentos e importes.
C


C
Imprime del sistema:
Reportes de Control.
Reportes de cifras de control electrónica TT.
Resumen de Transferencias.
Hoja Resumen del Proceso (HCOM).
Hoja de liquidación Final.
Matriz de Distribución de sobres.
Verifica que coincidan las cifras:
Del reporte resumen de las transferencias integradas al sistema, columna "presentado" con las de la tabulación de las Cartas de Presentación.
De la columna "anomalías", con las de los reportes ROE / ROED.
De la columna "aceptado", con las de la Hoja de Compensación, columna "aceptado", y con las del Reporte de Cifras de Control TT.
Coinciden
SI
D
NO
Determina diferencia y
Corrige si es error de la sucursal.
Si es error del sistema, reporta al Centro de Atención a sucursales Cecoban CASC y efectúa lo que le indiquen.
E


E
Imprime del sistema para todos los bancos, los reportes:
Hoja de Compensación.
ROE.
ROED.
D
Genera el sistema:
Archivos de datos para los bancos en diskette.
Cuenta los sobres de cada banco y verifica que coincidan con los registros en la Matriz de Distribución de Sobres.
SI
Coinciden
NO
Determina la diferencia y corrige.
Registra en hoja de Control de Entrega de resultados:
Número de sobres.
Hoja de compensación
Reporte de operaciones con anomalía para bancos cedentes ROE.
Reporte de anomalías para bancos girados ROED.
.Archivos con operaciones a su cargo.
F


F
Efectúa entrega de productos a los bancos y obtiene firma de acuse de recibo del delegado en hoja de control de entrega de resultados.
Arma un paquete con:
Hoja de control de entrega de resultados.
Resumen de las transferencias Ingresadas al sistema.
Resumen (HCOM) de compensación.
Hoja de Liquidación Final.
Reporte de Cifras de Control TT.
Cifras de control de reportes ROA y ROED.
Matriz de distribución de sobres.
Cartas de presentación.
Archiva el paquete.
Fin

2.6 OPCIONES DE SOLUCIÓN.

Selección de la Plataforma de Desarrollo de Software y Hardware.

Los modelos más importantes para el diseñador son el modelo de implantación de sistemas y el modelo de implantación de programas.

El modelo de implantación de sistemas, a su vez se divide en un modelo del procesador y uno de tareas. La primera tarea a la que nos enfrentamos es decidir como asignar el modelo esencial (la parte automatizada) a las piezas principales de hardware y software del sistema.

Para realizar estas asignaciones se deben de tener en cuenta varios factores, los principales son:

Criterios de selección.

- **Costo:** Dependiendo de la naturaleza del sistema, la solución más económica puede ser un grupo de micro computadoras de bajo costo; para otros sería más práctico y económico hacer la implantación con la infraestructura existente en la organización.

- **Eficiencia:** El tiempo de respuesta de los sistemas de línea es de suma importancia, por lo tanto se debe escoger procesadores y dispositivos de almacenamiento de datos suficientemente rápidos y poderosos para satisfacer los requerimientos de desempeño del sistema.

- **Seguridad:** El usuario final puede tener requerimientos de seguridad que dicten que algunos o todos los datos delicados se coloquen en lugares protegidos.

Restricciones Políticas y Operacionales.

La configuración de hardware puede verse influenciada también por restricciones políticas impuestas por el usuario final, por otros niveles de administración dentro de la organización o por el departamento encargado de todos los sistemas de cómputo.

Para la elección del software a emplear en el desarrollo del sistema, además de los criterios mencionados anteriormente, existen algunos más que deben tomarse en cuenta. Se ha estado escribiendo programas de computación desde que se desarrollaron las primeras computadoras de propósito general.

Los programas se escriben con lenguajes de programación, es conveniente agrupar los distintos lenguajes de programación en cuatro generaciones distintas:

Lenguajes de primera generación: Fueron los lenguajes de máquina que se usaron en los años 50. Los programadores que intentaban que la computadora hiciera algo útil codificaban sus instrucciones con unos y ceros binarios.

Lenguajes de segunda generación: Son los sucesores del lenguaje de máquina, generalmente se conocen como lenguajes de ensamble o ensambladores. Estos lenguajes son de bajo nivel en el sentido de que el programador tiene que escribir una declaración por cada instrucción de máquina. En lugar de pensar en términos del problema que se quiere resolver, se debe pensar en términos de la máquina.

Lenguajes de tercera generación: Son la norma actual. incluyen BASIC, COBOL, FORTRAN, PASCAL, C, y muchos más. Son de alto nivel en el sentido de que una sola declaración usualmente representa cinco o diez declaraciones de lenguaje ensamblador.

Los lenguajes de tercera generación también se caracterizan como lenguajes guiados por procedimientos. Requieren que el programador piense con cuidado la secuencia de los cálculos o procedimientos necesarios para lograr alguna acción.

Lenguajes de cuarta generación: Los lenguajes de cuarta generación o 4GLs, son la moda actual y son considerados como el desarrollo más importante en el campo del software en los últimos 20 años.

La mayor parte tiene características de programación estructurada ausentes en los lenguajes de tercera generación. En lo particular, la mayoría de los detalles tediosos de programación relacionados con introducir datos a la computadora se ocultan al programador.

Sin tomar en cuenta el lenguaje de programación que se use, existen puntos comunes que se deben considerar. los más comunes son:

- **Productividad:** Probablemente, la cuestión más importante de la programación actual sea la productividad: escribir más software, más rápidamente. Exceptuando casos raros de productividad se considera más importante actualmente que la eficiencia.

- **Eficiencia:** En algunas aplicaciones. la eficiencia sigue siendo de importancia. Esto sucede en muchos sistemas de tiempo real, y puede darse en otros tipos de sistemas que procesan grandes volúmenes de datos. La eficiencia usualmente entra en conflicto con otras metas: si se emplea mucho tiempo en el desarrollo de un programa eficiente, es probable que sea menos mantenible y menos transportable.

- **Corrección:** Se puede argumentar que esto es lo más importante, si el programa no funciona correctamente. no importa que tan eficiente sea. Se prefieren lenguajes de programación como ADA y PASCAL si la corrección es de importancia crítica, por

que son de tipos rígidos y el lenguaje revisa todo cuidadosamente para evitar referencias ilegales a los datos.

- **Portabilidad:** El usuario puede desear ejecutar el mismo sistema en varios tipos distintos de computadoras. Algunos lenguajes de programación son más portátiles que otros, esto es más cierto en lenguajes de tercera generación (C, PASCAL, COBOL, etc.) que en los de cuarta. No existe un lenguaje universalmente portátil, por ello, además del lenguaje de programación es necesario tener en cuenta el estilo de programación si la portabilidad es un factor importante.

- **Mantenimiento:** Debemos recordar que los sistemas viven durante mucho tiempo, por lo que el software debe tener mantenimiento.

Opciones de solución.

Actualmente se cuenta con una gran cantidad de opciones para la solución de problemas relacionados con el proceso de diseño e implantación de un sistema de cómputo. Esto se debe al gran desarrollo de la industria cibernética y se puede observar desde un nivel físico (diferentes tipos de hardware); pasando por un nivel medio o de interprete (diferentes sistemas operativos): hasta un nivel de desarrollo (diferentes herramientas de programación). Ahora, si tomamos en cuenta que dentro de cada una de nuestras opciones de desarrollo contamos con varias filosofías de programación, algoritmos, etc.; es obvio pensar que nuestra elección surgirá de entre muchas posibles.

Un enfoque para encontrar la mejor solución, es centrarse en el problema principal y a partir de él, derivar la opción más adecuada para el desarrollo; por supuesto previendo todos los requerimientos que debe tener un buen sistema. Como ejemplo de estos se tienen la relación costo-beneficio, el tiempo de entrega, el tiempo para adaptarse al sistema, la compatibilidad con otras plataformas, etc. Figura 2.6.1.

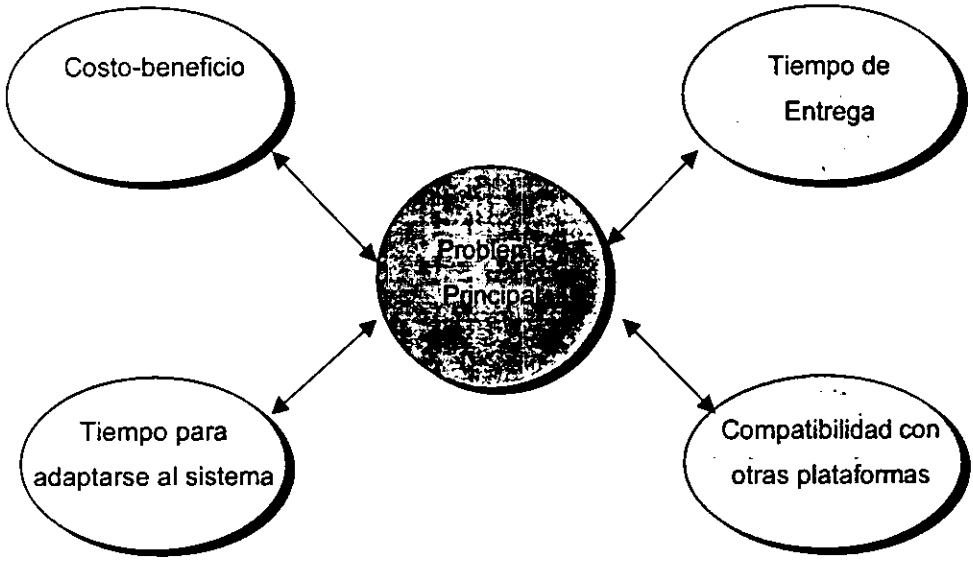


Figura 2.6.1. Requerimientos que debe tener un buen sistema

Es necesario saber que la elección de una solución, no es del todo libre; hay que considerar que se puede contar con la opción de utilizar la infraestructura actual o comprar nuevos recursos. En el caso de utilizar los recursos existentes, tenemos que las opciones disminuyen en número; pero dentro de estas se puede encontrar la solución óptima tomando en cuenta las políticas y normas de la empresa. es decir, proporcionar el mayor rendimiento con el mínimo de recursos.

Plataforma del sistema.

La plataforma del sistema se refiere al software y hardware sobre el cual se desarrollará y ejecutará el sistema de compensación electrónica de cheques. Las opciones que se tienen para el desarrollo del sistema son:

Ambiente DOS.

Este medio es muy útil y veloz, pero las herramientas gráficas y el software que existen para el desarrollo de aplicaciones, donde se requiere de una buena interfaz gráfica, son limitadas, puesto que dichas herramientas requieren de mucha programación.

Ambiente Windows.

La interfaz con el usuario es una de las herramientas más importantes de Windows, ésta consiste del uso de ventanas, menús y elementos gráficos que facilitan el uso de los sistemas. El manejo y uso de estos elementos es general para cualquier aplicación y muchas utilizan dibujos o iconos para representar unidades de disco, archivos, subdirectorios, comandos de operación del sistema y acciones.

Tipos de software para el desarrollo del sistema.

A continuación se presentan los puntos más importantes de varios productos para desarrollar la interface del usuario (front-end) y las rutinas propias del sistema (back-end).

El software que se estudiará para llevar a cabo la programación del sistema de compensación electrónica de cheques es el siguiente:

- Visual Foxpro
- Access 97
- Delphi
- Visual Basic

- C++

- Borland Pascal 7.0

Programas para desarrollar sistemas de bases de datos.

La aparición de la computadora personal en el mercado y su rápida evolución tecnológica, así como el abatimiento de costos, han propiciado su uso en gran escala. En consecuencia ha surgido una gran cantidad de programas para desarrollar bases de datos.

La demanda de acceso del usuario a bases de datos ha apresurado a los desarrolladores de software a diseñar interfaces realistas. Recientes adelantos técnicos (procesadores más veloces y eficientes, memoria expandida, monitores de alta resolución) han sido de gran ayuda.

El objetivo de un sistema de bases de datos es facilitar el acceso a los datos. Las vistas de alto nivel ayudan a lograrlo. Si el tiempo de respuesta para una consulta es demasiado largo, el valor del sistema se reduce. El funcionamiento del sistema depende de la eficiencia de las estructuras de datos utilizados para representar los datos y de qué tan eficiente pueda operar el sistema con esas estructuras. Como sucede en muchos otros aspectos de los sistemas de cómputo, deben hacerse concesiones, no solo en el espacio y el tiempo, sino también entre la eficiencia de un tipo de información y la de otro.

Uno de los aspectos más importantes de los manejadores de bases de datos más recientes es que todos logran que ese poder sea más accesible con una serie de menús, editores de pantalla y características automáticas que vuelvan relativamente simple la realización de operaciones relacionales.

Los sistemas manejadores de bases de datos relacionales (RDBMS) han sido altamente aceptados por la forma en que se manejan los datos. Los sistemas relacionales ofrecen los siguientes beneficios:

- Acceso sencillo a los datos.
- Flexibilidad en el modelado de los datos.
- Disminución de la redundancia de datos y el nivel de almacenamiento.
- Existe independencia entre el almacenamiento físico y el diseño lógico de datos.
- Alto nivel de manipulación de datos.

El manejador de bases de datos es un módulo de programas que constituyen la interfaz entre los datos de bajo nivel, almacenados en la base de datos y los programas de aplicación; por ejemplo, los que permiten hacer consultas.

VISUAL FOXPRO.

Microsoft Visual Foxpro ofrece a los desarrolladores las herramientas necesarias para manejar datos (como organizar tablas y ejecutar consultas), crear un sistema manejador de bases de datos relacionales integral (DBMS), o programar una aplicación completa de manejo de datos desarrollada para usuarios finales.

Características.

1. Capacidad de desarrollo de aplicaciones que permita incrementar la productividad. El lenguaje Xbase se ha diseñado para permitir la creación de objetos, clases y subclases. Los objetos pueden ser creados visualmente y reusados en cualquier

momento. Además Visual Foxpro permite a los usuarios conjuntar componentes (objetos) que también pueden ser reusados.

2. Proporcionar el mejor conjunto de herramientas para lograr un gran desempeño en el desarrollo cliente/servidor. Por medio de la tecnología de conectividad a bases de datos (Open Database Connectivity, ODBS), los usuarios pueden crear conexiones con otras bases de datos SQL, haciendo de Visual Foxpro una excelente herramienta para construir aplicaciones cliente/servidor.

3. Integrar Microsoft Visual Foxpro con la familia Microsoft, permitiendo el desarrollo de aplicaciones que tomen ventaja de otros productos y tecnologías. Visual Foxpro opera con otros programas de aplicación por medio de componentes OLE y APIs (Aplication Programming Interfaces).

4. Proporcionar una solución de escalabilidad para preservar inversiones existentes en Foxpro 2.x.

5. Programación orientada a objetos. El diseño orientado a objetos y la programación orientada a objetos representan un cambio en el enfoque de los procedimientos de programación normales. Las clases y los objetos son dos conceptos fundamentales de la programación orientada a objetos. Una clase contiene información de cómo debe ser y como debe comportarse un objeto.

6. Desarrollo de aplicaciones con manejo de eventos.

7. Herramientas de desarrollo visuales:

   - Project manager

   - Database designer

- Form designer

- Visual Class Designer

Visual Foxpro proporciona herramientas para construir aplicaciones que integren cliente/servidor e Internet. Las herramientas de diseño y su orientación a objetos, el lenguaje orientado a datos, el manejo rápido de datos y sus capacidades de creación de componentes.

Requerimientos de hardware.

- Computadora con procesador 386 ó superior.

- Windows 95 ó Windows NT 3.5 ó posterior.

- 8 MB en RAM (12 MB recomendable).

- 15 ó 80 MB disponibles en disco duro, dependiendo del tipo de instalación.

- Monitor VGA ó superior, se recomienda usar SVGA.

Ventajas.

Foxpro es un manejador de bases de datos bastante versátil y eficaz, entre las ventajas que ofrece se encuentran las siguientes:

- Su software también es válido para Windows.

- Cuenta con ambiente interactivo.

- Puede invocar al lenguaje C y al lenguaje ensamblador.

- Ofrece generador de informes y de etiquetas.

- Permite la fácil creación de menús.

- Admite programación para ambiente multiusuario.

- Acepta variables de cadena, ventanas e índices que solo están limitados por la capacidad de memoria de la computadora en que se este trabajando.

Desventajas.

Entre las desventajas que representa Foxpro, se encuentran las siguientes:

- La versión normal de Foxpro no cuenta con compilador.

- Para poder obtener programas autónomos *.EXE es necesaria la adquisición de un Kit de compilación.

ACCESS 97.

Cuenta con características que lo hacen un producto fácil de utilizar, potente y flexible. Es accesible para todos los niveles de usuarios, desde principiantes hasta diseñadores de bases de datos.

Microsoft Access es un programa para el manejo de bases de datos relacionales que permite el almacenamiento, agrupamiento y búsqueda rápida de todo tipo de datos, indispensables en las labores diarias de una empresa, que se combina con la facilidad de uso que ofrece Microsoft Windows.

Las bases de datos de Access están constituidas por diversos objetos, como lo son: las tablas, consultas, formularios, informes, macros y módulos.

La estructura de Access no es muy compleja, sin embargo, requiere de varios elementos que lo hacen que sea potente y al mismo tiempo fácil de manejar por los usuarios, desde bases de datos sencillas como una agenda o directorio telefónico hasta aquellas grandes y complejas como un inventario o una base de datos integral que controle las compras, ventas inventario y nómina de una empresa.

Características.

- Permite cambiar el nombre de la columna para que ajuste mejor.
- Es posible filtrar una tabla para presentar solamente la información que deseamos ver.
- Se puede esquematizar el diseño de nuestra base de datos o ver todas las relaciones a la vez, utilizando la ventana de relaciones.
- Permite ajustar la posición de los controles de un formulario desplazando un solo punto en la cuadrícula.
- Su mayor desventaja es que no puede crear archivos ejecutables.

Requerimientos de hardware y software de Microsoft Access 97.

- Computadora personal con procesador 486 ó superior.
- Sistema operativo Windows 85, Windows NT 3.51 ó posterior.
- 12 MB en RAM para Windows 95 y 16 MB para estaciones de trabajo Windows NT.
- De 28 a 60 MB libres en disco duro. Para una instalación típica se necesitan 40 MB.

- Drive para CD-ROM.

- CD-ROM que contenga Internet Explorer, manejadores adicionales y archivos con extensión AVI.

- Adaptador de vídeo VGA, como mínimo. Se recomienda SVGA a 256 colores.

- Microsoft Mouse, Microsoft IntteliMouse o cualquier señalador compatible.

Ahora, observaremos en cuadros comparativos los manejadores de bases de datos descritos anteriormente.

ACCESO A LA INFORMACIÓN	VISUAL FOXPRO	ACCESS 97
Habilidad para trabajar con varios formatos de datos.	✓	✓
Conectividad completa con todos los formatos de datos.	✓	
Crea nuevas tablas en cualquier formato.		✓
Salva las tablas en cualquier formato de base de datos.		✓
PRODUCTIVIDAD	**VISUAL FOXPRO**	**ACCESS 97**
Asistentes expertos para facilitar el desarrollo inicial de las aplicaciones.	✓	✓
Infobox para personalización de todos los objetos.	✓	✓
Asistentes expertos para la simplificación de tareas complicadas.	✓	✓
Asistente paso a paso con consultas complejas.	✓	✓
Consultas preprogramadas.		✓
Asistente de reportes paso a paso.	✓	✓
Asistente paso a paso para la creación de tablas cruzadas.		✓

PERSONALIZACIÓN		
Infobox para una fácil personalización de todos los objetos.	✓	✓
Asistentes expertos para construir rápidamente formas, reportes y demás.	✓	✓
Un verdadero generador de reportes para su edición rápida y precisa.	✓	✓
Variedad de validaciones de campos y valores predeterminados.	✓	✓
Soporte de lenguaje de programación visual.	✓	✓
OTRAS	VISUAL FOXPRO	ACCESS 97
Compatibilidad con dBASE para DOS.		✓
Lenguaje que soporte modelo de objetos (OOP).	✓	
Ayuda en línea completamente sensitiva al contexto.	✓	✓
Asistentes expertos.	✓	✓
Uso de controles.	✓	✓
Llama directamente a los APIs de Windows.	✓	✓
Acceso transparente a datos locales y SQL.	✓	✓

Lenguajes de programación (visualizadores).

Delphi.

Esta herramienta combina el poder de los compiladores tradicionales de tercera generación con el fácil uso rápido desarrollo del ambiente 4 GL. Esta basado en Pascal Orientado a Objetos(es una extensión significativa de Pascal 7.0 de Borland).

Requerimientos.

- Procesador 486 DX/66 Mhz ó superior.

- Microsoft Windows ó Windows NT 4.0 (Service Pack 2) ó NT 3.51 (Service Pack 5).

- 12 MB en RAM. Se recomienda 16 MB en RAM.

- 60 MB disponibles en disco duro para la instalación completa del programa.

- Drive para CD-ROM.

- Monitor VGA ó de mayor resolución.

- Mouse u otro dispositivo señalador.

- Soporte de red (cualquier red compatible con Windows).

Ventajas.

- El desempeño es significativamente mejor por que genera en la compilación, archivos ejecutables. Realiza enlaces inteligentes que activa la optimización de segmentos, por lo que el archivo ejecutable reduce su tamaño más del 30%, lo cual permite una carga rápida del archivo y una ganancia adicional en su desempeño.

- Puede compilar archivos ejecutables (.EXE) independientes así como reutilizar las bibliotecas de enlace dinámico (Dynamic Linked Libraries, DLL). Por último también permite a los programadores profesionales escribir directamente en código ensamblador para tener el control directo del microprocesador.

- La conectividad a la base de datos es nativa si se utiliza la base de datos de Borland. Sin embargo, también soporta enlaces a otros tipos de datos vía ODBC.

- Tiene más controles de construcción que aumentan el formato de los iconos, a través de una paleta de componentes multipágina. Tiene una extensa galería de plantillas.

- Aumenta la capacidad de colocación de los objetos.

- Aumenta la lista de propiedades de modificación.

- Tiene dos formas de sincronización de código de pantalla.

- Particiona funciones y eventos.

- La paleta de componentes es organizada en página de block de notas, desplegando los iconos en un simple renglón con formato de barra de lista.

- Incluye plantillas reconstruidas que hacen más fácil el desarrollo de aplicaciones estándar o componentes complejos como pantallas MDI, formas de bases de datos, diálogos multipáginas y cajas de listas dobles. La arquitectura es completamente extensible, permitiendo a los desarrolladores un fácil registro de sus propias plantillas dentro de la galería.

- Facilita el diseño visual con características tales como la alineación automática y el dimensionamiento del objeto.

- Provee un menú de propiedades que puede ser accesado directamente para hacer más eficientes e intuitivas las modificaciones.

- El código de edición de pantalla de Delphi sincroniza todas las representaciones de diseño visuales con el código básico. En otras palabras, como la aplicación es construida por la colocación de objetos en una forma, el código correspondiente es generado simultáneamente. No hay limitaciones, el código siempre esta accesible y los desarrolladores pueden intercambiar instantáneamente entre el editor de código y las herramientas de diseño, permitiendo un modo más eficiente para cada parte del proyecto.

- Delphi provee un rastreador (debugger) completo con interrupciones en condiciones de ejecución. La pantalla del rastreador y las vistas pueden ser guardadas de sesión en sesión, permitiendo a los desarrolladores crear un ambiente confortable. Incluye un poderoso navegador (browser) similar al utilizado en Borland C++, el cual provee en desplegado comprensivo de objetos y clases.

Las ventajas relacionadas con el manejo de base de datos son:

Los desarrolladores pueden colocar una base de datos como un campo de tal manera que puede tener un conjunto de propiedades y bidireccionar la comunicación vía ODBC compatible con la base de datos.

El componente de la base de datos puede ser usado como un mecanismo para navegar a través de la base de datos, utilizando flechas representando al siguiente y al registro previo.

Las consultas (queries) SQL pueden ser definidas por código. En la forma que una consulta puede estar dentro de la base de datos.

Maneja un soporte extenso de bases de datos que incluyen la base de datos de Borland (BDE) para Paradox y acceso a dBASE.

Desventajas.

- Es una herramienta poco difundida.

- Su límite de cadenas es de 255 caracteres, sin embargo, existen cadenas de terminación cero que pueden ser 65,535 caracteres de extensión, pero las operaciones que se pueden realizar con ellas no son tan poderosas como las de Visual Basic.

- El límite de 255 caracteres también se aplica a alguna propiedad de control.

- No existen arreglos redimensionables. Sin embargo, Delphi tiene colecciones que pueden cambiarse de tamaño, las cuales son más poderosas que los arreglos.

VISUAL BASIC.

Aunque exista una gran variedad de lenguajes visuales uno de los que han destacado sobretodo para el desarrollo de aplicaciones front-end es Microsoft Visual Basic.

Hace unos años, con Visual Basic los desarrolladores por primera vez pudieron desarrollar aplicaciones basadas en Windows fácil y rápidamente. Como resultado, actualmente los desarrolladores crean sus aplicaciones con un gran precedente de productividad.

La tendencia dominante de hoy en día es la utilización de las herramientas de desarrollo de aplicaciones generadas mediante programación visual (entendida como el uso de expresiones tales como dibujo, iconos, barras de menús, es decir, elementos gráficos) en el proceso de la programación de aplicaciones.

Microsoft Visual Basic es la forma más fácil y rápida de crear aplicaciones poderosas para sistemas operativos Microsoft Windows. La programación en Visual Basic permite crear aplicaciones útiles y robustas, que utilizan una interfaz gráfica hacia el usuario (Graphical User Interfase, GUI).

Se pueden crear interfases gráficas de usuario dibujando objetos en forma gráfica. Se deben asignar ciertas propiedades sobre estos objetos para definir su conducta y apariencia. A demás, para que la interfaz gráfica funcione, es necesario escribir el código correspondiente a los eventos que ocurren en la interfaz.

Requerimientos de hardware y software.

- Máquina compatible con IBM, procesador 386 ó superior.
- Disco duro con 50 MB libre como mínimo para la instalación completa.
- Drive de 3.5 " ó drive CD-ROM.
- 8 MB de RAM requeridos por Windows 95.
- Un Mouse u otro dispositivo señalador.
- Para versiones Visual Basic de 32 bits se debe contar con Windows 95 ó superior, ó Windows NT 3.51 ó superior.
- Para versiones Visual Basic de 16 bits, se debe contar con Windows 3.1 ó superior.

Ventajas.

- Es un lenguaje de programación completo y permite la construcción de programas estructurados.
- No es un lenguaje orientado a objetos porque carece de componentes de herencia y polimorfismo, aunque no de encapsulamiento.
- Su compatibilidad con las antiguas versiones de BASIC es casi completa; acepta instrucciones tradicionales como print, input o data aunque estas instrucciones no son muy útiles.
- Utiliza el tipo de dato "Wariant" el cual es un tipo de variable de datos que puede almacenar datos de cualquier tipo, ya sea numérico, cadenas o de fecha/tiempo.

- La programación en Visual Basic es por eventos, es decir, que se asocia código a las acciones posibles de los objetos de nuestra forma.

- Utilización de las denominadas Visual Basic Extension (VBX). Los VBX programas normalmente realizados en lenguaje C que realizan una función determinada y que pueden ser utilizados de manera directa en las aplicaciones realizadas en Visual Basic.

- Posee en librería de rutinas especializadas con la cual el programador puede crear sus propios VBX, estas librerías están basadas en el software llamado Microsoft Windows Software Developed Kit (SDK).

Desventajas.

- Utiliza Microsoft BASIC como lenguaje fundamental.

- En la compilación produce código semi-interpretado.

- El control de desplegado es un arreglo de iconos con una representación gráfica que no siempre es intuitiva. En otras palabras los desarrolladores pueden perder la búsqueda para el control de imágenes, por ejemplo muchos otros iconos pueden verse similares. Los desarrolladores deben tomar el control dentro de la forma para certificar su identidad.

- Los usuarios de Visual Basic pueden accesar a la lista de propiedades en un menú para seleccionar la opción para una propiedad particular y esto requiere la selección de cada campo para realizar los campos correspondientes.

- Provee un programa con capacidad de rastreo, tal como desplegados de variable, el cual es muy amigable. Sin embargo, la funcionalidad esta limitada puesto que no puede interrumpir una condición especifica.

- La capacidad de los "custom controls" que traen de fábrica, sobre todo si el Visual Basic va ha ser utilizado para el manejo de base de datos de gran tamaño, y obliga a usuarios la adquisición de custom controls de tercero fabricantes.

- Los requerimientos de hardware aumentan considerablemente si se utilizan demasiados custom controls en un solo programa.

PASCAL.

Pascal es un lenguaje de propósito general para la resolución de aplicaciones de todo tipo: gestión, científicas, de ingeniería, etc. Entre sus características más sobresalientes se encuentran las siguientes:

Características.

1. Lenguaje de propósito general, excelente para el aprendizaje de la programación.

2. Lenguaje procedural, estructurado y recursivo.

3. Tiene gran riqueza de tipos. Tipos de datos simples y estructurados, así como datos definidos por el usuario.

4. Gracias a su compilador produce programas ejecutables rápidos y eficientes.

5. Facilidad para realizar programación modular debido a la posibilidad de diseñar subprogramas o módulos del tipo procedimiento y función.

6. Biblioteca de utilerías.

7. Depurador.

8. Utiliza poco espacio de memoria.

9. Pascal considera a los dispositivos externos tales como la pantalla, teclado y la impresora, como si fueran archivos de entrada/salida, y esto debe tomarse en cuenta cuando se llevan a cabo programas que utilicen estos dispositivos.

Ventajas.

- Es un lenguaje procedural y estructurado, lo cual facilitará la programación del sistema.
- Es un lenguaje recursivo, lo cual es de utilidad para la realización de búsquedas.
- Tiene gran riqueza de tipos.
- Produce programas ejecutables rápidos y eficientes.
- Cuenta con un ambiente editor fácil de manejar.
- Permite el manejo de estructuras de datos.
- Permite el manejo de cadenas de una forma moderadamente fácil.
- Permite el manejo de los dispositivos de entrada-salida.

Desventajas.

- Pascal es un lenguaje de propósito general y no está especialmente enfocado en la manipulación de bases de datos.

- Si se usan estructuras de datos de tipo estático (arrays) se corre el riesgo de desperdiciar memoria ó de que haga falta.

- Sería necesaria la programación de cada una de las rutinas para la manipulación de bases de datos como borrar, agregar y modificar registros.

- También se realizaría la programación de rutinas para la manipulación de los dispositivos de entrada-salida.

Lenguaje C++

El lenguaje C fue creado por Dennis Ritchie en 1972 en los laboratorios Bell. Aún cuando C es un lenguaje de programación de alto nivel, también permite tener acceso a la programación a nivel bit, y esta ventaja lo hace muy valioso en la programación de sistemas.

Características.

1. Ofrece un extenso conjunto de facilidades para manejar una amplia gama de aplicaciones.

2. Maneja todos los tipos de organización de datos.

3. Tiene un conjunto completo de operadores.

4. Maneja una biblioteca de rutinas que facilita el manejo de entrada salida de datos.

5. Es eficaz y compacto.

6. Tiene un alto grado de transportabilidad.

7. Abundancia de tipos de datos.

8. Produce código altamente eficiente.

9. Está diseñado para soportar programación orientada a objetos.

Ventajas.

- Cuenta con utilerías que facilitan la entrada y salida de datos.
- Es un programa que permite la modularidad.
- Permite compilar su código fuente con lo cual se obtienen programas autónomos *.EXE.
- Tiene un gran número de datos entre los que destacan las estructuras de datos.
- Su código es compacto y eficiente.
- Tiene un alto grado de transportabilidad.
- Cuenta con un ambiente editor agradable.
- Permite la manipulación de dispositivos de entrada-salida de una forma sencilla por tratarlos igual que un archivo cualquiera.

Desventajas.

- Sería necesario escribir rutinas para la manipulación de bases de datos para borrar, agregar y modificar registros.

- Seria necesario la programación de las rutinas para generar informes y controlar los dispositivos de entrada-salida.

- Para optimizar el uso de la memoria se requeriría de estructuras dinámicas. lo cual haria compleja la programación.

La tabla siguiente resume las ventajas y desventajas del software analizado:

CARACTERISTICAS	Delphi	Visual Basic	PASCAL	C++
Permite programacion modular.	✓		✓	✓
Es recursivo.			✓	✓
Cuenta con riqueza de tipos.	✓	✓	✓	✓
Reduce archivos *.EXE.	✓	✓	✓	✓
Cuenta con editor propio.	✓		✓	✓
Permite el manejo de estructuras de datos.	✓	✓	✓	✓
Manejo fácil de cadenas.		✓		
Generador de informes.		✓		
Instrucciones directas de ordenamiento y búsqueda.		✓		
Rapidez para manejar archivos grandes.	✓			
Cuenta con instrucciones para manejar browsers.	✓	✓		✓

2.7 ELECCIÓN DE LA SOLUCIÓN OPTIMA.

Después del análisis de cada una de las opciones que teníamos como factibles para desarrollar el Sistema de Compensación Electrónica de Cheques, concluímos que a pesar de las grandes ventajas que nos ofrece otro software, la mejor solución para nuestro proyecto es Pascal Windows 7.0 para los programas de back-end, Delphi 4.0 para los programas visualizadores Front-end y el ambiente gráfico proporcionado por el sistema operativo Windows NT.

2.7.1 Pascal Windows 7.0 (Back-end).

Se utilizará Pascal Windows 7.0, porque además de ser un programa poderoso y fácil de usar, se encontró en el análisis realizado que no existen razones suficientes para la adquisición de un nuevo software.

Justificación

- La mayor justificación es que se tiene la Licencia de uso de Pascal Windows 7.0
- Es un lenguaje de la 3er generación de propósito general procedural y estructurado.
- Lenguaje recursivo con gran riqueza de tipos (datos simples, estructurados y del usuario).
- Programas ejecutables rápidos y eficientes (garantiza cierta autonomía e integridad a los sistemas).
- Cuenta con un ambiente editor fácil de manejar.
- Programación modular (Subprogramas, procedimientos y funciones).
- Cuenta con un compilador eficiente e independiente.
- Biblioteca de utilerías.
- Depurador.
- Permite el manejo de estructuras de datos.
- Permite el manejo de cadenas de caracteres de una forma fácil.
- Manejo de dispositivos de Entrada/Salida.

- Pascal Windows 7.0 emplea programación orientada a objetos, ahorrando memoria y tiempo.
- Con la programación orientada a objetos, el tamaño del programa principal disminuye considerablemente, dejando el trabajo a los objetos.
- Uso de tareas compartidos con Windows, multitarea.
- Su programación ObjectWindows permite crear Ventanas.
- Lectura de datos con ventanas de diálogo.
- Visualizar Gráficos. manejo de dibujo programado, líneas, rectángulos, etc.
- Manejo de mapas de bits.
- Uso del ratón y cursores.
- Manejo de menús y de iconos.
- Manejo de interrupciones del BIOS y del DOS.

2.7.2 Delphi 4.0 Programa para Desarrollar Visualizadores (Front-ends).

De las herramientas analizadas para desarrollar el front-end la que elegimos fue Delphi 4.0. (por estar basada en Object Pascal es una extensión significativa de Pascal 7.0 de Borland). La principal justificación es que al desarrollar el back-ends con Pascal 7.0, la herramienta que más se acopla para el desarrollo del Front-ends es Delphi, por ser un software que combina el poder de los compiladores tradicionales de tercera generación por su fácil uso y rápido desarrollo en ambiente Windows.

Justificación.

- Se posee la licencia para poder utilizarlo.

- El desempeño es significativamente mejor porque genera en la compilación, archivos ejecutables. Realiza enlaces inteligentes que activan la optimización de segmentos, por lo que el archivo ejecutable reduce su tamaño más del 30%, lo cual permite una carga rápida del archivo y una ganancia adicional en su desempeño.

- Puede compilar archivos ejecutables (.EXE) independientes, así como reutilizar las bibliotecas de enlace dinámico (Dynamic Linked Libraries, DLL). Por último, tambien permite a los programadores profesionales escribir directamente en código ensamblador para tener el control directo del microprocesador.

- La conectividad a la base de datos es nativa si se utiliza la base de datos de Borland. Sin embargo, también soporta enlaces a otros tipos de datos via ODBC.

- Tiene más controles de construcción que aumentan el formato de los iconos, a través de una paleta de componentes multipágina. Tiene una extensa galería de plantillas.

- Aumenta la capacidad de colocación de los objetos.

- Aumenta la lista de propiedades de modificación.

- Tiene dos formas de sincronización de código de pantalla.

- Particiona funciones y eventos.

- Los controles están referenciados como VCLs (Biblioteca Visual de Componentes) que soportan toda la funcionalidad de los estándares de Windows. Además soporta una tercera parte de VBXs, lo que provee el acceso a un intervalo de una tercera parte de componentes adicional.

- La paleta de componentes es organizada en páginas del block de notas, desplegando los iconos en un simple renglón con formato de barra de lista.

- Incluye plantillas preconstruidas que hacen más fácil el desarrollo de aplicaciones estándar o componentes complejos como pantallas MDI, formas de bases de datos, diálogos multipáginas y cajas de listas dobles. La arquitectura es completamente

extensible, permitiendo a los desarrolladores un fácil registro de sus propias plantillas dentro de la galería.

- Facilita el diseño visual con características tales como la alineación automática y el dimensionamiento del objeto.

- Provee un menú de propiedades que puede ser accesado directamente para hacer más eficientes e intuitivas las modificaciones.

- El código de edición de pantalla de Delphi sincroniza todas las representaciones de diseño visuales con el código básico. En otras palabras, como la aplicación es construida por la colocación de objetos en una forma, el código correspondiente es generado simultáneamente. No hay limitaciones, el código siempre está accesible y los desarrolladores pueden intercambiar instantámente entre el editor de código y las herramientas de diseño. permitiendo un modo más eficiente para cada parte del proyecto.

- Dephi provee un rastreador (debugger) completo con interrupciones en condiciones de ejecución. La pantalla del rastreador y las vistas pueden ser guardadas de sesión en sesión, permitiendo a los desarrolladores crear un ambiente confortable. Incluye un poderoso navegador (browser) similar al utilizado en Borlan C++, el cual provee un desplegado comprensivo de objetos y clases (incluyendo la capacidad de rastrear objetos y procedimientos virtuales).

- En el particionamiento de funciones, permite que la función sea colocada en un archivo local o en un DCU (archivo unitario de Pascal) el cual debe estar explicitamente referenciado por los archivos que lo usan.

Las ventajas relacionadas con el manejo de bases de datos son:

- Los desarrolladores pueden colocar una base de datos como un campo de tal manera que puede tener un conjunto de propiedades y bidireccionar la comunicación vía ODBC compatible con la base de datos.

- El componente de la base de datos puede ser usado como un mecanismo para navegar a través de la base de datos, utilizando flechas representando al siguiente y al registro previo.

- Las consultas (queries) SQL pueden ser definidas por código. En la forma que una consulta puede estar dentro de la base de datos.

- Maneja un soporte extenso de bases de datos que incluyen la base de datos de Borland (BDE) para Paradox y acceso a Dbase.

2.7.3 Sistema Operativo Windows NT.

El sistema operativo propuesto deberá ser amigable con el usuario a través de una interface gráfica. Por lo que se optó por la elección del sistema operativo Windows NT, dicho sea de paso el sistema podrá emplearse en cualquier ambiente de Windows9X.

Respecto a Windows, se puede decir que es el sistema operativo con más difusión y éxito que todos los demás; que tiene una amplísima cantidad y variedad de aplicaciones disponibles; que es un ambiente gráfico que permite tener y crear aplicaciones muy amigables, las cuales pueden ser utilizadas fácilmente por cualquier tipo de usuario acostumbrado a los ambientes gráficos. Aunado a esto se puede considerar que existe también una gran infraestructura para este sistema operativo y sus aplicaciones, lo que permite asegurar su permanencia en el mercado y en el uso cotidiano por tiempo todavía indefinido. Estas características hacen de Windows NT, el sistema operativo más recomendable tanto para realizar la integración de algún nuevo sistema, como de

la construcción de la aplicación que se tiene pensada para su difusión, ya que hoy en dia Windows señala el camino a seguir por todas las aplicaciones que se desarrollen en el futuro.

Otro factor importante a considerar es el usuario, tomando como base que la mayoría del personal que labora en el Cecoban recibió capacitación sobre ambiente Windows. Además de contar con la infraestructura para el ambiente en Windows NT, por lo que se optó por seleccionar este último.

Justificación.

Algunas otras características de Windows NT son:

- Extensibilidad. El código esta escrito para crecer cómodamente y cambiar según evolucionen las necesidades del mercado. Para ello emplea una estructura modular, utiliza objetos para representar los recursos del sistema, Drivers cargables, mecanismos de llamadas de procedimiento remoto (RPC).
- Facilidad de transporte. Dependiendo las necesidades del mercado, el código debe poder moverse fácilmente de un procesador a otro. Para ello emplea software C portable, emplea módulos de aislamiento del procesador para ser reemplazados por análogos para otros microprocesadores. Aislamiento de plataforma, para lo cual encapsula el código dependiente de la plataforma en librerías de enlace dinámica (DLL).
- Fiabilidad y robustez. Protección de mal funcionamiento interno y de fallos externos, para que las aplicaciones no puedan dañar ni al sistema operativo ni su funcionamiento.
- Compatibilidad. Bajo procesadores Intel proporciona compatibilidad binaria con software Microsoft existente, sistema operativo MS-DOS, Windows de 16 bits, OS/2 y LAN Manager, estos últimos empleando emuladores. Compatibilidad a nivel fuente con aplicaciones POSIX.

- Rendimiento. Gran eficacia y operatividad en aplicaciones con altas necesidades de procesamiento rápido y buenos tiempos de respuesta como en paquetes gráficos o análisis financiero.
- Reducción del número de pasos necesarios para la instalación el equipo.
- Las aplicaciones se cargan más rápido, identificando las más frecuentes y colocando en el mismo sitio los archivos necesarios para su inicio.
- La salida del sistema es mucho más rápida.
- Maneja en forma eficiente el espacio en disco.
- El asistente de mantenimiento calendariza y realiza automáticamente revisiones internas para que la PC siempre esté funcionando en las mejores condiciones.
- Acceso y uso más fácil de Internet.
- Mejorías en la herramienta para conexión telefónica a redes.
- Interfaces de red que le permiten enlazarse a diferentes tipos de redes e interaccionar con distintas clases de sistemas de ordenadores.
- Ayuda directa de HTML y ayuda del Web.
- La estructura de Windows NT se divide en dos modos: modo usuario y modo Kernel. Generalmente un sistema operativo se ejecuta en modo Kernel y las aplicaciones en modo usuario, sin embargo los substemas protegidos de Windows NT se ejecutan en modo usuario, lo cual permite modificar o agregar nuevos subsistemas sin afectar la integridad y en forma más rápida y eficiente.
- Los componentes del Kernel crean, eliminan y administran objetos y realizan el sumario de tipos de datos que representan los recursos del sistema.
- Refuerzan las normas de seguridad y resguarda los recursos del sistema operativo.
- Crean y finaliza procesos y tareas, suspende y continúa la ejecución de tareas: almacena y recupera información acerca de los procesos y tareas de Windows NT.
- Controla los mensajes entre un proceso cliente y un proceso servidor.
- Windows NT es un sistema operativo transportable.
- Windows NT es un sistema operativo de multiproceso simétrico que soporta múltiples entornos de sistemas operativos.
- Posee una interfaz gráfica de usuario (GUI) de Windows.

- Ejecuta programas basados en Win32, Windows de 16 bits, MS-DOS, POSIX y OS/2.
- Emplea principios avanzados de sistemas operativos como la memoria virtual, multitarea real.
- Es seguro, potente, confiable y flexible.

CAPÍTULO 3.

DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA.

3.1 BACK END PARA CADA MÓDULO

3.1.1 Diagrama de contexto

El diagrama de contexto es un caso especial de los diagramas de flujo de datos. mismos que se exponen en la siguiente sección (3.1.2). Se recomienda leer la información de dicha sección y después consultar el diagrama de contexto para el sistema de compensación electrónica de cheques que se muestra en la figura 3.1.2.1.

3.1.2 Diagrama de flujo de datos

Desde la perspectiva del análisis de sistemas, un diagrama de flujo de datos (DFD) es una representación gráfica del flujo de datos en un sistema. Un DFD consiste de una serie de diagramas que comienzan en un nivel conceptual alto y que progresa sucesivamente hacia un mayor detalle acerca del sistema bajo análisis. El nivel más alto, más abstracto, se conoce como diagrama de contexto; este establece el contexto del sistema bajo análisis. El objetivo del diagrama de contexto es representar el ámbito del sistema por medio de identificar los límites del mismo y sus entidades externas principales (aquellas que proveen la entrada y aceptan la salida). El siguiente nivel de detalle es el diagrama de flujo de datos de nivel cero; el cual identifica los principales eventos del negocio y los principales almacenes de datos dentro del sistema bajo análisis. Después tenemos los DFDs de nivel uno y así sucesivamente, cada uno mostrando más detalle hasta que los diagramas resultantes no puedan descomponerse más, esto ocurre por lo general, cuando tenemos ya sea sólo una entrada o una salida de un proceso.

El diagrama de flujo de datos es una técnica tradicional para modelar procesos, tiene la ventaja de ahorrar grandes costos originados por malentender los requerimientos del usuario. Los diagramas de flujos de datos o DFDs nos dan una vista de lo que los sistemas hacen con los datos sin mostrar un orden específico.

Un DFD documenta el procesamiento que lleva a cabo el sistema, esto lo hace por medio de mostrar el flujo de datos dentro, alrededor y fuera del sistema. Su propósito es servir como medio de comunicación entre los usuarios y el analista. Se usa un diagrama porque este es menos propenso a una mala interpretación de lo que es una descripción textual.

Los componentes de un diagrama de flujo de datos son los siguientes:

* El proceso, que indica la manera en que una o más entradas se transforman en salidas. Se representa por medio de un círculo.

* El flujo, se usa para mostrar el movimiento de datos entre los otros componentes (entidades externas, almacenes de datos y procesos). Se representa por medio de una flecha que entra o sale de un proceso

* El almacén de datos, se usa para modelar datos en reposo y se representa por medio de un rectángulo con los lados abiertos. Los almacenes de datos no pueden estar ligados directamente a otros almacenes de datos o a entidades externas sin un proceso que transforme los datos.

* La entidad externa, esta se refiere a una entidad con la que el sistema se comunica, pero que es exterior al mismo. Se representa por medio de un rectángulo.

Para mantener el diagrama simple y manejable se emplea una jerarquía o nivel de diagramas. Usualmente hay al menos dos niveles: el nivel de contexto y el nivel cero.

El diagrama de contexto únicamente muestra las entidades externas y un solo proceso junto con sus flujos de datos, esto es, ningún almacén de datos y ningún flujo interno de datos.

El diagrama de nivel cero expande el único proceso del diagrama de contexto y lo divide en varios procesos.

Algunos autores llaman al diagrama de contexto "diagrama de flujo de datos de nivel cero" y al diagrama de flujo de datos de nivel cero lo llaman "diagrama de flujo de datos de nivel 1". En el presente trabajo, al diagrama de contexto lo llamaremos diagrama de contexto y al primer diagrama de flujo de datos que se desarrolle a partir del diagrama de contexto le llamaremos DFD de nivel cero. De cualquier manera, el diagrama de contexto es el diagrama de flujo de datos de nivel conceptual más general en cualquier proceso.

A continuación se muestra el diagrama de contexto y los diagramas de flujo de datos para el sistema de compensación electrónica.

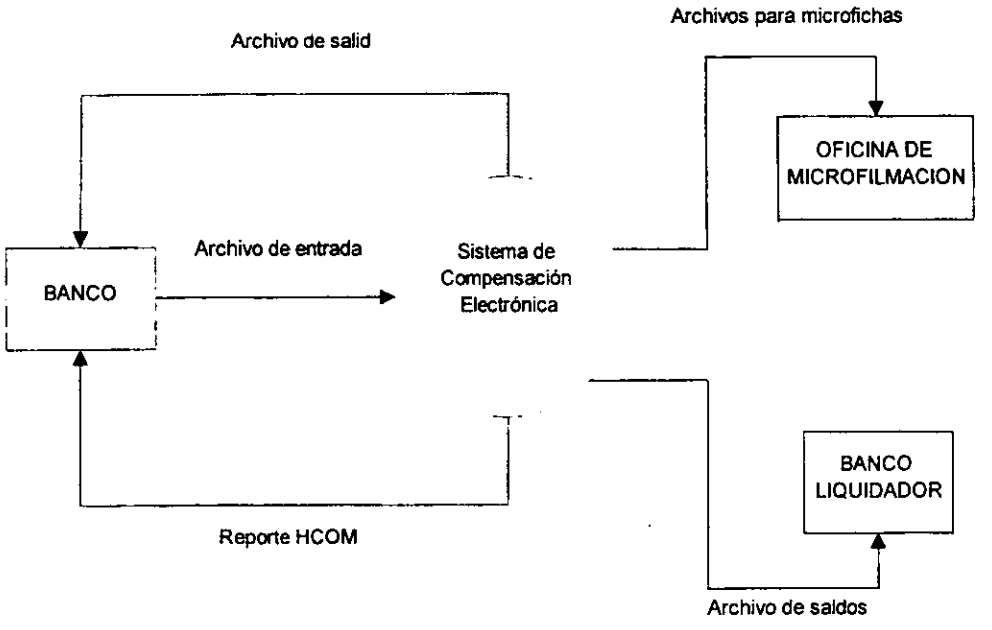


Figura 3.1.2.1 Diagrama de contexto para el sistema de compensación electrónica.

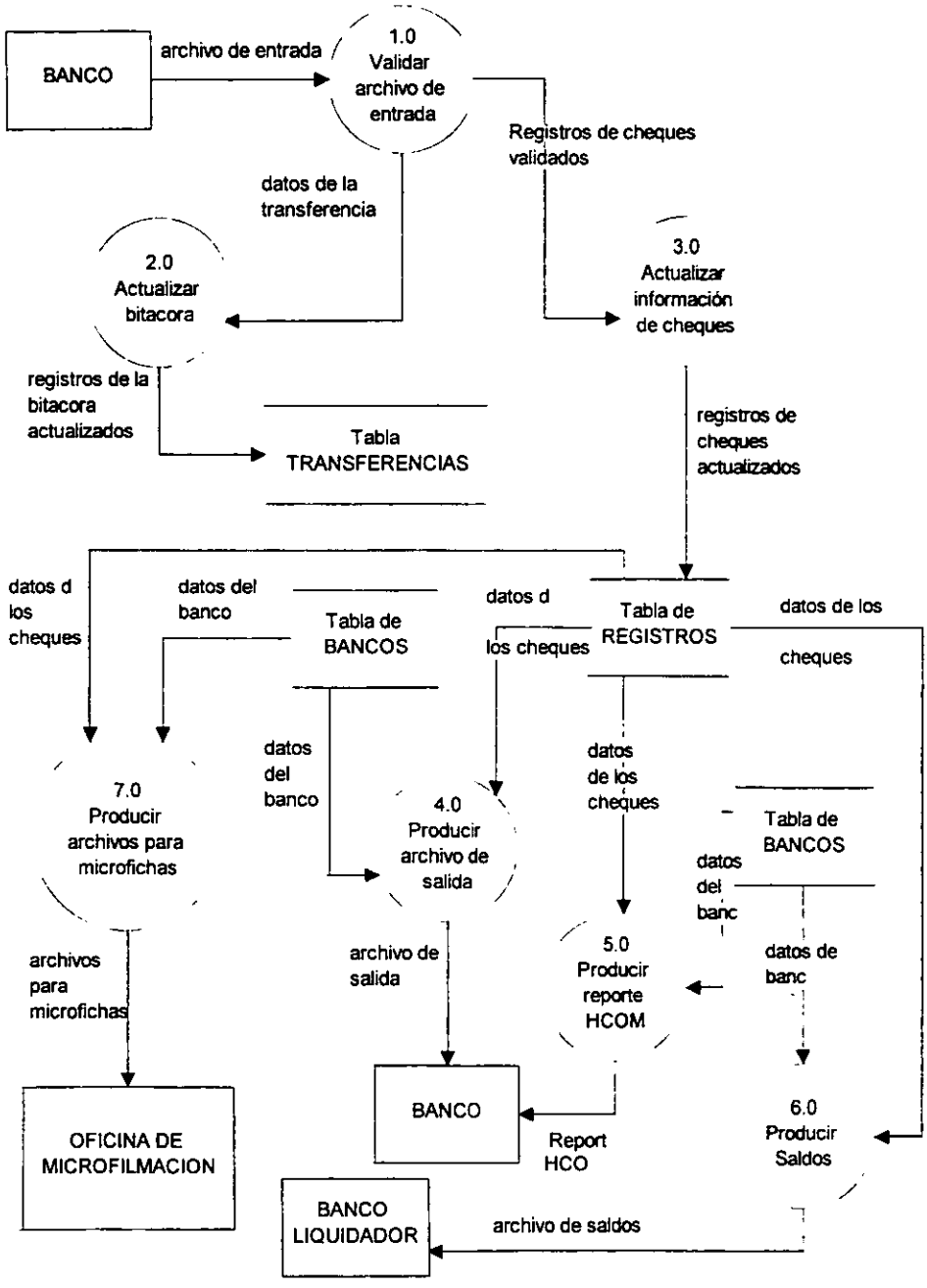


Figura 3.1.2.2 Diagrama de flujo de datos de nivel 0

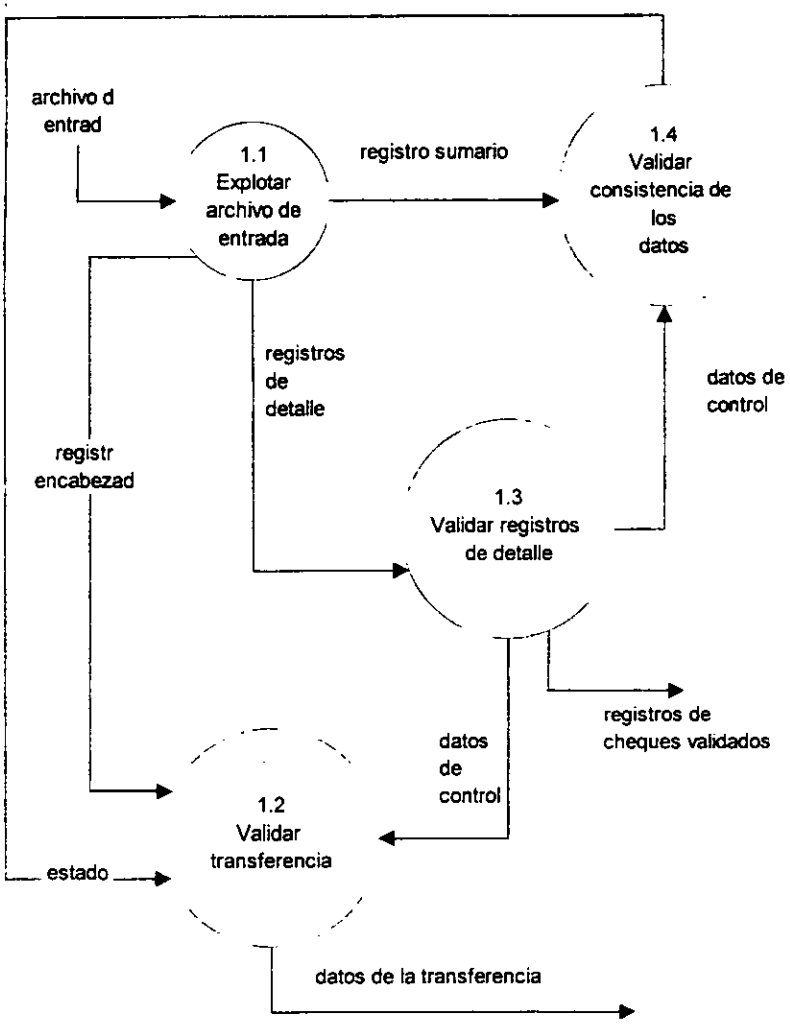


Figura 3.1.2.3 DFD de nivel 1 para el proceso 1.0

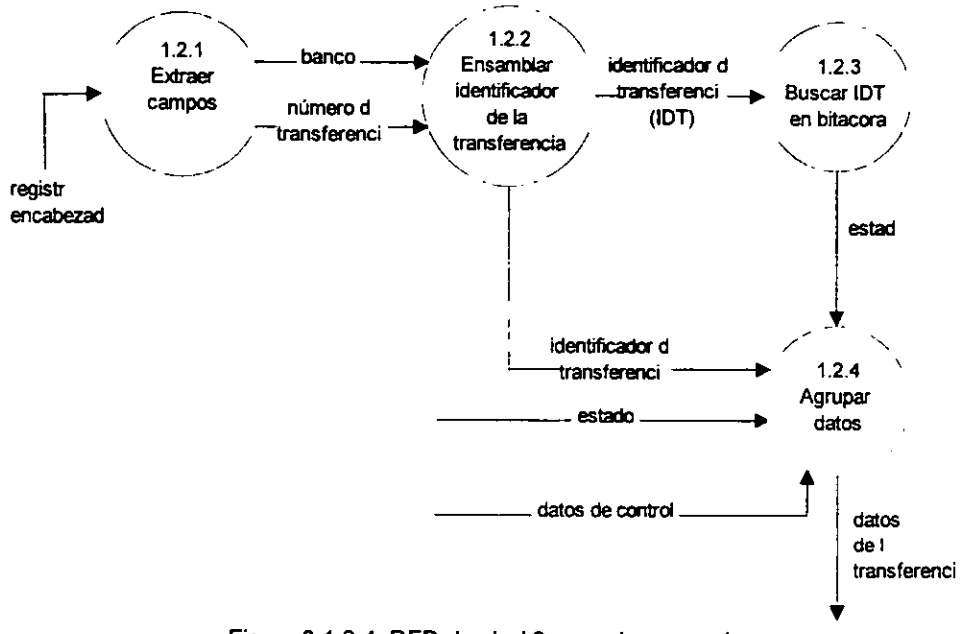


Figura 3.1.2.4 DFD de nivel 2 para el proceso 1.

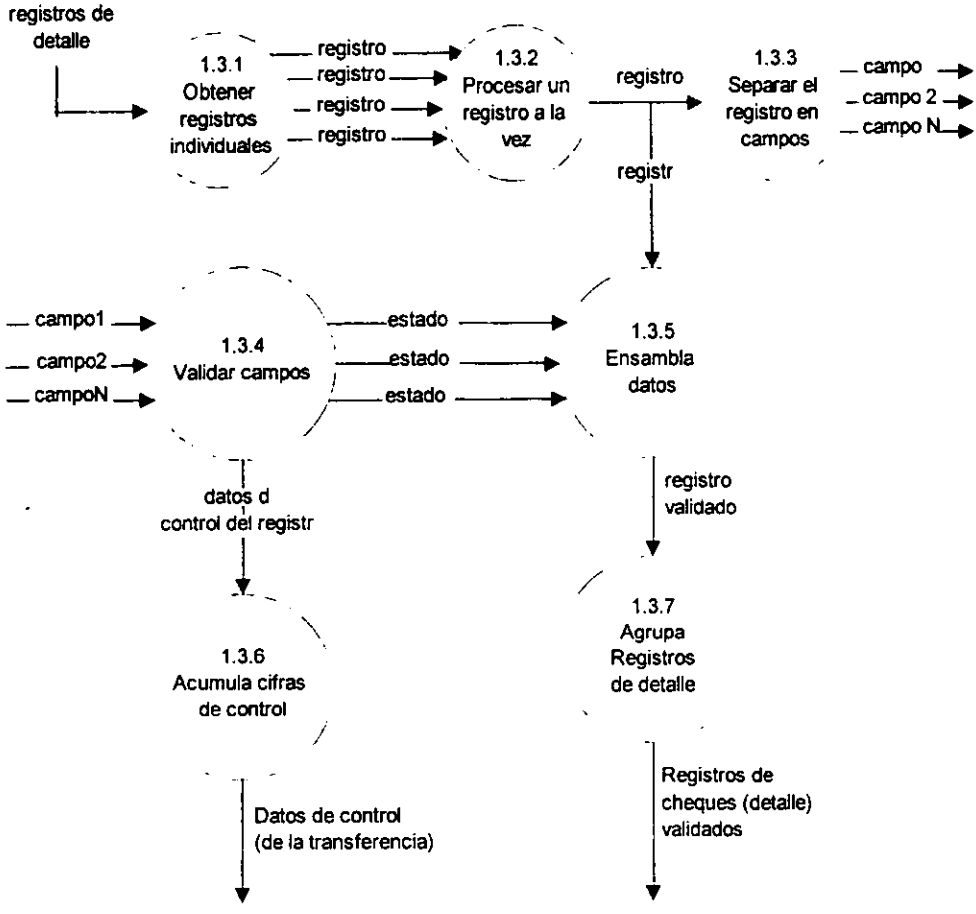


Figura 3.1.2.5 DFD de nivel 2 para el proceso 1.3

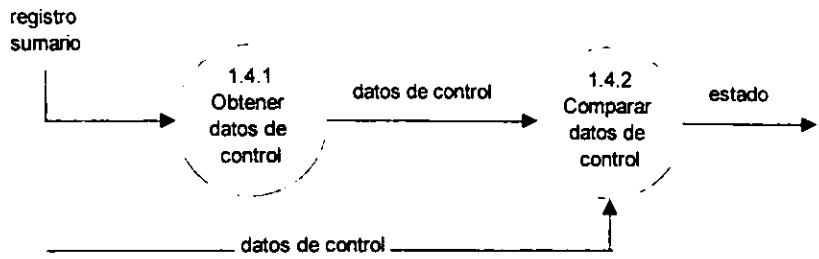


Figura 3.1.2.6 DFD de nivel 2 para el proceso 1.4

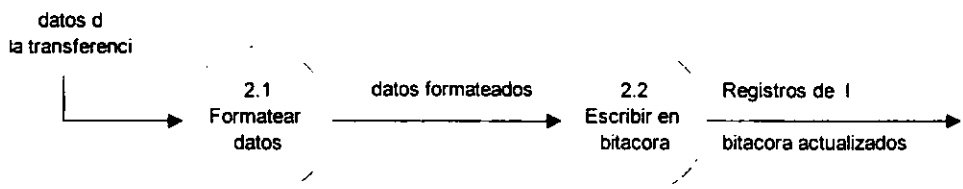


Figura 3.1.2.7 DFD de nivel 1 para el proceso 2.0

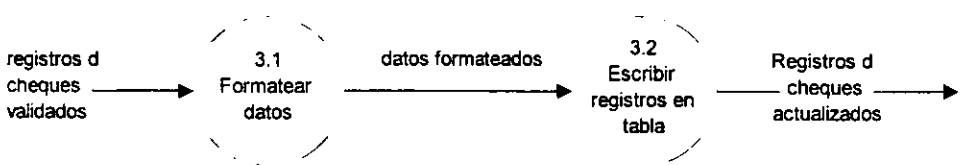


Figura 3.1.2.8 DFD de nivel 1 para el proceso 3.

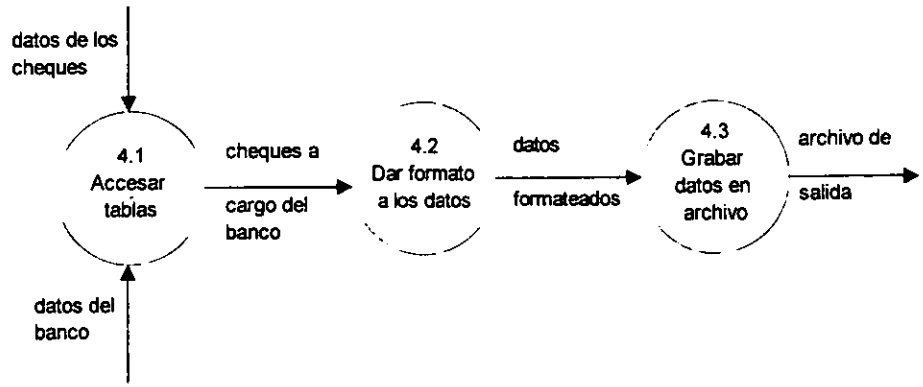


Figura 3.1.2.9 DFD de nivel 1 para el proceso 4.0

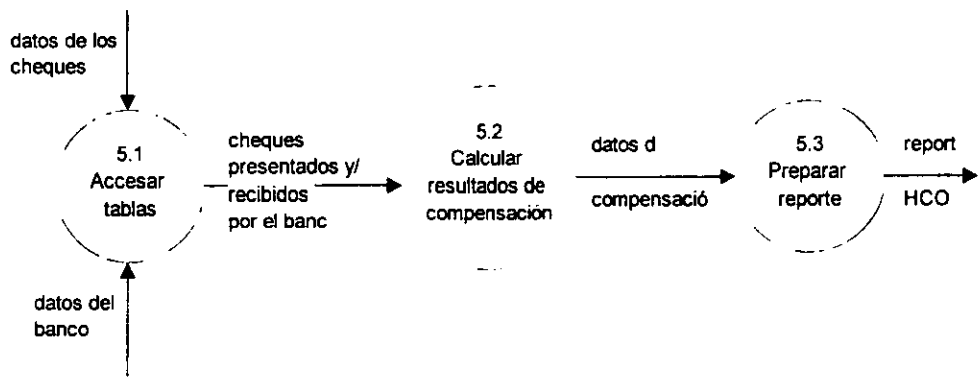


Figura 3.1.2.10 DFD de nivel 1 para el proceso 5.0

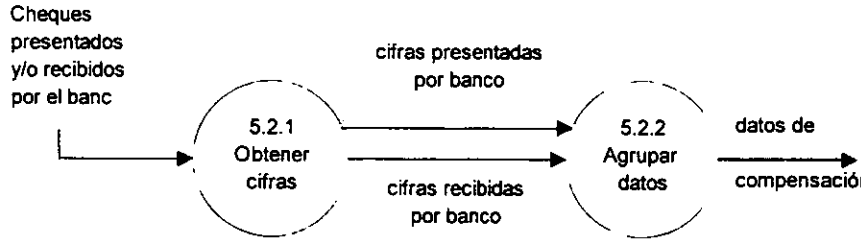


Figura 3.1.2.11 DFD de nivel 2 para el proceso 5.2

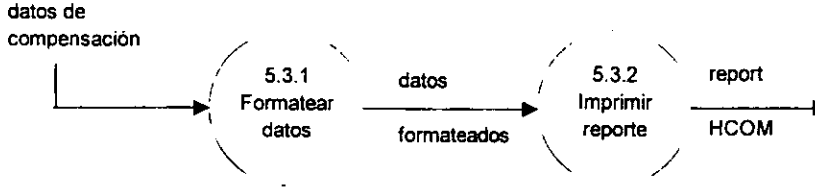


Figura 3.1.2.12 DFD de nivel 2 para el proceso 5.3

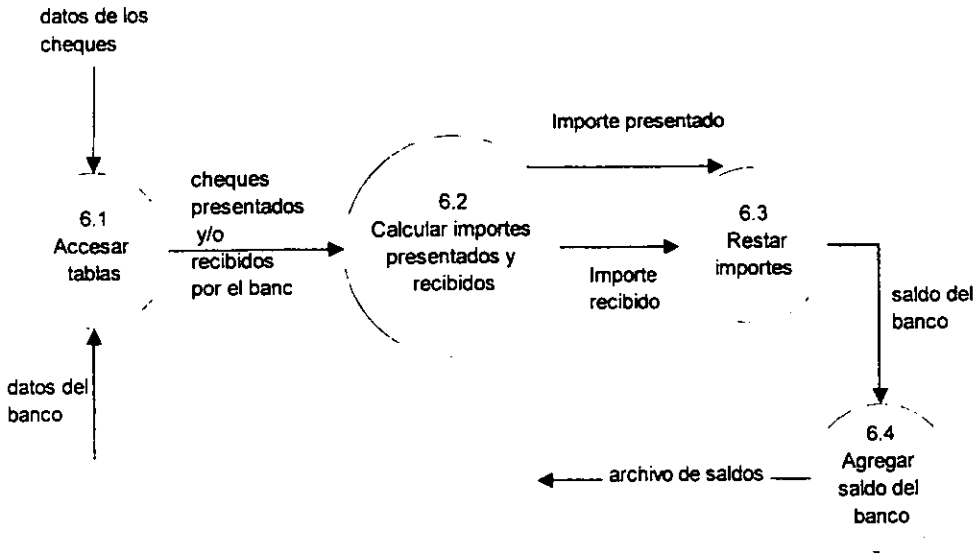


Figura 3.1.2.13 DFD de nivel 1 para el proceso 6.0

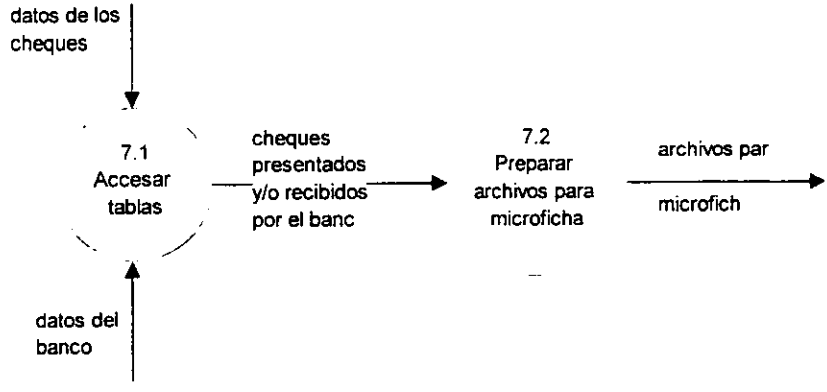


Figura 3.1.2.14 DFD de nivel 1 para el proceso 7.0

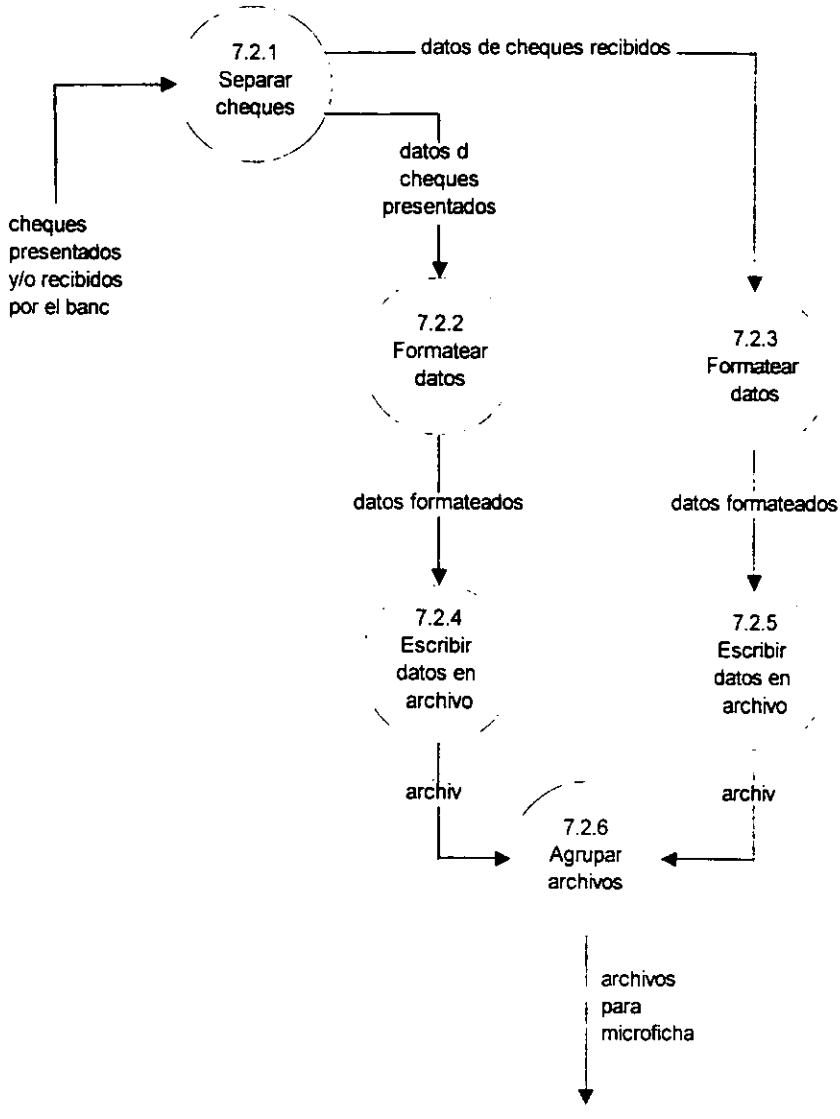


Figura 3.1.2.15 DFD de nivel 2 para el proceso 7.2

3.1.3 Diccionario de datos

En un sistema dado, es necesario proporcionar un enfoque organizado para representar las características de cada objeto de datos y elemento de control. Esto se realiza con el diccionario de datos.

Se ha propuesto el diccionario de datos como gramática casi formal para describir el contenido de los objetos definidos durante el análisis estructurado. Esta importante notación de modelado ha sido definida como especialidad dentro de los diccionarios de referencia que utilizamos en la vida diaria. El diccionario es una referencia de "datos acerca de los datos" recopilados por el analista de sistemas para guiarse durante el análisis y el diseño. Como documento, recopila, coordina y confirma lo que un término específico significa para la gente de la organización.

Los analistas de sistemas deben estar conscientes y catalogar los diversos términos que se refieren al mismo dato. Esto evitará duplicar esfuerzos, favoreciendo una mejor comunicación entre los departamentos de la organización, que comparten una misma base de datos y, a la vez, facilitar el mantenimiento.

Para el analista de sistemas es muy importante conocer qué tipos de datos componen el diccionario de datos, las convenciones utilizadas en ellos y la manera en que se desarrollan, de tal manera que le permita concebir el sistema y cómo es que éste trabaja.

Información que debe contener un diccionario de datos

Una manera de saber lo que debe contener el diccionario de datos, es visualizar como llegará a utilizarse. Es el elemento básico de referencia para localizar los nombres y atributos de los datos utilizados en todo el sistema de organización. Por esto se deben incluir todos los datos elementales.

Con el fin de ser de utilidad, los registros del diccionario de datos deben contener información referente a las categorías siguientes:

- El nombre y acrónimo del dato.
- La descripción del dato.
- Los datos elementales que se relacionan con el término.
- La longitud disponible en caracteres.

Nombre y acrónimo (alias)

El diccionario de datos debe contener el nombre de cada dato, esto es, la manera de denominar al dato en los programas, y su acrónimo. Todo esto debe quedar registrado en el diccionario de datos para facilitar la comunicación entre los departamentos y sus programas.

Descripción

Debe incluir una descripción textual del dato elemental, la cual debe ser concisa, pero informativa para cualquiera que la consulte.

Longitud del dato

Se refiere a la longitud permitida para el acceso a un dato elemental. La longitud siempre se da en función del número de caracteres impresos y no por la cantidad requerida de memoria.

Cuando el diccionario de datos se integra de manera correcta, es útil para el desarrollo del sistema, la modificación del mismo y su mantenimiento.

No.	Nombre y alias	Tipo	Long.	Descripción
1	Número de Banco NumeroBanco	Numérico	3	Número del banco, asignado por la Asociación de Banqueros de México.
2	Nombre del Banco NombreBanco	Alfabético	50	Nombre de la razón social del banco participante en la compensación electrónica.
3	Acrónimo del Banco AcronBanco	Alfabético	8	Nombre corto del banco participante en la compensación electrónica.
4	Nombre de Transferencia IdentificacionTrans	Alfanumérico	12	Formado de la siguiente manera EPPBBBAS.DDC donde PP = Plaza, BBB = Banco, S = Servicio, DD = Dia y C = Número consecutivo de transferencia.
5	Número Banco Cedente BancoCed	Numérico	3	Número del banco que identifica a la Institución, que en una plaza en particular, genera uno o más archivos y los presenta a la cámara de compensación para su proceso.
6	Documentos Aceptados DoctosAcep	Numérico	7	Total de operaciones que fueron aceptadas por el sistema después de la validación del archivo de entrada, es decir estas operaciones no fueron rechazadas por algún motivo.
7	Importe Aceptado ImpAcep	Numérico	18	Monto total de las operaciones que fueron aceptadas por el sistema después de la validación del archivo de entrada. Las dos últimas posiciones de la derecha corresponden a los centavos.
8	Documentos Rechazados DoctosRech	Numérico	7	Total de operaciones que fueron rechazadas por el sistema después de la validación del archivo de entrada, es decir fueron rechazadas por algún motivo.
9	Importe Rechazado ImpRech	Numérico	18	Monto total de las operaciones que fueron rechazadas por el sistema después de la validación del archivo de entrada. Las dos últimas posiciones de la derecha corresponden a los centavos.
10	Número de Sobres de la Transferencia SobPre	Numérico	3	Cantidad que indica el número de bancos girados diferentes que existen en un archivo de entrada. Los documentos de cada banco girado deben ir en un sobre separado.

11	Número Banco Girado BancoGir	Numérico	3	Identifica a una Institución que recibe cheques a su cargo, cedidos por otro de los bancos participantes (cedentes).
12	Total de Documentos Aceptados TotalDoctosAcep	Numérico	7	Total de operaciones aceptadas por el sistema, que son presentadas por un banco cedente a un banco girado en particular.
13	Total de Documentos Rechazados TotalDoctosRech	Numérico	7	Total de operaciones rechazadas por el sistema, que son presentadas por un banco cedente a un banco girado en particular.
14	Total de Importe Aceptado TotalImpAcep	Numérico	18	Monto total de las operaciones que fueron aceptadas por el sistema y que fueron presentadas por un banco cedente a un banco girado en particular. Las dos últimas posiciones de la derecha corresponden a los centavos.
15	Total de Importe Rechazado TotalImpRech	Numérico	18	Monto total de las operaciones que fueron rechazadas por el sistema y que fueron presentadas por un banco cedente a un banco girado en particular. Las dos últimas posiciones de la derecha corresponden a los centavos.
16	Número de Registro NumReg	Numérico	7	Número del registro dentro de la transferencia a la que pertenece.
17	Importe Importe	Numérico	15	Monto del cheque registrado, correspondiendo las dos últimas posiciones de la derecha a los centavos.
18	Estado Estado	Alfabético	1	Indica si el registro fue aceptado (A) o rechazado (R).
19	Datos Adicionales Datos	Alfanumérico	124	Contiene los datos del registro que formara parte del archivo de cheques a su cargo, para cada uno de los bancos girados.

3.1.4 Normalización

El construir una base de datos es un proceso de examinar los datos que son útiles y necesarios para una aplicación, después dividirlos en un formato de tablas y columnas relativamente simple. Hay dos puntos que se deben comprender acerca de las tablas y las columnas que son la esencia de cualquier base de datos.

El modelo más simple para una base de datos es un archivo plano, donde se tiene sólo una tabla la cual incluye campos para cada elemento que se necesita almacenar. El problema con una base de datos de archivo plano es que desperdicia espacio de almacenamiento y su mantenimiento es más difícil. La solución para los problemas que existen al manejar bases de datos de archivo plano es usar un modelo relacional para los datos, el cual consiste en tener los datos repartidos en varios archivos llamados tablas y establecer relaciones entre un campo o campos de una tabla con la información de las otras tablas. Esto nos da flexibilidad y eficiencia al mismo tiempo que contribuye a un mantenimiento del sistema menos costoso y más fácil.

* Las tablas almacenan datos acerca de una entidad. Una entidad puede ser una persona, una parte de una máquina, un libro, o cualquier otro objeto tangible o intangible. La consideración primaria es que una tabla contiene datos acerca de sólo una cosa.

* Las columnas, también llamadas campos, contienen atributos de la entidad. Así como una tabla contiene datos acerca de una sola entidad, cada columna deberá contener sólo un elemento de datos acerca de esa entidad. Si, por ejemplo, se esta creando una tabla de direcciones, no tiene sentido hacer que una sola columna contenga la ciudad, el estado, y el código postal cuando es igual de fácil crear tres columnas y registrar cada atributo separadamente.

La normalización se refiere al proceso de crear una estructura relacional para almacenar información que sea eficiente, confiable y flexible. La normalización de una base de datos es esencialmente el proceso de tomar una tabla ancha inicial con muchas columnas y rediseñarla como varias tablas relacionadas cada una de ellas conteniendo menos columnas que la tabla original.

Los teóricos de las bases de datos han dividido la normalización en varias reglas llamadas formas normales.

* Primera Forma Normal

Ningún grupo repetido.

* Segunda Forma Normal

Ninguno de los atributos que no sean llave dependen de una parte de la llave primaria.

* Tercera Forma Normal

Ningún atributo depende de otros atributos que no sean llave.

Adicionalmente, para que una base de datos esté en segunda forma normal, primero debe estar en primera forma normal, y para estar en tercera forma normal, debe cumplir los requisitos para la primera y segunda formas normales. Existen también formas adicionales de normalización, pero raramente se aplican.

Comencemos a aplicar estas reglas, para normalizar la base de datos usada por el sistema de compensación electrónica de cheques. Vamos a partir de nuestra "tabla ancha", que llamaremos base de datos de archivo plano (flat-file database).

BASE DE DATOS DE ARCHIVO PLANO

CAMPOS:

Número Banco Cedente
Nombre Banco Cedente
Acrónimo Banco Cedente
Nombre Transferencia 1
Nombre Transferencia 2
...
Nombre Transferencia N
Documentos Aceptados Transferencia 1
Documentos Rechazados Transferencia 1
Importe Aceptado Transferencia 1
Importe Rechazado Transferencia 1
Sobres Presentados Transferencia 1
Documentos Aceptados Transferencia 2
Documentos Rechazados Transferencia 2

Importe Aceptado Transferencia 2
Importe Rechazado Transferencia 2
Sobres Presentados Transferencia 2
...
Documentos Aceptados Transferencia N
Documentos Rechazados Transferencia N
Importe Aceptado Transferencia N
Importe Rechazado Transferencia N
Sobres Presentados Transferencia N
Número Banco Girado 1
Total Documentos Aceptados Banco Girado 1
Total Documentos Rechazados Banco Girado 1
Total Importe Aceptado Banco Girado 1
Total Importe Rechazado Banco Girado 1
Número Banco Girado 2
Total Documentos Aceptados Banco Girado 2
Total Documentos Rechazados Banco Girado 2
Total Importe Aceptado Banco Girado 2
Total Importe Rechazado Banco Girado 2
...
Número Banco Girado N
Total Documentos Aceptados Banco Girado N
Total Documentos Rechazados Banco Girado N
Total Importe Aceptado Banco Girado N
Total Importe Rechazado Banco Girado N
Registro 1 de la Transferencia 1
Registro 2 de la Transferencia 1
...
Registro N de la Transferencia 1
Banco Girado del registro 1 de la Transferencia 1
Banco Girado del registro 2 de la Transferencia 1
...
Banco Girado del registro N de la Transferencia 1
Importe del registro 1 de la Transferencia 1
Importe del registro 2 de la Transferencia 1
...
Importe del registro N de la Transferencia 1
Estado del registro 1 de la Transferencia 1
Estado del registro 2 de la Transferencia 1
...
Estado del registro N de la Transferencia 1
Datos adicionales del registro 1 de la Transferencia 1
Datos adicionales del registro 2 de la Transferencia 1
...
Datos adicionales del registro N de la Transferencia 1

Registro 1 de la Transferencia 2
Registro 2 de la Transferencia 2
...
Registro N de la Transferencia 2
Banco Girado del registro 1 de la Transferencia 2
Banco Girado del registro 2 de la Transferencia 2
...
Banco Girado del registro N de la Transferencia 2
Importe del registro 1 de la Transferencia 2
Importe del registro 2 de la Transferencia 2
...
Importe del registro N de la Transferencia 2
Estado del registro 1 de la Transferencia 2
Estado del registro 2 de la Transferencia 2
...
Estado del registro N de la Transferencia 2
Datos adicionales del registro 1 de la Transferencia 2
Datos adicionales del registro 2 de la Transferencia 2
...
Datos adicionales del registro N de la Transferencia 2
...
Registro 1 de la Transferencia N
Registro 2 de la Transferencia N
...
Registro N de la Transferencia N
Banco Girado del registro 1 de la Transferencia N
Banco Girado del registro 2 de la Transferencia N
...
Banco Girado del registro N de la Transferencia N
Importe del registro 1 de la Transferencia N
Importe del registro 2 de la Transferencia N
...
Importe del registro N de la Transferencia N
Estado del registro 1 de la Transferencia N
Estado del registro 2 de la Transferencia N
...
Estado del registro N de la Transferencia N
Datos adicionales del registro 1 de la Transferencia N
Datos adicionales del registro 2 de la Transferencia N
...
Datos adicionales del registro N de la Transferencia N

Apliquemos ahora la primera forma normal para eliminar de nuestra base de datos de archivo plano los grupos repetidos.

PRIMERA FORMA NORMAL (1FN) Ningún grupo repetido. Los grupos repetidos los movemos a otra tabla, dando como resultado las dos tablas siguientes.

TABLA 1

Número Banco Cedente
Nombre Banco Cedente
Acrónimo Banco Cedente

TABLA 2

Nombre Transferencia
Número Banco Cedente
Documentos Aceptados Transferencia
Documentos Rechazados Transferencia
Importe Aceptado Transferencia
Importe Rechazado Transferencia
Sobres Presentados Transferencia
Número Banco Girado
Total Documentos Aceptados para Banco Girado
Total Documentos Rechazados para Banco Girado
Total Importe Aceptado para Banco Girado
Total Importe Rechazado para Banco Girado
Número de Registro
Banco Girado del Registro
Importe Registro
Estado del registro
Datos Adicionales del Registro

En la TABLA 1 el campo llave es el campo Número Banco Cedente, en la TABLA 2 la llave es el campo Nombre Transferencia.

SEGUNDA FORMA NORMAL (2FN) Ninguno de los atributos que no son llave dependen de una parte de la llave primaria.

La segunda forma normal realmente sólo aplica a tablas donde la llave primaria esta definida por dos o más columnas. La esencia es que si hay columnas las cuales

pueden ser identificadas únicamente por parte de la llave primaria, estas necesitan estar en su propia tabla. En este caso como la llave de la TABLA 2 no es compuesta, nuestro diseño se encuentra ya en segunda forma normal.

TERCERA FORMA NORMAL (3FN) Ninguno de los atributos depende de otros atributos que no son llave. Esto significa que todas las columnas en la tabla contienen datos acerca de una entidad que esta definida por la llave primaria. Las columnas en la tabla deben contener datos acerca de sólo una cosa. Esta es realmente una extensión de la Segunda Forma normal (ambas se usan para mover columnas que deben ir en su propia tabla).

Observemos que la TABLA 1 ya se encuentra en tercera forma normal puesto que cumple con la condición de que ninguno de los atributos depende de otros atributos que no son la llave. Los campos no llave de la TABLA 1 son:

Nombre Banco Cedente
Acrónimo Banco Cedente

Estos campos dependen de la llave primaria que es el Número Banco Cedente.

En la TABLA 2 los campos no llave son:

01 Número Banco Cedente
02 Documentos Aceptados Transferencia
03 Documentos Rechazados Transferencia
04 Importe Aceptado Transferencia
05 Importe Rechazado Transferencia
06 Sobres Presentados Transferencia
07 Número Banco Girado
08 Total Documentos Aceptados para Banco Girado
09 Total Documentos Rechazados para Banco Girado
10 Total Importe Aceptado para Banco Girado
11 Total Importe Rechazado para Banco Girado
12 Número de Registro
13 Banco Girado del Registro

14 Importe Registro
15 Estado del registro
16 Datos Adicionales del Registro

Los campos 02, 03, 04, 05 y 06 dependen todos de la llave primaria, así que se quedarán en esta tabla. Los campos 07, 08, 09, 10 y 11 dependen del campo 01 que no es la llave primaria de esta tabla, así que deben ir en otra tabla. Los campos 13,14, 15 y 16 dependen del campo 12 que tampoco es llave, así que se irán a otra tabla. Tenemos como resultado de la tercera forma normal cuatro tablas.

TABLA 1

Número Banco Cedente
Nombre Banco Cedente
Acrónimo Banco Cedente

TABLA 2

Nombre Transferencia
Número Banco Cedente
Documentos Aceptados Transferencia
Documentos Rechazados Transferencia
Importe Aceptado Transferencia
Importe Rechazado Transferencia
Sobres Presentados Transferencia

TABLA 3

Número Banco Cedente
Número Banco Girado
Total Documentos Aceptados para Banco Girado
Total Documentos Rechazados para Banco Girado
Total Importe Aceptado para Banco Girado
Total Importe Rechazado para Banco Girado

TABLA 4

Número de Registro
Nombre Transferencia
Banco Girado del Registro
Importe Registro

Estado del registro
Datos Adicionales del Registro

Nótese que cada vez que se crea una nueva tabla como resultado de la normalización, se agrega un campo a dicha tabla que sirve como una liga con la tabla relacionada. Ahora que tenemos las tablas normalizadas asignémosle a cada una un nombre apropiado. Asignemos además un nombre adecuado a cada campo, de tal manera que sea el que usemos como base para la implementación.

BANCOS

Numero de Banco
Acrónimo del Banco
Nombre de Banco

TRANSFERENCIAS

Nombre Transferencia
Número Banco Cedente
Documentos Aceptados
Documentos Rechazados
Importe Aceptado
Importe Rechazado
Sobres Presentados

INTERCAMBIOS

Número Banco Cedente
Número Banco Girado
Total Documentos Aceptados
Total Documentos Rechazados
Total Importe Aceptado
Total Importe Rechazado

REGISTROS

Número de Registro
Nombre Transferencia
Banco Girado
Importe
Estado

Datos Adicionales

Tenemos a continuación un ejemplo de como se vería la base de datos con algunos datos incluidos dentro de la misma.

TABLA BANCOS

BANCO	ACRÓNIMO	NOMBRE
002	BANAMEX	BANCO NACIONAL DE MÉXICO, S.A.
003	SERFIN	BANCA SERFIN S.A.
007	CITIBANK	CITIBANK MÉXICO, S.A.
012	BANCOMER	BANCOMER, S.A.
021	BITAL	BANCO INTERNACIONAL, S.A.
044	INVERLAT	BANCO INVERLAT, S.A.

TABLA TRANSFERENCIAS

NOMBRE TRANS	BANCO CED	DOCTOS ACEP.	DOCTOS RECH	IMPORTE ACEP	IMPORTE RECH.
E01002A1.251	002	2	0	$ 27,076.57	$ 0.00
E01012A1.251	012	2	0	$ 45,378.97	$ 0.00
E01021A1.251	021	2	0	$ 93,456.78	$ 0.00
E01002A1.252	002	1	1	$ 120,000.00	$ 800.00
E01003A1.251	003	2	0	$ 42,467.60	$ 0.00

TABLA REGISTROS

NOMBRE TRANS.	NUM. REGISTRO	BANCO GIRADO	IMPORTE	ESTADO	DATOS ADICIONALES
E01002A1.251	0000001	012	$ 1,845.34	A	020000001.....
E01002A1.251	0000002	003	$ 25,231.23	A	020000002.....
E01012A1.251	0000001	021	$ 44,378.97	A	020000001.....
E01012A1.251	0000002	044	$ 1,000.00	A	020000002.....
E01021A1.251	0000001	002	$ 15,000.50	A	020000001.....
E01021A1.251	0000002	003	$ 78,456.28	A	020000002.....
E01002A1.252	0000001	012	$ 800.00	R	020000001.....
E01002A1.252	0000002	044	$120,000.00	A	020000002.....
E01003A1.251	0000001	002	$ 32,467.60	A	020000001.....
E01003A1.251	0000002	002	$ 10,000.00	A	020000002.....

TABLA INTERCAMBIOS

BCO CED	BCO GIR	TOTAL DOC ACEP	TOTAL DOC RECH	TOTAL IMP ACEP	TOTAL IMP RECH
002	012	1	1	$ 1,845.34	$ 800.00
002	003	1	0	$ 25,231.23	$ 0.00
012	021	1	0	$ 44,378.97	$ 0.00
012	044	1	0	$ 1,000.00	$ 0.00
021	002	1	0	$ 15,000.50	$ 0.00
021	003	1	0	$ 78,456.28	$ 0.00
002	044	1	0	$ 120,000.00	$ 0.00
003	002	2	0	$ 42,467.60	$ 0.00

Para la construcción de la base de datos se uso Borland Pascal Windows 7.0, puesto que esta es una base de datos donde la mayoría de la información se crea nuevamente cada día de proceso, así que no se requiere un RDBMS (Sistema Manejador de Bases de Datos Relacionales) para llevar a cabo la administración de los datos. A continuación se explica de manera breve las principales acciones que se realizan con los datos.

Para obtener un resumen de las transferencias presentadas por un banco, se solicita al usuario el número del banco y con esta llave foránea se localizan secuencialmente en la tabla TRANSFERENCIAS los registros correspondientes de ese banco. Esta tabla contiene por lo general menos de 50 registros y en un caso muy extremo no consistiría de más de 150, así que tomando en cuenta la velocidad de las máquinas actuales y la eficiencia del compilador usado, las búsquedas en esta tabla resultan prácticamente instantáneas. Finalmente al usuario se le muestran los totales de documentos e importes de cada una de las transferencias de ese banco. Esta consulta se realiza cuando el usuario selecciona desde la interface visual del sistema, la opción "Consultar transferencias".

Para saber todos los documentos e importe que un banco ha presentado a otros bancos y lo que ha recibido de los otros bancos, igualmente se hace uso de la clave del

banco y con esta llave foránea se accesa la tabla INTERCAMBIOS para localizar todos los bancos girados a los cuales el banco cedente les ha presentado información. Para saber lo que ha recibido de cada banco, el número de banco se utiliza ahora como número de banco girado y se buscan todos los registros que correspondan a dicho banco girado. Esta tabla puede ir de un tamaño normal de 500 registros a un máximo de 3,000 (no depende del volumen de información sino del número de bancos participantes). La consulta anterior se activa cuando el operador selecciona la opción "Reporte Hoja de Compensación" para un banco en particular o para todos los bancos desde la interface del usuario.

Para generar los archivos de salida por ejemplo, buscaríamos en la tabla REGISTROS todos los registros que coinciden con el banco girado para el cual queremos generar el archivo de cheques a su cargo. En esta tabla si dependemos del volumen de información ya que tenemos que recorrer de principio a fin la tabla de los registros, pero aprovechamos la ocasión para generar de una sola pasada los archivos a su cargo de todos los bancos, así que no tenemos que recorrerla una vez para cada banco. La generación de los archivos de cheques a su cargo se activa por medio de la selección de algunas opciones del sistema, como por ejemplo la opción "Archivo de Salida en Disquete", donde se solicita al usuario el número de banco del cual quiere generar en disquete el archivo de cheques a su cargo.

Por último, la tabla BANCOS, es la única que no se crea desde cero cada vez que iniciamos un proceso. En esta tabla se da mantenimiento a los bancos que se dan de alta o de baja del servicio de compensación electrónica de cheques. La actualización de esta tabla se hace por medio de una opción de la interface de usuario llamada "Mantenimiento al Catálogo de Bancos".

3.1.5 Diagrama entidad relación

Los diagramas de entidad relación o diagramas E-R son parte de las herramientas conceptuales que existen para describir datos y las relaciones que existen entre ellos. El modelo entidad relación esta basado en una percepción del mundo como si consistiese de una colección de objetos básicos (entidades) y relaciones entre estos objetos.

* Una entidad es un objeto distinguible que existe.
* Cada entidad tiene asociados con ella un conjunto de atributos que la describen.
* Una relación es una asociación entre varias entidades.

La estructura lógica completa de una base de datos puede ser expresada gráficamente por un diagrama entidad relación.

Las entidades corresponden a las tablas en un sistema de base de datos relacional y se representan por medio de rectángulos.

Los atributos se muestran como óvalos que se conectan por medio de una linea al rectángulo de la entidad correspondiente.

Las relaciones que son asociaciones entre las entidades, se representan por un diamante con líneas que pasan a través de él y que conectan a las entidades.

El grado de la relación se representa de la siguiente manera:

Uno a Uno (1:1)
Uno a muchos (1:M)
Muchos a muchos (M:N)

El diagrama de entidad relación para el sistema de compensación electrónica de cheques se muestra en la figura 3.1.5.1 (se muestran sólo los atributos principales).

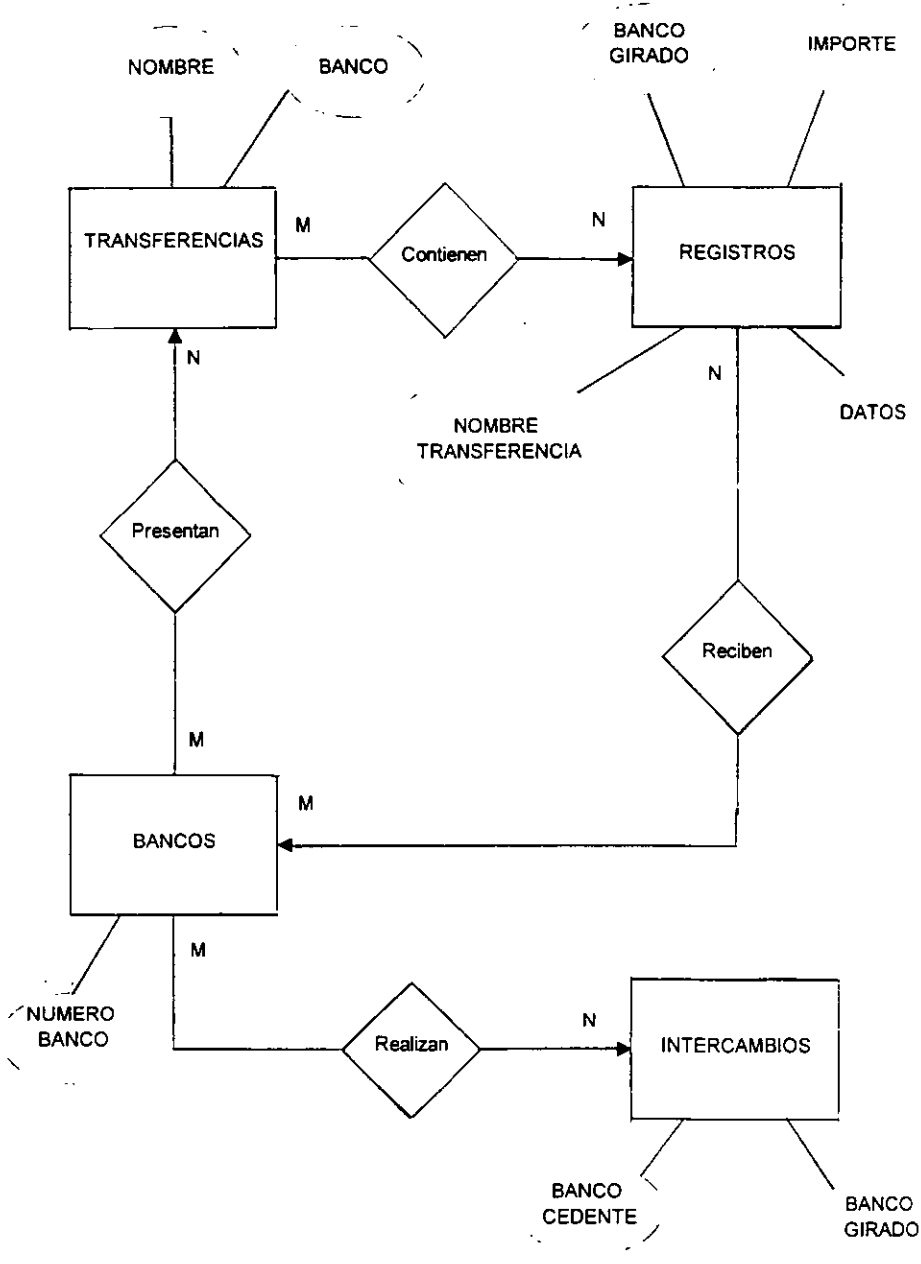


Figura 3.1.5.1 Diagrama Entidad-Relación del sistema de compensación electrónica.

3.2 GENERACIÓN DE CÓDIGO PARA EL PROCESAMIENTO DE LA INFORMACIÓN.

En el apéndice D. se muestra el código de todo el sistema, además se explica de manera general para que sirve cada una de las unidades programadas.

3.3 DISEÑO Y CONSTRUCCIÓN DEL FRONT END

Introducción al ambiente de desarrollo de aplicaciones mediante Delphi

Delphi se divide en tres secciones, el compilador (con su ligador), la biblioteca, y el IDE (ambiente de desarrollo integrado, o Integrated Development Environment). El compilador/ligador es un programa que crea el archivo ejecutable de Windows estilo Intel, sin ningún interprete de por medio. La biblioteca es código que nos permite usar todas las capacidades de Delphi. La biblioteca esta escrita en su totalidad en Object Pascal (es una biblioteca "de clases" estilo Microsoft Fundation Classes, MFC [Clases de la Fundación Microsoft], llamada Visual Component Library [biblioteca de componentes visuales]), y es totalmente orientada a objetos.

Una nota interesante es que el IDE esta desarrollado en Delphi, y utiliza las mismas bibliotecas que se utilizan para compilar los programas. El IDE es utilizado para realizar el diseño y la programación de la aplicación, cuando se inicia Borland Delphi se observa el IDE como se muestra en la figura 3.3.1.

Como se ve, Delphi creó automáticamente una ventana titulada Form1 (Forma1), esta tiene algunos elementos comunes a las ventanas de Windows, es el plano de fondo para los controles, se puede cambiar el tamaño y las propiedades de la misma utilizando el ratón. Los puntos que se visualizan nos ayudan a alinear los controles que coloquemos dentro.

Para empezar a diseñar una ventana (forma) dentro de Delphi, lo primero que tenemos que hacer es colocar los controles sobre una forma, diseñando los elementos que el usuario tiene que utilizar para comunicarse con la aplicación.

Obsérvese que la pantalla principal del IDE tiene una zona llamada barra de controles que contiene lo que se conoce como paletas de controles. Estas paletas de controles están identificadas por unas etiquetas que dicen Standard, Additional, Win32, etc. Estas

paletas seleccionan un conjunto de controles determinado, la paleta standard muestra los controles más comunes.

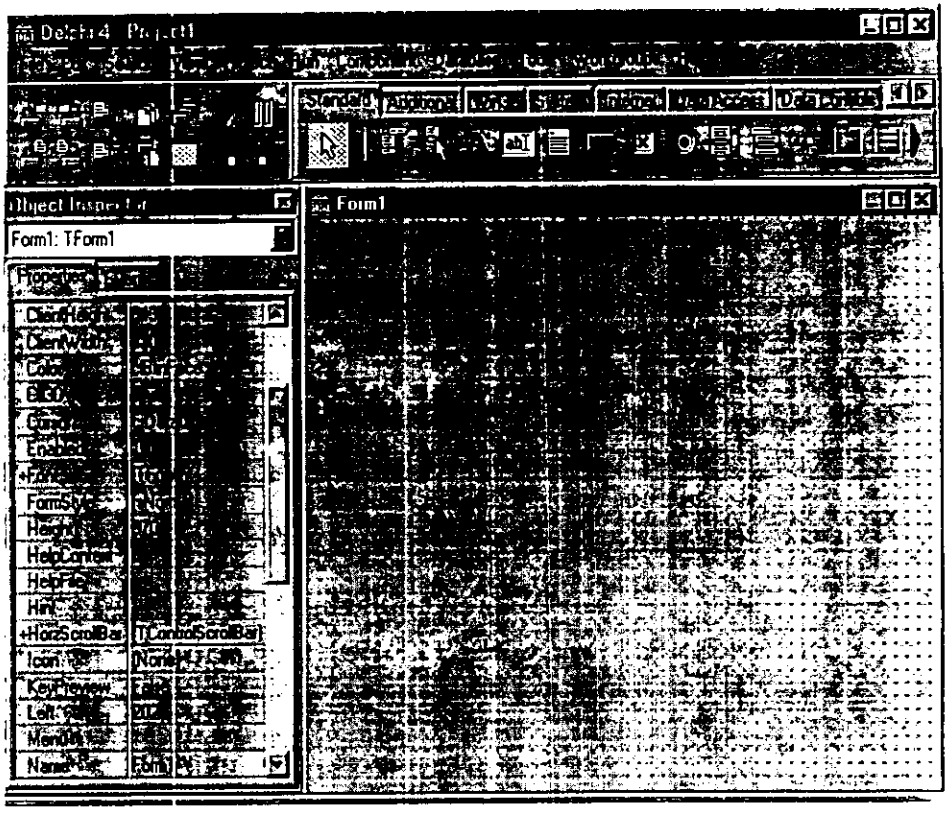


Figura 3.3.1 Ambiente de desarrollo Integrado (IDE)

Para agregar un control se emplea la barra de controles, esta crea un control único. Para nuestra aplicación, necesitamos colocar controles que nos permitan "navegar" por la aplicación. Esto lo realizaremos por medio de una imagen gráfica (icono) que cuando sea seleccionada con el ratón, lleve a cabo determinada acción. Para colocar un control de imagen, se elige de la paleta de controles "Additional" (Adicional) el control "Image" (imagen) como se describe a continuación:

Se selecciona el sexto control de la paleta additional (Image), ver figura 3.3.2.

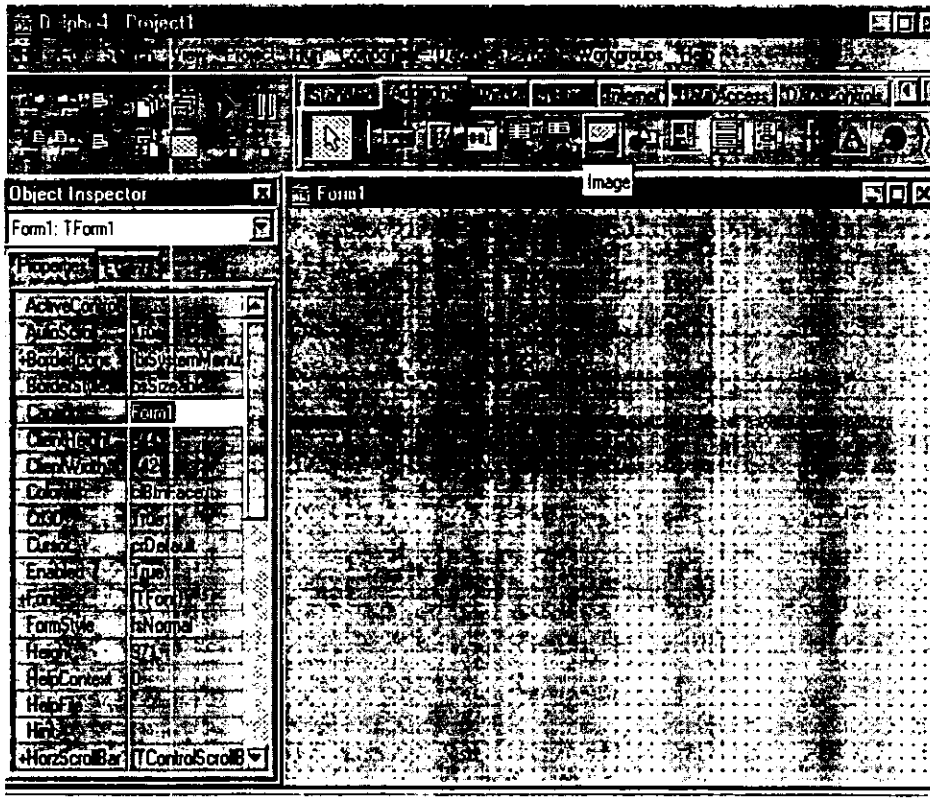


Figura 3.3.2 Selección del control "Imagen".

Ahora, se pulsa dos veces sobre el componente, con lo anterior, el control se despliega en la forma, este control nos servirá para crear el ícono (véase la figura 3.3.3).

Ahora tenemos el control de imagen en nuestra forma de diseño, sin embargo este no tiene la presentación adecuada para nuestros fines, así que procedemos a modificar las características del control que acabamos de colocar.

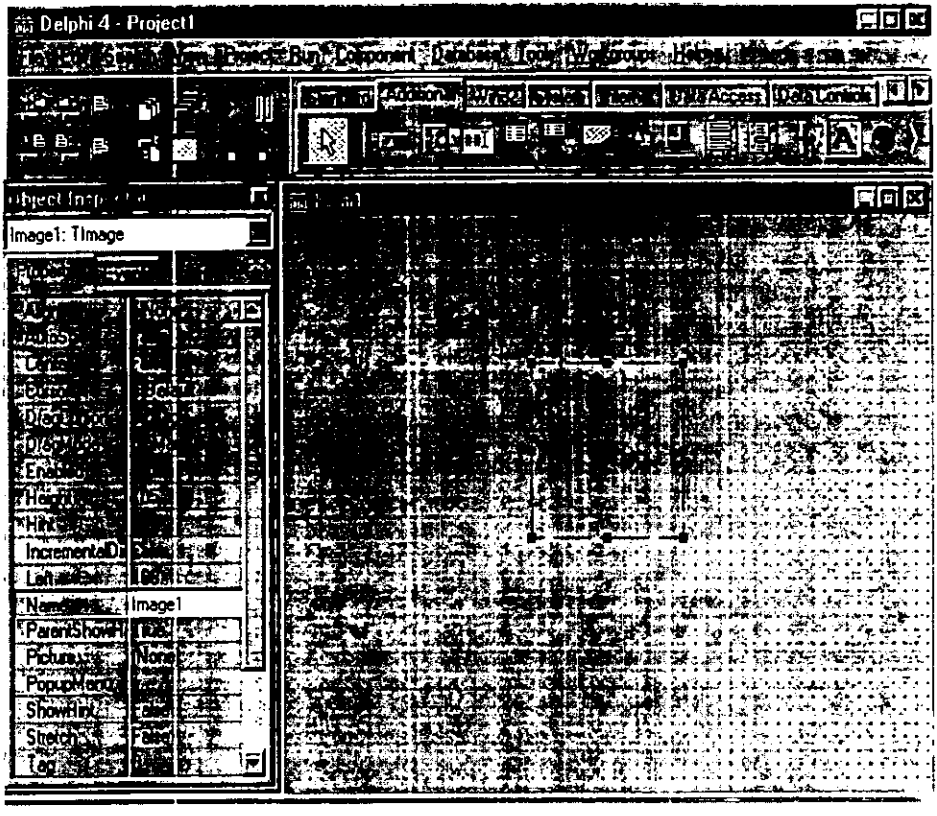


Figura 3.3.3 Control "Imagen" sobre la forma de diseño.

En primer término, para asignar una imagen a nuestro control, hacemos doble click en el mismo, con lo que aparece una pantalla que se conoce como editor de imágenes (Picture Editor). En esta ventana se elige la opción Load (cargar) para poder seleccionar la imagen deseada (véase la figura 3.3.4). En este caso seleccionaremos un ícono, que es como el botón de avance de una grabadora. Después de que hemos cargado la imagen hacemos click en el botón Ok (Aceptar) para regresar a la forma de diseño, la que ahora tendrá el aspecto que se muestra en la figura 3.3.5.

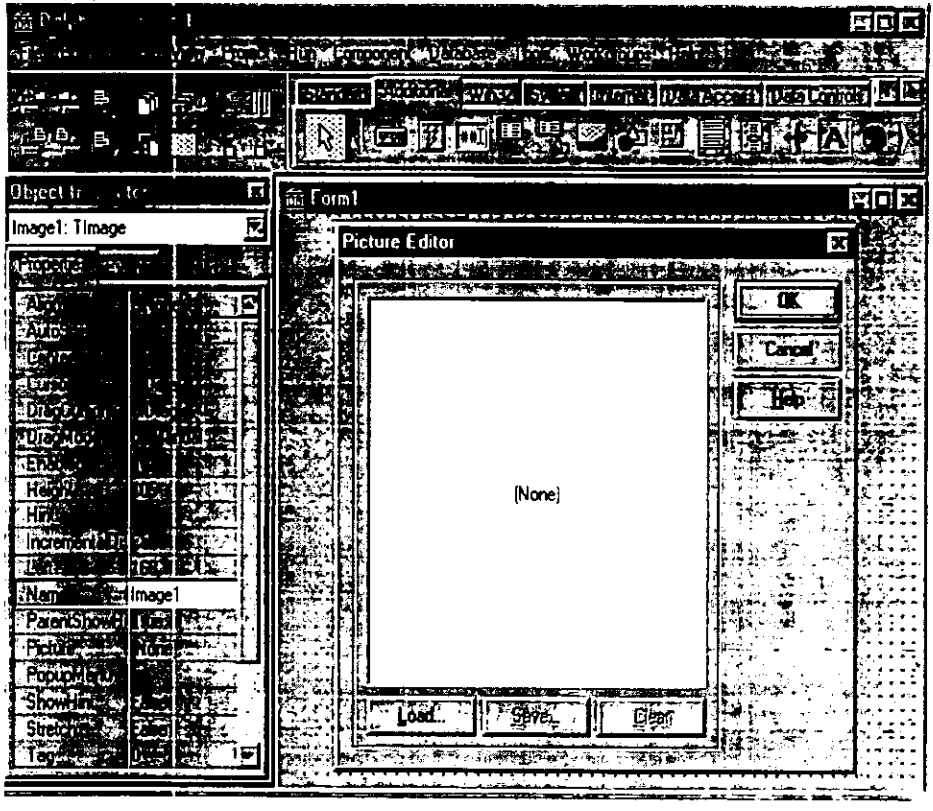


Figura 3.3.4 Asignación del ícono al control.

La modificación de las propiedades de cualquier objeto dentro del proyecto, se puede hacer mediante el uso de una herramienta llamada "Object Inspector" (Inspector de Objetos), el cual se muestra en la figura 3.3.5.

Ahora, vamos a modificar varias de las propiedades del icono con el que estamos trabajando. Primeramente modifiquemos la opción AutoSize (Tamaño Auto-ajustable) con lo cual el control tomará las dimensiones de la imagen asignada. Establezcamos la propiedad de tal manera que su valor sea "true" (verdadero).

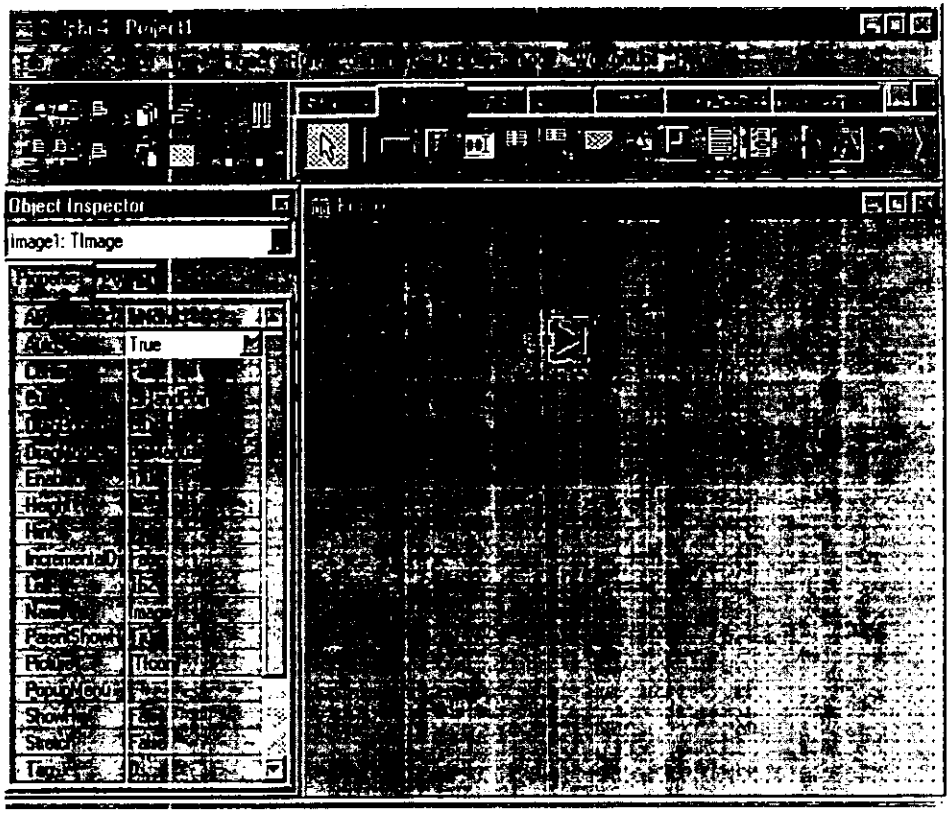


Figura 3.3.5 Inspector de objetos.

A continuación cambiemos la propiedad Cursor, por medio de la cual, el cursor cambiará de apariencia cuando el usuario coloque el apuntador del ratón sobre la imagen. En este caso vamos a escoger un apuntador en forma de mano con el dedo índice apuntando hacia arriba, es decir seleccionemos el valor crHandPoint (véase la figura 3.3.6).

Modifiquemos ahora la propiedad llamada Hint (Indicación), con esto, cuando el usuario coloque el apuntador del ratón sobre el ícono, se desplegará un mensaje que dará información de la acción que lleva a cabo dicho control. En este caso asignemos a esta

propiedad la leyenda "Siguiente", para indicar que este botón es para avanzar a la siguiente ventana de la aplicación (véase la figura 3.3.7).

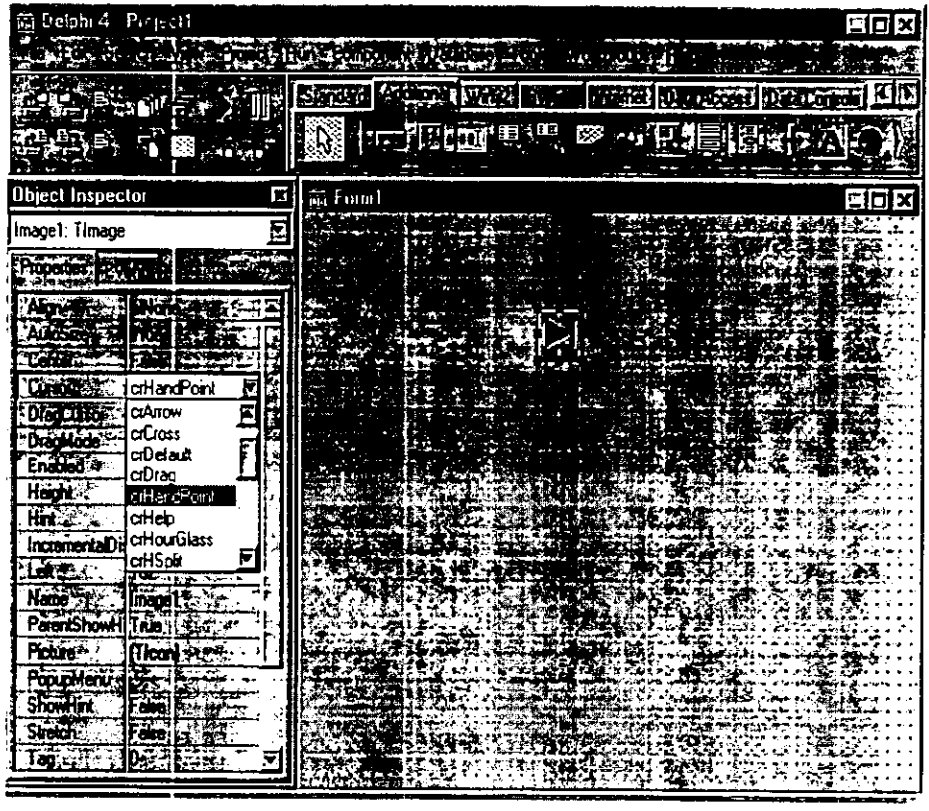


Figura 3.3.6 Modificación de la propiedad Cursor.

Finalmente, en lo que respecta a este control, modifiquemos la propiedad ShowHint (Mostrar indicación), establezcamos su valor a true (verdadero), para que muestre la leyenda que colocamos anteriormente (véase la figura 3.3.8).

Para crear otros controles similares al anterior, realizamos para cada uno, todos los pasos anteriormente descritos. En esta ventana en particular se desea colocar otros dos botones de grabadora, uno para retroceder y otro para detener (terminar) la aplicación. Además se agregó otro control de este tipo para llevar a cabo la acción de inicializar el sistema.

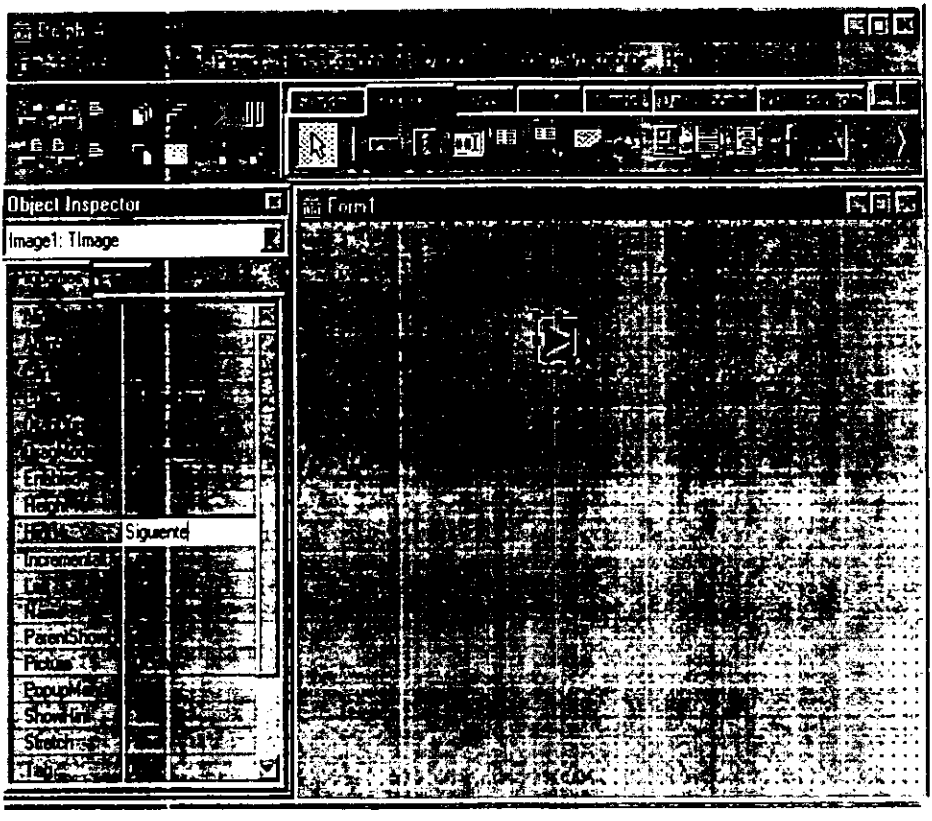


Figura 3.3.7 Modificación de la propiedad Hint.

Ahora coloquemos algunos controles más a nuestra forma. Seleccionemos de la paleta de controles "Standard" el control llamado "Panel"; hacemos doble click con el ratón sobre el control y una vez sobre la forma lo colocamos en la parte inferior de la misma. Coloquemos otros dos controles de este tipo y agrupémoslos en la parte inferior de la forma, como se muestra en la figura 3.3.9.

Ahora, modifiquemos la propiedad caption (título) de estos controles para dejarla en blanco, es decir, que inicialmente no muestren ningún titulo o mensaje. El primero de estos controles nos servirá para mostrar un mensaje breve que indique, con una o dos palabras, la acción que lleva a cabo el ícono sobre el cual se posiciona el ratón.

También servirá para indicar que acción se esta realizando en un momento dado. La propiedad caption de este control se modificará en tiempo de ejecución, de la siguiente manera:

Panel1.Caption := *Mensaje*.

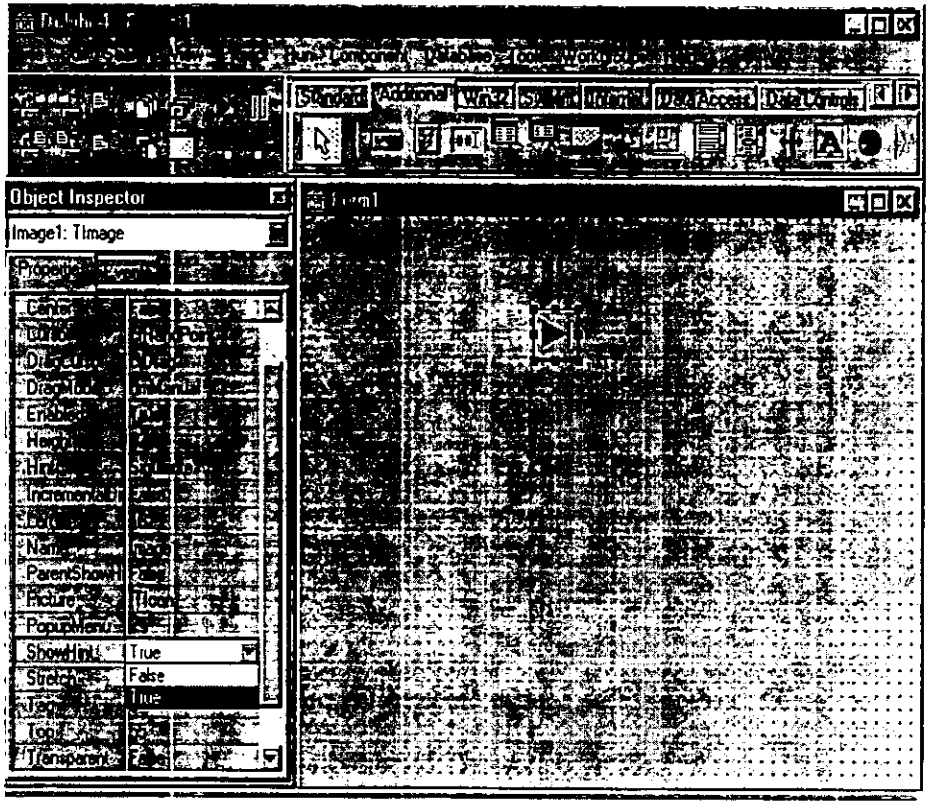


Figura 3.3.8 Modificación de la propiedad ShowHint.

Los otros dos paneles serán utilizados para mostrar la fecha del proceso y la fecha de aplicación respectivamente. Las propiedades "Caption" de estos controles, también serán modificadas en tiempo de ejecución según la fecha con la que esté trabajando el sistema. De esta manera el operador podrá saber, en todo momento, en que fechas se encuentra trabajando.

Es importante mencionar que el contenido de la propiedad caption no identifica a un componente, sino que los componentes son identificados por su nombre (propiedad name). Delphi asigna automáticamente el nombre a los controles, pero por supuesto que se puede poner el nombre que uno desee.

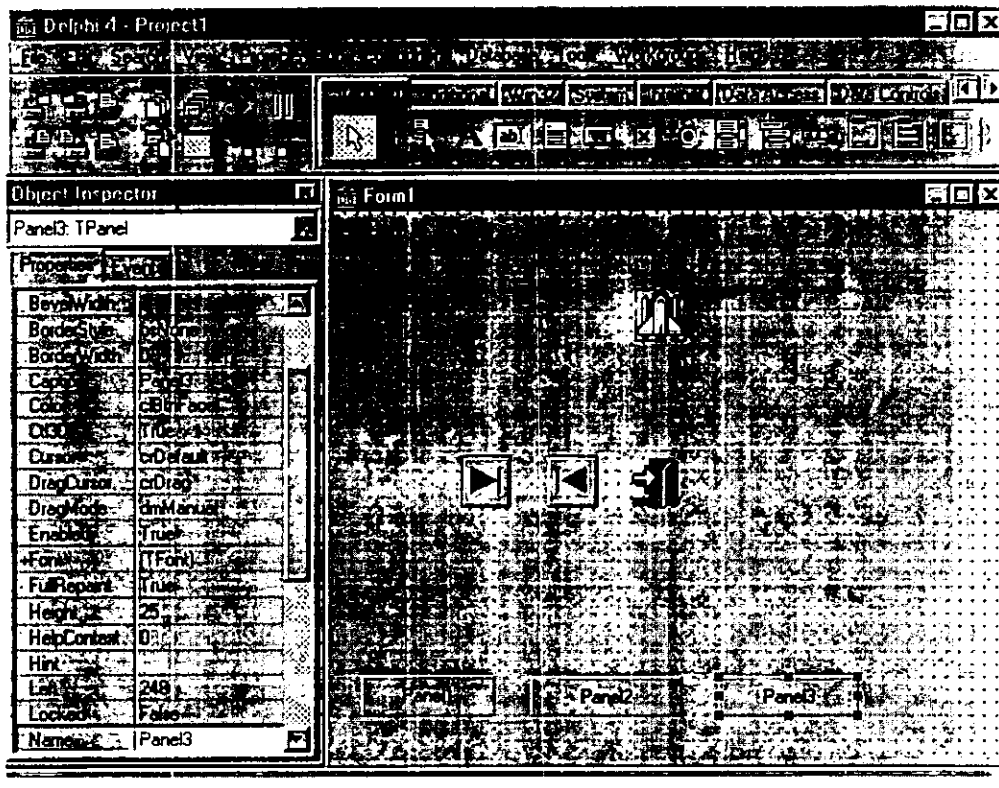


Figura 3.3.9 Se agrega los controles "Panel"

Una vez terminado el diseño de los controles gráficos, modifiquemos la propiedad caption de la forma1. Le asignamos el título a la ventana, en este caso "Inicialización". Asimismo, modifiquemos la propiedad BorderStyle (estilo del borde) que por omisión permite cambiar las dimensiones de la ventana con el ratón. En este caso no queremos que el usuario pueda modificar el tamaño de las ventanas por lo cual establecemos esta

propiedad al valor bsSingle, con el cual se dibuja un borde sencillo alrededor de la ventana y no permite que se cambien las dimensiones de la misma (véase la figura 3.3.10).

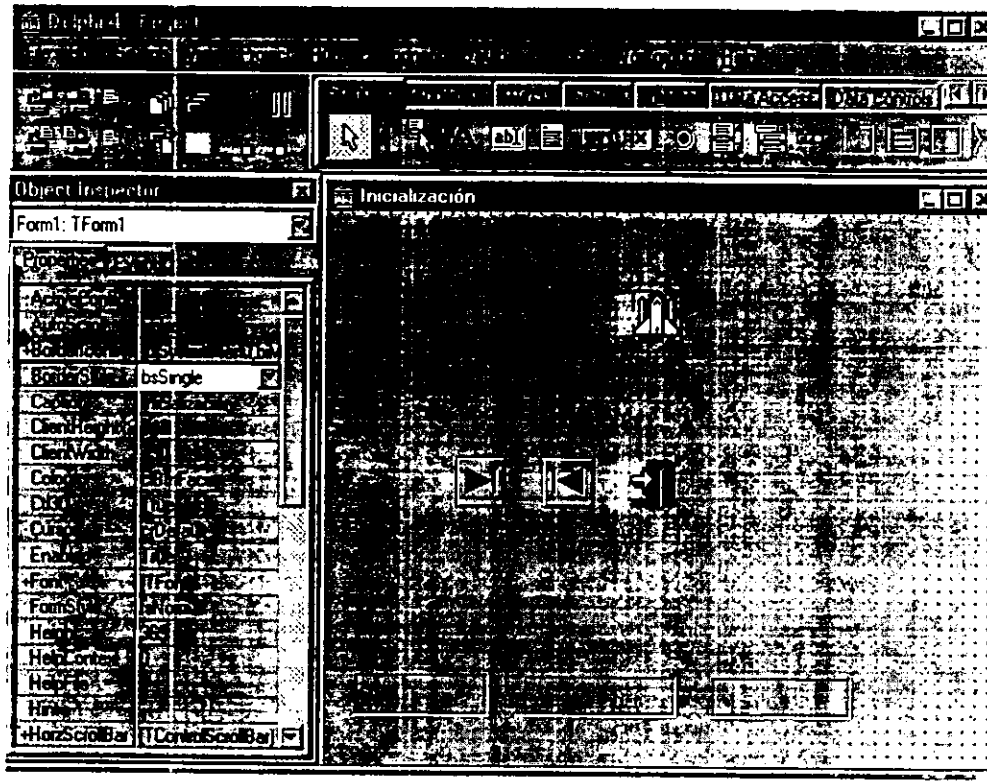


Figura 3.3.10 Modificación de la propiedad BorderStyle.

Otra propiedad de la forma1, que vamos a modificar, es BorderIcons (Iconos del Borde), con esto establecemos que controles de la ventana estarán disponibles para el usuario. En este caso, queremos que el usuario pueda minimizar la ventana (por si desea cambiar de aplicación), y que pueda cerrarla (por si desea terminar la aplicación de esta manera). Sin embargo, no permitiremos que pueda maximizar la ventana, porque la apariencia de la misma se distorsionaría, así que esta propiedad la establecemos a un valor de False (falso), de tal manera que este ícono no aparezca en la parte superior derecha de la ventana. Por supuesto, los otros dos controles que estarán disponibles

para el usuario, se establecen a un valor true (verdadero), para que sean visibles y puedan seleccionarse (véase la figura 3.3.11).

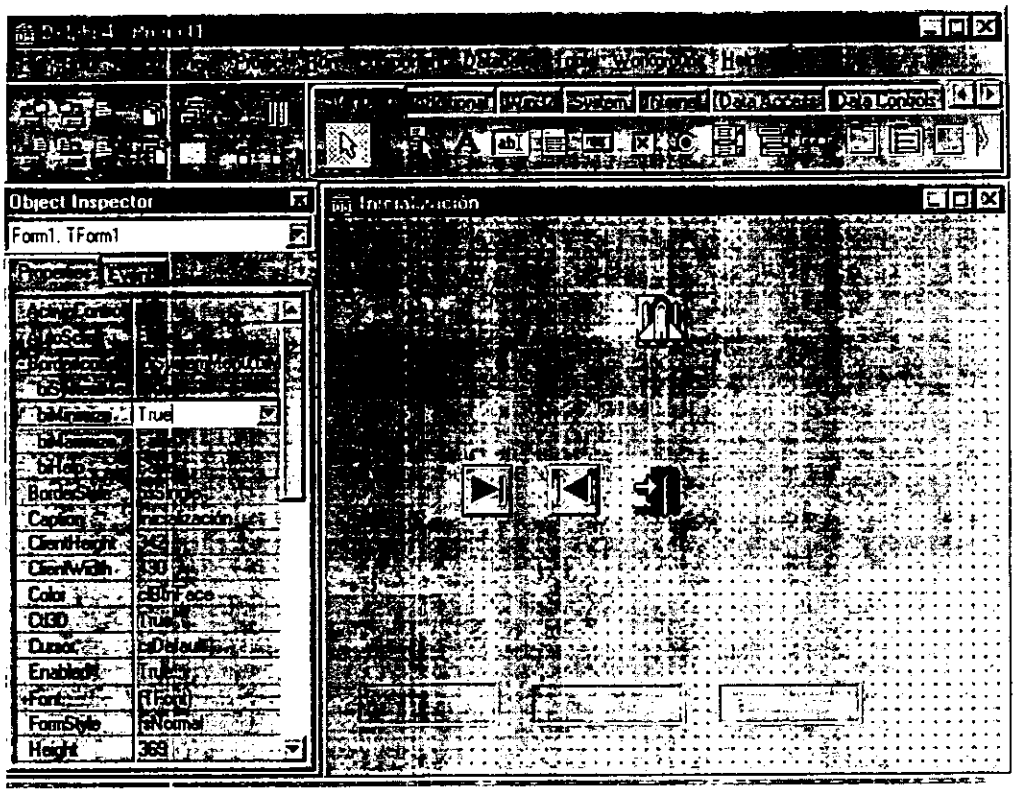


Figura 3.3.11 Modificación de la propiedad BorderIcons.

Finalmente, para terminar con esta forma, arrastramos y colocamos cada elemento en su posición final, las dimensiones de la ventana se ajustan posicionandose con el ratón en los bordes de la misma y reduciendo o ampliando el tamaño hasta que tenga la apariencia deseada, esto en realidad es muy sencillo con el uso de herramientas tales como Delphi (véase la figura 3.3.12).

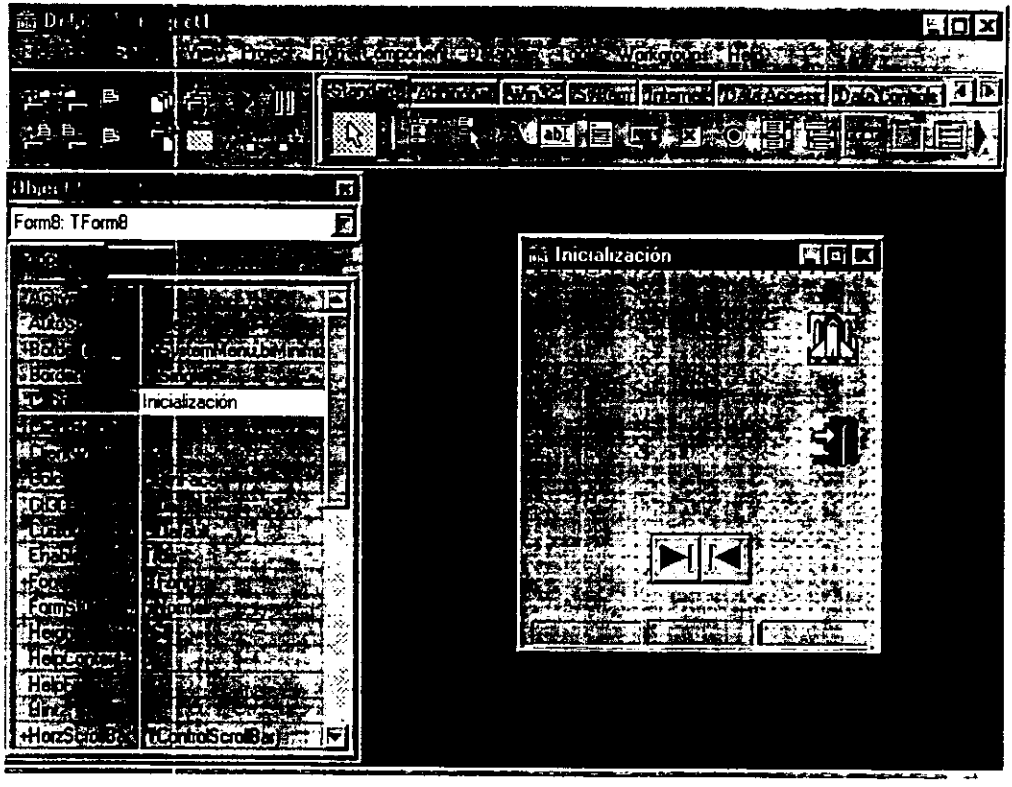


Figura 3.3.12 Apariencia final de la forma de diseño.

De la misma manera en que se hizo anteriormente, se crearon otras ventanas que contienen este tipo de controles y que nos permiten navegar por el sistema. De esta manera podemos pasar de una fase del proceso de compensación electrónica a la siguiente, o a la anterior, según se necesite. En las figuras 3.3.13 a 3.3.19 se muestran las principales ventanas que conforman la aplicación. Para más información sobre las opciones de cada forma, consultar el manual de usuario.

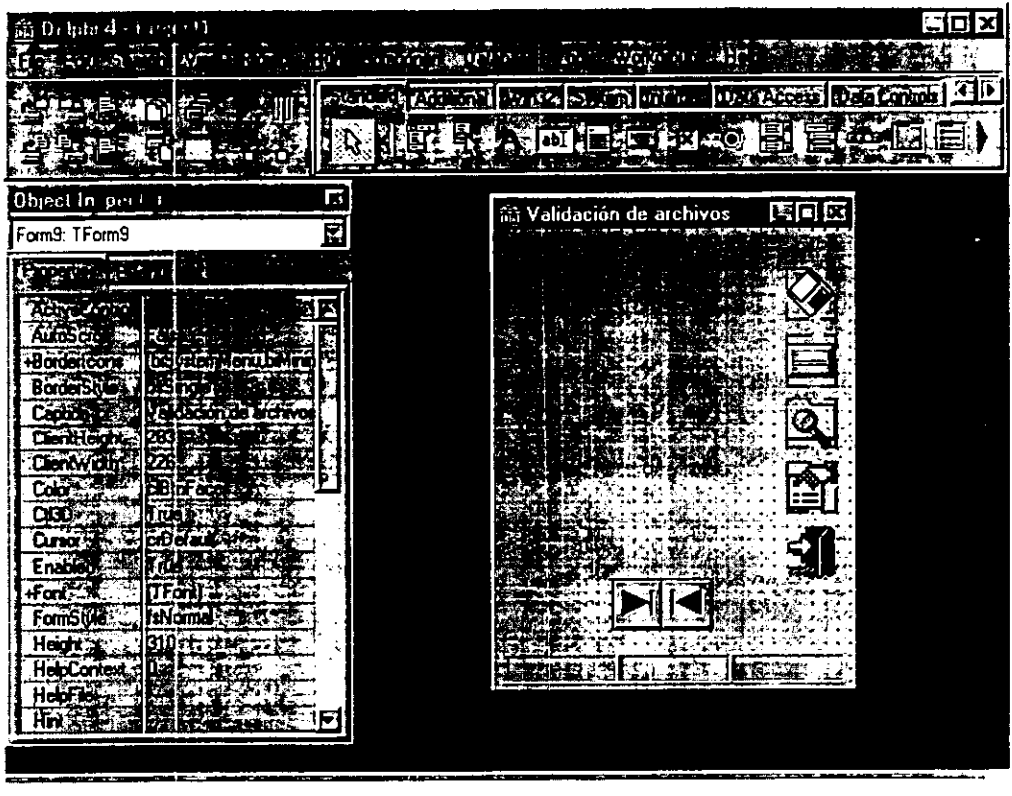


Figura 3.3.13 Ventana de Validación.

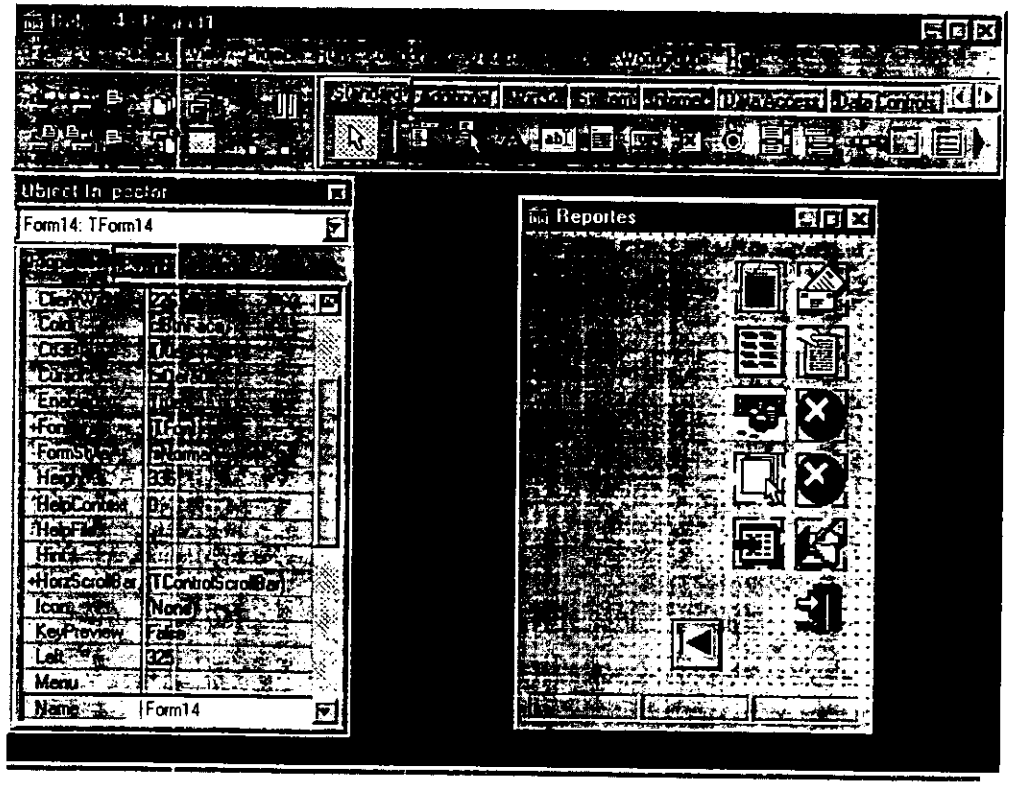


Figura 3.3.14 Ventana de Reportes.

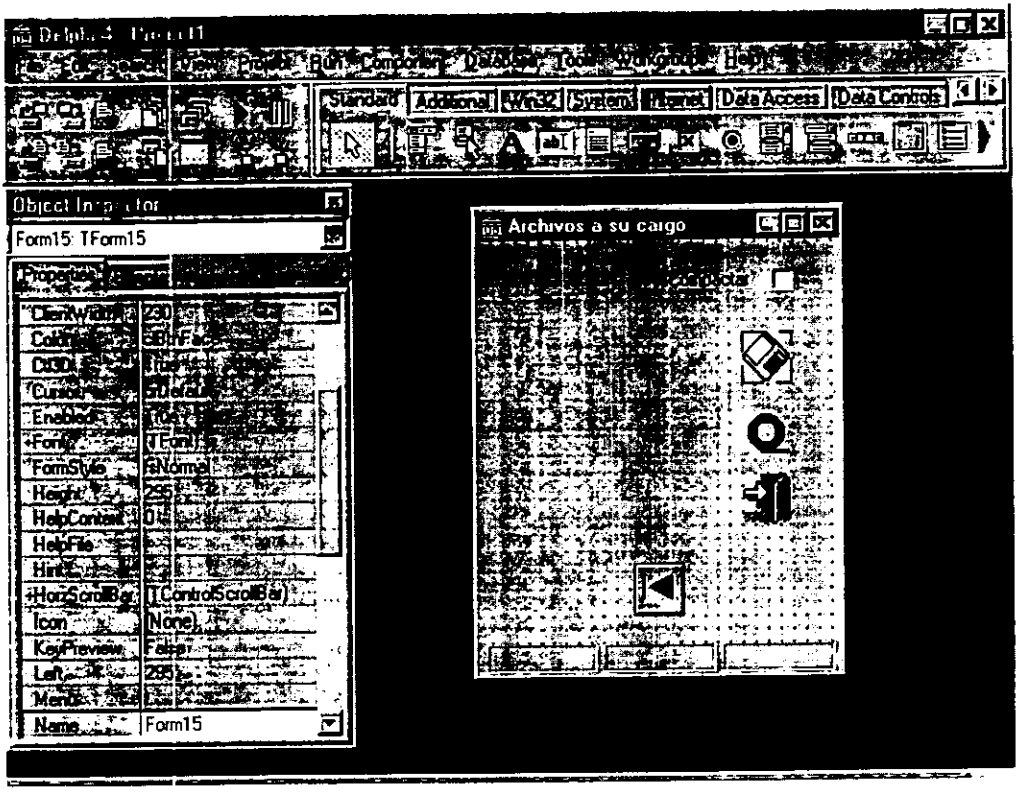


Figura 3.3.15 Ventana de "Salidas" (archivos a su cargo).

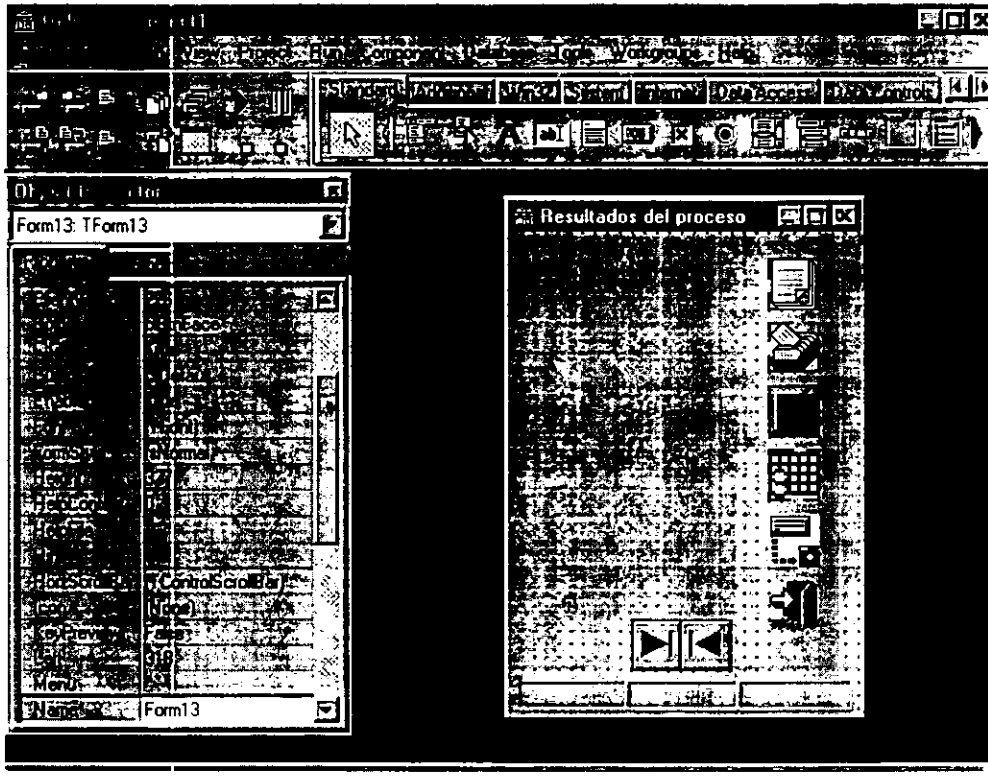


Figura 3.3.16 Ventana para generar los "Resultados" del proceso.

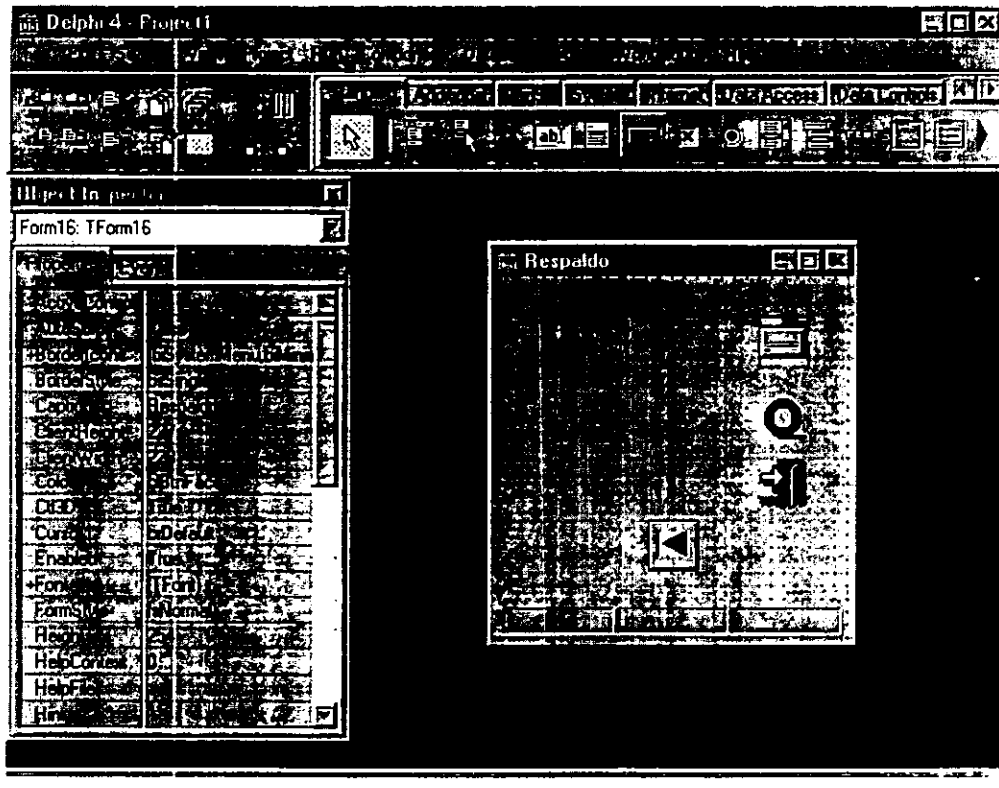


Figura 3.3.17 Ventana para realizar el "Respaldo" de la información.

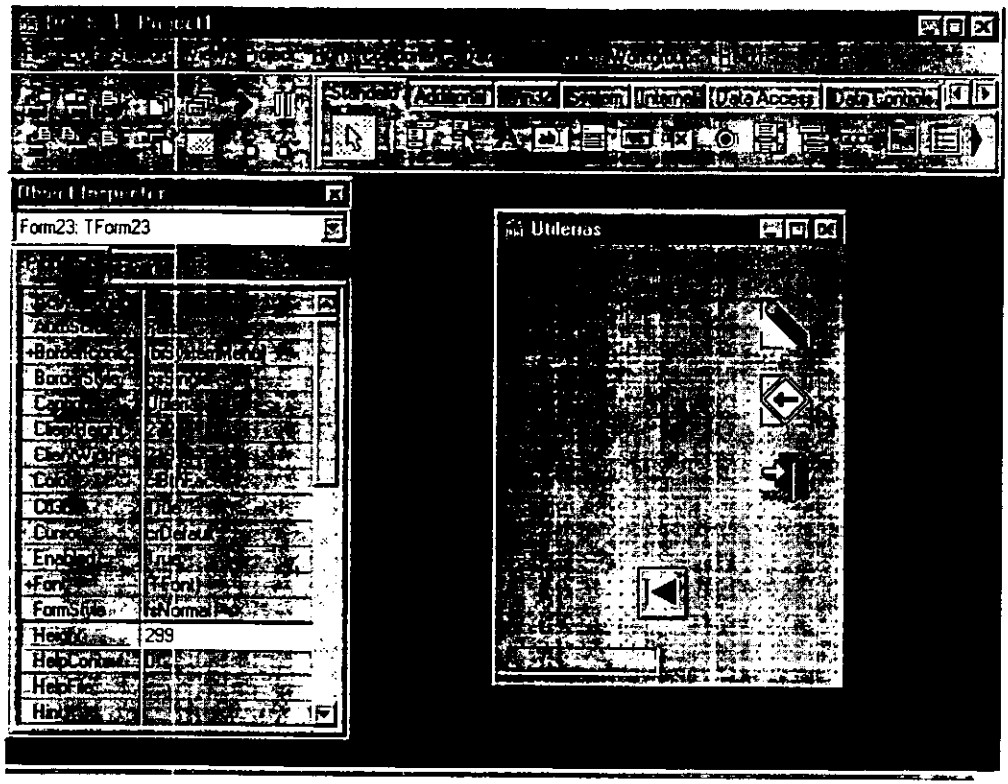


Figura 3.3.18 Ventana de "Utilerías".

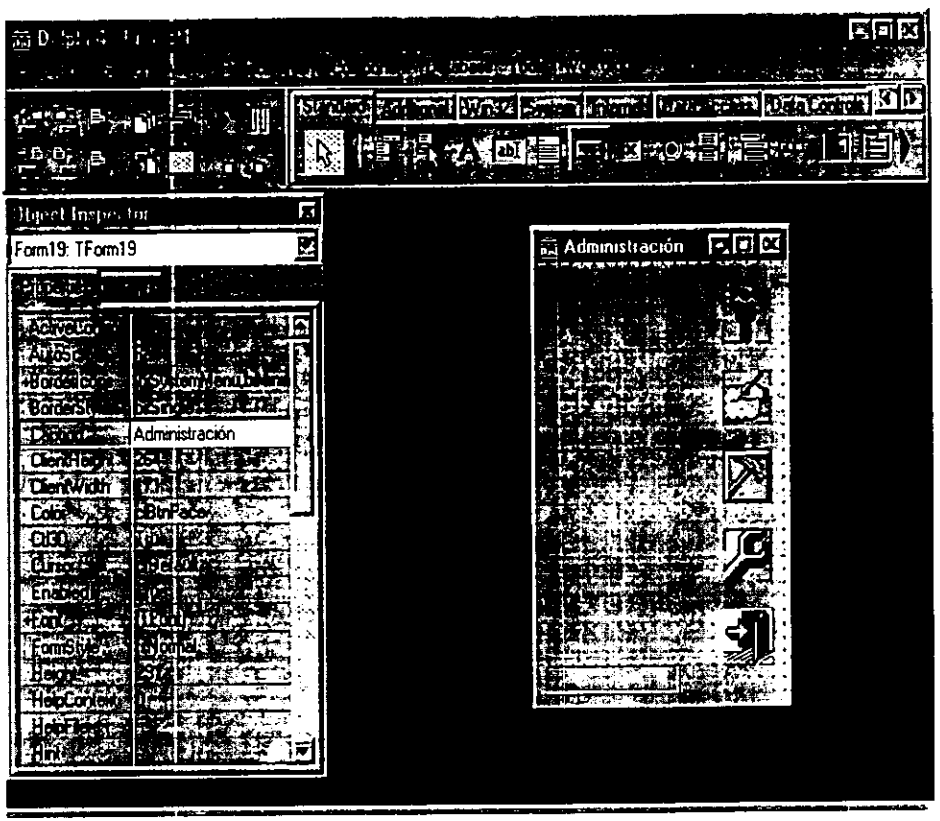


Figura 3.3.19 Ventana de "Administración".

El usuario de la aplicación hará uso de las ventanas mostradas anteriormente para operar el sistema. En algunas opciones, se le pide al usuario que proporcione algún dato de entrada. Por lo general este dato de entrada es el número de uno de los bancos participantes en la compensación electrónica de cheques. Para llevar a cabo la captura de este dato, se creó una forma, que permite seleccionar el número de banco del cual se desea obtener información. Para agregar una nueva forma al proyecto se usa la

opción New Form (Nueva Forma), la cual se encuentra en el menú File (Archivo) del IDE de Delphi. En la figura 3.3.20 se muestra la opción New Form y la nueva forma creada (Form2).

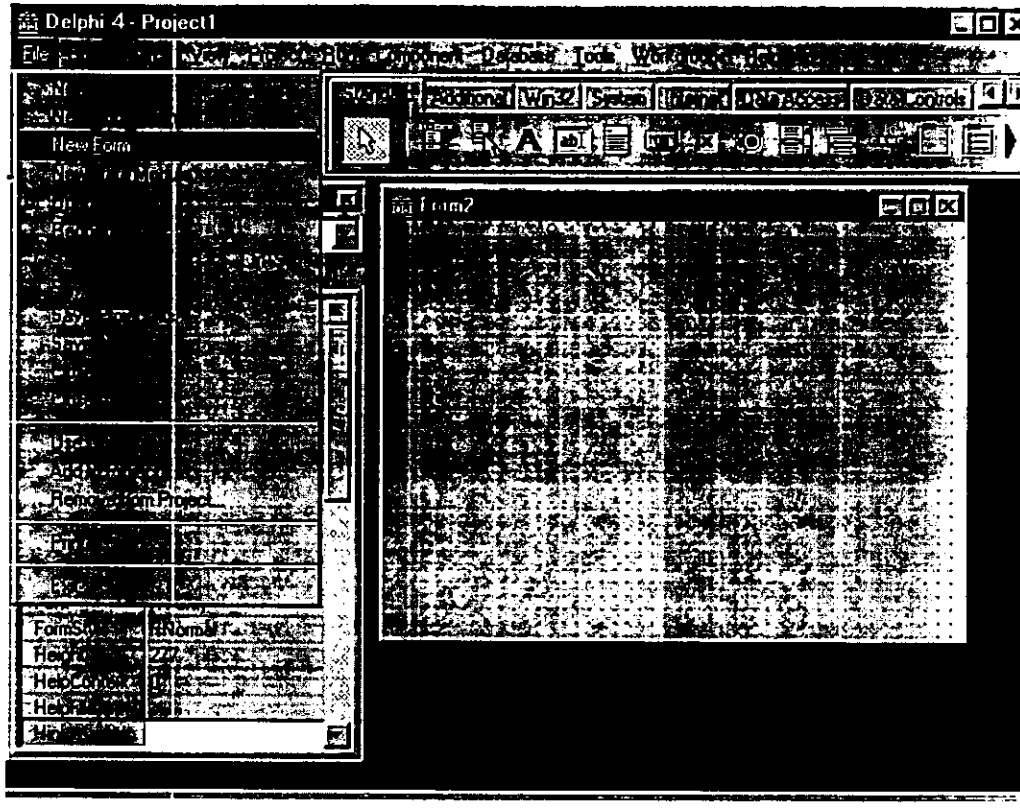


Figura 3.3.20 Creación de una nueva forma

Dentro de la forma2, vamos a incluir un control llamado ComboBox (Caja Combinada), el cual nos permitirá seleccionar de una lista, el banco a procesar. Igual que como se describió anteriormente, para incluir un ComboBox en nuestra forma2, hacemos doble click en el elemento ComboBox de la paleta de controles standard y Delphi lo coloca automáticamente como se muestra en la figura 3.3.21.

Para este control, la lista de elementos se puede especificar mediante la propiedad Items (elementos). Al seleccionar esta propiedad aparece el String List Editor (Editor de Lista de Cadenas), donde se pueden ir tecleando los elementos de la lista. Sin embargo, en esta aplicación, los elementos de la lista serán agregados en tiempo de ejecución, haciendo el uso de la tabla de bancos. Cada banco en la tabla será agregado como un elemento de la lista del ComboBox.

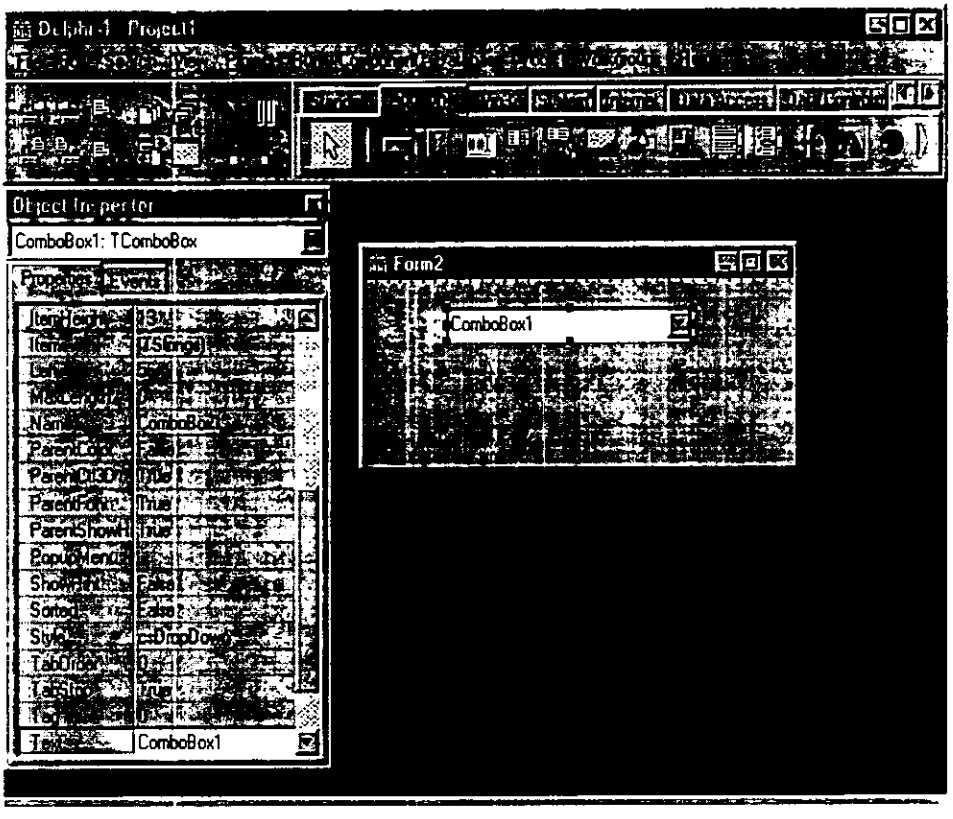


Figura 3.3.21 Control de ComboBox (caja combinada).

Se puede hacer referencia a cada elemento de la lista mediante un índice y de esta manera asignarle un valor. Tenemos algo como lo siguiente:

```
Form2.ComboBox1.Items[Indice] := StrBanco;
```

Donde StrBanco es una cadena de caracteres denotando el nombre del banco e Indice es una variable para llevar el control del elemento de la lista al que nos estamos refiriendo.

Dado que esta forma de captura puede ser usada en varios contextos, el titulo de la misma será modificado, de acuerdo a su uso, en tiempo de ejecución. Para realizar lo anterior se modifica la propiedad caption de la forma2, de la siguiente manera:

Form2.Caption := 'Consulta de transferencias por banco cedente';

El texto a la derecha de la asignación se puede modificar de acuerdo al contexto en que se esté trabajando.

En esta forma2 vamos a incluir además dos controles de "botón gráfico" (BitBtn), para verificar si el usuario quiere continuar con el proceso o quiere cancelarlo. Cualquier control que se desee incluir en cualquiera de las formas, se inserta haciendo doble click sobre el mismo, como se ha explicado anteriormente. La figura 3.3.22 muestra la inclusión de estos dos botones dentro de la forma2.

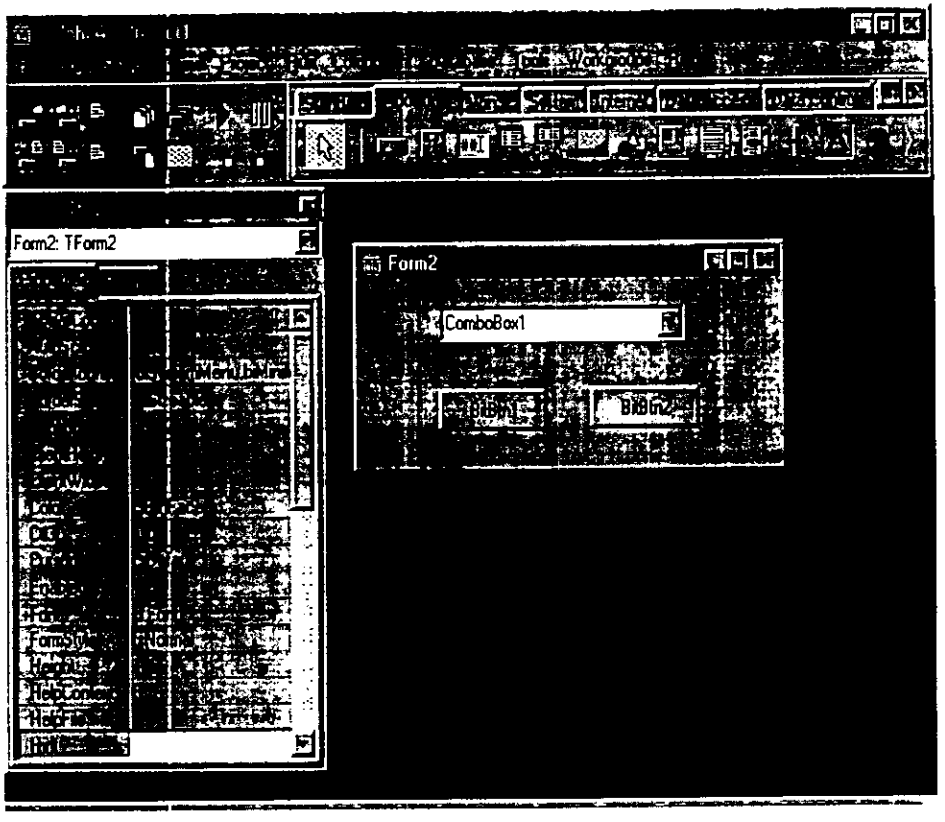


Figura 3.3.22 Controles BitBtn.

Vemos que por omisión Delphi le pone el título a estos botones, en este caso a uno lo titula BitBtn1 y al otro le pone BitBtn2. Para cambiar el título de estos controles, nuevamente hacemos uso de la propiedad caption. Al botón1 le pondremos la leyenda Aceptar, y al botón2 le asignaremos la leyenda Cancelar. El símbolo "&" antes de la palabra Aceptar indica que este control puede ser seleccionado al oprimir las teclas ALT + A (letra que esta enseguida del símbolo "&" y que además aparece subrayada). Adicionalmente, para que aparezca sobre cada botón un icono que represente la acción a realizar, se modifica la propiedad Kind (tipo). Al primer botón le asignamos el tipo bkOK y al segundo el valor bkCancel, con estas modificaciones los botones tienen el aspecto que se muestra en la figura 3.3.23.

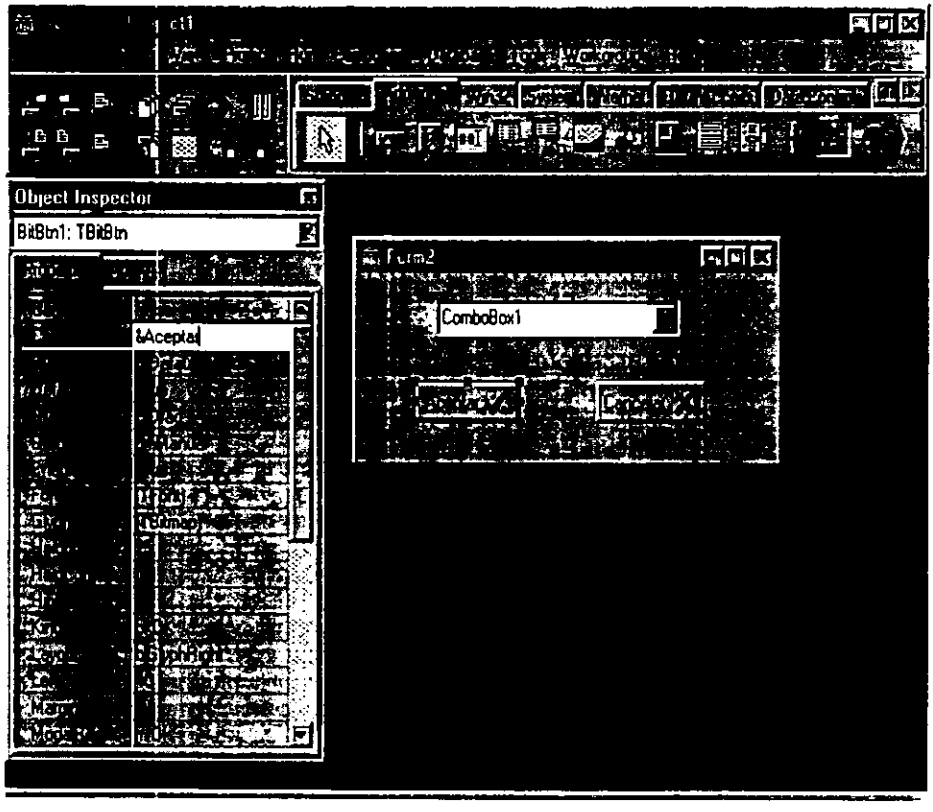


Figura 3.3.23 Botones de Aceptar y Cancelar.

Existen diferentes formas que se han incluido en el proyecto, para su creación se han utilizado los mismos procedimientos descritos hasta ahora, tanto para la creación de nuevas formas como para la inclusión de controles dentro de las formas.

En las figuras 3.3.24 a 3.3.30 se muestra la mayoría del resto de las formas incluidas en el proyecto. Se incluye después de cada forma, una breve descripción de la misma.

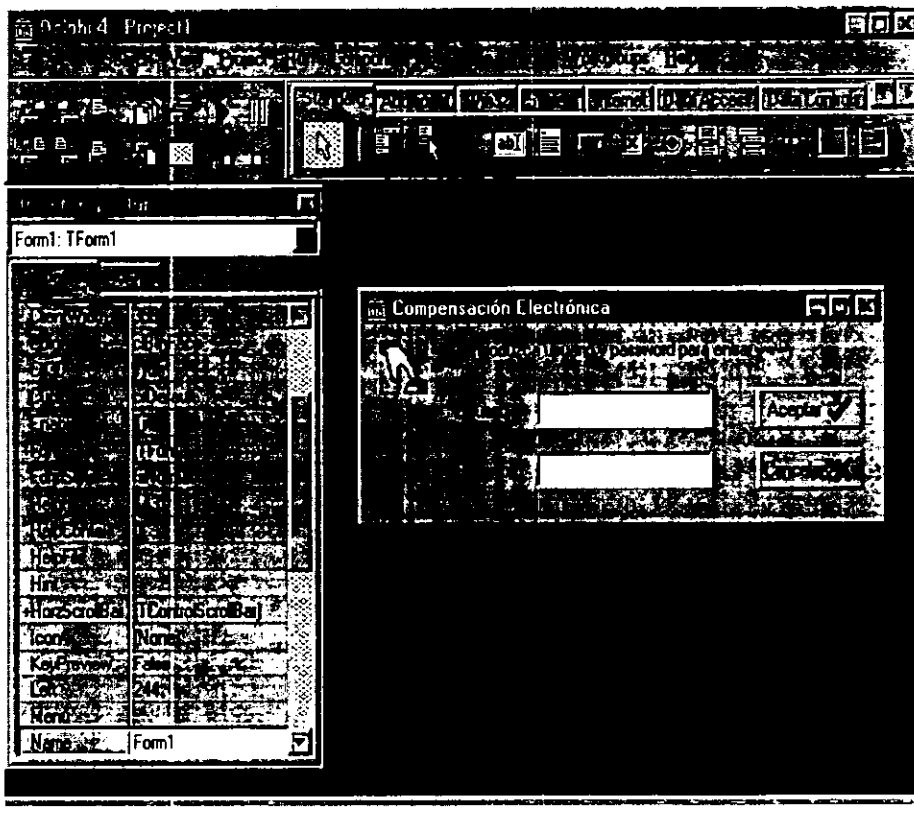


Figura 3.3.24 Forma para solicitar el usuario y password.

Para tener acceso al sistema debe proporcionarse una clave de usuario junto con el password (contraseña) correspondiente. El sistema valida la información tecleada y dependiendo de los derechos del usuario le permite llevar a cabo la operación del sistema o el mantenimiento del mismo.

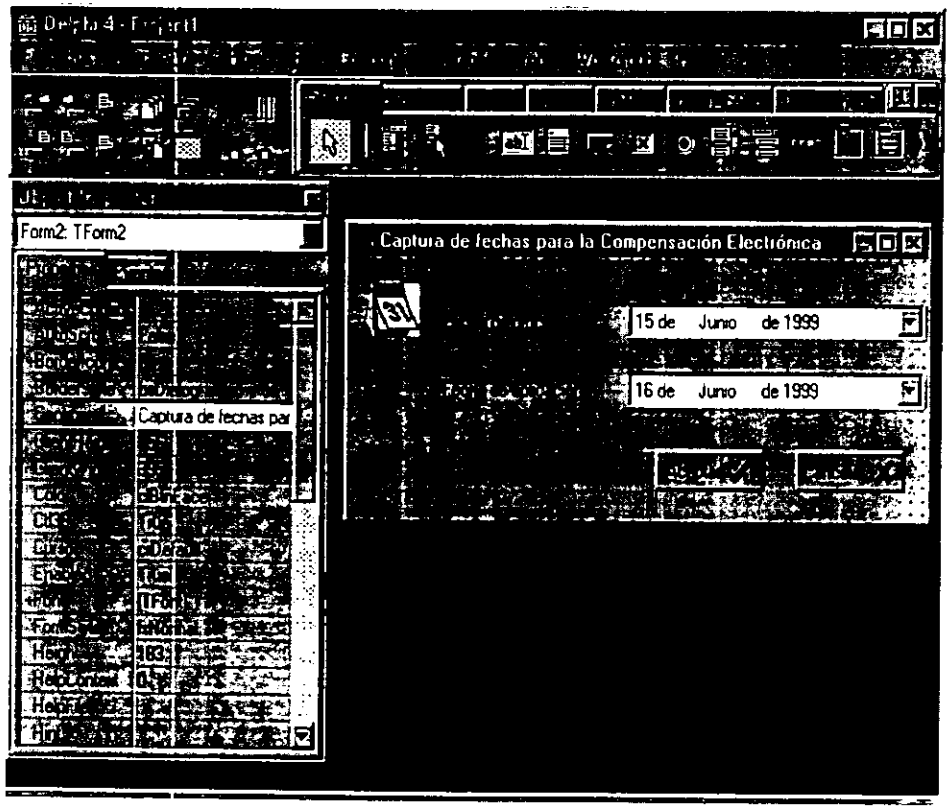


Figura 3.3.25 Forma para validar las fechas.

En la fase inicial del proceso de compensación electrónica aparece la forma de la figura 3.3.25, esta sirve para seleccionar las fechas de proceso y de aplicación. Con base a esta información, el sistema realiza las validaciones de cada uno de los archivos que son ingresados al sistema en la etapa de validación, verificando que la información contenida en los campos correspondientes del archivo, cumplan con las fechas especificadas. Asimismo, con estas fechas se generan los distintos reportes y archivos de salida del sistema.

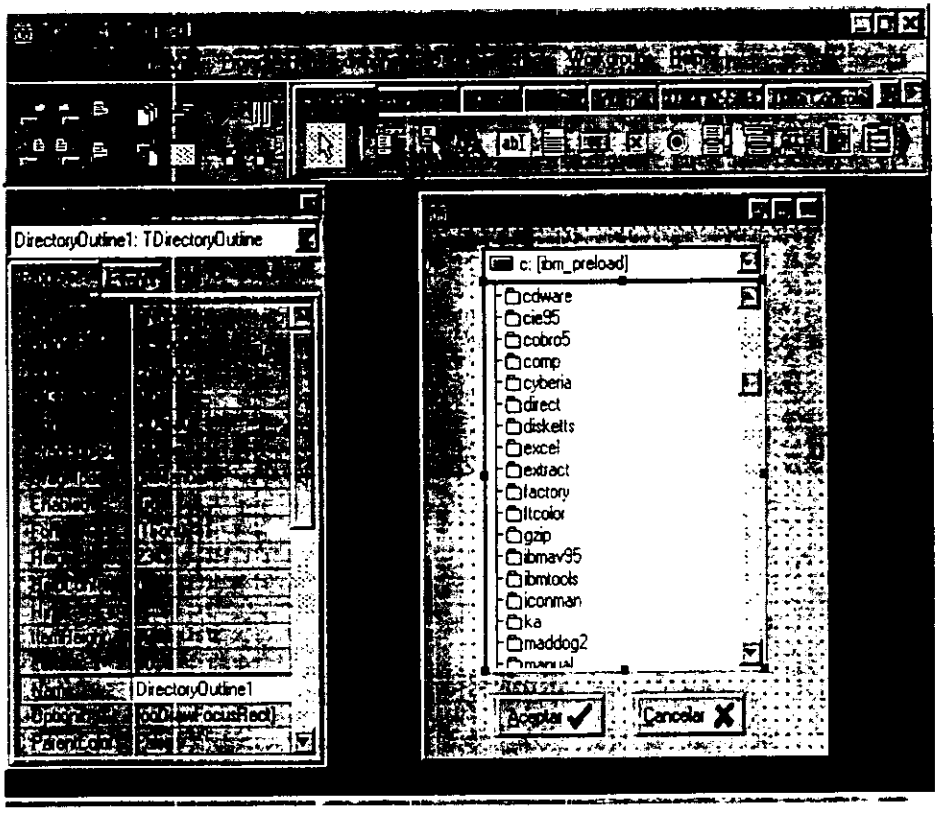


Figura 3.3.26 Forma para respaldar los archivos de entrada.

Una vez que se ha concluido con el proceso de compensación en un día cualquiera, se realiza el respaldo de la información correspondiente a los archivos de entrada al sistema. Por medio de la forma que se muestra en la figura 3.3.26, el usuario puede indicar en que directorio debe respaldarse la información.

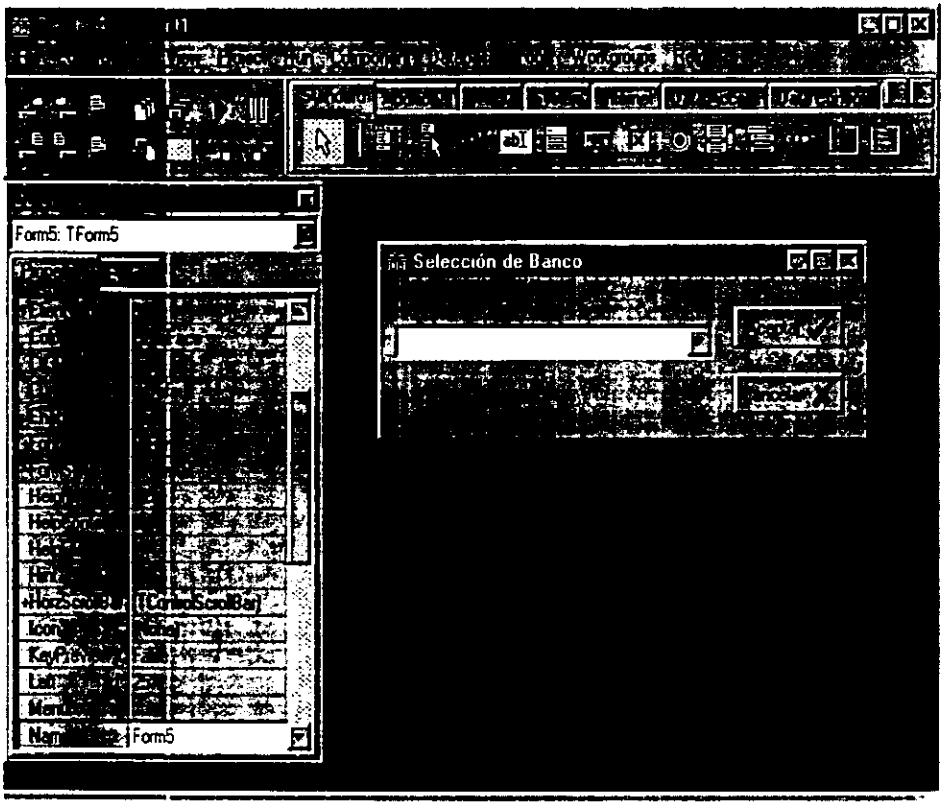


Figura 3.3.27 Forma para seleccionar el banco.

Esta es la forma que nos permite seleccionar un banco. La utilizamos como forma genérica en las diferentes opciones que requieren que el usuario especifique el banco, por ejemplo para generar un archivo de salida o un reporte.

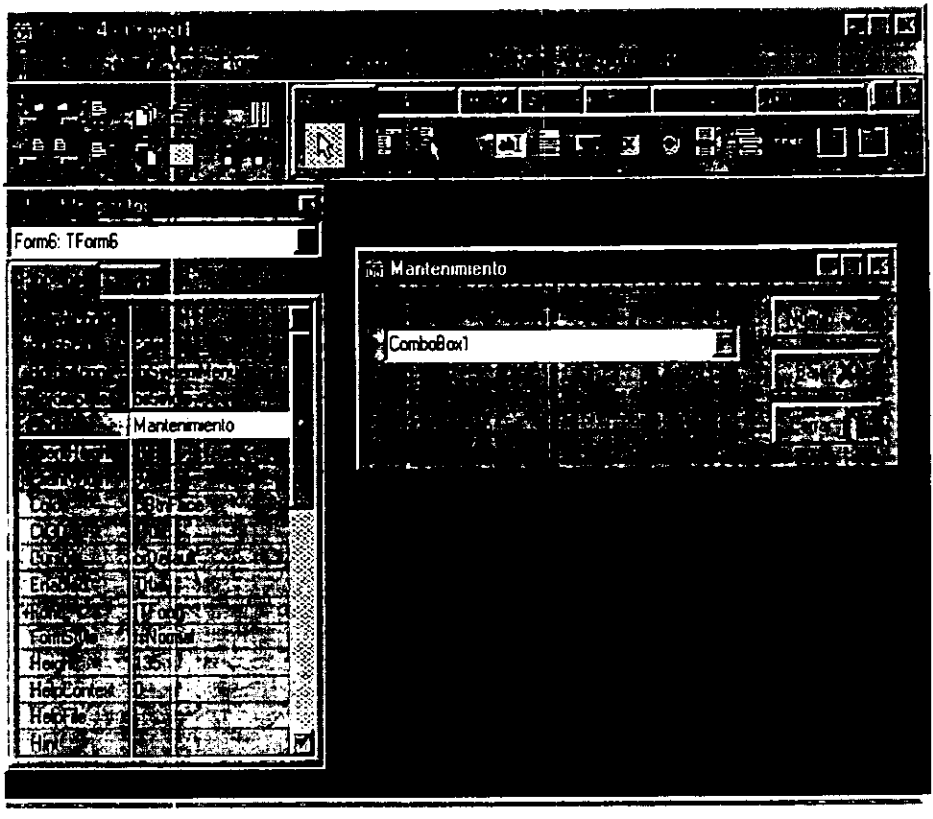


Figura 3.3.28 Forma para el mantenimiento de catálogos.

Esta forma se utiliza para llevar a cabo acciones de mantenimiento del sistema. Se pueden dar de alta o de baja a los bancos, o a los usuarios, según se requiera.

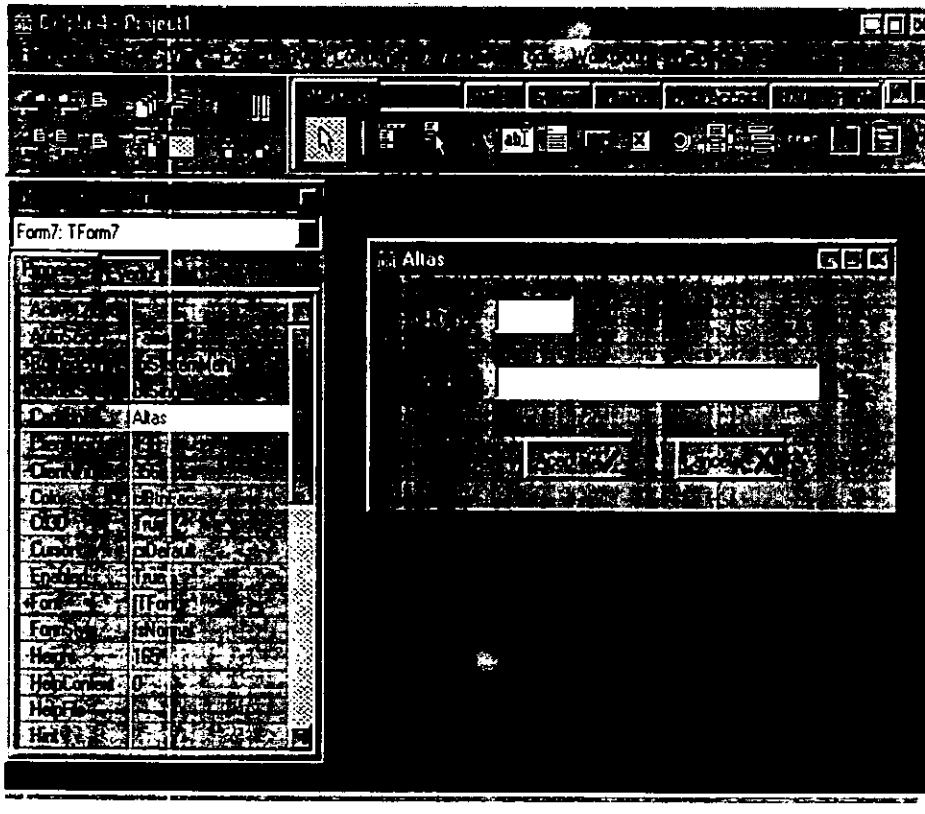


Figura 3.3.29 Forma para dar de alta nuevos bancos.

Cuando se requiere dar de alta un nuevo banco, se utiliza la forma que se muestra en la figura 3.3.29, donde se debe proporcionar el número y el nombre del nuevo banco.

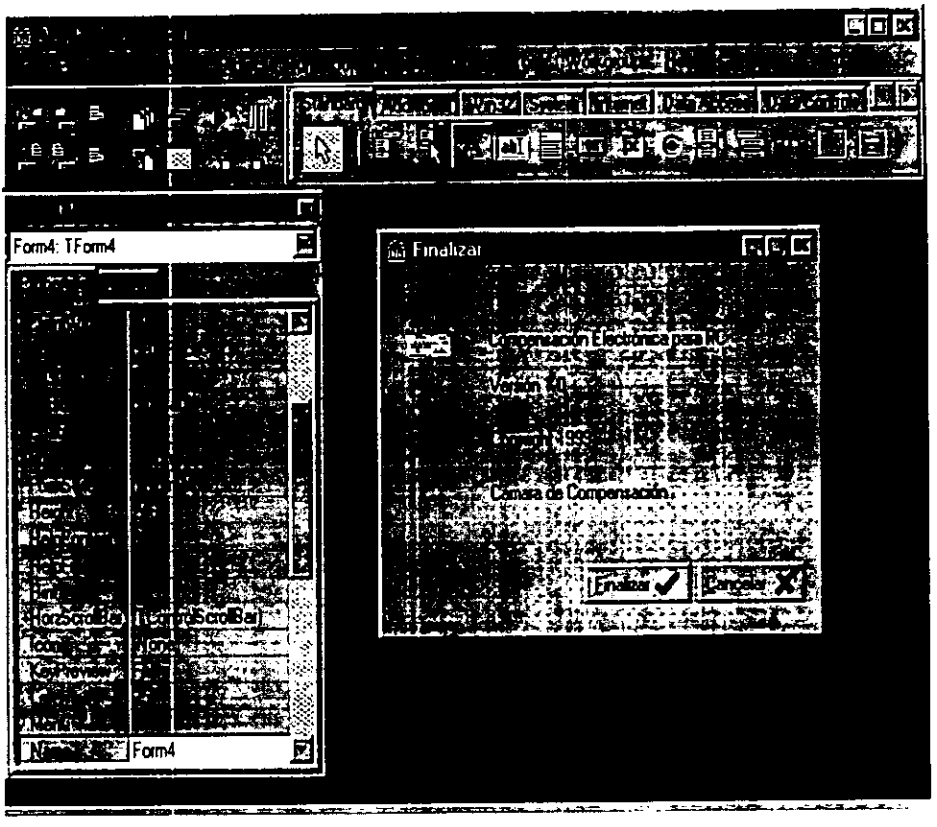


Figura 3.3.30 Forma para finalizar la aplicación.

La forma de la figura 3.3.30 es la que aparece al final de todas las ventanas, cuando se navega por el sistema usando el ícono "siguiente".

Por último, cabe mencionar que a cada una de las formas que se diseñan con el IDE de Delphi, se les asigna un archivo de texto que contiene el código generado de manera automática. A continuación se describe a grandes rasgos. la estructura de un archivo de proyecto generado por Delphi.

El programa principal de Delphi es un archivo de texto ASCII con extensión DPR. Esta extensión quiere decir Delphi project (proyecto de Delphi).

Para cada una de las ventanas que se diseñan en el IDE, Delphi crea una unidad. Una unidad es un archivo individual (también de texto ASCII) que representa en general a un objeto, o a una agrupación lógica de funciones. En el caso de los objetos que son formas, Delphi también crea un archivo ".DFM" (Delphi Form[Forma de Delphi]) para guardar la apariencia del diseño de las mismas.

El siguiente ejemplo muestra el archivo DPR que se crea cuando se inicia Delphi. En general se pueden realizar programas muy complejos sin jamás modificar el archivo de proyecto DPR, pero es importante saber como funciona. El archivo de proyecto Project1.DPR tiene la siguiente apariencia.

```
program Project1;

uses
  Forms,
  Unit1 in 'Unit1.pas' {Form1};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.
```

Analicemos este archivo. La sentencia Program Project1, nos dice que el programa se llama Project1. El nombre del programa debe ser el mismo que el nombre del archivo (sin extensión .DPR).

La sección "uses" nos dice qué archivos son usados. Forms (formas) es la biblioteca de Delphi que contiene los comandos para crear y manipular ventanas, que en Delphi se llaman formas. El otro renglón en la sección "uses" nos dice que la unidad llamada "Unit1" se encuentra en el archivo Unit1.pas, y Delphi nos hace un comentario (entre llaves), que dice que la forma en esta unidad se llama Form1.

$R, es una directiva del compilador. Una directiva es un comando para el compilador, no para el IDE. Cuando el compilador encuentra la directiva {$R *.RES}, un archivo de recursos con el mismo nombre del programa (pero con extensión .RES) es compilado junto con el programa. Un archivo de recursos (.RES) es un archivo con especificaciones para usar y manejar textos en general, iconos y en algunos casos cajas de diálogo o menús.

Una vez que Delphi ha especificado todo lo que va a usar el proyecto, el programa inicializa la aplicación (Application.Initialize), crea la forma Form1 a partir de la definición de objeto Tform1 usando el método CreateForm, y "ejecuta" la aplicación (Application.Run). Cualquier forma creada con "CreateForm" permanece "viva", aunque podría estar invisible, hasta que el método Application.Run termina. Después de esto, como todo programa de D.O.S., la aplicación termina.

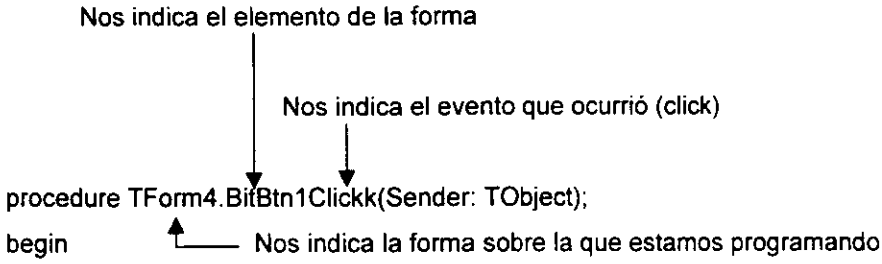
¿Por qué Application.Run? "Run" (correr o ejecutar) en Delphi es un método de la aplicación que hace al programa entrar en el ciclo de mensajes de Windows. El ciclo de mensajes es un ciclo estilo "Do While" que recibe mensajes de Windows (clicks del ratón, tecleos, etc.) y procesa los mensajes en una manera consistente para todas las ventanas de la aplicación. En los viejos dias de la programación para Windows, los programadores, tenían que escribir el ciclo de mensajes ellos mismos. Pero hoy en día, bibliotecas de clases como MFC y la VCL de Delphi manejan la tediosa programación del ciclo de mensajes.

Las unidades (units) que genera Delphi para cada forma contienen las definiciones de los tipos de datos correspondientes a los controles utilizados, además de los procedimientos creados durante el diseño de las formas. El código completo de este proyecto se puede ver en el apéndice D. A continuación, se muestra y analiza el código de la forma 4.

```
unit Unit4;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, StdCtrls, Buttons;

type
  Tform4 = class(TForm)
    BitBtn1: TBitBtn;
    Label1: TLabel;
    Image1: TImage;
    procedure TForm4.BitBtn1Clickk(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form4: Tform4;

implementation
```

```
{$R *.DFM}
procedure TForm4.BitBtn1Clickk(Sender: TObject);
begin
  Form4.Close;
end;

end.
```

Al principio de esta "unit4" (unidad4) tenemos, como en el caso del archivo de proyecto, la cláusula 'Uses' (usar) para hacer uso de ciertas funciones y procedimientos que residen en las unidades incluidas a continuación de dicha cláusula. Estas líneas son creadas automáticamente por Delphi, de acuerdo a los controles que se incluyen en la forma. Véase por ejemplo que se han incluido las unidades Controls (controles), Graphics (Gráficas), y Forms (formas), entre otras. Estas unidades las utiliza Delphi para crear las ventanas y los controles incluidos. A continuación viene la sección Type (tipo), donde se declara un tipo de objeto que corresponde al diseño de la forma 4. Por cada control incluido en la forma se declara un elemento dentro del tipo de objeto Tform4 que a su vez es un objeto. Por ejemplo el objeto Image1 es declarado por Delphi automáticamente como del tipo Timage (Tipo imagen). Recordemos que en esta ventana hemos incluido un botón cuya única finalidad es, que cuando el usuario pulse en él con el ratón, la ventana se cierre. Cuando el usuario hace click con el ratón sobre el botón, Windows detecta esto y genera un mensaje que es enviado a nuestra aplicación. La aplicación interpreta este mensaje y en base a eso se sigue una acción determinada. Este mensaje que se produce se puede asociar con el evento de que el usuario ha realizado una acción, y desde el punto de vista de programación en Delphi, decimos que ocurrió el evento "click en el botón 1". Cuando ocurre un evento en alguno de los controles de la forma, el programador puede asignar código a dicho evento. Para unir la acción del usuario (evento) con lo que se desea que realice el programa en consecuencia, hacemos uso del IDE de Delphi. Hacemos doble click en el botón1 que en este caso fue titulado como Aceptar, y Delphi nos muestra lo siguiente:



Aquí tenemos el encabezado de un procedimiento de Pascal orientado a objetos (object pascal) seguido de las palabras reservadas Begin (inicio) y End (fin) que nos indican donde comienza y termina el procedimiento. Entre estas dos palabras reservadas el programador puede incluir el código necesario, para que se lleven a cabo algunas tareas de programación. En el caso anterior únicamente hemos colocado la instrucción Form4.Close (Form4.Cerrar) con lo que indicamos que cuando ocurra un click del mouse en el botón1, se cierre la forma4.

Podemos ver que la mayor parte del código para interactuar con la interface gráfica es escrito por Delphi, pero las acciones especificas a realizar deben ser escritas por el programador. A veces el código que genera el programador excede con mucho al que es realizado por Delphi, pero de cualquier manera resulta de gran ayuda contar con este tipo de ambiente de desarrollo para Windows, ya que sin esta ayuda la tarea de programar el ambiente gráfico resulta muy tediosa y tardada.

3.4 DISEÑO E IMPLEMENTACIÓN DE RUTINAS DE DIAGNOSTICO Y EVALUACIÓN DE LOS DATOS.

3.4.1 Introducción.

La realización de pruebas de software, en el ciclo de vida del desarrollo de sistemas tiene gran relevancia, debido al costo en la inversión tanto en recursos humanos, económicos y de tiempo. Lo anterior no implica en forma principal, la detección de inconsistencias por mal funcionamiento en el sistema, si no más bien radica, en el objetivo de entregar un producto de calidad, que ahorre costos de tiempo, esfuerzo y dinero, al efectuarse un menor mantenimiento del sistema, al momento de ser implantado.

La fase de pruebas puede llevar mucho tiempo, si es que no se hicieron eficientemente, el análisis, el diseño y la programación, así como si no se realizo un trabajo adecuado en estas fases, puede volverse iterativo. Las primeras pruebas muestran la presencia de errores, y las posteriores verifican si los programas corregidos funcionan correctamente.

En el momento en que se pone un sistema en operación, para que pueda ser utilizado por el cliente, se presentan diferentes dudas:

- ¿El sistema hace lo que se pidió?
- ¿Soportará la carga de trabajo planteada?
- ¿El sistema puede fallar?

Estas son las preguntas más comunes que se plantean al momento de instalar un sistema por primera vez. Es por lo anterior que se realiza el proceso de pruebas, las cuales disminuyen el riesgo de falla al operar el sistema.

La prueba de los programas es la técnica de confirmación del sistema, ésta se debe realizar antes de entregar el sistema al usuario. Las pruebas consisten en ejercitar el programa utilizando datos similares a los datos reales y observar los resultados e interpretar estos para detectar errores o insuficiencias en el sistema. Es importante comprender que las pruebas nunca demuestran que un programa este correcto. Es probable que existan fallas aún después de la prueba más compleja. Un plan de pruebas bien elaborado brindará como resultado la presencia de errores y de esta manera se podrá proceder a su depuración. El plan deberá indicar el nombre de la prueba, los datos de entrada, el objetivo de la prueba y los datos de salida, además de que es recomendable que se lleve a cabo con el personal que no haya participado durante la programación, pero que conozca la operación del negocio que utilizará el sistema.

3.4.1.1 Flujo de información para las pruebas.

En la figura 3.4.1.1.1 se tienen dos entradas: La configuración del software que incluya la especificación de requisitos, la especificación del diseño y el código fuente; la configuración de prueba que incluye un plan y un procedimiento de pruebas y se evalúan los resultados esperados contra los generados. Cuando se descubren errores se inicia un proceso de depuración que tiene como finalidad realizar las correcciones pertinentes.

A medida que se recompilan y se evalúan los resultados de la prueba se puede determinar una medida cualitativa de calidad y fiabilidad del software. Si se encuentran frecuentemente errores que requieran modificaciones en el diseño, la calidad y fiabilidad del software está en entre dicho, de tal forma que es necesario seguir realizando pruebas. Si por el contrario, el funcionamiento del software parece ser el correcto y los errores que se encuentran son menores, se pudiese pensar que calidad y fiabilidad del software son aceptables, o que las pruebas fueron inadecuadas, ya que no permitieron descubrir errores importantes. Así mismo, si durante la prueba no se descubren errores,

quedará la sospecha de que las pruebas no fueron adecuadas y que el software puede estar defectuoso, para lo cual se considera que estos defectos pudiesen ser descubiertos por el usuario y deberán ser corregidos en la fase de mantenimiento o garantía.

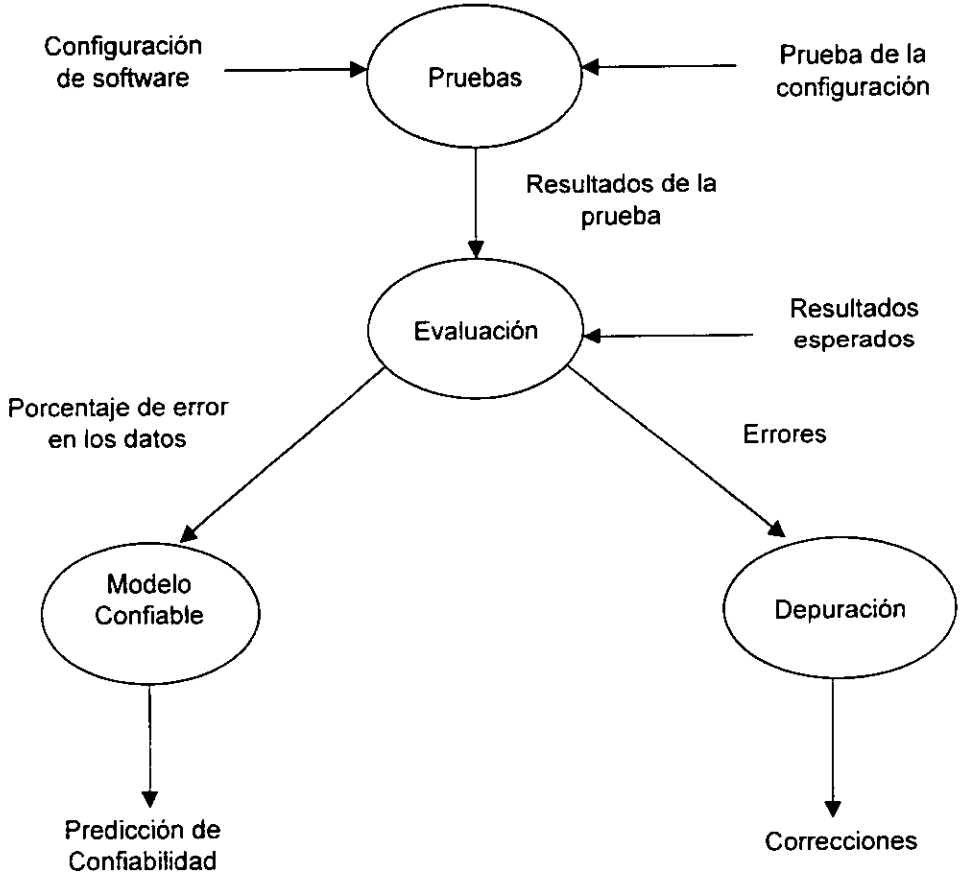


Figura 3.4.1.1.1 Flujo de Información para las pruebas

3.4.2 Tipo de pruebas.

3.4.2.1 Prueba Funcional.

Esta es la forma más común de prueba; su propósito es asegurar que el sistema realiza sus funciones normales de manera correcta. Así. los casos de prueba se desarrollan y se alimentan al sistema. Las salidas son examinadas para ver si son correctas.

En esta sección se comprobaron los resultados obtenidos tanto del sistema anterior como del nuevo sistema, presentandose algunas diferencias, mismas que fueron corregidas, lo que dio como resultado que el nuevo sistema cumpliera con las especificaciones que dieron origen a su desarrollo.

3.4.2.2 Prueba de Recuperación.

El propósito de este tipo de prueba es asegurar que el sistema pueda recuperarse adecuadamente de diversos tipos de fallas. Esto es de particular importancia en los sistemas en línea de grandes dimensiones, así como sistemas que trabajen en tiempo real.

En esta sección se realizaron pruebas de interrupción de energía eléctrica, durante el proceso de compensación comprobandose que este tipo de fallas no afecta directamente al proceso, es decir no hay perdida de información. Sin embargo hay que hacer la aclaración que la Cecoban cuenta con equipos no-break, lo que evitan que se presenten este tipo de problemas.

3.4.2.3 Prueba de Desempeño.

Su propósito es asegurar que el sistema pueda manejar el volumen de datos y transacciones de entrada especificados en el modelo de implantación del usuario, además de asegurar que tenga el tiempo de respuesta requerido.

En esta etapa de pruebas, lo que se busco es ejecutar la aplicación en situaciones extremas, es decir en una jornada normal del proceso se llevan a cabo en promedio 150,000 a 200,000 movimientos diarios. Para la prueba se llevaron a cabo 500,000 movimientos sin que esto representará alguna degradación del funcionamiento del sistema.

3.4.2.4 Prueba Exhaustiva.

Esta prueba consiste en generar casos de prueba que permitan cubrir cada entrada así como cada una de las combinaciones posibles de las situaciones que el sistema pudiera enfrentar, de tal forma que se puede asegurar un comportamiento adecuado.

3.4.3 Técnicas de prueba.

En cuanto a las técnicas para las pruebas del software existen dos enfoques denominados pruebas de caja negra y pruebas de caja blanca.

3.4.3.1 Prueba de Caja Negra.

Esta prueba pretende demostrar que la entrada se acepta en forma adecuada y que se produce una salida correcta, así como la integridad en la información externa se mantiene. La prueba de la caja negra examina algunos aspectos del modelo fundamental del sistema sin tomar mucho en cuenta la estructura lógica del sistema.

Esta prueba pretende determinar:

a) Funciones incorrectas o ausentes.
b) Errores de interface.
c) Errores en la estructura de datos o en el acceso a las bases de datos externas.
d) Errores de rendimiento.
e) Errores de inicio y de terminación.

3.4.3.2 Prueba de Caja Blanca.

Esta prueba se basa en un minucioso examen de los detalles procedurales. Se comprueban los caminos lógicos del software proponiendo casos de prueba que provoquen que se cumplan conjuntos específicos de condiciones examinando el comportamiento del programa en varios puntos para determinar si el resultado real coincide con el esperado. El inconveniente de la prueba es que para algunos casos es complicado probar todos los caminos lógicos. Algunas características que se deben cumplir son las siguientes:

- Garantizar que se ejecuten por lo menos una vez todos los caminos independientemente de cada módulo.
- Ejercitar todas las decisiones lógicas. verdaderas o falsas.
- Ejecutar todos los ciclos en sus límites y con sus límites operacionales.
- Ejecutar las estructuras internas de datos para asegurar su validez.

3.4.4 Estrategias para Pruebas.

3.4.4.1 Prueba de Unidad.

Las partes que conforman la prueba de unidad son las siguientes:

- *Interface*. Se comprueba que la información fluye de forma adecuada hacia y desde la unidad o programa que está siendo probado. Se examinan las estructuras de los

datos locales para asegurar que los datos que se mantienen temporalmente conservan su integridad durante todos los pasos de ejecución. Se prueban las condiciones límite para asegurar que el programa funciona correctamente en los límites establecidos como restricciones del procesamiento. De esta manera se prueban todos los caminos de la estructura de control con el fin de asegurar que todas las instrucciones del programa se ejecuten al menos una vez.

- *Manejo de Errores.* Entre los errores potenciales que se comprueban, pueden ser los siguientes:

  - Descripción ininteligible del error.
  - El error señalado no corresponde con el error encontrado.
  - La condición de error hace que intervenga el sistema antes que el mecanismo de manejo de errores.
  - El procesamiento de la condición excepcional es incorrecto.
  - La condición de error no proporciona suficiente información para ayudar a la localización del error.

- *La prueba de límites.* Verifica los valores máximos y mínimos permitidos, los valores de los datos por debajo y por encima de los máximos y mínimos son apropiados para descubrir estos errores.

3.4.4.2 Pruebas de Integración.

Una vez que se han realizado las pruebas de unidad o de programas por separado, el paso siguiente es ponerlos ahora en interacción, es decir, realizar la prueba de integración. Para esta prueba se deben considerar los siguientes puntos:

- Un programa no debe tener efectos adversos o inadvertidos sobre otros.
- Las funciones cuando se combinan, pueden no producir la función principal deseada.

- Las estructuras de datos globales pueden presentar problemas.
- Los datos se pueden perder en alguna interface.

Para este tipo de pruebas, existen dos formas de atacarlas: En forma ascendente y en forma descendente. El enfoque ascendente empieza por probar módulos individuales pequeños separadamente, luego los módulos individuales se combinan para formar unidades cada vez más grandes que se probarán en conjunto: esto se conoce como pruebas de subsistemas. Finalmente, todos los componentes del sistema se combinan para probarse, esto se conoce como prueba de sistema y suele estar seguido por las pruebas de aceptación, estas se realizan mediante el empleo de datos generados por la organización encargada de construir el sistema. La prueba de aceptación del sistema se efectúa con datos reales, que a menudo descubren errores en la definición de requerimientos del sistema.

El enfoque de prueba descendente empieza con un esqueleto del sistema, es decir la estrategia de prueba supone que se han desarrollado los modelos ejecutivos de alto nivel del sistema, pero que los de bajo nivel existen sólo en los módulos que no procesan nada. La selección de la estrategia de integración depende de las caracteristicas del software y, a veces, del plan del proyecto. En términos generales se puede utilizar un planteamiento combinado.

3.4.4.3 Pruebas de Validación.

Después de la prueba de integración el paso siguiente es la prueba de validación. La prueba de validación es la que se obtiene cuando el software desarrollado funciona de acuerdo a las expectativas razonables del cliente. La validación del software se obtiene aplicando una serie de pruebas que demuestran la conformidad con los requerimientos.

3.4.4.4 Pruebas del Sistema.

Para esta parte se deberán considerar los siguientes puntos:

3.4.4.4.1 Prueba de recuperación.

El propósito de este tipo de prueba es asegurar que el sistema pueda recuperarse adecuadamente de diversos tipos de fallas. Las pruebas de recuperación pueden requerir que el equipo que realiza el proyecto simule o provoque fallas de hardware, de corriente, en el sistema operativo, etc. Este tipo de prueba puede ser manual o automática. Si la recuperación es automática se deberá evaluar que el sistema se reinicie correctamente, si la recuperación requiere de la intervención humana, hay que evaluar los tiempos promedio de recuperación para determinar si esta dentro de los límites aceptables.

3.4.4.4.2 Prueba de seguridad.

Consiste en verificar los mecanismos de protección incorporados al sistema. Estos mecanismos tienen la finalidad de proteger al sistema contra acciones impropias de gente que traten de perjudicar u obtener información en forma ilícita.

3.4.4.4.3 Prueba de resistencia.

En este tipo de prueba, se ejecuta el sistema de forma que demande recursos en cantidades anormales, es decir, diseñar pruebas especiales que generen 10 interrupciones por segundo, pero sólo una o dos son normales, incrementar la frecuencia de datos de entrada en un orden de magnitud con el fin de comprobar como responden las funciones de entrada, ejecutar casos de prueba que puedan dar problemas con el esquema de gestión de memoria y diseñar casos de prueba que produzcan excesivas búsquedas de información.

3.4.4.4.4 Pruebas de rendimiento.

Está diseñada para obtener el rendimiento del sistema en tiempo de ejecución dentro del contexto de un sistema integrado. Las pruebas de rendimiento a menudo se realizan de forma paralela con las pruebas de resistencia. La instrumentación consiste en monitorear los intervalos de ejecución, los sucesos ocurridos y muestran de los estados de la maquina en funcionamiento normal. De esta forma se pueden descubrir situaciones que lleven a degradaciones y posibles fallos del sistema.

3.4.5 Pruebas al sistema.

Las pruebas respecto al sistema representan una revisión final de las especificaciones técnicas, del diseño y de asegurar que se cumplan los requerimientos definidos por los usuarios para su uso y desempeño. Se evaluaron los siguientes puntos para la realización de estas pruebas:

- Evaluación de los filtros de información para verificar que las validaciones establecidas en el sistema sean correctas.
- Evaluación con respecto al comportamiento del flujo de la información; es decir asegurarse que el sistema produzca las salidas apropiadas de acuerdo a diferentes entradas.
- Asegurarse que el sistema cumpla con los requerimientos definidos.
- Validación del flujo de información con respecto al flujo de pantallas para que sea adecuado en el desempeño del sistema.

Se tomaron en cuenta las siguientes preguntas para la realización de las pruebas de interface:

Para las ventanas:
¿El tamaño de las ventanas puede ser ajustado, moverse y desplegarse?

¿Se encuentra accesible la información dentro de la ventana a través del mouse y las teclas de funciones?

¿Se generan las pantallas adecuadamente cuando se sobre-escribe y se vuelve a abrir?

¿Operan todas las funciones relacionadas con la ventana?

¿Están disponibles y desplegados apropiadamente en la ventana todos los menús emergentes, barras de herramientas, barras deslizables, cuadros de diálogos, botones. iconos y otros controles importantes?

¿Cuándo se despliegan varias ventanas se representan adecuadamente el nombre de cada ventana?

¿Está resaltada adecuadamente la ventana activa?

Entrada de datos:

¿Se repiten y son introducidos adecuadamente los datos alfanuméricos en el sistema?

¿Se reconocen los datos no validos?

¿Son intangibles los datos de entrada de datos?

En las pruebas con el usuario se evaluaron los siguientes puntos:

- Funcionalidad de botones para verificar que realicen la operación correcta y presentar la pantalla correspondiente para lo cual fueron diseñados.
- La interface gráfica con el usuario cumplió con los estándares.
- Revisar que no se encuentren errores ortográficos en las pantallas y en los mensajes de error.
- Verificar que los mensajes de error correspondan al suceso.
- Validar que los mensajes de error sean claros.
- Validación de campos de tipo fecha según el formato establecido en los requerimientos.
- Validación de los campos numéricos.

- Verificar que la información sea la correspondiente al evento, como por ejemplo la generación de los archivos de salida para los diferentes bancos.

Finalmente y sobre la base de los conceptos anteriormente descritos al sistema se le efectuaron pruebas de integración y aceptación, las cuales consistieron respectivamente en:

Pruebas de Integración.

En la integración de los módulos del sistema de "Compensación Electrónica de Cheques para PC", se utilizo primordialmente la estrategia ascendente, debido a que cada integrante del equipo de desarrollo probó individualmente cada módulo desarrollado y después se fueron uniendo en los submódulos del sistema hasta lograr la integración final del sistema con sus respectivas pruebas.

Pruebas de Aceptación.

Implico la planeación y ejecución de pruebas funcionales de desempeño y de resistencia para demostrar que el sistema implantado satisface los requisitos que dieron origen al sistema.

En las pruebas de aceptación para el sistema "Compensación Electrónica de Cheques para PC", se efectuaron dos conjuntos de pruebas: aquellas desarrolladas por el propio equipo de desarrollo del sistema y las que realizaron los usuarios. En ambos casos se incorporaron pruebas de unidad y de aceptación así como pruebas especificas de los controles y resultados de salida del sistema. Se realizaron pruebas de cada uno de los módulos, en cada una de sus diferentes opciones.

Con los resultados de las pruebas desarrolladas por los usuarios se pudo determinar que el sistema cumple con los requisitos que ellos mismos establecieron en la fase de análisis y con los resultados de las pruebas desarrolladas por el equipo de trabajo se

determino que la integración de todos los módulos que forman al sistema, es satisfactoria.

Las pruebas constaron de diferentes ejemplos de cada una de las operaciones que realiza el sistema, dentro de ellos se tenian casos críticos para evaluar el comportamiento del sistema. Ademas se ejecutaron todos los procesos del sistema obteniendo en todas las pruebas un resultado correcto, con lo cual los usuarios mostraron gran confianza en el sistema.

3.4.5 Aprobación del usuario.

Para la aprobación del sistema, es necesario considerar los objetivos de la empresa que lo solicito, sus recursos y sus funciones básicas, áreas de responsabilidad, departamentos o divisiones.

En base a la determinación de requerimientos, podemos analizar el modelo propuesto para obtener la aprobación del usuario. El objetivo es aprobar formalmente cada fase del ciclo de vida de un proyecto.

3.4.5.1 Caracteristicas.

Es un documento requerido, preparado por el líder del proyecto y debe haber un documento de aprobación por cada fase del proyecto. Contiene los siguientes puntos:

Referencia	Descripción
(I)	Nombre de la persona que firmará la aprobación, así como su puesto o área que representa.
(II)	Nombre del líder del Proyecto o en su defecto la persona responsable de la aprobación que se solicita.
(III)	Nombre de la fase del proyecto que se va a probar.
(IV)	Fecha en la cual se elabora el documento de aprobación.
(V)	Cuerpo del documento. En esta parte se redacta la solicitud de aprobación con los pormenores deseados.
(VI)	Fecha máxima en la cual se espera obtener la aprobación para continuar con el proyecto.
(VII)	Nombre y firma del Líder del Proyecto que solicita la aprobación.
(VIII)	En esta parte se indicará si la solicitud se aprueba o rechaza. En último caso se deberán indicar las causas en que se funda.
(IX)	En este espacio podrá utilizarlo el que autorizo la solicitud para hacer comentarios o sugerencias respecto de la solicitud.
(X)	Fecha en que se tiene planeado iniciar la siguiente fase. Esta dependerá del resultado de la aprobación, si esta fue rechazada, posiblemente se tendrá la necesidad de replanear.
(XI)	Nombre(s) y Firma(s) de la(s) persona(s) que autoriza(n) la solicitud, así como su puesto o área que los faculta para dar autorización.

Dadas las caracteristicas anteriores, presentamos un ejemplo de los formatos de solicitud de aprobación de los usuarios:

Formato : **Aprobación del menú principal del sistema.**

A : Cámara de Compensación Bancaria. (I)
De : ---------- (II)
Fase : Pruebas de aceptación (III)
Fecha : 14 de junio de 1999 (IV)

Solicitud de Aprobación. (V)

Se solicita el visto bueno (Vo.Bo.) del Menú Principal de la aplicación y poder continuar con las demás etapas del proyecto.

Este menú principal consta de lo siguiente.

Opción	Descripción
Archivo	Dada las características de la aplicación, esta opción sirve para salir de la aplicación.
Inciar	Inicializa los archivos del sistema para el proceso de compensación.
Validación	Valida la ubicación de los archivos a procesar.
Reportes	Generación de reportes y procesos del sistema
Salidas	Generación de los archivos de salida para los bancos respectivos.
Saldos	Generación del archivo de liquidación de los resultados de la compensación, que se enviará al Banco de México.
Respaldos	Respaldo de los archivos del proceso para futuras aclaraciones.
Utilerías	Herramientas que apoyan el proceso de compensación.
Mantenimiento	Permita la actualización del catalogo de bancos y modificación de los códigos de transacción.
Ayuda	Ayuda para el operador del sistema.

El periodo acordado para efectuar las pruebas es del _____ al _____, el cual concluyó exitosamente el día _____

De acuerdo a lo convenido, esperamos su respuesta a más tardar el día: _____ (VI)

Atentamente

_____
Líder de Proyecto (VII)

Informe de Aprobación. (VIII)

[ ] Se aprueba
[ ] Se rechaza, por las siguientes razones:

1.

2.

Comentarios. (IX)

Fecha de inicio de la siguiente fase _____ (X)

Firma de conformidad.

_____
CECOBAN (XI)

3.4.6 Mantenimiento del Sistema.

El mantenimiento es una actividad constante que conservará al Sistema Compensación Electrónica de Cheques, en los niveles máximos de eficiencia sin rebasar las limitaciones de los costos. Tiene por objeto reducir los errores provenientes del diseño, los ocasionados por los cambios ambientales y mejorar los servicios y el alcance del sistema. Estas actividades se clasifican en:

- Correctivo.
- Preventivo
- Perfectivo
- Adaptativo (aumentativo y tecnológico).

3.4.6.1 Correctivo.

La primera actividad del mantenimiento se da ocasionalmente cuando la prueba del software no haya descubierto todos los errores latentes de un sistema. Durante su uso del sistema se encontrarán errores, los cuales deben ser informados al equipo de desarrollo. El proceso que incluye él diagnostico y corrección de uno o más errores se denomina mantenimiento correctivo.

3.4.6.2 Preventivo.

En este tipo de mantenimiento es para prevenir errores. Se da cuando cambia el software para mejorar la funcionalidad de la aplicación. También se puede considerar el mantenimiento a la información que se maneja para actualizarla y los resultados dados por el sistema sean los correctos.

3.4.6.3 Perfectivo.

Esta actividad de mantenimiento se da cuando un paquete de software tiene éxito. A que medida que se usa el software, se reciben de los usuarios recomendaciones sobre nuevas posibilidades acerca de modificaciones a funciones ya existentes. Para satisfacer estas peticiones se lleva a cabo el mantenimiento perfectivo.

3.4.6.4 Adaptativo.

En este tipo de mantenimiento se encuentran implícitos al aumentativo y el tecnológico.

La vida útil estimada del software de aplicación puede fácilmente sobrepasar los diez años. pero considerando la evolución del ambiente. en la practica éste puede volverse obsoleto. Por lo tanto, el mantenimiento adaptativo es una actividad que modifica al software para que las interacciones sean adecuadas con su entorno cambiante.

El mantenimiento adaptativo se debe a cambios en el ambiente del programa y a la adaptación de nuevas unidades o módulos.

Un estudio realizado por Lientz y Swanson (1980), establecieron que alrededor del 65% del mantenimiento era perfectivo, el 18% adaptativo y el 17% correctivo.

3.4.6.4.1 Aumentativo.

Este tipo de mantenimiento se da cuando se incluyen nuevas funciones que no se contemplaron al inicio del desarrollo del sistema y surgen como una necesidad del usuario.

3.4.6.4.2 Tecnológico.

Esta actividad que contribuye al mantenimiento, se da debido a todo cambio importante en la informática. Si en un ciclo de 36 meses surgen nuevas generaciones de hardware, regularmente aparecen nuevos sistemas operativos o nuevas versiones de los antiguos; y frecuentemente se mejoran o modifican los equipos periféricos y otros elementos del sistema.

3.5 PRUEBAS, FACTIBILIDAD TECNICA Y OPERATIVA.

3.5.1 Factibilidad Técnica.

3.5.1.1 Antecedentes.

Como se describió en el capitulo 2, este sistema sustituye al que se procesaba en un equipo de cómputo "mainframe" de la marca Honeywell, el cual por sus costos de mantenimiento y de operación, resultaba no redituable. Por otro lado la dependencia que se tenia con el proveedor dadas las continuas fallas de dicho equipo, ocasionaba que al área responsable de llevar a cabo el proceso de compensación bancaria tuviese que invertir horas extras de trabajo para obtener los resultados del proceso.

3.5.1.2 Hardware.

El sistema para su funcionamiento requiere como mínimo un equipo de cómputo con procesador Pentium II a 166 Mhz, 16 Mb en RAM, 2 Gb en disco, básicamente, sin embargo y tomando en cuenta el avance tecnológico en la materia, un equipo de mayores caracteristicas garantizaria la funcionalidad de la aplicación por lo menos durante tres o cuatro años. La Cámara de Compensación Bancaria cuenta con este tipo de equipos, por lo que no implico un costo adicional por la implementación de esta solución.

3.5.1.3 Herramienta para el desarrollo de la solución.

Considerando la experiencia del personal del área de informática en la Cámara de Compensación Bancaria, respecto a los productos Turbo Pascal Windows y Delphi, así como se cuenta con las licencias correspondientes, lo que represento un ahorro económico ya que no se tuvo que invertir en los siguientes rubros: capacitación y compra de las licencias de dichos productos.

3.5.1.4 Sistema Operativo.

Considerando que ésta Empresa cuenta con las facilidades de utilizar Windows 95 o Windows NT, y dadas las características funcionales que presenta Windows NT, se opto por utilizar este sistema operativo en el equipo donde será ejecutada la aplicación motivo de este proyecto.

3.5.1.5 Costos estimados del proyecto.

Se debe tener en cuenta que la estimación de costos de un producto de programación es una tarea muy complicada puesto que es difícil hacer estimaciones exactas durante las fases de planeación y de análisis de un desarrollo, debido a la gran cantidad de factores desconocidos en ese momento, por lo que se considera un factor muy importante que contribuye a los retrasos de entrega y sobregiro en presupuestos tan comunes en los proyectos de programación.

3.5.1.5.1 Factores en el costo del software.

Existen muchos factores que influyen en el costo de un producto de programación, pero los principales son:

- Capacidad del programador o programadores.
- Complejidad del producto.
- Tamaño del programa y tiempo disponible.
- Confiabilidad requerida.
- Nivel tecnológico.

Capacidad del programador. La producción y mantenimiento de productos de programación son tareas laboriosas por lo que la productividad y la calidad son

funciones directas de la capacidad y esfuerzo individual. Existen dos aspectos fundamentales y relacionados con la capacidad del programador: la competencia global del individuo y su familiaridad con el área particular de la aplicación.

Se entiende como competencia global, a la capacidad para escribir programas de computadora correctos y considerar que las variaciones en la productividad de la programación es un factor significativo para la estimación de costos. Y se cree que en proyectos muy grandes las diferencias individuales tienden a compensarse, pero en proyectos de cinco programadores o menos la diferencia puede ser muy importante.

Para el aspecto de familiaridad con el área de aplicación, se tiene una gran ventaja, ya que el personal involucrado en el proyecto esta relacionado con el tema, lo que facilito el desarrollo de la aplicación.

Complejidad del producto. Existen tres categorías para los productos de programación: programas de aplicación, en los que se incluyen procesamientos de datos y programas científicos; programas de apoyo, como compiladores, ligadores y sistemas de inventarios; y por último programas de sistema, como sistemas de bases de datos, sistemas operativos y sistemas de tiempo real.

Por lo que podemos clasificar el proyecto dentro de la categoría de programas de aplicación y considerando que el nivel de complejidad, en lo que respecta al desarrollo general, no es grande, se debieron de implementar controles del proceso, dada la importancia que reviste la aplicación.

Tamaño del producto y tiempo disponible. Un proyecto grande de programación es generalmente más caro que uno pequeño, sin embargo y dada la relevancia de esta aplicación, se puede considerar como un proyecto de gran envergadura.

Generalmente la estimación del tiempo de desarrollo de un programa se basa tanto en las líneas de código fuente y el esfuerzo total en meses del programador, pero uno de

los errores comunes en la estimación de lineas de código fuente de un producto de programación es subestimar la cantidad de código de servicio requerido; este es la parte del código fuente que permite el manejo de entradas y salidas, comunicación interactiva con el usuario, manejo de errores. etc. En base a proyectos anteriores se puede mencionar que algunas veces el código de servicio llega a sèr más del 50% e incluso hasta el 90% del código total del producto.

Nivel de confiabilidad requerido. La confiabilidad de un producto de programación puede definirse como la probabilidad de que un programa desempeñe una función requerida bajo ciertas condiciones especificas y durante cierto tiempo. La confiabilidad puede expresarse en términos de exactitud. firmeza, cobertura y consistencia del código fuente. Las caracteristicas de la confiabilidad pueden instrumentarse en un producto de programación, pero existe un costo asociado con el aumento del nivel de análisis, diseño, instrumentación y esfuerzo de verificación y validación que debe aportarse para asegurar alta confiabilidad.

El nivel de confiabilidad deseado debe establecerse durante la fase de planeación al considerar el costo de las fallas del programa: en algunos casos, las fallas pueden causar al usuario pequeñas inconveniencias. mientras que en otros tipos de productos puede generarse una pérdida financiera.

Teniendo en cuenta los puntos anteriormente descritos. es obvio que el proceso de compensación electrónica de cheques, debe tener un grado de confiabilidad muy grande, ya que como se menciono previamente la importancia de la aplicación es sumamente grande y relevante.

Nivel tecnológico. El nivel de tecnología empleado en un proyecto de programación se refleja en el lenguaje utilizado, la maquina abstracta (tanto el equipo como los programas de apoyo), las practicas y las herramientas de programación utilizadas. Se sabe que el número de líneas de código fuente escritas por día es, por completo, independientemente del lenguaje utilizado. y que las proposiciones escritas en un

lenguaje gráfico como lo son Borland Pascal Windows y Delphi, suelen generar varias instrucciones a nivel máquina. Los lenguajes modernos de programación brindan características adicionales para mejorar la productividad y confiabilidad del producto de programación; entre estas características están, la verificación de tipos de datos, la abstracción de datos, la compilación separada, el manejo de excepciones y de interrupciones, así como los mecanismos de concurrencia.

3.5.1.5.2 Técnicas de Estimación de Costos del Software.

La estimación de costos puede llevarse a cabo en forma *jerárquica hacia abajo* o en forma *jerárquica hacia arriba*, estas dos formas se describen a continuación:

La estimación jerárquica hacia abajo se enfoca primero a los costos del nivel del sistema, así como los costos de manejo de configuración, del control de calidad, de la integración del sistema, del entrenamiento y de las publicaciones de la documentación. Los costos del personal relacionado se estiman mediante el examen del costo de proyectos anteriores que resulten similares.

En la estimación jerárquica hacia arriba, primero se estima el costo del desarrollo de cada módulo o subsistema; tales costos se integran para obtener un costo total. Esta técnica tiene la ventaja de enfocarse directamente a los costos del sistema, pero se corre el riesgo de despreciar diversos factores técnicos relacionados con algunos módulos que se desarrollarán. La técnica subraya los costos asociados con el desarrollo independiente de cada módulo o componente individual del sistema, aunque puede fallar al no considerar los costos del manejo de la configuración o del control de calidad.

En la práctica, ambas técnicas deben desarrollarse y compararse para que iterativamente se eliminen las diferencias obtenidas.

En la actualidad se tienen técnicas como el llamado Juicio Experto y la Técnica DELFI. las cuales consisten en:

3.5.1.5.3 Juicio Experto.

La técnica más utilizada para la estimación de costos es el uso del juicio experto, que además es una técnica de tipo jerárquica hacia abajo. El juicio experto se basa en la experiencia, en el conocimiento anterior y en el sentido comercial de uno o más individuos dentro de la organización.

La mayor ventaja del juicio experto, que es la experiencia, puede llegar a ser su debilidad; al experto puede confiarse de que el proyecto sea similar al anterior; pero bien puede suceder que haya olvidado algunos factores que ocasionan que el sistema nuevo sea significativamente diferente; o quizás, el experto que realiza la estimación no tenga la experiencia en ese tipo de proyecto; por lo que generalmente se forman grupos de personas para que entre todos se establezca la estimación.

La mayor desventaja de la estimación en grupo es el efecto que la dinámica interpersonal del grupo pueda tener en cada uno de los individuos; los miembros de un grupo pueden ser inocentes respecto a factores de tipo político, a la presencia de alguna autoridad dentro del grupo, o al dominio de un miembro del grupo con una fuerte personalidad.

3.5.1.5.4 Estimación del Costo por la Técnica DELFI.

La técnica DELFI puede adaptarse a la estimación de costos de la siguiente manera:

1. Un coordinador proporciona a cada experto la documentación con la definición del sistema y una papeleta para que escriba su estimación.

2. Cada experto estudia la definición y determina su estimación en forma anónima: los expertos pueden consultar con el coordinador, pero no entre ellos.
3. El coordinador prepara y distribuye un resumen de las estimaciones efectuadas, incluyendo cualquier razonamiento extraño efectuado por alguno de los expertos.
4. Los expertos realizan una segunda ronda de estimaciones, otra vez anónimamente, utilizando los resultados de la estimación anterior. En los casos que una estimación difiera mucho de las demás, se podrá solicitar que también en forma anónima el experto justifique su estimación.
5. El proceso se repite tantas veces como se juzgue necesario, impidiendo una discusión grupal durante el proceso.

Es posible que después de varias rondas de estimaciones no se llegue a un consenso; en ese caso, el coordinador deberá analizar los aspectos relacionados con dada experto para determinar las causas de tales diferentes. Puede ser que el coordinador tenga que recabar información adicional y presentársela a los expertos con fin de resolver las diferencias en los puntos de vista.

3.5.2 Factibilidad Operativa.

3.5.2.1 Introducción.

Para determinar si el sistema será benéfico para la empresa, se tiene que determinar los requerimientos operativos de esta, por lo que hay que planear algunos pasos a seguir para asegurar el logro de los objetivos planteados, destacando los siguientes:

- Verificación cuidadosa de la calidad y la integración (no tener omisiones).
- Identificación de todos los elementos claves para las etapas de diseño y transición.
- Exactitud en la información de volúmenes de funciones y datos.

- Control del equipo para mantener un trabajo detallado, sin olvidar la planeación de tiempo establecido.

- Fuentes de información (datos a reunir, existentes en sistemas manuales y en sistemas automatizados).

- Estado de los datos, su preparación y conversación.

- Métodos de extracción de datos, su complejidad y esfuerzo necesario.

- Contribución del grupo de desarrollo.

- Recursos, tareas y calendarios.

3.5.2.2 Beneficios para la empresa.

Dentro de los beneficios que obtendrá la Cámara de Compensación Bancaria con este sistema, se encuentran:

Menores costos por concepto de operación y mantenimiento del equipo de cómputo en donde se ejecuta la aplicación.

Mayor certidumbre en el proceso de compensación electrónica de cheques, al ser este sistema confiable y presentar mayores características operativas que su antecesor.

Para el usuario final, la operación del sistema será amigable y bajo un ambiente gráfico.

El mantenimiento del sistema no representará para ésta Empresa, altos costos, debido a que parte del personal que desarrollo este proyecto labora en dicha empresa.

Se eliminara la dependencia que se tenia con el proveedor del equipo en donde se procesaba el anterior sistema.

MANUAL TÉCNICO Y DE USUARIO

MANUAL TÉCNICO

El sistema de compensación electrónica de cheques funciona de una manera tal que las posibles contingencias se reducen al mínimo. Es importante mencionar aquí que este sistema no opera basándose en información de días pasados, sino que se limita a procesar la información correspondiente a un día en particular.

El sistema cuenta además con varios controles que reducen la posibilidad de contingencias, si existe alguna inconsistencia en los resultados producidos por el sistema, la persona que realiza las funciones de control puede darse cuenta del error al observar los reportes que produce el sistema de compensación electrónica para este fin. En los reportes de control se notaría una diferencia de cifras lo cual indica que algo ha salido mal o que existe alguna inconsistencia en la información.

En ocasiones el problema se debe a que algún banco presenta incorrectamente su información, en este caso es posible eliminar del sistema estos datos incorrectos, mediante el uso de la opción "Borrar Transferencia" de la ventana "Utilerías".

A veces puede suceder que exista inseguridad por parte del operador del sistema, en cuanto a los resultados producidos. Supongamos que la PC llegó a un punto en que no respondía y se tuvo que apagar cuando se estaban generando los resultados de la compensación, supongamos también que en este caso los reportes de control muestran una inconsistencia en cuanto a las cifras que se esperaba obtener como resultado. En esta situación el problema se puede solucionar usando otra de las opciones de la ventana "Utilerías" llamada "Reproceso". Con esta opción, el sistema vuelve a ingresar automáticamente toda la información de entrada que se tenía hasta el momento de la contingencia y a partir de ella obtiene los resultados de compensación partiendo de cero. Con la acción anterior se pueden eliminar cualquier tipo de inconsistencias en los resultados de la compensación, además de que la recuperación del proceso se lleva a cabo en un tiempo razonablemente corto, no excediendo nunca

más allá de los quince minutos en las plazas de alto volumen (más de 200,000 cheques) y no más de 5 minutos en las plazas con volúmenes de información pequeños (hasta 80,000 cheques). Cabe mencionar aquí, que en el sistema anterior, en caso de requerirse un reproceso de la información, la pura acción de inicializar el sistema consumía de 20 a 25 minutos, más el tiempo que se llevara el validar los archivos de entrada y obtener los resultados de compensación, que en conjunto sumarían alrededor de 60 minutos o más.

En este sistema debido a su diseño y a los requisitos que debe cumplir para su operación, no se permite modificar archivos de datos o parámetros mientras se está en la producción real. El sistema debe garantizar por sí mismo que producirá los resultados esperados en todas las ocasiones sin necesidad de mantenimiento y sin necesidad de soporte en caso de contingencias.

El mantenimiento se reduce a actualizar los catálogos de los bancos participantes en el proceso de compensación electrónica, los datos que identifican a la plaza (esto se hace sólo una vez, después de instalarse el sistema) y otros datos sobre la manera en que el sistema ha de producir los resultados para los usuarios finales (bancos). No es necesario indexar tablas ni dar mantenimiento a bases de datos. Asimismo, dado que no tenemos una base de datos con la información acumulada de días anteriores, no necesitamos hacer respaldos de archivos o bases de datos. Sin embargo, sí se realiza diariamente el respaldo de los archivos de entrada al sistema, pero esto es parte del proceso de operación cotidiano y no entraría dentro del manual técnico sino que se encuentra especificado en el manual de usuario.

La peor situación en la que nos podríamos encontrar en caso de alguna contingencia sería la de tener que volver a instalar el sistema y configurarlo nuevamente, sin embargo esta situación no se presenta, porque en cada sucursal que se instala este sistema, la instalación se lleva a cabo en dos PCs distintas, una donde se realiza el proceso en condiciones normales (llamada PC de producción) y en otra que se utilizaría

en caso de alguna contingencia. Es decir, se cuenta con un sistema completo de respaldo que incluye el hardware y el software necesario para llevar a cabo la compensación en la PC destinada para este fin. En este caso, el usuario no tendría que llevarse los datos ya procesados de una PC a la otra, sino que tendría que validar nuevamente la información de entrada en la otra PC y obtendría los mismos resultados que en la PC de producción cuando esta funciona en condiciones normales. Para esto, al instalar el sistema en alguna sucursal se hace una instalación idéntica del sistema en las dos máquinas, siendo iguales cada catálogo y opción configurable del sistema en las dos PCs.

Como podemos ver, el funcionamiento de este sistema se concentra en producir los resultados diarios de la compensación de una manera ágil y confiable y no se concentra en la acumulación de los datos de compensación obtenidos en días anteriores. Lo anterior con el fin de garantizar una operación con el mínimo de contingencias y de riesgos. Sin embargo, este sistema sí produce algunos resultados que son la entrada a otros sistemas que se preocupan de la acumulación de los datos, como lo son los sistemas de estadísticas y los de facturación. Además produce resultados que son microfilmados para su uso posterior al día del proceso, y en caso de requerirse información de un día anterior se puede realizar un reproceso, fuera de las horas de producción o en una PC de pruebas, con la información del día requerido. El reproceso se puede hacer a partir de los respaldos de archivos de entrada, producidos por el mismo sistema de compensación electrónica.

De esta manera se puede concluir en esta sección, que este sistema ofrece una mayor capacidad y flexibilidad en los casos de contingencia que el sistema que existe actualmente en operación. Lleva a cabo esta función de una manera más fácil, rápida y con menor costo.

Hagamos un resumen de todo lo anterior y pongámoslo en forma de un procedimiento de operación.

1. Antes de la operación normal del día actualizar los catálogos de bancos según se requiera.
2. Siempre se debe inicializar el sistema antes de empezar el proceso del día.
3. Si un banco presenta alguna transferencia incorrecta y se desea eliminar del sistema use la opción "Borrar Transferencia" de la ventana "Utilerías".
4. Si hubo alguna inconsistencia en los resultados de la compensación a causa de apagar el equipo, use la opción "Reproceso" de la ventana "Utilerías".
5. Si persisten los problemas del punto 4, inicialice el sistema de la PC de respaldo y valide ahí la información de entrada.
6. Si los puntos 4 y 5 no solucionaron el problema, tome su disco de instalación original e instale el sistema nuevamente. A continuación procese la información de entrada de manera normal.

Con los puntos anteriores quedaría solucionado cualquiera de los problemas normales que se pudieran presentar durante la operación normal del sistema.

MANUAL DE USUARIO

I. Introducción

En cada plaza, el usuario es responsable de proporcionar los servicios de compensación referidos en este manual, podrá procesar los archivos con operaciones que presenta un banco a las demás instituciones bancarias participantes.

II. Funcionamiento

En el icono llamado COMPENSACIÓN ELECTRONICA mismo que se encuentra en el escritorio de Windows, haga doble click para iniciar la operación del sistema.

Una vez realizada la acción anterior, el sistema mostrará una ventana como la que se ilustra en la figura 2.1 a continuación:

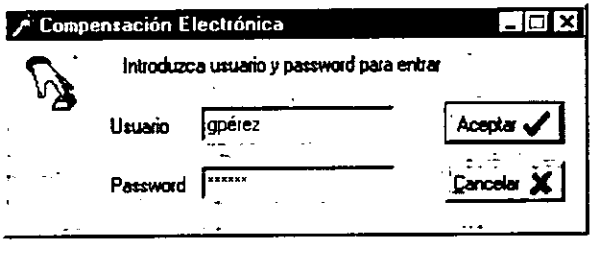


Figura 2.1 Ventana de clave de acceso.

La ventana de la figura anterior permitirá introducir una clave de usuario y una contraseña para poder accesar a la pantalla principal del sistema. Para accesar se deben realizar los siguiente pasos:

Seleccione el recuadro que corresponde al usuario y teclee la clave de usuario.

Seleccione ahora la casilla correspondiente a la contraseña y escriba la clave.

Para continuar seleccione con el ratón el botón de "Aceptar": en caso contrario seleccione "Cancelar".

Si la contraseña y la clave del usuario son correctos, el sistema desplegará la pantalla siguiente (Ver figura 2.2)

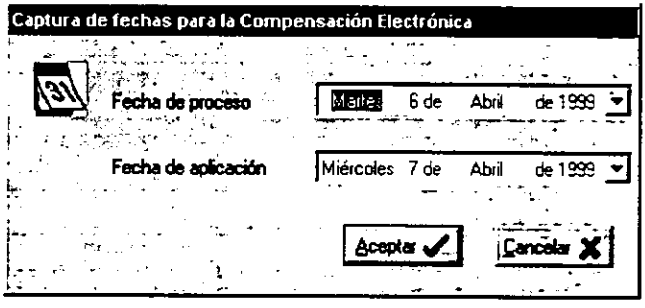


Figura 2.2 Captura de fechas.

Al efectuar lo anterior, el programa solicita las fechas del proceso de compensación y de la aplicación respectivamente, tal como se muestra en la figura anterior (se despliega por omisión la fecha del sistema para el proceso, que debe ser la fecha del día), y la fecha de aplicación, misma que es el día hábil bancario siguiente a la fecha de compensación (cabe mencionar que el sistema pedirá forzosamente estas fechas).

Si no se desea cambiar algún dato de la fecha bastará con oprimir la opción "Aceptar".

Si desea salir del sistema en este momento, debe elegir la opción "Cancelar" para terminar la ejecución del programa.

i) Inicialización

Posteriormente, aparecerá la ventana que se muestra en la figura 2.3 para inicializar el sistema.

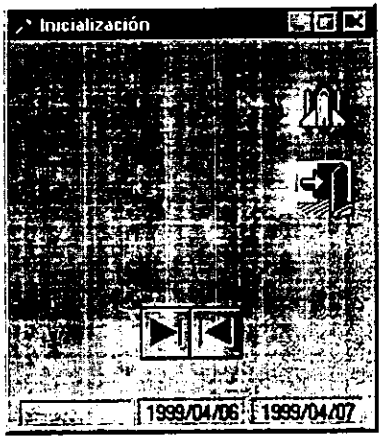


Figura 2.3 Inicialización del Sistema.

Para llevar a cabo la inicialización del sistema, bastará con seleccionar el icono de inicializar mediante el ratón.

Esta opción inicializa los archivos del sistema para comenzar el proceso correspondiente al día. Se debe utilizar esta opción, al iniciar el proceso.

ii) Validación

Una vez ejecutada la acción anterior se despliega la ventana siguiente (Ver figura 2.4), que es la ventana correspondiente para la validación de los diversos archivos de entrada de cada uno de los bancos que presentan información.

Es necesario que el usuario indique la fuente (unidad, dispositivo o ruta) que contiene el archivo que desea validar; para esto, el programa muestra tres opciones, mismas que se explican a continuación

Unidad de disquete A:

Utilice esta opción en caso de que el archivo se encuentre en la unidad A, como se muestra en la figura 2.4.

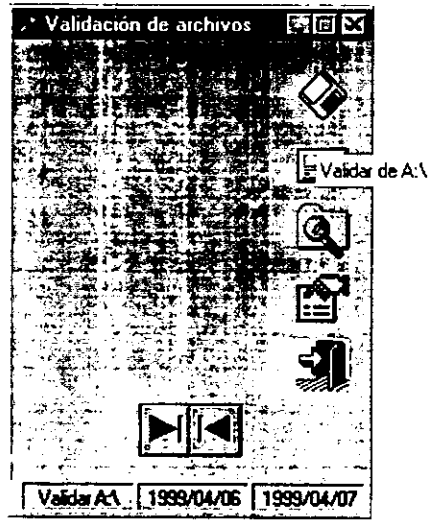


Figura 2.4 Validación de archivos de la Unidad A:\

Validar desde C:\STI\ARCHIVOS

En caso de que el archivo sea transmitido por modem, este se encontrará en el directorio C:\STI\ARCHIVOS, con esta opción el usuario podrá seleccionar el archivo deseado (ver figura 2.5).

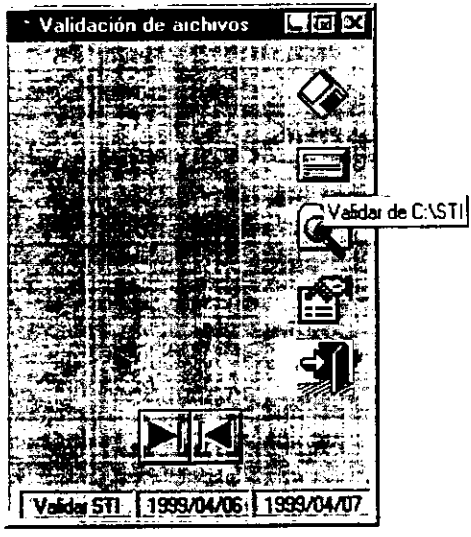


Figura 2.5 Validación de archivos de disco duro.

Explorar

En caso de que el archivo se encuentre en el disco duro (unidad C), con esta opción el usuario podrá indicar la ruta o trayectoria "path" (ver figura 2.5).

Unidad de Cinta

Esta opción se pueden utilizar, en caso de que el archivo sea entregado en una cinta magnética. (Ver Figura 2.6); por tanto antes de elegir esta opción, verifique que la cinta está lista en la unidad correspondiente

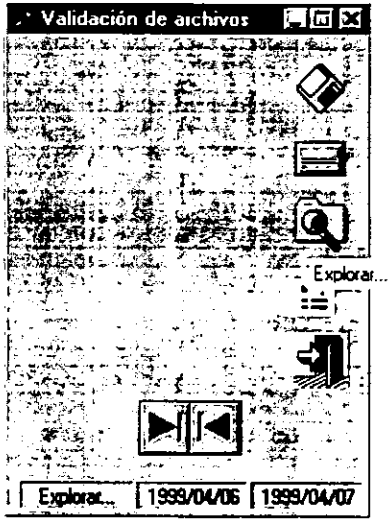


Figura 2.6 Validando archivos de Unidad de Cinta.

Nota: En la mayoría de las sucursales regionales, no se cuenta con dispositivos manejadores de cinta y no aplican las opciones que usan unidades de cinta que aparecen en este manual.

iii) Consultar transferencia por banco

Para poder consultar una transferencia, hacer doble click en el icono indicado en la pantalla de la figura 2.7, el sistema solicitará el número del banco respectivo. Después de introducir el dato anterior, el programa desplegará la información de entrada del banco indicado.

En caso de que el banco elegido no tenga transferencias, el programa desplegará un mensaje de falta de datos.

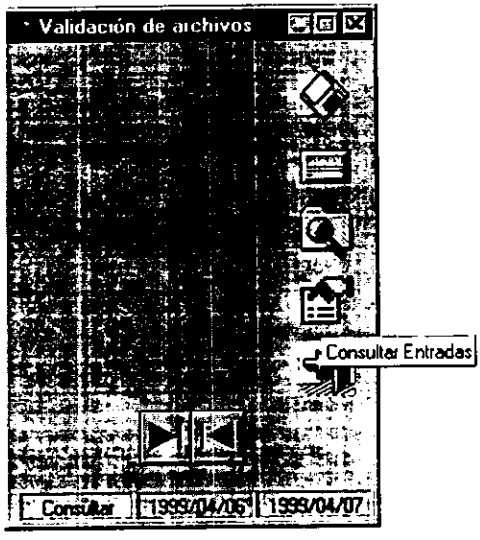


Figura 2.7 Consulta de los archivos de entrada.

Una vez que se ha terminado la validación y consulta de archivos, continuamos con el proceso del sistema. Elegimos el icono "Siguiente" o "Anterior" y hacemos click para proseguir o retroceder según sea el caso (ver figura 2.8)

Cabe mencionar que estos dos iconos de avance y retroceso se emplearán a lo largo de toda la aplicación para cambiar de una pantalla a otra.

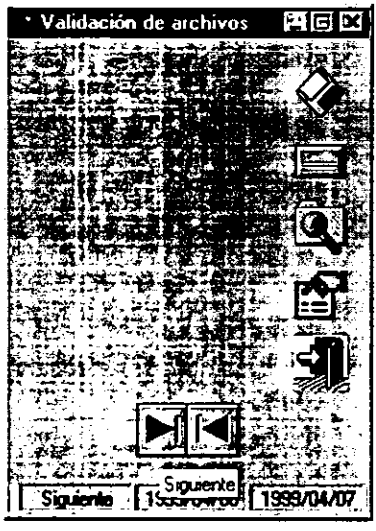


Figura 2.8 Ventana que ilustra como continuar.

iv) Resultados del Proceso

En esta sección el usuario procederá a ejecutar ciertos procesos para la obtención de los resultados del proceso de la compensación bancaria de cheques. La ventana a continuación muestra las diversas opciones que nos permitirán obtener los resultados correspondientes a cada banco (ver figura 2.9).

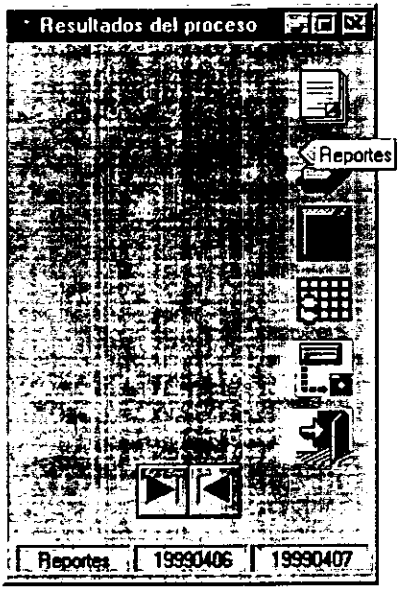


Figura 2.9 Resultados del proceso.

v) Reportes

Al seleccionar el icono de reportes, se despliega una ventana con una serie de iconos concernientes a los diversos procesos y reportes correspondientes al proceso de la compensación bancaria. A continuación mostramos toda la gama de posibles reportes que se pueden generar (Ver figura 2.10). Para ejecutar cualquiera de las opciones bastará con seleccionar mediante el ratón y hacer doble click a la opción deseada.

Carta Presentación:

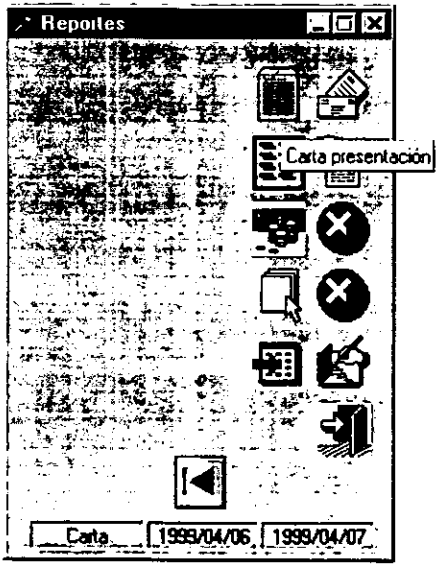
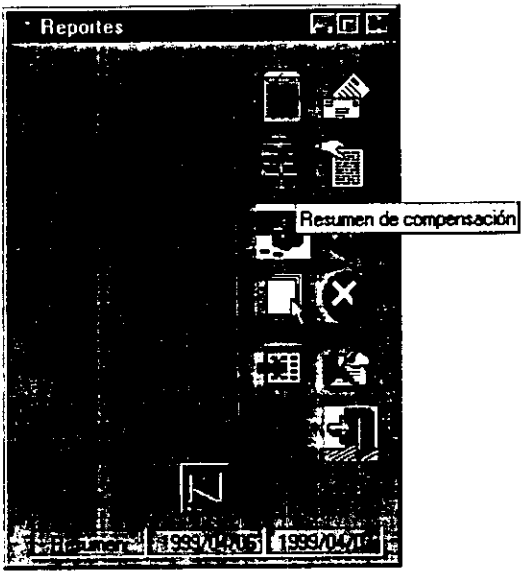


Figura 2.10 Reporte Carta Presentación.

En caso de que el delegado de algún banco, presente erróneamente la carta de presentación de su información, el sistema permitirá generar el reporte de los datos de entrada de dicho banco. Para ejecutar este reporte se hará doble click en el icono correspondiente de la figura anterior.

Resumen de Compensación.

La figura 2.11 muestra el icono que permite imprimir el reporte global de la información que los bancos han presentado y recibido.



La figura 2.11 Resumen de Compensación.

Hoja de Compensación (Ejemplo)

RESUMEN DE COMPENSACION ELECTRONICA PLAZA : MEXICO, D.F. FECHA : 1999/04/12 HORA : 21:09

CLAVE		ACEPTADO DOCTOS	ACEPTADO IMPORTE	RECIBIDO DOCTOS	RECIBIDO IMPORTE	DIFERENCIA
002	BANAMEX	46,355	644,279,001.34	32,007	465,200,981.52	179,078,019.82
003	SERFIN	7,836	145,642,459.75	15,180	192,079,017.65	-46,436,557.90
007	CITIBAN	6,369	90,432,462.62	5,174	103,821,512.45	-13,389,049.85
011	CONFIA	9,363	127,192,283.76	9,293	112,930,155.41	14,262,128.35
012	BANCOME	42,637	589,470,761.21	30,143	353,656,043.41	235,814,717.79
013	INDUSTR	52	793,088.67	32	126,443.13	666,645.54
014	SANT-ME	14,901	207,965,630.41	16,290	214,456,091.14	-6,490,460.73
017	BBV	17,495	215,185,512.01	19,642	275,330,639.88	-60,145,127.87
019	BANEFA	399	12,902,408.71	1,043	8,957,867.20	3,944,541.51
021	BITAL	43,607	604,538,769.45	38,532	415,766,886.63	188,771,882.82
025	SURESTE	278	2,286,269.12	829	20,062,979.06	-17,776,709.94
030	BAJIO	522	5,838,815.85	160	3,268,794.52	2,570,021.33
032	IXE	920	25,564,115.39	1,199	10,700,849.98	14,863,265.41
036	INBURSA	155	6,298,618.37	1,148	37,394,870.50	-31,096,252.13
037	INTERAC	0	0.00	75	935,969.62	-935,969.62
042	MIFEL	0	0.00	733	19,753,961.59	-19,753,961.59
044	INVERLA	14,680	181,310,611.53	17,062	273,664,388.58	-92,353,777.05
059	INVEX	0	0.00	38	368,863.01	-368,863.01
062	AFIRME	673	14,842,953.02	660	10,342,371.66	4,500,581.36
068	PROMEX	4,941	56,646,191.26	5,049	58,104,291.51	-1,458,100.25
072	BANORTE	0	0.00	10,252	207,078,896.06	-207,078,896.06
102	ABN	0	0.00	5	972,149.24	-972,149.24
106	AMERICA	0	0.00	11	242,417.45	-242,417.45
107	BOSTON	0	0.00	16	416,444.97	-416,444.97
108	TOKIO	0	0.00	56	656,770.63	-656,770.63
113	DRESDNE	0	0.00	13	189,628.97	-189,628.97
119	REPUBLI	0	0.00	97	6,987,181.16	-6,987,181.16
149	BANRURA	0	0.00	176	1,400,351.77	-1,400,351.77
161	BANCREC	0	0.00	11,470	136,323,133.74	-136,323,133.74
T O T A L		216,385	2,931,189,952.44	216,385	2,931,189,952.44	0.00

TOTAL NEGATIVO : -644,471,803.93 TOTAL POSITIVO : 644,471,803.93

Opción de Reportes, Póliza de Liquidación final.

La figura 2.12 muestra el icono a emplear al optar por la generación de la póliza.

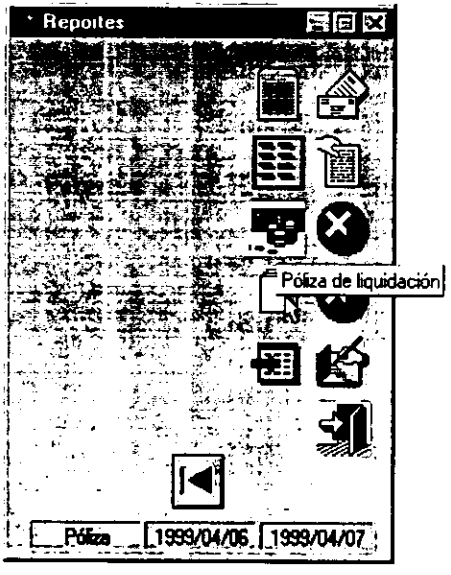


Figura 2.12 Póliza de Liquidación.

Póliza (Ejemplo)

HOJA DE LIQUIDACION FINAL DE COMPENSACION ELECTRONICA PLAZA : MEXICO, D.F. FECHA : 1999/04 11

TRANSITO	SALDOS DEUDORES	DETALLE	TRANSITO	SALDOS ACREEDORES
		BANCO NACIONAL DE MEXICO, S.A.	002	[illegible]
003	46,434,837.97	BANCA SERFIN, S.A.		
007	13,34[illegible], 4[illegible].95	CITIBANK MEXICO, S.A.		
		BANCA CONFIA, S.A	011	[illegible],126.36
		BANCOMER, S.A.	012	238,314,717.79
		BANCO INDUSTRIAL, S.A.	013	666,669.54
014	6,491,441.73	BANCO SANTANDER MEXICANO, S.A.		
017	60,141,127.67	BANCO BILBAO VIZCAYA MEXICO, S.A.		
		BANCO DEL EJER. FZA. AEREA Y ARM, S.N.C.	019	3,944,541.51
		BANCO INTERNACIONAL, S.A.	021	[illegible],662.82
025	17,[illegible]6,[illegible]9.94	BANCO DEL SURESTE, S.A.		
		BANCO DEL BAJIO, S.A.	030	[illegible],[illegible]1.33
		I.E BANCO, S.A.	032	14,663,265.41
036	31,094,281.13	BANCO INBURSA, S.A.		
037	935,969.62	BANCO INTERACCIONES, S.A.		
042	19,[illegible]61,59	BANCA MIFEL, S.A.		
044	92,353,[illegible].05	BANCO INVERLAT, S.A.		
059	369,663.01	BANCO INVEX, S.A.		
		BANCA AFIRME, S.A.	062	[illegible],561.34
068	1,455,[illegible].25	BANCA PROMEX, S.A.		
072	207,[illegible],896.06	BANCO MERCANTIL DEL NORTE, S.A.		
102	9[illegible],149.24	BK. AMRO BANK (MEXICO), S.A.		
106	240,417.45	BANK OF AMERICA MEXICO, S.A.		
107	416,444.97	BANCO DE BOSTON, S.A.		

108	656,[illegible]	BANK OF TOKIO MEXICO, S.A	
113	169,[illegible]	DRESDNER BANK MEXICO, S.A.	
119	6,987,181.14	REPUBLIC NATIONAL BANK OF N.Y. MEX, S.A.	
149	1,406,351.[illegible]	BANCO NACIONAL DE CREDITO RURAL, S.N.C.	
161	136,323,233.74	BANCRECER, S.A.	
	644,471,603.93	SUMAS IGUALES	644,471,803.93

SUMA NUM. DE TRANSITO CON SALDOS DEUDORES
1,385

SUMA NUM. DE TRANSITO CON SALDOS ACREEDORES
202

Opción de Reportes, Reportes de Control

A continuación indicamos el icono a elegir para la obtención de los reportes de control como se ve en la figura 2.13.

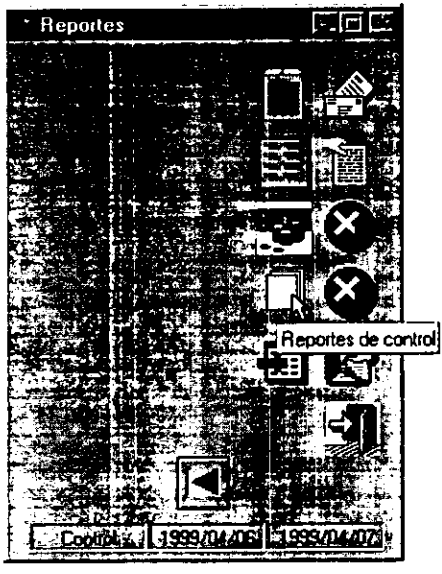


Figura 2.13 Reportes de Control.

Reporte de Cifras de Control de la compensación (Ejemplo)

REPORTE DE CIFRAS DE CONTROL DE LA COMPENSACION ELECTRONICA PLAZA : MEXICO, D.F. FECHA : 1999/04/1[illegible]

CLAVE	INSTITUCION	DOCTOS	IMPORTE
002	BANCO NACIONAL DE MEXICO, S.A.	32,007	465,200,981.52
003	BANCA SERFIN, S.A.	15,180	192,079,017.65
007	CITIBANK MEXICO, S.A.	5,174	103,821,512.45
011	BANCA CONFIA, S.A.	9,293	112,930,155.41
012	BANCOMER, S.A.	30,143	353,656,043.41
013	BANCO INDUSTRIAL, S.A.	32	126,443.13
014	BANCO SANTANDER MEXICANO, S.A.	16,290	214,456,091.14
017	BANCO BILBAO VIZCAYA MEXICO, S.A.	19,642	275,330,639.88
019	BANCO DEL EJER, FZA. AEREA Y AFM. S.N.C.	1,043	8,957,867.20
021	BANCO INTERNACIONAL, S.A.	38,532	415,766,886.63
025	BANCO DEL SURESTE, S.A.	829	20,062,979.06
030	BANCO DEL BAJIO, S.A.	160	3,268,754.52
032	IXE BANCO, S.A.	1,199	10,700,849.96
036	BANCI INBURSA, S.A.	1,148	37,394,870.50
037	BANCO INTERACCIONES, S.A	75	935,969.62
042	BANCA MIFEL, S.A.	733	19,753,961.59
044	BANCI INVERLAT, S.A.	17,062	273,664,388.58
059	BANCO INVEX, S.A.	38	368,863.01
062	BANCA AFIRME, S.A.	660	10,342,371.66
068	BANCA PROMEX, S.A.	5,049	58,104,291.51
071	BANFAIS, S.A.	4,013	47,600,642.03
072	BANCO MERCANTIL DEL NORTE, S.A.	6,068	157,651,940.87
086	BANCO DEL CENTRO, S.A	171	1,826,313.16
102	ABN AMRO BANK (MEXICO , S.A	5	972,149.24
106	BANK OF AMERICA MEXICO, S.A	11	242,417.45
107	BANCO DE BOSTON, S.A.	16	416,444.97
108	BANK OF TOKIO MEXICO, S.A	56	656,770.63
113	DRESDNER BANK MEXICO, S.A.	13	189,628.97
119	REPUBLIC NATIONAL BANK OF N.Y. MEX, S.A.	97	6,987,181.16
143	BANCO DE CREDITO RURAL DEL NORTE, S.N.C.	1	14,190.20
144	BANCO DE CREDITO RURAL DEL CENTRO, S.N.C.	13	163,998.19
146	BANCO DE CREDITO RURAL DEL GOLFO, S.N.C.	6	27,570.99
147	BANCO DE CREDITO RURAL DEL CENTRO SUR, S.N.C	40	352,817.76
148	BANCO DE CREDITO RURAL DEL NOROESTE, S.N.C	1	2,265.67
149	BANCO NACIONAL DE CREDITO RURAL, S.N.C.	94	789,781.94
158	BANCO DE CREDITO RURAL DEL ISTMO, S.N.C.	4	11,996.23
160	BANCO DE CREDITO RURAL DEL PENINSULAR, S.N.C	1	3,286.11
161	BANCRECER, S.A.	11,470	136,323,133.74
162	BANCO DE CRED. RURAL DEL CENTRO NORTE, S.N.C	1	8,587.01
165	BANCO DE CRED. RURAL DEL PACIFICO SUR, S.N.C	11	25,855.38
	T O T A L :	216,365	2,931,189,952.44

Reporte Resumen de Entradas

La figura 2.14 muestra el icono que permite imprimir el reporte global de la información que los bancos han presentado. Asimismo, este reporte sirve para verificar que todos los archivos presentados, hayan sido procesados y considerados dentro de todo el proceso de compensación.

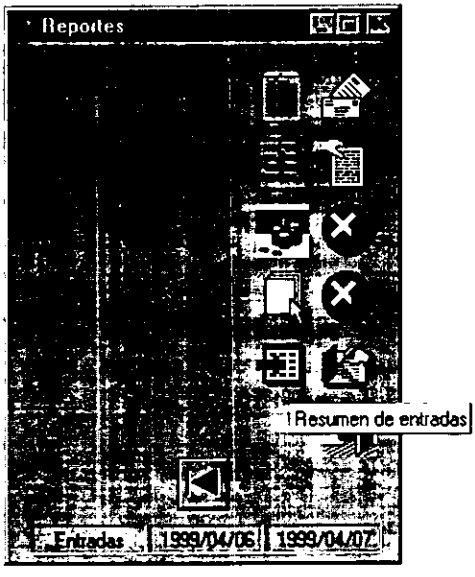


Figura 2.14 Resumen de entradas.

Para procesar este reporte bastará hacer doble click en el icono mostrado en la figura anterior (ver figura 2.14).

Hoja de Control de Entrega de resultados de la Compensación Electrónica

HOJA DE CONTROL DE ENTREGA DE RESULTADOS DE LA COMPENSACION ELECTRONICA MEXICO, D.F. Fecha : 1999/04/12

		R E C I B I D O S			N O M B R E
CLAVE	INSTITUCION	SOBRES	DOCTOS	IMPORTE	
002	BANCO NACIONAL DE MEXICO, S.A.	14	32,007	465,200,981.52	_____
003	BANCA SERFIN, S.A.	15	15,180	192,039,017.65	_____
007	CITIBANK MEXICO, S.A.	15	5,174	103,021,512.45	_____
011	BANCA CONFIA, S.A.	15	9,293	122,930,155.41	_____
012	BANCOMER, S.A.	15	30,143	353,656,043.41	_____

013	BANCO INDUSTRIAL, S.A.	8	32	126,443.13	
014	BANCO SANTANDER MEXICANO, S.A.	16	16,290	214,456,091.14	
017	BANCO BILBAO VIZCAYA MEXICO, S.A.	16	19,642	275,330,639.88	
019	BANCO DEL EJE, FIA. AEREA Y ARM, S.N.C	14	1,043	8,957,867.20	
021	BANCO INTERNACIONAL, S.A.	16	38,532	415,766,886.63	
025	BANCO DEL SURESTE, S.A.	10	829	20,062,979.06	
030	BANCO DEL BAJIO, S.A.	9	160	3,268,794.52	
032	IXE BANCO, S.A.	12	1,199	10,700,849.98	
036	BANCO INBURSA, S.A.	12	1,148	37,394,870.50	
037	BANCO INTERACCIONES, S.A.	12	75	935,969.62	
042	BANCA MIFEL, S.A.	12	733	19,753,961.59	
044	BANCO INVERLAT, S.A.	16	17,062	273,664,388.58	
059	BANCO INVEX, S.A.	5	38	368,883.01	
062	BANCA AFIRME, S.A.	10	660	10,342,371.66	
069	BANCA PROMEX, S.A.	16	5,049	58,104,291.51	
071	BANPAIS, S.A	16	4,013	47,600,842.03	
072	BANCO MERCANTIL DEL NORTE, S.A	17	6,069	157,651,940.87	
086	BANCO DEL CENTRO, S.A.	8	171	1,826,313.16	
102	ABN AMRO BANK (MEXICO), S.A.	2	5	972,149.24	
106	BANK OF AMERICA MEXICO, S.A.	1	11	242,417.45	
107	BANCO DE BOSTON, S.A.	4	16	416,444.97	
108	BANK OF TOKIO MEXICO, S.A	7	56	656,770.63	
113	DRESDNER BANK MEXICO, S.A.	5	13	189,628.97	
119	REPUBLIC NATIONAL BANK OF N.Y. MEX, S.A	7	97	6,987,181.16	
143	BANCO DE CREDITO RURAL DEL NORTE, S.N.C	1	1	14,192.20	
144	BANCO DE CREDITO RURAL DEL CENTRO, S.N.C	1	13	163,998.10	
146	BANCO DE CREDITO RURAL DEL GOLFO, S.N.C	2	6	27,570.99	
147	BANCO DE CREDITO RURAL DEL CENTRO SUR S.N.C	3	41	352,817.78	
148	BANCO DE CREDITO RURAL DEL NOROESTE, S.N.C	-	-	2,265.67	
149	BANCO NACIONAL DE CREDITO RURAL, S.N.C	6	55	789,781.94	
150	BANCO DE CREDITO RURAL DEL ITSMO, S.N.C	-	4	11,996.23	
160	BANCO DE CREDITO RURAL DEL PENINSULAR S.N.C	1	1	3,286.51	
161	BANCRECER, S.A	17	11,471	136,323,133.74	
162	BANCO DE CRED RURAL DEL CENTRO NORTE S.N.C	1	1	8,587.00	
163	BANCO DE CRED RURAL DEL PACIFICO SUR S.N.C	2	11	25,855.35	

Reporte Matriz de Sobres

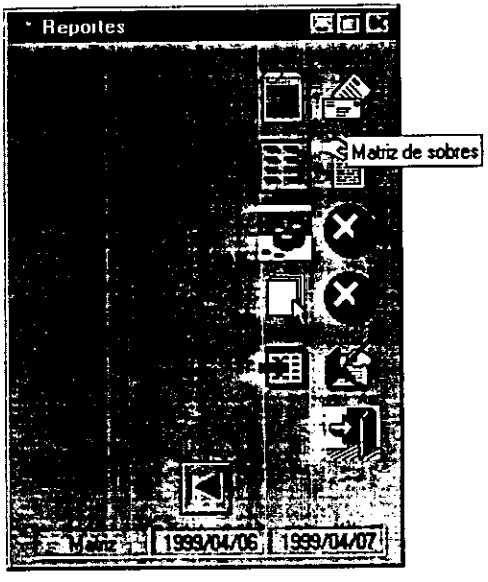


Figura 2.15 Matriz de sobres.

Este reporte nos indica cuantos sobres presenta y recibe cada banco (ver figura 2.15).

Reporte Hoja de Compensación

Vale la pena mencionar que, las opciones [Hoja de Compensación]. [ROE (Reporte de Operaciones Erróneas)] y [ROEC (Reporte de Operaciones Erróneas a Cargo)], permiten al usuario imprimir los reportes respectivos de tres maneras (distintos rangos para bancos), que mencionaremos más adelante. A continuación en la figura 2.16. Se muestra el icono para el reporte hoja de compensación:

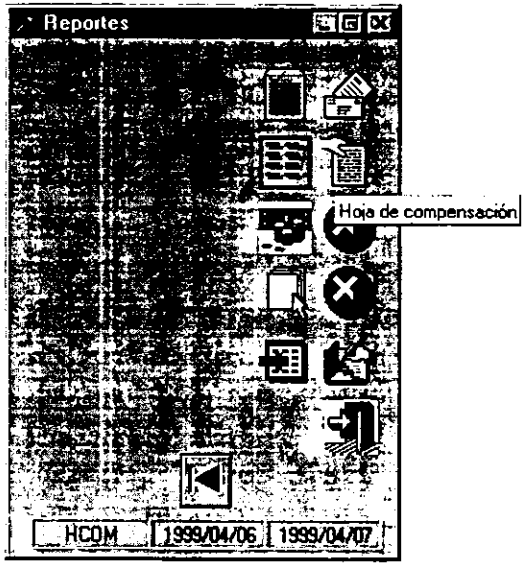


Figura 2.16 Hoja de compensación.

Antes de que el programa proceda con la generación del reporte respectivo, independientemente de la opción que se elija, pedirá el número del banco que se desea, tal como se muestra en la figura 2.17

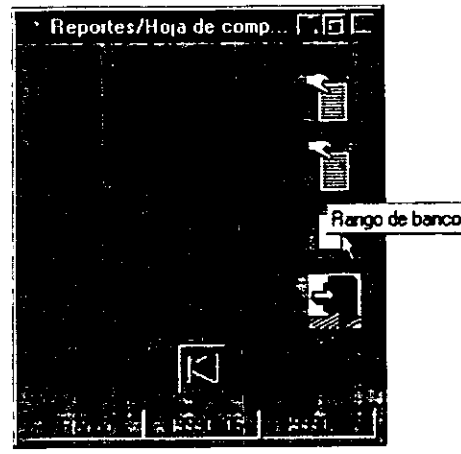


Figura 2.17 Opción de Rango de Bancos.

Otras opciones para elegir el o los bancos que se requieran en el proceso de la compensación, se muestran en la figura 2.18 siguiente:

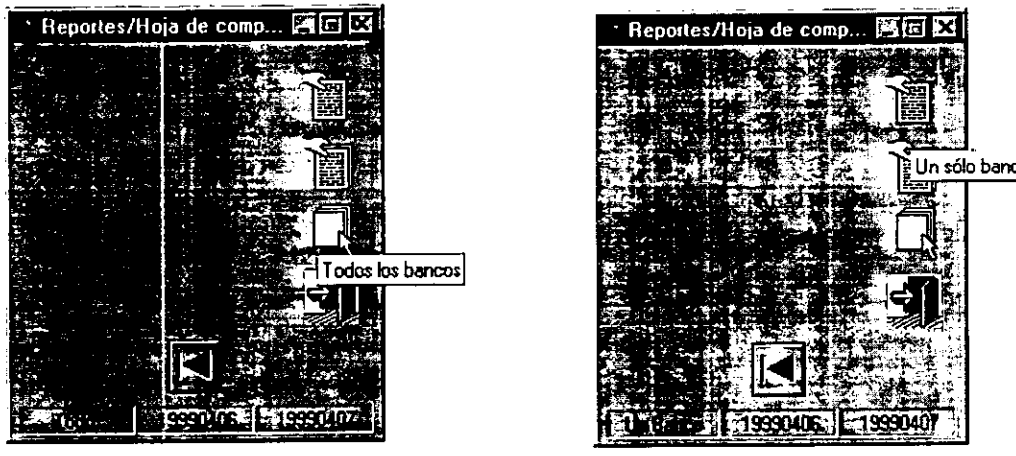


Figura 2.18 Opciones para bancos.

Reportes de operaciones presentadas con Anomalías

Con esta opción se generan los archivos con el detalle de las operaciones presentadas con anomalías para cada banco. Al seleccionar esta opción, aparece el menú que se muestra en la figura 2.19

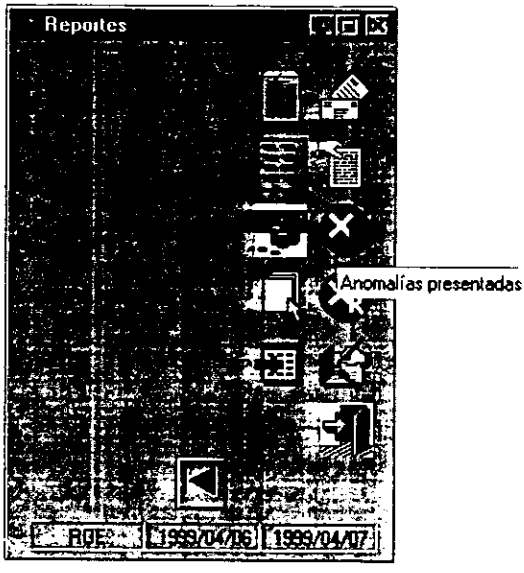


Figura 2.19 Reportes presentados con anomalías.

Al igual que en la generación del reporte de la hoja de compensación, basta con seleccionar la opción y dar el banco o rango de bancos, para generar el reporte.

(Nota: Con la finalidad de exponer un panorama global, el detalle respectivo se muestra reducido de tal manera que el usuario pueda tener una idea completa de su contenido)

ROE – Operaciones erróneas (Ejemplo)

```
CECOBAN, S.A. DE C.V.                                                    HORA : 19:33
REPORTE DETALLE DE OPERACIONES PRESENTADAS CON ANOMALIAS                  1999/04/12   HOJA 0001
-------------------------------------------------------------------------------------------------
PLAZA                    :
INSTITUCION CEDENTE      :      007 CITIBANK MEXICO, S.A.
TRANSFERENCIA            :  1130001
FECHA DE INTERCAMBIO     :  1999/04/12
TIPO DE OPERACION        :      040 CHEQUES DE COBRO INMEDIATO
-------------------------------------------------------------------------------------------------
          INSTITUCION    REF  LOC                                    CAMPOS C/ ERROR          I        I M P O R T E
SECUENCIA   EMI   REC    LOTE    SEC  TT  PPP     CUENTA        CHEQUE     DI  DP  SSS  1 2 3 4 5 6 7 8  TRANSITO  ?
-------------------------------------------------------------------------------------------------
 0000290    007   072   0110029 0003  51  115  00000006733705  0000004462  0   8   557                          8,248.00
EL "DIGITO VERIFICADOR DE INTERCAMBIO" ES INVALIDO
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0000296    007   072   0110029 0009  51  115  00000012669275  0000004946  7   0   262                          2,176.14
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0001670    007   014   0110025 0100  51  115  00605001104512  0000000287  5   9   055                          4,500.00
EL "DIGITO VERIFICADOR DE INTERCAMBIO" ES INVALIDO
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0001959    007   014   0110025 0122  51  115  00655000257327  0000000976  8   8   86.                          7,631.65
EL "DIGITO VERIFICADOR DE INTERCAMBIO" ES INVALIDO
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0003335    007   014   0110053 0007  51  115  00570006053491  0000000088  0   3   010                         23,500.00
EL "DIGITO VERIFICADOR DE INTERCAMBIO" ES INVALIDO
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0003477    007   017   0110048 0015  51  115  00001100555352  0000501080  0   8   022                         23,112.31
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0004009    007   032   0110031 0046  51  115  00000019212217  00022733194  8   1   449                         60,212.00
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0004194    007   032   0110031 0050  51  115  00000010179121  00019733323  6   7   511                          7,570.99
EL "DIGITO VERIFICADOR DE INTERCAMBIO" ES INVALIDO
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0004327    007   021   0110047 0171  51  999  00040006830902  0009207959  4   5   977                          6,000.00
EL "DIGITO VERIFICADOR DE INTERCAMBIO" ES INVALIDO
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0004377    007   021   0110047 0181  51  999  00400048524224  0006268823  8   2   699                         10,000.00
EL "DIGITO VERIFICADOR DE INTERCAMBIO" ES INVALIDO
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0004658    007   002   0110058 0050  51  999  00032600982258  0000000221  4   7   006                            725.47
EL "DIGITO VERIFICADOR DE INTERCAMBIO" ES INVALIDO
-------------------------------------------------------------------------------------------------
 0004672    007   021   0110056 0050  51  999  00021020299998  0005769193  3   9   973                          4,724.00
EL "DIGITO VERIFICADOR DE INTERCAMBIO" ES INVALIDO
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------
```

ROEC – Reporte de Operaciones Erróneas a Cargo

Con esta opción se podrán generar los reportes de anomalías a cargo de cada banco. Para seleccionar esta opción haga doble click sobre el icono que se muestra en la figura 2.20

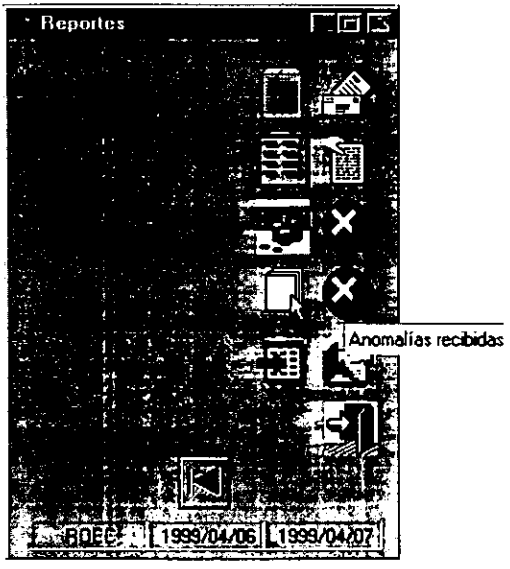


Figura 2.20 Reporte Anomalías recibidas.

Después de indicar el banco o rango de bancos correspondiente se imprime los reportes generados.

ROEC – Reporte de Operaciones Erróneas a Cargo (Ejemplo)

```
CECOBAN, S.A. DE C.V                                                                    HORA   11:52
REPORTE DETALLE DE OPERACIONES A SU CARGO CON ANOMALIAS                    1999/04/12   HOJA 0001
-------------------------------------------------------------------------------------------------------------------------
PLAZA                    AREA METROPOLITANA DE LA CIUDAD DE MEXICO, D.F
INSTITUCION DESTINO      007 CITIBANK MEXICO, S.A.
TRANSFERENCIA            003007
FECHA DE INTERCAMBIO     1999/04/12
TIPO DE OPERACION        040 CHEQUES DE COBRO INMEDIATO
-------------------------------------------------------------------------------------------------------------------------
          INSTITUCION    REF  LOC                                                  CAMPOS C/ ERROR                 E      I M P O R T E
SECUENCIA    EMI   REC    LOTE    SEC  TT  FPP     CUENTA        CHEQUE     Di  Dp  SSS  1 2 3 4 5 6 7 8  TRANSITO  T
-------------------------------------------------------------------------------------------------------------------------
 0000252    003   007   0020013 0006  51  115  0000203157029  0001400818  6   1   000                                    85
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------------------------------
 0000254    003   007   0020013 0008  51  115  0000030171039  0000052482  3   4   210                                 21,600
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------------------------------
 0000257    003   007   0020013 0010  51  115  0000203157029  0001400853  6   1   000                                 36,973
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------------------------------
 0000816    003   007   0020013 0034  51  115  0000030171039  0000052478  3   1   000                                  1,847
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------------------------------

    NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 003 =   0004  IMPORTE :              61,305.37

-------------------------------------------------------------------------------------------------------------------------
 0006983    011   007   0031024 0001  51  115  0009710100001  0000069063  2   9   210                                125,400
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------------------------------
 0007282    011   007   0031035 0001  51  115  0009710100001  0000068988  2   3   210                                 15,897
EL "DIGITO VERIFICADOR DE PREMARCADO" ES INVALIDO
-------------------------------------------------------------------------------------------------------------------------

    NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 011 =   0002  IMPORTE :             141,297.07
```

```
---------------------------------------------------------------------------------------------------------------------
0005987    014   007   8010001 0125  01        00-634646-051  0000000066                     010010077          490.00
EL VALOR DEL CAMPO "CLAVE DE TRANSACCION" ES INVALIDO
---------------------------------------------------------------------------------------------------------------------

   NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 014 =   0001  IMPORTE :              490.00

---------------------------------------------------------------------------------------------------------------------
0000330    032   007   0000001 0329  55  115  0005710500007  0000023175  6  1  222                          9,068.25
EL VALOR DEL CAMPO "CLAVE DE TRANSACCION" ES INVALIDO
---------------------------------------------------------------------------------------------------------------------

   NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 032 =   0001  IMPORTE :              9,068.25

---------------------------------------------------------------------------------------------------------------------
0000123    044   068   0010001 0001  55  115  0000041126221  0002044218  7  5  622                          2,242.16
EL VALOR DEL CAMPO "CLAVE DE TRANSACCION" ES INVALIDO
---------------------------------------------------------------------------------------------------------------------
0001300    044   068   0080001 0006  55  115  0000041126221  0002046475  7  2  687                          6,618.00
EL VALOR DEL CAMPO "CLAVE DE TRANSACCION" ES INVALIDO
---------------------------------------------------------------------------------------------------------------------
0001301    044   068   0090001 0007  55  115  0000041126221  0002046474  7  1  752                          2,781.00
EL VALOR DEL CAMPO "CLAVE DE TRANSACCION" ES INVALIDO
---------------------------------------------------------------------------------------------------------------------
0003140    044   068   0240001 0003  55  115  0000041126221  0002047503  7  6  782                         15,000.00
EL VALOR DEL CAMPO "CLAVE DE TRANSACCION" ES INVALIDO
---------------------------------------------------------------------------------------------------------------------

   NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 044 =   0004  IMPORTE :              26,641.16

---------------------------------------------------------------------------------------------------------------------
                         SUBTOTALES Y GRAN TOTAL

    NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 002 =   0002  IMPORTE :             4,780.42
    NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 003 =   0001  IMPORTE :             1,800.00
    NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 011 =   0003  IMPORTE :            29,416.06
    NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 012 =   0001  IMPORTE :               390.00
    NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 021 =   0002  IMPORTE :             4,164.00
    NUMERO DE OPERACIONES RECHAZADAS DEL BANCO 044 =   0004  IMPORTE :            26,641.16

                   -------------------------------------------------------------------
                   TOTAL OPERACIONES RECHAZADAS : 0013  IMPORTE :                 66,591.62
```

Reporte Acuse de Recibo

La figura 2.21 nos muestra la pantalla para generar el reporte del acuse de recibo.

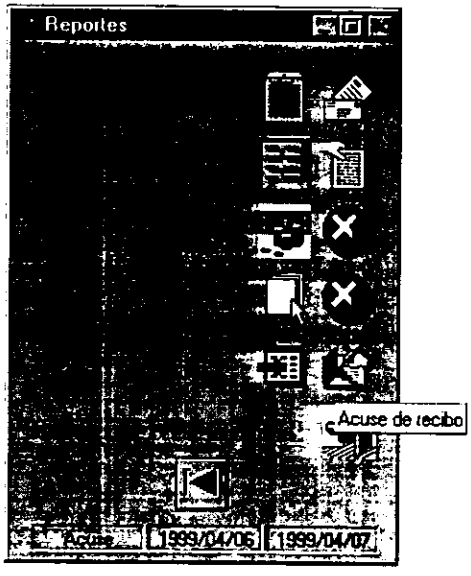


Figura 2.21 Reporte de Acuse de recibo.

Una vez que se han generado los reportes hacemos doble click sobre el icono de retroceso, para regresar a la pantalla anterior.

Habiendo regresado a la pantalla de resultados mostrada antes, se procede a generar las salidas del proceso de compensación en medio magnético y reportes que se le entregará al delegado del banco, por lo que dependiendo de las necesidades correspondientes para cada banco se generarán los resultados ya sea en disquete, cinta, para llevarlo a cabo, primero se elige el icono de archivos de cheques a su cargo como se muestra a continuación (ver figura 2.22).

Figura 2.22 Resultados - Archivo de cheques a su cargo.

Acto seguido se indica en que medio magnético se requieren los resultados (ver figura 2.23):

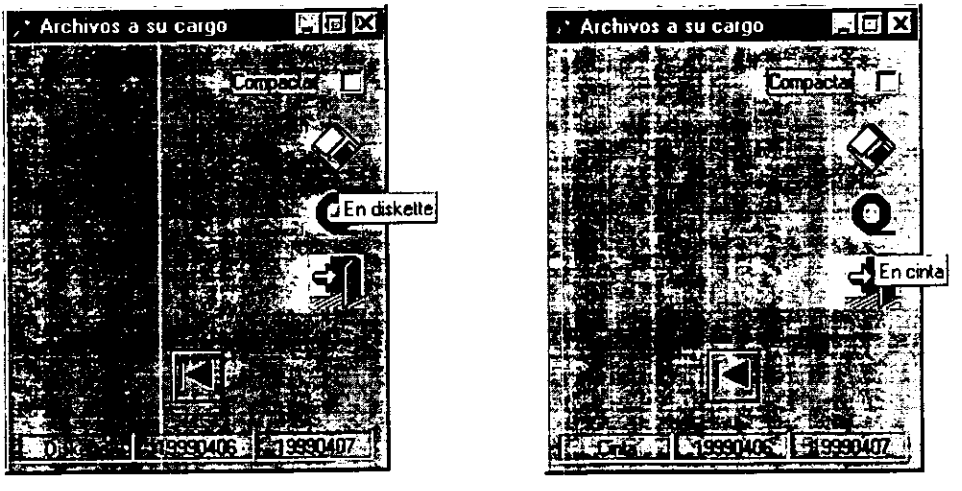


Figura 2.23 Resultados en medios magnéticos.

Si se desea generar los resultados en formato compactado, se marcará la opción que se muestra en la figura 2.24.

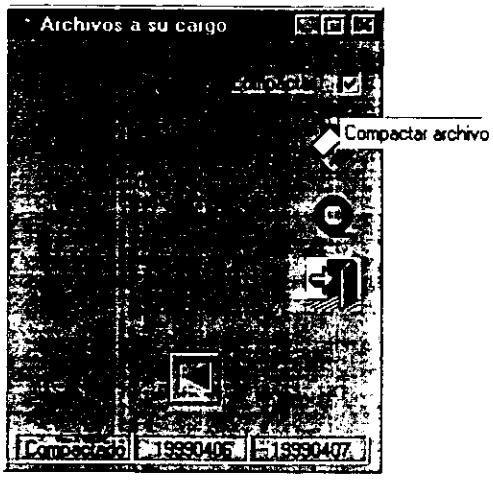


Figura 2.24 Ventana que muestra el icono para formato compactado.

Haciendo uso de las teclas de retroceso, volvemos a la ventana anterior para continuar con la generación de saldos.

vi) Saldos

La figura 2.25 nos muestra los pasos a seguir para la generación de saldos. Esta opción nos permite obtener los archivos para efectuar la liquidación de los resultados de la compensación a través del Banco de México (banco liquidador).

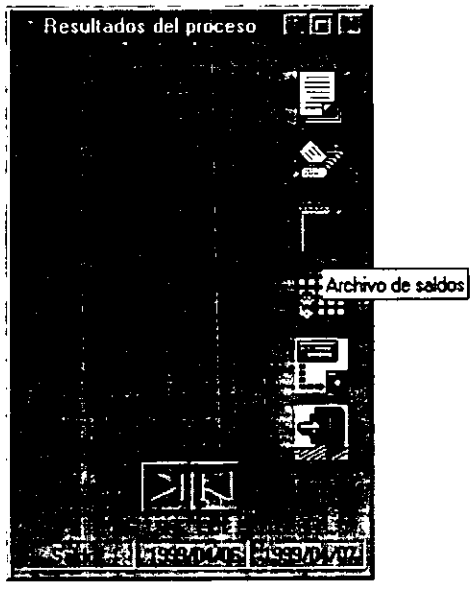


Figura 2.25 Saldos resultado del proceso.

El nombre del archivo generado, está compuesto de la siguiente manera: SPPCCBA1.DDC

Donde:

S Carácter "S" (salida).

PP Plaza de compensación.

CCBA Constante.

1 Número de servicio.

DD Es el día del proceso.

C Número consecutivo del proceso.

Acto seguido aparecerá una pantalla donde solicita el directorio para grabar el archivo de saldos como se muestra a continuación en la figura 2.26.

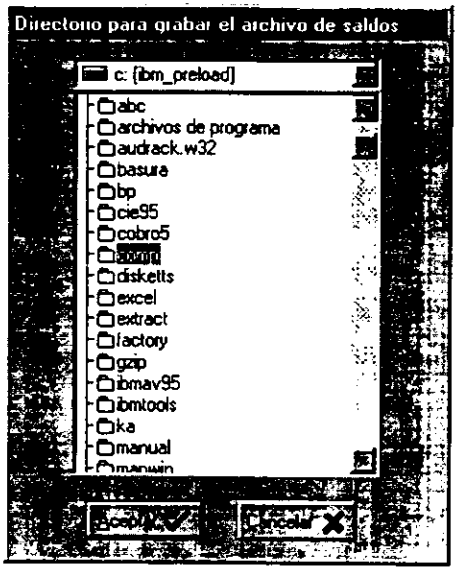


Figura 2.26 Grabar el archivo de Saldos.

Se especifica la trayectoria y se selecciona la opción "Aceptar", o en caso contrario, "Cancelar".

vii) Microfichas

Una vez terminado el proceso, habiendo revisado que los resultados hayan sido los correctos, se procede a la generación de los archivos para microfichas. En la cámara de compensación se almacenan los resultados en microfichas, para el caso de que algún banco requiera una aclaración posterior. Por lo que se ejecuta el siguiente proceso que se ilustra en la figura 2.27.

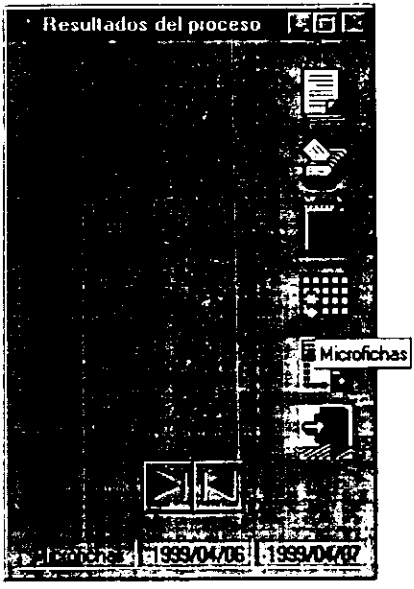


Figura 2.27 Generación de microfichas.

viii) Respaldo

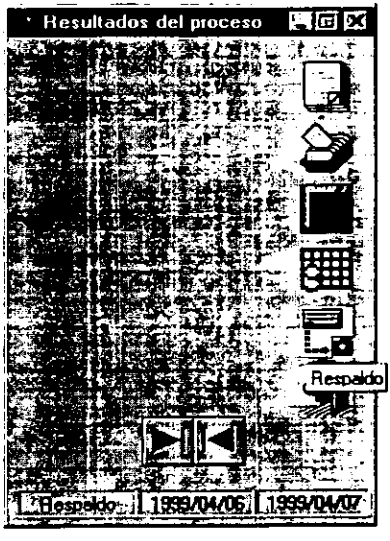


Figura 2.28 Respaldo de resultados.

Esta opción se usa para guardar o respaldar los datos de entrada en un medio magnético, además de servir para posible aclaración con algún banco. Se puede efectuar el respaldo en disco o en cinta (ver figura 2.29).

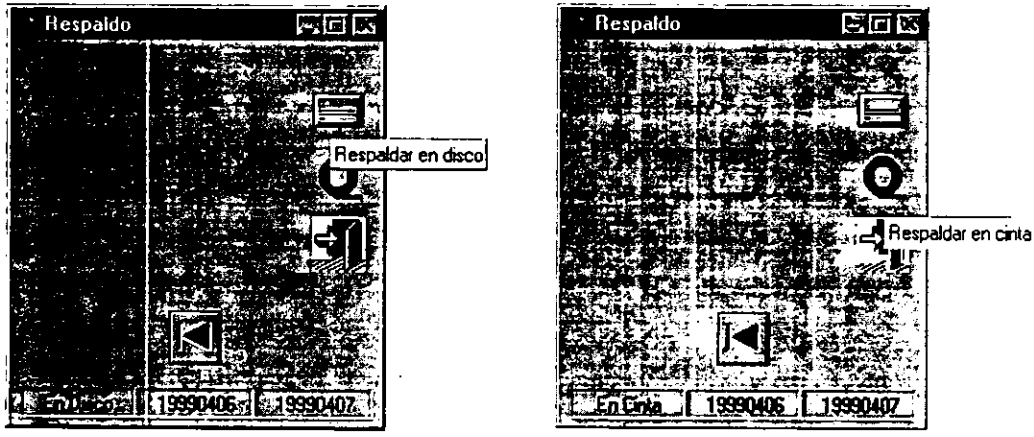


Figura 2.29 Medio magnético a emplear en el respaldo.

Después se puede seleccionar "salir" o "regresar", según sea el caso.

ix) Administración del Sistema

El Administrador del sistema será el encargado de registrar, modificar y consultar tanto a los usuarios como a los bancos, por lo que en primer lugar se requerirá de una medida de seguridad, que consiste en teclear una clave que lo autorice para realizar ciertas acciones. La figura 2.30 muestra la pantalla que se despliega para el acceso del administrador:

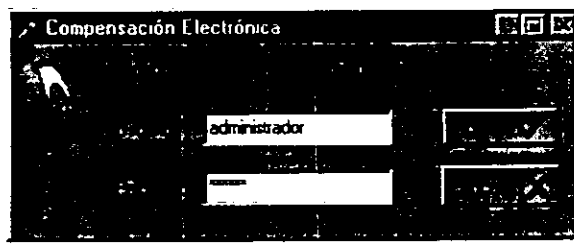


Figura 2.30 Acceso al administrador del sistema.

En esta sección se realiza el registro, modificación, eliminación y consulta de los usuarios que tendrán acceso al sistema, ver figura 2.31

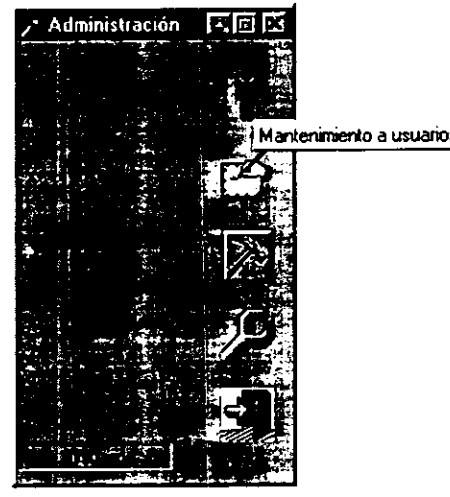


Figura 2.31 Mantenimiento a usuarios.

Se podrán llevar a cabo las siguientes operaciones: alta y baja de usuarios como podemos ver en la figura 2.32

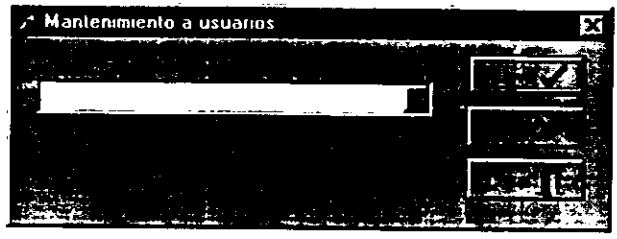


Figura 2.32 Ventana de altas y bajas de usuarios.

Una vez que han sido dados de alta los usuarios se procederá a darles una clave de acceso para que puedan ejecutar el sistema. (Ver figura 2.33)

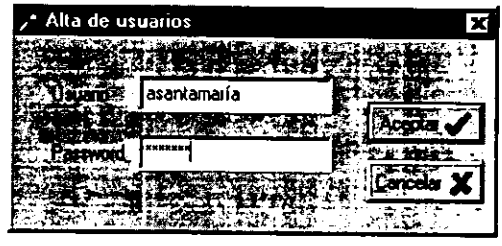


Figura 2.33 Alta de usuarios.

Otra de las funciones del Administrador del sistema es configurar el sistema de compensación bancaria para las diversas plazas ubicadas en cualquier parte de nuestro país, por lo que habrá de personalizarse el sistema después de instalarse como se ilustra a continuación (ver figura 2.34).

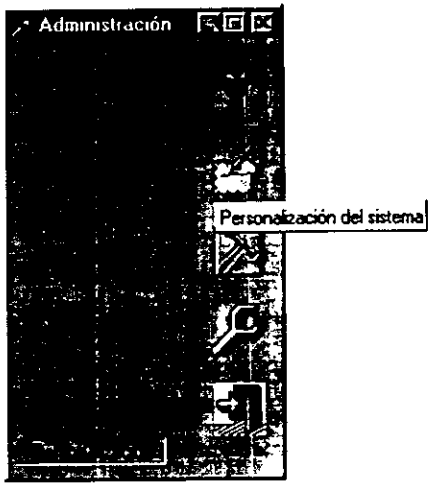


Figura 2.34 Personalización del sistema.

Para ello es necesario registrar algunos datos (ver figura 2.35):

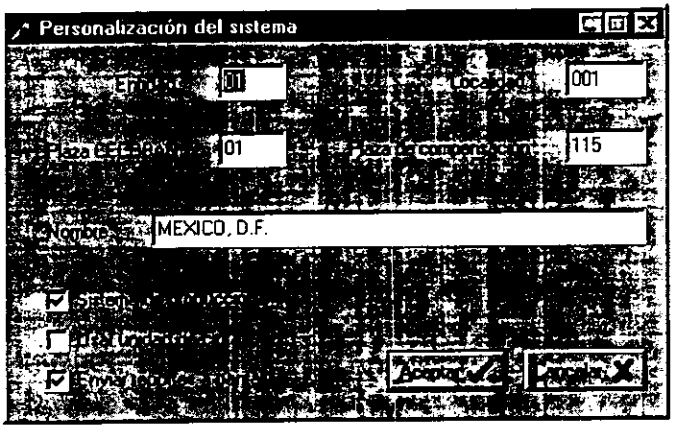


Figura 2.35 Datos para la personalización del sistema.

x) Mantenimiento (bancos)

En esta sección el administrador del sistema se encarga de actualizar la información concerniente a los bancos que requieran incorporarse al proceso de compensación electrónica (ver figura 2.36)

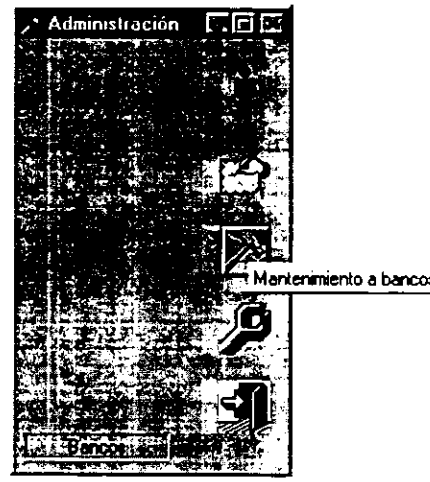


Figura 2.36 Mantenimiento a bancos.

Se pueden realizar altas y bajas de bancos (ver figuras 2.37 y 2.38).

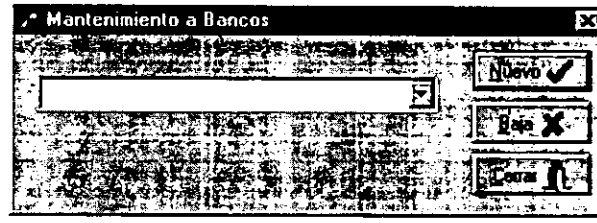


Figura 2.37 Alta y Baja de Bancos.

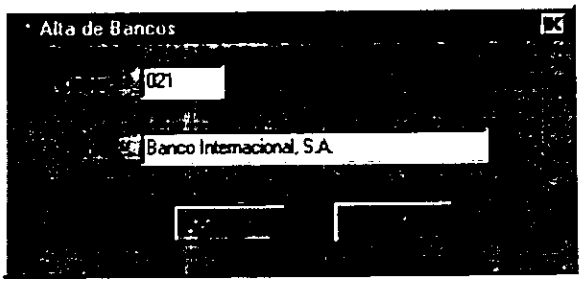


Figura 2.38 Alta de Bancos.

xi) Utilerías

El sistema incluye un conjunto de utilerías, mismas que se pueden visualizar al seleccionar la opción respectiva, tal como se muestra en la figura 2.39

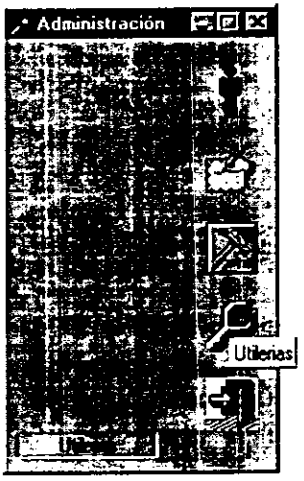


Figura 2.39 Ventana de Utilerías.

Para ejecutar cada una de las alternativas que conforman el conjunto de las utilerías, se selecciona la opción y se hace click.

Borrar transferencia

Para poder borrar una transferencia, el programa solicita el número del banco respectivo (ver figura 2.40)

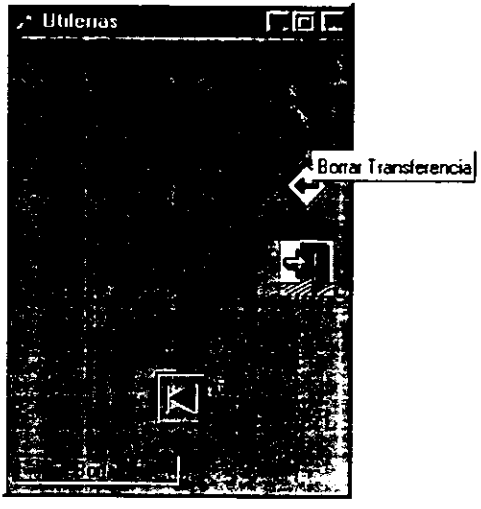


Figura 2.40 Borrar transferencia.

Después de introducir el dato del banco, el sistema avisará la terminación de la operación.

En caso de que el banco elegido no tenga transferencias, el programa enviará un mensaje de falta de datos.

Recuperar todo el proceso

La figura 2.41 nos muestra la pantalla que permite recuperar el proceso, en caso de haber algún error u omisión, en la información enviada por los bancos.

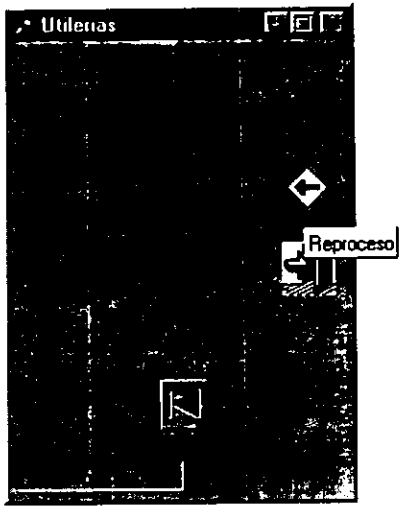


Figura 2.41 Reproceso.

xii) Finalizar

Esta opción se usa para terminar una sesión del sistema, como se muestra en la figura 2.42

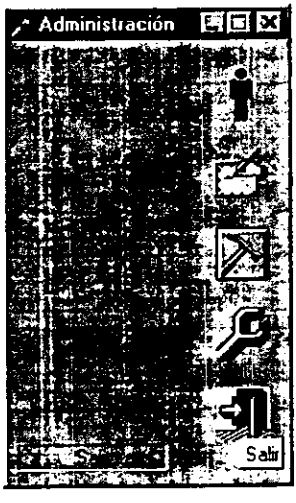


Figura 2.42 Finalizar el sistema.

Acto seguido pide la confirmación de la salida, desplegando la ventana de la figura 2.43

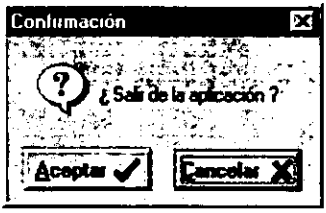


Figura 2.43 Confirmación de la salida del sistema.

CONCLUSIONES

CONCLUSIONES

El presente trabajo tuvo como finalidad la implementación del Sistema de Compensación Electrónica de Cheques, considerando en todo momento que la migración al nuevo sistema tenía que realizarse de manera estratégica y mediante la utilización de tecnología de punta que asegurará el éxito del proyecto, pero sobre todo, se tradujera en beneficios para mejorar el funcionamiento del Cecoban.

Un aspecto que se cuido mucho fue el emplear un software de nueva tecnología, por lo que empleamos Delphi con metodología orientada a objetos, dando como resultado front–ends amigables al usuario. Así como el empleo de un lenguaje de programación estructurado y modular, de fácil mantenimiento para las rutinas del back-end, aparte de que se contaba con la experiencia necesaria para su rápido desarrollo.

La selección de la herramienta Delphi versión 4.0, para el desarrollo del "Sistema de Compensación Electrónica de Cheques", garantiza su funcionalidad ya que dicha herramienta cumple con los estándares de programación vigentes, lo que permitirá al usuario final una sencilla operación del sistema, así como un fácil mantenimiento por parte del personal encargado de esta tarea.

En lo referente al sistema operativo optamos por Windows NT, dado que permite una mayor capacidad de usuarios que se pueden relacionar más fácilmente con el mismo por su interfaz amigable y por las ventajas que el sistema operativo representa, para su desenvolvimiento y conectividad futura en diversas plataformas.

Una de las características importantes del sistema es que no utiliza bases de datos, lo que elimina cualquier tipo de mantenimiento a las mismas, y más que desventaja se tiene una ventaja ya que los archivos que son generados pueden ser cargados a cualquier base de datos si se desea. La justificación de no usar las bases de datos, es el no tener información que sea acumulable por más de un día y no hacer uso de varias

llaves para búsquedas u ordenamiento de datos, además quisimos mantener el mismo tipo de archivo como es entregado por los bancos, para no pasarlo a base de datos y al final regresarlo a un archivo plano nuevamente.

Se pueden plantear otras opciones de solución para este sistema de compensación electrónica que use por ejemplo base de datos y una arquitectura cliente-servidor, aunque por el momento, la solución desarrollada en este trabajo cumple con las expectativas y funciona de manera adecuada y confiable.

Todo lo anterior pudo realizarse, debido a las herramientas empleadas para elaborar dicho sistema. La tecnología hoy en día, permite entender, por medio de objetos, el ambiente que rodea a un sistema antes de desarrollarlo o de mejorarlo, permitiendo su construcción de una manera más sencilla, rápida y flexible. Las técnicas orientadas a objetos, tienen como objetivo principal ayudar a la creación de sistemas más humanizados con los cuales el usuario se sienta más confiado y pueda tener la seguridad de trabajar en un ambiente amigable que le dé la confianza para desarrollar su trabajo de manera más placentera.

Nos dimos cuenta de la importancia de tener un área de sistemas actualizada, y más aún cuando se maneja información que requiere ser procesada rápidamente para obtener resultados más precisos y eficaces y que no pueden esperar al día siguiente. Y vencer el temor a emplear nuevas herramientas por desconocer su uso.

Esto proporciona una mejor visión a posibles actualizaciones futuras, ya que se han buscado las herramientas que permitan una migración de forma sencilla y segura, pensando en el próximo milenio.

Una ventaja que nos brinda el sistema desarrollado es que no es necesaria una capacitación especial para que el personal pueda operar el sistema. Además, el sistema anterior fue comprado en el extranjero y el personal tenía que viajar para recibir

la capacitación adecuada, con el sistema desarrollado, se eliminó la dependencia con el proveedor extranjero, ya que el software desarrollado cumple con todos las expectativas del cliente, como son rentabilidad, flexibilidad, eficiencia y productividad.

El haber participado en el desarrollo de esta aplicación, que permitió a Cecoban, disminuir en gran medida los costos de operación que representaba el sistema anterior, así como el incrementar la confiabilidad del proceso de Compensación Electrónica de Cheques, tan importante para el Sistema Financiero Mexicano, consideramos que resulto benéfico el esfuerzo realizado tanto para conseguir una mayor excelencia, como para la empresa que nos brindo las facilidades para desarrollar el presente trabajo, quien obtuvo un producto necesario para su operación.

El cliente quedó muy satisfecho, ha promovido más el servicio de compensación electrónica y ha aumentado por lo menos ocho veces los volúmenes de información procesados a nivel nacional, además de que ha reducido los costos de mantenimiento de hardware de $ 750.000 mensuales a $ 25.000 mensuales. Los ingresos producidos por este servicio han pasado de alrededor de $ 1,500,000 a $ 6,000,000 mensuales. Sin tomar en cuenta otros ahorros, por ejemplo en sistemas de aire acondicionado, energía eléctrica, fuentes ininterrumplibles de energía, espacio para las instalaciones y gastos de capacitación.

También este sistema da muchas opciones para la solución de contingencias. El sistema anterior no tenía un respaldo de hardware. Este sistema, ya que corre en PC, puede instalarse en cualquier lugar. En cada sucursal donde se instala se hace en dos PCs, una de las cuales es respaldo de la otra. Asimismo se puede tener un sistema instalado en un edificio fuera de la cámara de compensación y realizar ahí el proceso en caso de que por alguna razón no se pudiera tener acceso a las instalaciones de la cámara de compensación, como podría suceder en el caso de un desastre natural, una manifestación, un acto de terrorismo, etc.

Se puede concluir que con la experiencia adquirida al desarrollar este sistema se podría a su vez desarrollar un sistema de compensación automatizada que pudiera competir con otros sistemas existentes hechos en el extranjero.

En resumen podemos decir que se redujeron las ventanas de tiempo en todo el proceso, se redujeron los altos costos de mantenimiento, de consumibles, de insumos, de personal, además, actualmente ya se tienen equipos y sistemas de compensación electrónica de respaldo en pc´s, cumple con todas las especificaciones para operar antes durante y después del año 2000, es sencillo de operar y puede ser instalado en equipos pc`s lo que hizo posible el poderlo utilizar a nivel nacional, que para la empresa era un reto ya que debía aprovechar la apertura de nuevas plazas y dar el servicio, lo cual no hubiera sido posible e incosteable al usar los mainframes con los que contaba.

Un punto que es importante señalar como experiencia laboral y de cualquier proyecto, es el trabajo en equipo, esto nos permitió realizar el proyecto en menos tiempo y con mejores resultados, ya que todos los integrantes aportaron mayores y mejores ideas para un fin común, el trabajo en equipo conjunta la experiencia e ingenio de cada integrante logrando el éxito en el proyecto y, la satisfacción del cliente por el cumplimiento de los objetivos.

Al trabajar en equipo se puede abarcar más material en menos tiempo, lográndose con esto una mayor producción. Esto se puede aplicar a cualquier proyecto que requiera terminarse en un tiempo limitado.

El desarrollo de este sistema nos permitió aplicar los conocimientos adquiridos en nuestra formación académica, dando como resultado una aplicación que satisface una necesidad en beneficio de la comunidad bancaria, necesaria en el desarrollo económico del país.

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 en el campo de trabajo, requieren ser resueltos con una calidad y eficacia que permita seguir adelante con nuestro desempeño laboral.

Con respecto a la formación que la Facultad de Ingeniería nos brinda podría mejorarse si se incluyeran materias con herramientas comerciales actuales, o seminarios con proyectos que se usen en la industria, lo cual elevaría el nivel competitivo de sus egresados.

BIBLIOGRAFÍA

BIBLIOGRAFÍA

Ingeniería de Software
Roger S. Pressman
Ed. Mc. Graw Hill.

Análisis y Diseño de Sistemas
Kendall y Kendall
Ed. Prentice Hall

Contabilidad
Elias Lara Flores
Ed. Trillas

Metodología de la Programación Estructurada
Ma. Dolores Alonso
Silvia Romero
Ed. Paraninfo

Estructura Lógica y Diseño de Programas
Alan Cohen
Ed. Paraninfo

Lenguaje de Programación Estructurada y Modular
D.G.S.C.A.
Ed. UNAM

Turbo Pascal para Windows
Francisco Pascual González
Ed. Editorial Addison-Wesley Iberoamericana

El libro de Windows NT
Helen Custer
Ed. Mc. Graw Hill

Windows 95
Adrian King
Ed. Mc. Graw Hill

Structured design
Edward Yourdon & Larry Constantine
Ed. Yourdon Press.

Introduction to computers and data processing
Shelly & Cashman
Ed. Anaheim Publishing Company.

Selección de equipo de procesamiento electrónico de datos para el centro de compensación bancaria
Banco de México, Unidad de Sistemas 1977.

Especificaciones del sistema de compensación automatizada de cheques
Banco de México, 1979.

Estándares para el proceso automatizado de cheques
Asociación Mexicana de Bancos 1989

Manual de especificaciones de los caracteres E13-B
Deluxe Mexicana, S.A. de C.V. 1990

Delphi 4
Francisco Charte Ojeda
Ed. Anaya Multimedia

Aprendiendo Delphi en 14 dias
Osier, Batson y Groban
Ed. Borland Press

http:/www.datamodel.org/NormalizationRules.html
5 Rules of data normalization

http:/www.citilink.com/~jgarrick/vbasic/database/rdbms.html
The relational database model

http:/www.citilink.com/~jgarrick/vbasic/database/fundamentals.html
Fundamentals of database desing

http:/www.cs.sfu.ca./CC/354/zaiane/material/notes/chapter1/node7.html
The entity-relationship model

http:/www.island-data.com/downloads/papers/normalization.html
Normalization is a nice theory

http:/www.istis.unomaha.edu/cmit/bsad8030/chp47sld001.htm
Information and databases

http:/www.spea.indiana.edu/jgant/lect5_dataflow/sld001.htm
Process modeling

http:/www.hha.dk/businesscomputing/week09/slides_week09/sld005.htm
Data flow diagrams

http:/www.ie.eng.ua.edu/current_courses/ie4647ppt/IE464-5/sld001.htm
Data flow diagramming

http:/www.arrakis.es/~eb1fts/delphi/delphi.htm
Curso de Delphi

http:/members.home.net/dmartinez/development/delphi/s_tutor/indice.html
Tutor de Delphi

Apéndice A

Apéndice A.

Especificaciones del papel.

El propósito de establecer especificaciones del papel para cheques es el de asegurar que los documentos sean capaces de soportar adecuadamente los rigores del manejo manual y los sucesivos procesos de lectura y clasificación a que son sometidos.

El volumen de documentos que se maneja hoy por hoy requiere de procesos operativos eficientes. Esta eficiencia obliga, a su vez, al establecimiento de valores mínimos de resistencia mecánica y características físicas.

Adicionalmente a los valores proporcionados por la norma ANSI (American National Standard Institute) (Instituto Nacional de Estándares Americanos.), hay otras características en el papel cuya consideración es importante para el impresor de cheques.

Tipos de papel.

Se permite la utilización de los papeles comúnmente utilizados para la fabricación de formas, papeles bond, autocopiantes y otros papeles especiales, siempre que cumplan con las especificaciones señaladas.

El papel para cheques puede o no tener reactivo de seguridad, esto quiere decir que se le ha adicionado un agente reactivo, esta reacción es visible a simple vista y consiste en la aparición, ante la aplicación de hipoclorito de sodio, de una mancha de tono café o azul.

Marca de agua.

Es un logotipo, texto o imagen integrada al cuerpo del papel durante su fabricación, el cual es creado mediante variaciones en la concentración de las fibras de celulosa. Estas variaciones hacen que la marca de agua se haga visible al observar la hoja de papel a trasluz.

Papeles cubiertos.

El papel que con aplicación de resinas o arcillas en su superficie. o el papel al que se ha adherido películas plásticas o metálicas no debe ser utilizado en la fabricación de cheques.

Cuando el documento es cortado de tal forma que la orientación de sus fibras es paralela a la dimensión mayor del documento, se dice que el papel está "al hilo". Si está cortado de tal forma que la orientación de sus fibras es paralela a la dimensión menor, se dice que el papel está a "contra hilo". En la fabricación de cheques se recomienda la utilización "al hilo" del papel.

Partículas metálicas.

Se reconoce que la presencia de partículas magnetizables en la masa del papel puede interferir con la lectura de los caracteres E13-B.

Peso Base (Gramaje).

Es el peso, medido en gramos, de una hoja de papel con una superficie de un metro cuadrado.

El peso base del papel para cheques debe ser 90 gramos por metro cuadrado con tolerancia de ± 5%.

Lisura.

Es la resistencia que ofrece el papel al paso del aire a lo largo de la hoja. Las unidades designadas en la escala del instrumento Sheffield representan el número de centímetros cúbicos de aire por minuto que circula sobre una pulgada cuadrada de la superficie del papel.

Mientras más irregular es la superficie el papel. más alto es el flujo de aire y por lo tanto más alta es la medida de resistencia.

La lisura se relaciona con el premarcado y posmarcado de los caracteres magnetizables. También afecta la eficiencia en el manejo del documento, cuando este es transportado mecánicamente por el equipo de clasificación.

Espesor.

Es el grueso de la hoja de papel. La medida del espesor es importante, así como su rango de variación.

El espesor es una característica del papel relevante en la impresión de caracteres magnéticos.

Porosidad.

Es la resistencia del papel al paso del aire bajo una opresión específica como el tiempo promedio en segundos, que se requiere para desplazar 100 ML de aire a través de una pulgada cuadrada de papel bajo una presión de 4.88 pulgadas de agua.

Mientras más baja es la lectura, más poroso. o abierto, es el papel. Si el valor de la resistencia al paso del aire es excesivamente bajo, las máquinas lectoras que utilizan succión de aire para separar o transportar el documento puede llegar a ofrecer

dificultades, ya que es posible que, al ser succionados, los documentos se adhieran unos a otros, lo cual ocasiona una alimentación defectuosa o de más de un documento simultáneamente.

La porosidad es también una característica del papel relevante en la impresión de caracteres magnéticos.

Resistencia al levantamiento (cera).

Es la resistencia que ofrecen las fibras de papel a ser separadas de la masa, cuando se separa una gota de cera que ha sido previamente aplicada a la superficie y a la que las fibras se adhieren.

La resistencia al levantamiento es una característica del papel relevante en la impresión de caracteres magnéticos.

Coeficiente de fricción.

Es la resistencia relativa a la moción de dos hojas de papel que están en contacto entre sí. Una hoja de papel es adherida a una superficie lisa y horizontal. Sobre esta se coloca una segunda hoja de papel libremente y sobre ambas un contrapeso. El coeficiente de fricción en equivalente a la tangente del ángulo del plano inclinado en el cual el desplazamiento ocurre.

El coeficiente de fricción es una característica del papel importante para el posmarcado y el manejo mecánico de los documentos en las máquinas clasificadoras.

Reflectancia.

Es la reflectancia relativa de una superficie de papel iluminada, blanca o de color, tal como es vista por el ojo humano.

La reflectancia mínima aceptable es 60%.

Esta característica del papel es relevante para la lectura OCR de los documentos.

Opacidad.

Es la propiedad del papel que impide el paso a través suyo de los rayos de luz, no dejando ver los cuerpos que tienen detrás; la opacidad es la que determina el poder cubriente.

Esta característica del papel es relevante para la lectura OCR de los documentos.

Resistencia al rasgado.

Es la fuerza promedio, medida en gramos, para rasgar una hoja de papel, una vez que se ha iniciado un primer rasgado. Mientras más alta la lectura, más alto es la resistencia del papel al rasgado.

El rasgado es una medida de la resistencia física del papel y se relaciona con su habilidad para soportar el arranque, el paro y las altas velocidades de los sistemas transportadores de las máquinas clasificadoras.

Rigidez.

Es la resistencia a la flexión que el papel es capaz de soportar, en ambas direcciones, al ser defléctado por un pequeño péndulo.

La capacidad del documento para ser manejado por el equipo clasificador esta relacionada con su rigidez. El papel con valores excesivamente bajos de rigidez ocasionará atoramiéntos en el equipo clasificador. El valor de rigidez es menor a contra hilo. Esta es la razón por la que los documentos a contra hilo, documentos donde el hilo del papel esta paralelo a la medida corta del documento, requieren de consideraciones especiales para asegurar una correcta rigidez.

Resistencia a la tensión

Es la resistencia que ofrece una probeta de papel hasta el punto de rotura, cuando es sometida a un movimiento de tensión por sus extremos.

Mientras más alta es la lectura, más resistente es el papel a la tensión.

Unidades de medida utilizadas en las especificaciones de papel.

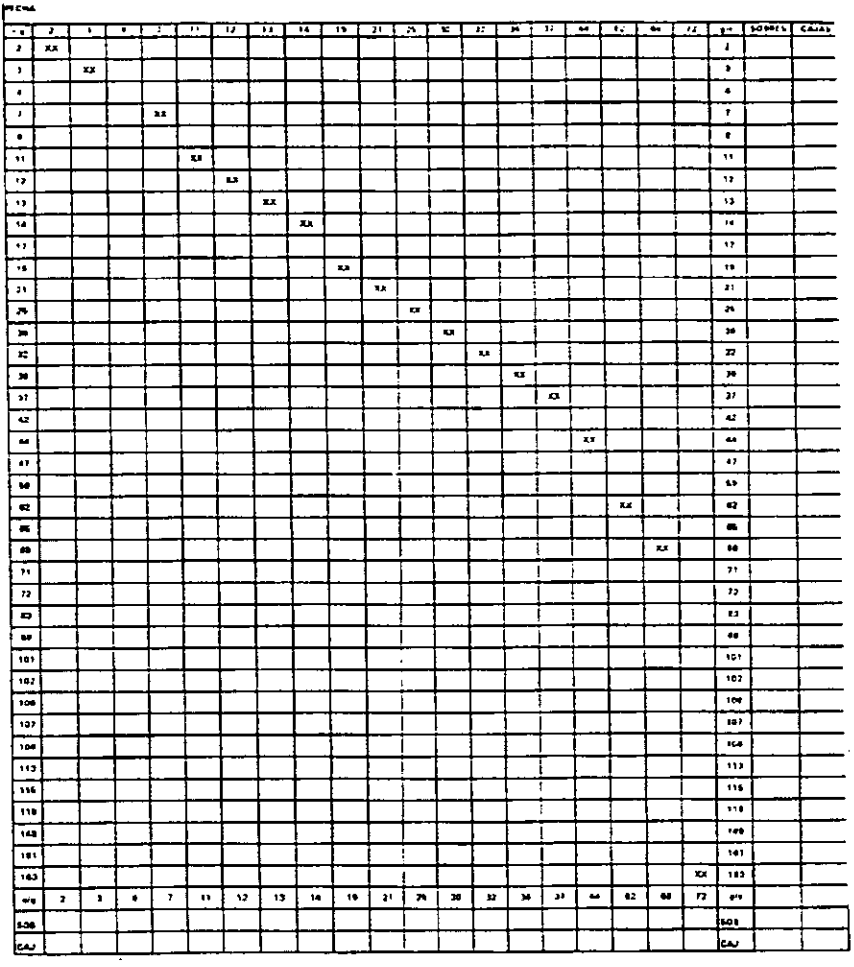
Lisura	Unidades Sheffield
Espesor	Milésima de pulgada
Peso base (Gramaje)	Gramos por metro cuadrado
Porosidad	Segundos en densitómetro Gurley Hill 100 ml
Resistencia al rasgado	Gramos Elmendorf
Resistencia a la tensión	Kilográmetros probeta de 15 mm.
Rigidez Taber	Gramos fuerza por centímetro
Reflectancia	Porcentaje
Blancura	Porcentaje
Opacidad	Porcentaje
Resistencia al levantamiento superficial	Núm Dennison

Apéndice B

Apéndice B

Reporte en Excel de la matriz de distribución

MATRIZ DE DISTRIBUCION
T. E. I.

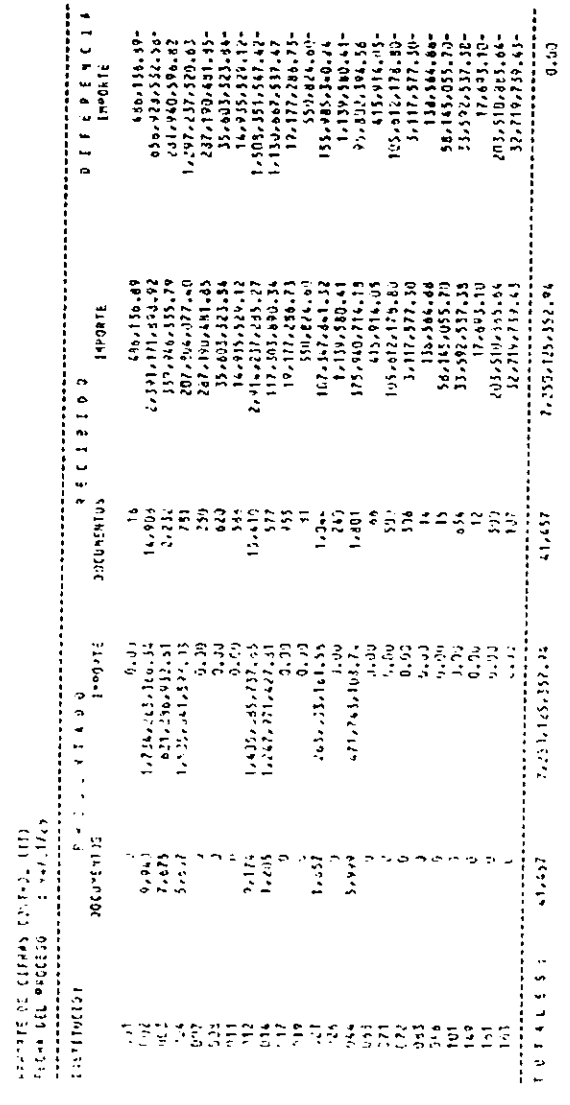


Acuse de recibo en Excel utilizado por la cámara de compensación:

ACUSE DE RECIBO			
BANCO		NOMBRE	FIRMA
001	BANXICO		
002	BANAMEX		
003	SERFIN		
006	BANCOMEXT		
007	CITIBANK		
011	CONFIA		
012	BANCOMER		
013	INDUSTRIAL		
014	MEXICANO		
017	BBV		
019	BANEFA		
021	BITAL		
025	SURESTE		
030	BAJIO		
032	IXE		
036	INBURSA		
037	INTERACCIONES		
042	MIFEL		
044	INVERLAT		
047	PROMOTOR		
058	BANREGIO		
059	INVEX		
062	AFIRME		
068	PROMEX		
072	BANORTE		
102	AMRO		
106	BOFA		
107	BOSTON		
108	TOKIO		
113	DRESNER		
115	FUJI		
119	REPUBLIC		
135	NAFINSA		
149	BANRURAL		
161	BANCRECER		

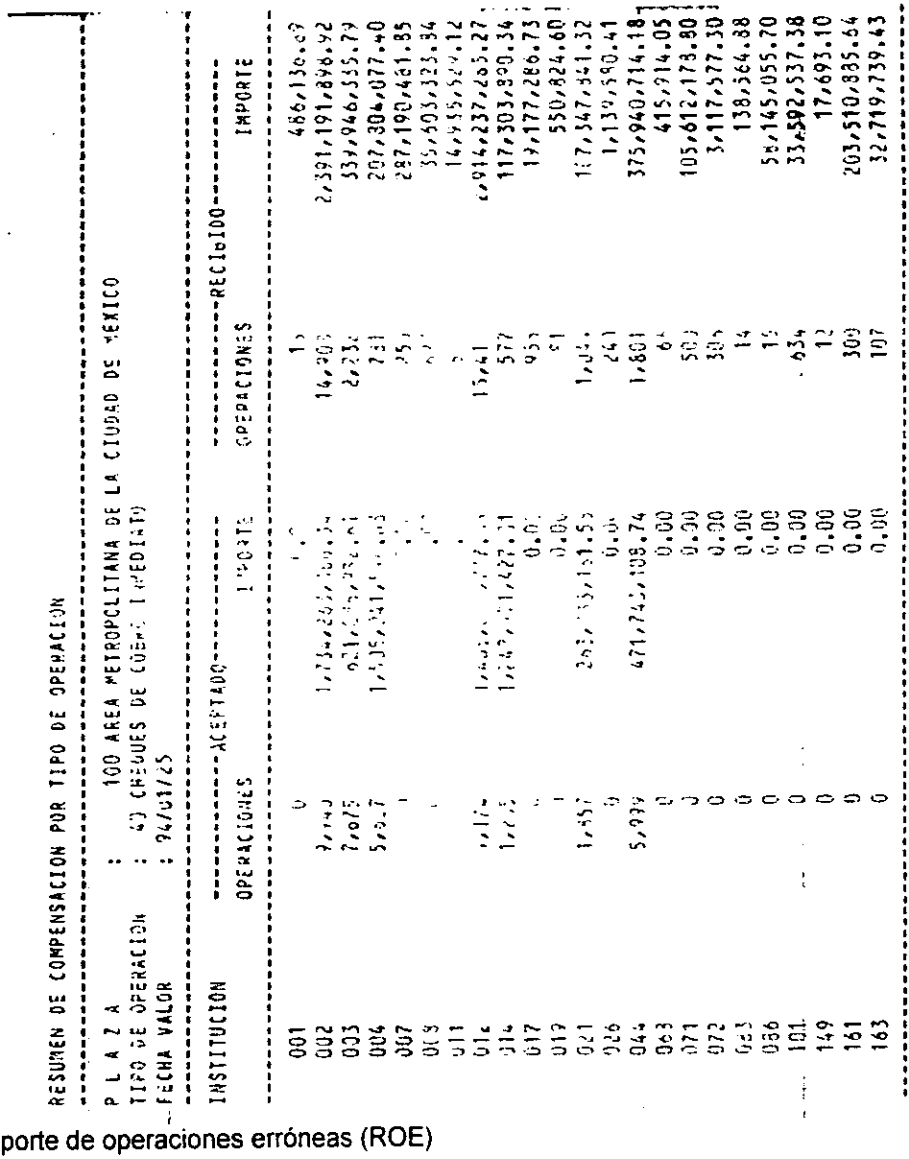
Reporte de cifras de control (TT).

Este reporte detalla el número de operaciones presentadas y recibidas por cada banco, la información se obtiene a partir de los archivos de salida.



Hoja resumen de compensación:

Este reporte detalla el número de operaciones presentadas y recibidas por cada banco, la información se obtiene a partir de los archivos de entrada.



RESUMEN DE COMPENSACION POR TIPO DE OPERACION

P L A Z A : 100 AREA METROPOLITANA DE LA CIUDAD DE MEXICO
TIPO DE OPERACION : 40 CHEQUES DE CUENTA INDEPENDIENTE
FECHA VALOR : 94/01/25

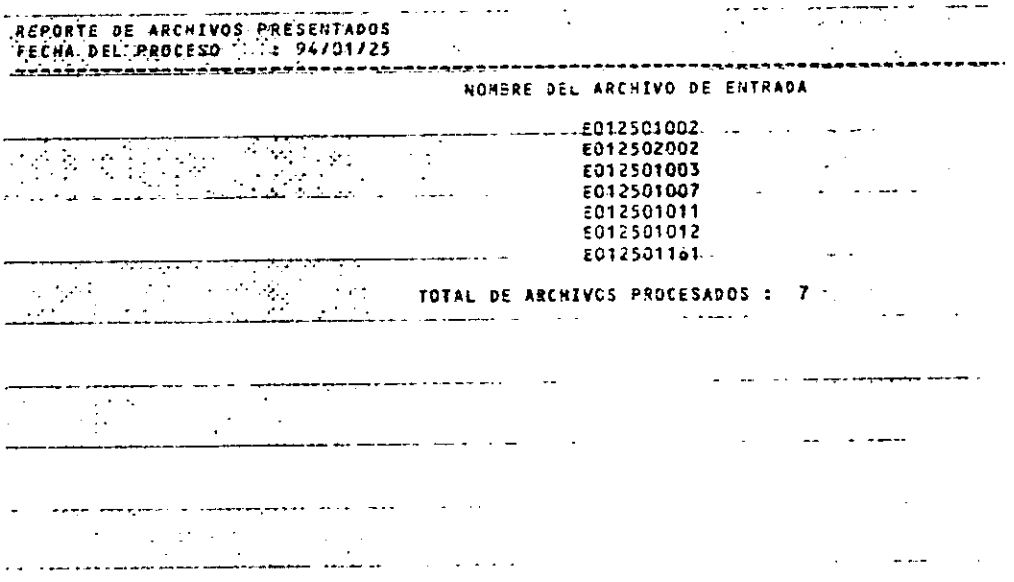
INSTITUCION	ACEPTADO		RECIBIDO	
	OPERACIONES	IMPORTE	OPERACIONES	IMPORTE
001	0	[illegible]	15	486,136.69
002	7,[illegible]	[illegible]	14,[illegible]	2,391,191,898.92
003	7,[illegible]	[illegible]	2,[illegible]	533,946,555.79
004	5,[illegible]	[illegible]	7[illegible]	207,304,077.40
007	1	[illegible]	[illegible]	287,190,461.85
009	[illegible]	[illegible]	[illegible]	35,[illegible]23.34
011	[illegible]	[illegible]	[illegible]	14,[illegible]
014	[illegible]	[illegible]	15,441	2,914,237,[illegible]
016	1,[illegible]	[illegible]	577	117,303,890.34
017	[illegible]	0.00	953	13,177,286.73
019	1	0.00	51	550,824.60
021	1,457	2[illegible]	1,[illegible]	167,347,[illegible]
026	0	0.00	241	1,139,[illegible]
044	5,939	471,[illegible]	1,801	375,940,714.18
063	0	0.00	6[illegible]	415,914.05
071	3	0.00	503	105,612,178.80
072	0	0.00	385	3,117,577.30
083	0	0.00	14	138,364.88
086	0	0.00	15	58,145,055.70
101	0	0.00	634	33,592,537.38
149	0	0.00	12	17,693.10
161	0	0.00	309	203,510,885.64
163	0	0.00	107	32,719,739.43
T O T A L E S :	41,457	7,250,125,352.34	41,497	7,250,125,352.34

Reporte de operaciones erróneas (ROE)

Este reporte contiene el detalle de los errores que tuvieron los registros en los archivos que presento cada banco, también detalla el número de operaciones e importe rechazados.

Reporte de archivos procesados.

Este reporte indica el número y nombre de los archivos que fueron procesados.



REPORTE DE ARCHIVOS PRESENTADOS
FECHA DEL PROCESO : 94/01/25

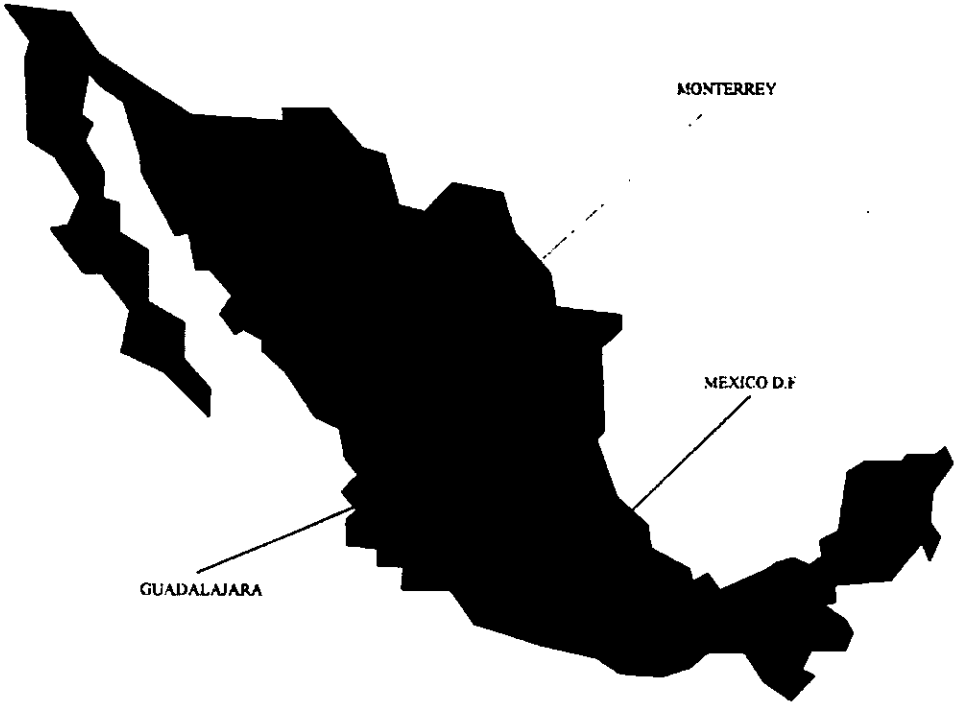
NOMBRE DEL ARCHIVO DE ENTRADA

E012501002
E012502002
E012501003
E012501007
E012501011
E012501012
E012501161

TOTAL DE ARCHIVOS PROCESADOS : 7

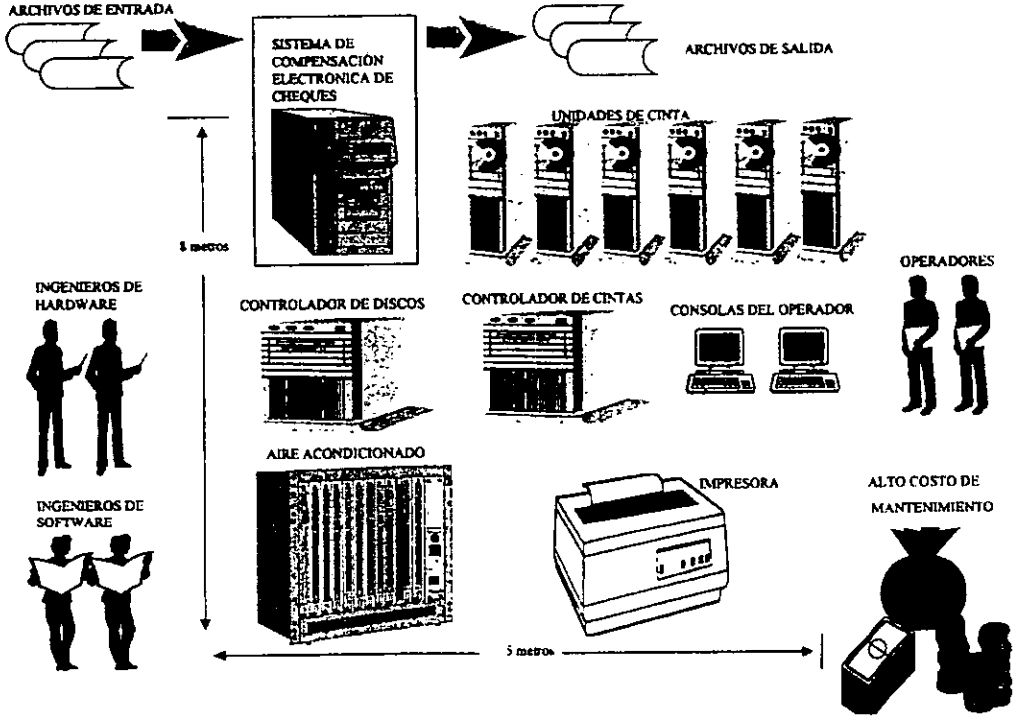
El proceso de compensación electrónica solamente se realiza en México, Monterrey y Guadalajara.

SITUACIÓN ACTUAL
(COBERTURA)



Se requiere de instalaciones amplias, adecuadas y de personal calificado.

SITUACION ACTUAL



Apéndice C

COMPENSACION ELECTRONICA DE CHEQUES

REGISTRO ENCABEZADO DE ENTRADA "Ee"

	ZONA TEI													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	TIPO DE REGISTRO	NUMERO DE SECUENCIA	INSTITUCION EMISORA	SENTIDO	CPI	SERVICIO TEI	NUMERO DE TRANSFERENCIA	ESPACIO IDT	FECHA DE PRESENTACION	USO FUTURO TEI	TIPO DE ARCHIVO	CODIGO DE FORMATO	TIPO DE MONEDA	USO FUTURO BANCO
	"01"	VARIABLE	NUMERO DE TRANSITO EELLLBBBO	"E"	VARIABLE	"I"	VARIABLE DONN	VARIABLE	VARIABLE AAMMDD	VARIABLE	VARIABLE	VARIABLE	VARIABLE	VARIABLE
	NUMERICO	NUMERICO	NUMERICO	ALFANUMERICO	NUMERICO	NUMERICO	NUMERICO	ALFANUMERICO	NUMERICO	ALFANUMERICO	NUMERICO	NUMERICO	NUMERICO	ALFANUMERICO
	2	6	9	1	2	1	4	3	6	15	1	1	1	66
	1-2	3-8	9-17	18-18	19-20	21-21	22-25	26-28	29-34	35-49	50-50	51-51	52-52	53-120

COMPENSACION ELECTRONICA DE CHEQUES

REGISTRO ENCABEZADO DE SALIDA "Es"

	ZONA TEI								ZONA BANCO			
1	2	3	4	5	6	7	8	9	10	11	12	13
TIPO DE REGISTRO	NUMERO DE SECUENCIA	INSTITUCION RECEPTORA	SENTIDO	SISTEMA TEI	SERVICIO TEI	NUMERO DE TRANSFERENCIA	FECHA DE GENERACION	USO FUTURO TEI	TIPO DE ARCHIVO	CODIGO DE FORMATO	TIPO DE MONEDA	USO FUTURO BANCO
"09"	"000001"	NUMERO DE TRANSITO EELLLBBBD	"S"	"01" "02" "03"	"I"	VARIABLE DDNNNNNN	VARIABLE AAMMDD	ESPACIOS	VARIABLE	VARIABLE	VARIABLE	ESPACIOS
NUMERICO	NUMERICO	NUMERICO	ALFANUMERICO	NUMERICO	NUMERICO	NUMERICO	NUMERICO	ALFANUMERICO	NUMERICO	NUMERICO	NUMERICO	ALFANUMERICO
2	6	9	1	2	1	7	6	15	1	1	1	68
1-2	3-8	9-17	18-18	19-20	21-21	22-28	29-34	35-49	50-50	51-51	52-52	53-120

COMPENSACION ELECTRONICA DE CHEQUES

REGISTRO DE DETALLE "D"

	ZONA TEI						
CAMPO	1	2	3	4	5	6	7
	TIPO DE REGISTRO	NUMERO DE SECUENCIA	CODIGO DE OPERACIÓN	FECHA DE TRANSFERENCIA	INST. EMISORA (CEDENTE)	INSTITUCION RECEPTORA (GRADO)	IMPORTE
	"02"	VARIABLE	"40"	VARIABLE AAMMDD	NUMERO DE TRANSITO EELLLBBBD	NUMERO DE TRANSITO EELLLBBBD	VARIABLE
	NUMERICO	NUMERICO	NUMERICO	NUMERICO	NUMERICO	NUMERICO	NUMERICO
	2	6	2	6	9	9	15
	1-2	3-8	9-10	11-16	17-25	26-34	35-49

	ZONA BANCO												
CAMPO	8	9	10		11	12		13				14	15
	REFERENCIA DE LOCALIZACION		UNIDAD DE PREPARACION			CAMPO A NUESTRO CARGO							
	INSTITUCION PREPARADORA	FECHE DE PREPARACION	LOTE DE SALIDA	SECUENCIA DE SALIDA	CLAVE DE TRANSACCION	NO. DE CTA.	NO. CHEQUE	DIGITO DE INTERCAMBIO	DIGITO DE PREMARCADO	CODIGO DE SEGURIDAD	INFORMACION ADICIONAL	MOTIVO DE DEVOLUCION	FECHA DE TRANSFERENCIA INICIAL
	NUMERO DE TRANSITO EELLLBBB	VARIABLE MMDD	VARIABLE	VARIABLE	VARIABLE	VARIABLE	VARIABLE	VARIABLE	VARIABLE	VARIABLE	ESPACIOS	VARIABLE	VARIABLE AAMMDD
	NUMERICO	NUMERICO	NUMERICO	NUMERICO	NUMERICO	NUMERICO	NUMERICO	ALFANUMERICO	ALFANUMERICO	ALFANUMERICO	ALFANUMERICO	NUMERICO	NUMERICO
	8	4	7	4	2	13	10	1	1	3	10	2	6
POSICION	50-57	58-61	62-68	69-72	73-74	75-87	88-97	98-98	99-99	100-102	103-112	113-114	115-120

COMPENSACION ELECTRONICA DE CHEQUES

REGISTRO SUMARIO "S"

	1	2	3	4	5	6	7	8
	TIPO DE REGISTRO	NUMERO DE SECUENCIA	CODIGO DE OPERACIÓN	FECHA DE TRANSFERENCIA	TOTAL REGISTROS	IMPORTE	USO FUTURO TEI	USO FUTURO BANCO
	"09"	VARIABLE	"40"	VARIABLE	VARIABLE	VARIABLE	VARIABLE	VARIABLE
	NUMERICO	NUMERICO	NUMERICO	NUMERICO	NUMERICO	NUMERICO	ALFANUMERICO	ALFANUMERICO
	2	6	2	6	5	18	11	71
	1-2	3-8	9-10	11-16	17-21	22-39	40-50	51-120

Apéndice D

```
program Project1;

uses
  Forms,
  Unit1 in 'Unit1.pas' {Form1},
  Unit2 in 'Unit2.pas' {Form2},
  Datos in 'Datos.pas',
  Inicia in 'Inicia.pas',
  Unit3 in 'Unit3.pas' {Form3},
  Unit4 in 'Unit4.pas' {Form4},
  Unit5 in 'Unit5.pas' {Form5},
  Unit6 in 'Unit6.pas' {Form6},
  Unit7 in 'Unit7.pas' {Form7},
  pantalla3 in 'pantalla3.pas' {Form8},
  Unit9 in 'Unit9.pas' {Form9},
  Unit10 in 'Unit10.pas' {Form10},
  Unit11 in 'Unit11.pas' {Form11},
  Unit12 in 'Unit12.pas' {Form12},
  Unit13 in 'Unit13.pas' {Form13},
  Unit14 in 'Unit14.pas' {Form14},
  Unit15 in 'Unit15.pas' {Form15},
  Unit16 in 'Unit16.pas' {Form16},
  Unit17 in 'Unit17.pas' {Form17},
  Unit18 in 'Unit18.pas' {Form18},
  PaswUnit in 'Paswunit.pas',
  Unit19 in 'Unit19.pas' {Form19},
  Unit20 in 'Unit20.pas' {Form20},
  Unit21 in 'Unit21.pas' {Form21},
  Unit22 in 'Unit22.pas' {Form22},
  Unit23 in 'Unit23.pas' {Form23},
  Cuentas2 in 'Cuentas2.pas',
  Calculos in 'Calculos.pas',
  CargaDat in 'Cargadat.pas',
  Datos3 in 'Datos3.pas',
  Unit24 in 'Unit24.pas' {Form24},
  Imprime2 in 'Imprime2.pas',
  Disper4 in 'Disper4.pas',
  FmxUtils in 'FMXUtils.pas';

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.CreateForm(TForm2, Form2);
  Application.CreateForm(TForm3, Form3);
  Application.CreateForm(TForm4, Form4);
  Application.CreateForm(TForm5, Form5);
  Application.CreateForm(TForm6, Form6);
  Application.CreateForm(TForm7, Form7);
  Application.CreateForm(TForm8, Form8);
```

```
  Application.CreateForm(TForm9, Form9);
  Application.CreateForm(TForm10, Form10);
  Application.CreateForm(TForm11, Form11);
  Application.CreateForm(TForm12, Form12);
  Application.CreateForm(TForm13, Form13);
  Application.CreateForm(TForm14, Form14);
  Application.CreateForm(TForm15, Form15);
  Application.CreateForm(TForm16, Form16);
  Application.CreateForm(TForm17, Form17);
  Application.CreateForm(TForm18, Form18);
  Application.CreateForm(TForm19, Form19);
  Application.CreateForm(TForm20, Form20);
  Application.CreateForm(TForm21, Form21);
  Application.CreateForm(TForm22, Form22);
  Application.CreateForm(TForm23, Form23);
  Application.CreateForm(TForm24, Form24);
  form2.hide;
  Application.Run;
end.
```

```
unit Unit2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ComCtrls, Buttons, ExtCtrls;

type
  TForm2 = class(TForm)
    DateTimePicker1: TDateTimePicker;
    DateTimePicker2: TDateTimePicker;
    Label1 : TLabel;
    Label2 : TLabel;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    Image1: TImage;
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn1MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure BitBtn2MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

implementation
uses datos, Unit1, pantalla3;

{$R *.DFM}

procedure TForm2.BitBtn1Click(Sender: TObject);
Var
   year,month,day:word;
   strmonth,strday : String[2];
begin
  DecodeDate(DateTimePicker1.date, Year, Month, Day);
  strmonth:=IntToStr(Month);
  strday:=IntToStr(Day);
  if length(strday) = 1 Then
     strday:='0'+strday;
  if length(strmonth) = 1 Then
     strmonth:='0'+strmonth;
  CFechaPanel := IntToStr(Year)+'/'+Strmonth+'/'+strday;
  CFecha := IntToStr(Year)+Strmonth+strday;
```

```
  DecodeDate(DateTimePicker2.date, Year, Month, Day);
  strmonth:=IntToStr(Month);
  strday:=IntToStr(Day);
  if length(strday) = 1 Then
     strday:='0'+strday;
  if length(strmonth) = 1 Then
     strmonth:='0'+strmonth;
  FechaDepPanel := IntToStr(Year)+'/'+Strmonth+'/'+strday;
  FechaDep := IntToStr(Year)+Strmonth+strday;
  Form2.hide;
  Form8.Panel2.Caption := CFechaPanel;
  Form8.Panel3.Caption := FechaDepPanel;
  Form8.show;
end;

procedure TForm2.BitBtn2Click(Sender: TObject);
begin
  Form2.close;
  Form1.show;
end;

procedure TForm2.BitBtn1MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   BitBtn1.Cursor := crHandPoint;
end;

procedure TForm2.BitBtn2MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   BitBtn2.Cursor := crHandPoint;
end;

end.
```

```
unit Unit3;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Grids, Outline, DirOutln, FileCtrl, Buttons;

type
  TForm3 = class(TForm)
    DriveComboBox1: TDriveComboBox;
    DirectoryOutline1: TDirectoryOutline;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    procedure Button2Click(Sender: TObject);
    procedure DriveComboBox1Change(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form3: TForm3;

implementation

uses Unit1, Unit12, Unit13;

{$R *.DFM}

procedure TForm3.Button2Click(Sender: TObject);
begin
   Form3.Hide;
   Form3.Close;
   If Form3.Caption = 'Directorio para realizar el respaldo' Then
      Application.MessageBox('Respaldo en disco cancelado',
                    'Información',mb_Ok+mb_IconInformation)    { produce la caja de diálogo }
   Else
      Application.MessageBox('Generación de saldos cancelada',
                    'Información',mb_Ok+mb_IconInformation);   { produce la caja de diálogo }
end;

procedure TForm3.DriveComboBox1Change(Sender: TObject);
begin
   Try
      DirectoryOutline1.Drive := DriveComboBox1.Drive;
   Except
```

```
      Form12.Caption := 'Error';
      Form12.label1.Caption := '        Error de acceso';
      Form12.Showmodal;
      Form12.Caption := 'Información';
   End;
end;

procedure TForm3.Button1Click(Sender: TObject);
begin
   If Form3.Caption = 'Directorio para realizar el respaldo' Then
      Form1.Panel3.Caption := ' Estado : Respaldando...'
   Else
      Form1.Panel3.Caption := ' Estado : Generando archivo...';
   Form1.Panel3.Refresh;
   Form3.Hide;
   Form3.Close;
   If Form3.Caption = 'Directorio para realizar el respaldo' Then
      {realiza el respaldo...}
   Else
      {realiza la generación del archivo de saldos};
   Form1.Panel3.Caption := '';
   Form1.Panel3.Refresh;
   If Form3.Caption = 'Directorio para realizar el respaldo' Then
      Application.MessageBox('Terminó el respaldo',
                  'Información',mb_Ok+mb_IconInformation)   { produce la caja de diálogo }
   Else
      Application.MessageBox('Terminó la generación de saldos',
                  'Información',mb_Ok+mb_IconInformation);  { produce la caja de diálogo }
   Form1.Panel3.Caption := ' Estado : En espera de alguna opción';
   Form1.Panel3.Refresh;
end;

procedure TForm3.BitBtn1Click(Sender: TObject);
begin
   If Form3.Caption = 'Directorio para realizar el respaldo' Then
      Form13.Panel1.Caption := 'Respaldando'
   Else
      Form13.Panel1.Caption := 'Generando';
   Form13.Panel1.Refresh;
   Form3.Hide;
   Form3.Close;
   If Form3.Caption = 'Directorio para realizar el respaldo' Then
      {realiza el respaldo...}
   Else
      {realiza la generación del archivo de saldos};
   Form13.Panel1.Caption := '';
   Form13.Panel1.Refresh;
   If Form3.Caption = 'Directorio para realizar el respaldo' Then
      Begin
         Form12.label1.Caption := '       Terminó el respaldo';
         Form12.showmodal;
```

```
      End
   Else
      Begin
         Form12.label1.Caption := 'Termina producción de saldos';
         Form12.showmodal;
      End;
end;

procedure TForm3.BitBtn2Click(Sender: TObject);
begin
   Form3.Hide;
   Form3.Close;
   If Form3.Caption = 'Directorio para realizar el respaldo' Then
      Begin
         Form12.label1.Caption := 'Respaldo en disco cancelado';
         Form12.showmodal;
      End
   Else
      Begin
         Form12.label1.Caption := 'Generación de saldos cancelada';
         Form12.showmodal;
      End;
end;

end.
```

```
unit Unit15;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
  TForm15 = class(TForm)
    Image2: TImage;
    Image3: TImage;
    Image4: TImage;
    Image5: TImage;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    CheckBox1: TCheckBox;
    procedure Image1MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image2MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image3MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image4MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image5MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure FormMouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image3Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Image2Click(Sender: TObject);
    procedure CheckBox1MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form15: TForm15;

implementation

uses Unit5, Unit10, Unit13, FMXUtils,
     Datos, Datos3, CargaDat;

{$R *.DFM}
```

```
procedure TForm15.Image1MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Siguiente';
end;

procedure TForm15.Image2MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Anterior';
end;

procedure TForm15.Image3MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Terminar';
end;

procedure TForm15.Image4MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Diskette';
end;

procedure TForm15.Image5MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Cinta';
end;

procedure TForm15.FormMouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := '';
end;

procedure TForm15.Image3Click(Sender: TObject);
begin
   if form10.showmodal = mrOK Then  { produce la caja de diálogo }
     begin
      Application.Terminate;
     end;
end;

procedure TForm15.FormClose(Sender: TObject; var Action: TCloseAction);
begin
   if form10.showmodal = mrOK Then  { produce la caja de diálogo }
      Application.Terminate
```

```
   Else
      Action := caNone;
end;

procedure TForm15.Image2Click(Sender: TObject);
begin
   Form15.Hide;
   Form13.Panel2.Caption := CFechaPanel;
   Form13.Panel3.Caption := FechaDepPanel;
   Form13.Show;
end;

procedure TForm15.CheckBox1MouseMove(Sender: TObject; Shift: TShiftState;
  X, Y: Integer);
begin
   Panel1.Caption := 'Compactado';
end;


Procedure ASuCargoEnDiskette(Cadaux : String; OpcionCompactar : ShortInt);
Var
   Directorio : String;
Begin
   Directorio := 'SALIDAS\';
   If OpcionCompactar = 1 Then
      Directorio := 'SALIDAS.ZIP\';
   If (ExisteArchivo(MasterDir+'SALIDAS\'+'S'+SistemaTEI+Cadaux+'A1.'+Copy(CFecha,7,2)+NumeroProcesoStr)) And
      (TamanioArchivo > 260) Then
      Begin
         CopyFile(MasterDir+Directorio+'S'+SistemaTEI+Cadaux+'A1.'+
                  Copy(CFecha,7,2)+NumeroProcesoStr,
                  'A:\'+'S'+SistemaTEI+Cadaux+'A1.'+ Copy(CFecha,7,2)+NumeroProcesoStr);
         CmdLine := 'A:\000001.TMP';
         DeleteFile(CmdLine);
      End
End;



procedure TForm15.Image4Click(Sender: TObject);
begin
   Form5.showmodal;
   If Cadaux <> '000' Then
      Begin
         If DriveReady('A:000001.TMP') Then
            Begin
               If form15.CheckBox1.Checked = False Then
                  ASuCargoEnDiskette(Cadaux, 0) { 0 = no compactado }
               Else
                  ASuCargoEnDiskette(Cadaux, 1); { 1 = compactado }
            End
```

```
        End;
end;

end.
```

```
unit Unit14;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls;

type
  TForm14 = class(TForm)
    Image6: TImage;
    Image5: TImage;
    Image4: TImage;
    Image2: TImage;
    Image3: TImage;
    Panel1: TPanel;
    Panel3: TPanel;
    Image7: TImage;
    Image8: TImage;
    Panel2: TPanel;
    Image9: TImage;
    Image10: TImage;
    Image11: TImage;
    Image12: TImage;
    Image13: TImage;
    procedure Image1MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image2MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image3MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image4MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image5MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image6MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image7MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image8MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image9MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image10MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image11MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image12MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image13MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
```

```
    procedure FormMouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Image2Click(Sender: TObject);
    procedure Image3Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Image10Click(Sender: TObject);
    procedure Image11Click(Sender: TObject);
    procedure Image12Click(Sender: TObject);
    procedure Image6Click(Sender: TObject);
    procedure Image5Click(Sender: TObject);
    procedure Image7Click(Sender: TObject);
    procedure Image13Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form14: TForm14;

implementation

{$R *.DFM}

Uses Datos, Imprime2, Disper4, Calculos,
     Unit13, Unit10, Unit17;

procedure TForm14.Image1MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Siguiente';
end;

procedure TForm14.Image2MouseMove(Sender: TObject; Shift: TShiftState; X,
                          Y: Integer);
Begin
   Panel1.Caption := 'Anterior';
End;

procedure TForm14.Image3MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Terminar';
end;

procedure TForm14.Image4MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Carta';
end;
```

```
procedure TForm14.Image5MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Resumen';
end;

procedure TForm14.Image6MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Póliza';
end;

procedure TForm14.Image7MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Control';
end;

procedure TForm14.Image8MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Entradas';
end;

procedure TForm14.Image9MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'Matriz';
end;

procedure TForm14.Image10MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'HCOM';
end;

procedure TForm14.Image11MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'ROE';
end;

procedure TForm14.Image12MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := 'ROEC';
end;

procedure TForm14.Image13MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
```

```
begin
   Panel1.Caption := 'Acuse';
end;

procedure TForm14.FormMouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
   Panel1.Caption := '';
end;

procedure TForm14.Image2Click(Sender: TObject);
begin
   Form14.Hide;
   Form13.Panel2.Caption := CFechaPanel;
   Form13.Panel3.Caption := FechaDepPanel;
   Form13.Show;
end;

procedure TForm14.Image3Click(Sender: TObject);
begin
   if form10.showmodal = mrOK Then  { produce la caja de diálogo }
     begin
      Application.Terminate;
     end;
end;

procedure TForm14.FormClose(Sender: TObject; var Action: TCloseAction);
begin
   if form10.showmodal = mrOK Then  { produce la caja de diálogo }
      Application.Terminate
   Else
      Action := caNone;
end;

procedure TForm14.Image10Click(Sender: TObject);
begin
   Form14.Hide;
   Form17.Panel2.Caption := CFecha;
   Form17.Panel3.Caption := FechaDep;
   Form17.Caption := 'Reportes/Hoja de compensación';
   Form17.Show;
end;

procedure TForm14.Image11Click(Sender: TObject);
begin
   Form14.Hide;
   Form17.Panel2.Caption := CFecha;
   Form17.Panel3.Caption := FechaDep;
   Form17.Caption := 'Reportes/Anomalías presentadas';
   Form17.Show;
end;
```

```
procedure TForm14.Image12Click(Sender: TObject);
begin
   Form14.Hide;
   Form17.Panel2.Caption := CFecha;
   Form17.Panel3.Caption := FechaDep;
   Form17.Caption := 'Reportes/Anomalías recibidas';
   Form17.Show;
end;


procedure TForm14.Image6Click(Sender: TObject);
begin
   Dispersa;
   CalculaTotalArreglo;
   If (TotalImpre <> 0) Or (TotalImrec <> 0) Then
      ReporteImpresoPoliza;
end;


procedure TForm14.Image5Click(Sender: TObject);
begin
   Dispersa;
   CalculaTotalArreglo;
   If (TotalImpre <> 0) Or (TotalImrec <> 0) Then
      ReporteImpresoResumen;
end;


procedure TForm14.Image7Click(Sender: TObject);
begin
   Dispersa;
   ImprimeTT;
   CifrasRoe;
end;

procedure TForm14.Image13Click(Sender: TObject);
begin
   Dispersa;
   ImprimeAcuseDeRecibo;
end;

end.
```

```
Unit Calculos;

Interface

Uses Datos, Datos3, Datos4;


   Procedure IniciaArreglo;
   Procedure CargaFusionados;
   Procedure CalculaTotalArreglo;
   Procedure CargaTodoUnidoArchivos;
   Procedure CargaTodoUnidoReportes;
   Procedure SumaDatosAceptado(j : Integer);
   Procedure CalculaTotalBanco(i : Integer);
   Procedure CalculaPresentadoB1aB2(i, j : Integer);
   Procedure FormateaParteEntera(Var ParteEntera : String);


Implementation


Procedure IniciaArreglo;
Var
   i, j : Integer;
Begin
   For i := NUM_BANCOS+1 To MAX_ARREGLO Do
         ArregloBancosCuentas.Buffer[i].Banco.NumeroBanco := 0;
   For i := 1 To MAX_ARREGLO Do
      Begin
         ArregloBancosCuentas.Buffer[i].Banco.NumeroSec := 1;
         ArregloBancosCuentas.Buffer[i].Banco.ImporteSumario := 0.00;
      End;
End;




Procedure CargaFusionados;
Var
   i, j : Integer;
   IndiceFusionados : Integer;
   BancoAbsorbido   : String [3];
   LineaFusionados  : String;
Begin
   NUM_FUSIONES := 0;
   For i := 1 To MAX_BANCOS Do
      For j := 1 To MAX_FUSIONADOS Do
         Fusiones[i,j] := '000';
   Assign(Fusionados,MasterDir+'FUSION.DAT');
```

```
   Reset(Fusionados);
   IndiceFusionados := 0;
   While Not EOF (Fusionados) Do
      Begin
         NUM_FUSIONES := NUM_FUSIONES + 1;
         IndiceFusionados := IndiceFusionados + 1;
         ReadLn(Fusionados,LineaFusionados);
         Fusiones[IndiceFusionados,1] := Copy(LineaFusionados,1,3);
         For j := 2 To MAX_FUSIONADOS Do
            Begin
               BancoAbsorbido :=Copy(LineaFusionados,(j-1)*3+1,3);
               If BancoAbsorbido <> '000' Then
                  Fusiones[IndiceFusionados,j] := BancoAbsorbido;
            End;
      End; {While not}
Close(Fusionados);
End;



Procedure CargaTodoUnidoArchivos;
Var
   i, j                : Integer;
   BancosChicos        : Text;
   LineaBancosChicos   : String;
   LineaFusionados     : String;
Begin
   CargaFusionados;
   Assign(BancosChicos,MasterDir+'BANCOCHI.DAT');
   For i := 1 To MAX_BANCOS Do
      TodoUnido[i] := 0;
   For j := 1 To NUM_FUSIONES Do
      Begin
         Reset(BancosChicos);
         While Not EOF(BancosChicos) Do
            Begin
{              LineaBancosChicos[8] := ' ';}
               ReadLn(BancosChicos, LineaBancosChicos);
               If Fusiones[j,1] = Copy(LineaBancosChicos,1,3) Then
                  If LineaBancosChicos[8] = 'A' Then
                     TodoUnido[j] := 1;
            End;
      End;
   Close(BancosChicos);
End;
```

```
Procedure CargaTodoUnidoReportes;
Var
   i, j                : Integer;
   BancosChicos        : Text;
   LineaBancosChicos   : String;
   LineaFusionados     : String;
Begin
   CargaFusionados;
   Assign(BancosChicos,MasterDir+'BANCOCHI.DAT');
   For i := 1 To MAX_BANCOS Do
      TodoUnido[i] := 0;
   For j := 1 To NUM_FUSIONES Do
      Begin
         Reset(BancosChicos);
         While Not EOF(BancosChicos) Do
            Begin
{              LineaBancosChicos[9] := ' ';}
               ReadLn(BancosChicos, LineaBancosChicos);
               If Fusiones[j,1] = Copy(LineaBancosChicos,1,3) Then
                  If LineaBancosChicos[9] = 'R' Then
                     TodoUnido[j] := 1;
            End;
      End;
   Close(BancosChicos);
End;



Procedure FormateaParteEntera(Var ParteEntera : String);
{ Formatea la cadena ParteEntera, con comas }
Var
   i, j : Integer;
   ParteEnteraFormato : String;
Begin
   i:=Length(ParteEntera);
   j:=0;
   ParteEnteraFormato:='';
   If i > 3 Then
      Begin
         While (i > 0)  Do
            Begin
               If j = 3 Then
                  Begin
                     ParteEnteraFormato := ',' + ParteEnteraFormato;
                     j:=0;
                  End
               Else
                  Begin
```

```
                    ParteEnteraFormato := ParteEntera[i] + ParteEnteraFormato;
                    i:=i-1;
                    j:=j+1;
                End;
            End;
         ParteEntera:=''+ParteEnteraFormato;
      End;
End;


Procedure ObtenDoctosAceptado;
Var
   j : ShortInt;
   Caracter : Char;
Begin
   Seek(Aceptado, PosicionAceptado);
   DoctosAceptadosParaSumaStr := '';
   For j := 1 To 7 Do
      Begin
         Read(Aceptado, Caracter);
         DoctosAceptadosParaSumaStr := DoctosAceptadosParaSumaStr + Caracter;
      End;
   Val(DoctosAceptadosParaSumaStr,DoctosAceptadosParaSuma,Code);
End;




Procedure ObtenImporteAceptado;
Var
   j : ShortInt;
   Caracter : Char;
Begin
   Seek(Aceptado, PosicionAceptado+7);
   ImporteAceptadoParaSumaStr := '';
   For j := 1 To 18 Do
      Begin
         Read(Aceptado, Caracter);
         ImporteAceptadoParaSumaStr := ImporteAceptadoParaSumaStr + Caracter;
      End;
   Val(ImporteAceptadoParaSumaStr,ImporteAceptadoParaSuma,Code);
End;


Procedure SumaDatosAceptado(j : Integer);
Begin
   Assign(Aceptado,MasterDir+'CMP\'+BancoAceptado+'.ACP');
```

```
   Reset(Aceptado);
   EndPointGirado := j;
   If EndPointGirado <> 0 Then
      Begin
         PosicionAceptado := (EndPointGirado-1) * ACEPTADO_REG_SIZE;
         ObtenDoctosAceptado;
         ObtenImporteAceptado;
      End;
   Close(Aceptado);
End;



Procedure CalculaTotalBanco(i : Integer);
Var
   j : Integer;
Begin
   TotalDocpre  := 0;
   TotalImpre   := 0.0;
   TotalDocrec  := 0;
   TotalImrec   := 0.0;
   { **** Calcula Total Presentado **** }
   For j := 1 To  NUM_BANCOS Do
      Begin
         BancoAceptado := Copy(ArregloBancosCuentas.Buffer[i].Banco.AcronBanco,1,3);
         SumaDatosAceptado(j);
         If  (DoctosAceptadosParaSuma <> 0) And
             (ImporteAceptadoParaSuma <> 0) Then
            Begin
               TotalDocpre  := TotalDocpre  + DoctosAceptadosParaSuma;
               TotalImpre   := TotalImpre   + ImporteAceptadoParaSuma;
            End;
      End;{For}
   { **** Calcula Total Recibido **** }
   For j := 1 To  NUM_BANCOS Do
      Begin
         BancoAceptado := Copy(ArregloBancosCuentas.Buffer[j].Banco.AcronBanco,1,3);
         SumaDatosAceptado(i);
         If  (DoctosAceptadosParaSuma <> 0) And
             (ImporteAceptadoParaSuma <> 0) Then
            Begin
               TotalDocrec  := TotalDocrec  + DoctosAceptadosParaSuma;
               TotalImrec   := TotalImrec   + ImporteAceptadoParaSuma;
            End;
      End;{For}
End;
```

```
Procedure CalculaPresentadoB1aB2(i, j : Integer);
Begin
   TotalDocpre  := 0;
   TotalImpre   := 0.0;
   { **** Calcula Total Presentado Por El Banco1 Al Banco2 **** }
   BancoAceptado := Copy(ArregloBancosCuentas.Buffer[i].Banco.AcronBanco,1,3);
   SumaDatosAceptado(j);
   If (DoctosAceptadosParaSuma <> 0) And (ImporteAceptadoParaSuma <> 0) Then
      Begin
         TotalDocpre  := TotalDocpre  + DoctosAceptadosParaSuma;
         TotalImpre   := TotalImpre   + ImporteAceptadoParaSuma;
      End;
End;



Procedure CalculaTotalArreglo;
Var
   i, j : Integer;
Begin
   TotalDocpre  := 0;
   TotalImpre   := 0.0;
   TotalDocrec  := 0;
   TotalImrec   := 0.0;
   { **** Calcula Total Presentado **** }
   For i := 1 To NUM_BANCOS Do
      For j := 1 To NUM_BANCOS Do
         Begin
            BancoAceptado := Copy(ArregloBancosCuentas.Buffer[i].Banco.AcronBanco,1,3);
            SumaDatosAceptado(j);
            If  (DoctosAceptadosParaSuma <> 0) And
                (ImporteAceptadoParaSuma <> 0) Then
               Begin
                  TotalDocpre  := TotalDocpre  + DoctosAceptadosParaSuma;
                  TotalImpre   := TotalImpre   + ImporteAceptadoParaSuma;
               End;
         End;{ For j }
   { **** Calcula Total Recibido **** }
   For i := 1 To NUM_BANCOS Do
      For j := 1 To  NUM_BANCOS Do
         Begin
            BancoAceptado := Copy(ArregloBancosCuentas.Buffer[j].Banco.AcronBanco,1,3);
            SumaDatosAceptado(i);
            If  (DoctosAceptadosParaSuma <> 0) And
                (ImporteAceptadoParaSuma <> 0) Then
               Begin
                  TotalDocrec  := TotalDocrec  + DoctosAceptadosParaSuma;
```

```
                TotalImrec    := TotalImrec    + ImporteAceptadoParaSuma;
              End;
        End;{ For j }
End;


Begin
End.
```

```
Unit Cuentas2;

Interface

   Procedure ValidaArchivo;


Implementation

Uses Datos4, Calculos, CargaDat, Datos, Datos3,
     SysUtils, Unit24;

Const

(** TAMAYO DEL ARREGLO DE ERRORES Y NUMERO MAXIMO DE ERRORES **)
   NUM_ERRORES = 38;

(** ARCHIVO PARA COMPARATIVO Y EL DE LO ACEPTADO **)
   ACEPTADO_REG_SIZE = 25;
   COMPARATIVO_REG_SIZE = 25;

(** DATOS DEL ARCHIVO DE BANCOS **)
   BANK_FILE_RECORD_SZ = 60;
   INIC_NUM_BANCO = 1;
   FIN_NUM_BANCO  = 3;
   INIC_ACRON_BANCO = 4;
   FIN_ACRON_BANCO  = 12;
   INIC_NOMBRE_BANCO =  13;
   FIN_NOMBRE_BANCO  =  60;

(** DATOS DE TECLAS **)
   TAB_KEY           =  #09;
   ENTER_KEY         =  #13;

(*** DATOS DEL CURSOR ***)

   CURSOR_OFF_L = $00;
   CURSOR_OFF_H = $10;
   CURSOR_CAPTURA_L          =  7;   {Comienzo}
   CURSOR_CAPTURA_H          =  6;   {Final}


(** OTRAS CONSTANTES **)
   NUM_PLAZAS_NO_ABM_MAX    = 540;
   NUM_PLAZAS_ABM_MAX       =  20;

   NUM_PLAZAS_NO_ABM  : Integer  =   0;
   NUM_PLAZAS_ABM     : ShortInt =   0;
```

```
(*****  INFORMACION DEL ARCHIVO DE PLAZAS NO ABM NOABM.DAT  *****)
   NO_ABM_RECORD_SZ            =  9;


(*****  INFORMACION DEL ARCHIVO DE PLAZAS ABM PLAZABM.DAT  *****)
   ABM_RECORD_SZ               = 5;

(*****  INFORMACION DEL ARCHIVO DE PESOS  *****)
   PESOS_FILE_RECORD_SZ        = 36;


(*****  VECTOR PARA CALCULAR DIGITO VERIFICADOR DEL TRANSITO  *****)

   PesoTransitos : Array [1..8] Of ShortInt = (3,7,1,3,7,1,3,7);


(********* CONSTANTES PARA EL ROE ***************)

  Linea1 : String =  'CECOBAN, S.A. DE C.V.                                          '+
'                                                                                  ';
  Linea2A = 'REPORTE DETALLE DE OPERACIONES ';
  Linea2B : String [13] = 'PRESENTADAS C';
  Linea2C : String [72] = 'ON ANOMALIAS                                          '+
                          '                    ';
  Linea2D : String [10] = '  /  /  ';     {fecha de presentaci¢n}
  Linea2E = '    ';
  Linea2F = 'HOJA ';
  Linea2G : String [04] = '0000';         {n£mero de hoja}

  Linea3A = 'PLAZA                 :          AREA METROPOLITANA DE LA CIUDAD '+
  'DE ';
  Linea3B : String[25] = '';
  Linea3C = '                                                  ';

  Linea4A = 'INSTITUCION ';
  Linea4B : String [16] = 'CEDENTE  :      ';
  Linea4C : String [60] = '';             { adherente y nombre del banco }
  Linea4D = '                                                  ';

  Linea5A = 'TRANSFERENCIA         :  ';
  Linea5B : String[07] = '';              {transferencia ddnn ¢ ddnnnnn}
  Linea5C = '                                                  '+
            '                                                  ';
  Linea6A = 'FECHA DE INTERCAMBIO :  ';
  Linea6B : String[10] = '          ';    {fecha de intercambio}
  Linea6C = '                                                  '+
            '                                                  ';
  Linea7 : string = 'TIPO DE OPERACION    :      040 CHEQUES DE COBRO INMEDIATO';
  Linea8  = '           INSTITUCION    REF  LOC                                    '+
```

```
                '                    CAMPOS C/ ERROR              E'+
                '       I M P O R T E';
  Linea9   = 'SECUENCIA   EMI   REC    LOTE    SEC  TT  PPP     CUENTA        CHEQUE'+
             '    Di  Dp  SSS  1 2 3 4 5 6 7 8  TRANSITO   T      '+
             '                        ';
  Linea10A = '                          TOTAL OPERACIONES RECHAZADAS : ';
  Linea10B : String [04] = '';  {n£mero de operaciones rechazadas}
  Linea10C = '   IMPORTE : ';
  Linea10D : String [24] = '';    {importe total de operaciones rechazadas}
  Linea10E = '                                                 ';
  Linea11 = 'TIP    NUM     INS      IDENTIFICACION          FECHA          TIPO  '+
             '                                                 '+
             '                      ';
  Linea12 = 'REG    SEC     EMI      TRANSFERENCIA        PRESENTACION    ARCHIVO '+
             '                                            '+
             '                      ';
  Linea13 = 'TIP NUM   TIPO         TOTAL             I M P O R T E             '+
             '                                            '+
             '                    ';
  Linea14 = 'REG SEC   OPER         REGISTROS                                  '+
             '                                  '+
             '                    ';
  LineaG   = '-------------------------------------------------------------'+
'-------------------------------------------------------------------------';
  LineaG2 = '                            ---------------------------------------'+
'---------------------------------                                        ';

  LINEAS_EN_REPORTE     = 56;
  LINEAS_EN_ENCABEZADO  = 12;
  FormFeed              = #12;


Var
   PlazasNoABMFile      : Text;
   PlazasABMFile        : Text;
   PesosFile            : Text;
   BankFile             : Text;
{   LineaPlaza : String;
   NombrePlaza : String[30];
   NumeroPlaza : String[5];
   NuevaPlaza  : String[3];
   SistemaTEI  : String[2];
   NamePlaza : Text;
}
   Aceptado             : File Of Char;
   Comparativo          : File Of Char;
   BankReg              : String[BANK_FILE_RECORD_SZ];
   Entrada              : Text;
   Errores              : Text;
   Errores2             : Text;
```

```
Errores3                : Text;
Errores4                : Text;
ArchivoRoed             : Text;
Salida                  : Text;
Bitacora  : Text;
CmdLine : String;
LineasASumar : ShortInt;
PosicionAceptado     : LongInt;
PosicionComparativo : LongInt;
DoctosAceptadosParaSumaStr  : String[07];
ImporteAceptadoParaSumaStr : String[18];
DoctosAceptadosParaSuma      : LongInt;
ImporteAceptadoParaSuma     : Extended;
DoctosComparativoParaSumaStr  : String[07];
ImporteComparativoParaSumaStr : String[18];
DoctosComparativoParaSuma      : LongInt;
ImporteComparativoParaSuma     : Extended;
DatosSumario       : String[25];
BancoRoe           : String[03];
BancoGiradoDetalle : String[03];
Di, Dp             : String[01];
TransitoNFormato   : String[09];
UnidadDisco        : Char;
LimiteBack         : ShortInt;
Contador           : ShortInt;
NumGuiones         : ShortInt;
Code               : Integer;
Invalida           : ShortInt;
InvalidaTemp       : ShortInt;
ArregloErrores     : Array [1..NUM_ERRORES] Of ShortInt;
FinDetalle         : ShortInt;
EraDuplicada       : ShortInt;
PesosReg           : String[PESOS_FILE_RECORD_SZ];
NoABMReg           : String[NO_ABM_RECORD_SZ];
ABMReg             : String[ABM_RECORD_SZ];
PlazasNoABM        : Array [1..NUM_PLAZAS_NO_ABM_MAX,1..2] Of Integer;
PlazasABM          : Array [1..NUM_PLAZAS_ABM_MAX] Of Integer;
NumBancoGir        : Integer;
NumBancoGir2       : Integer;
CuentaLineas       : ShortInt;
NumPagina          : ShortInt;
TotalRechazos      : Integer;
BanderaHeader      : ShortInt;
TotalRegsDetalle   : Longint;
TotalRegsDetalleStr  : String [07];
TotalRegsDetalleStr2 : String;
ImporteNum         : Extended;
ImporteTotal       : Extended;
ImporteRechazos    : Extended;
OperacionesErroneas : Integer;
PorcentajeErrores : Real;
```

```
   ErrorHeader        : ShortInt;
   ErrorDetalle       : ShortInt;
   ErrorSumario       : ShortInt;
   IdentificacionTrans : String [07];
   TotalSumarioStr    : String [18];
   TotalSumarioStr2   : String;
   NombreArchivo      : String[20];
   PlazaCheque        : String [05];
   Transito           : String [05];
   LineaArchivos      : String[20];
   LineaEntrada       : String[124];
   LineaEntrada2      : String;
   LineaReporte       : String[136];
   Transferencias     : Array [1..60] Of String[07];
   EndPointCedHeader  : Integer;
   NumBancoRoe        : Integer;
   EndPointBancoRoe   : Integer;
   RemesaDup : ShortInt;
   TransitoHeader     : String[03];
   HuboRoe            : String[01];
   Strx, Pstrx : String;
   Secuencia          : LongInt;
   EndPointGirado     : Integer;
   TotalDoctosAcep    : LongInt;
   TotalImporteAcep   : Extended;
   TotalDoctosAcepStr : String[07];
   TotalDoctosAcepStr2 : String;
   TotalImporteAcepStr: String[18];
   TotalImporteAcepStr2 : String;
   TotalImporteRechazoStr  : String;
   TotalImporteRechazoStr2 : String;
{   TamanioArchivo      : LongInt;}
{   Prueba_o_Real : String[1];}
   PreprocesoFile     : Text;
   LineaPreproceso    : String;
   Recupera           : String[1];
   Hora               : String [5];



Procedure AbreArchivo;
Var
   j         : Integer;
   VSal      : File Of Char;
   Caracter  : Char;
   Posicion  : Integer;
   Tamanio   : LongInt;
Begin
   If ExisteArchivo(MasterDir+'BACKUP\COBRO1') Then
      Begin
```

```
            Assign(VSal,MasterDir+'BACKUP\COBRO1');
            Reset(VSal);
            Tamanio := FileSize(VSal);
            Repeat
               Seek(VSal,Tamanio-140);
               LineaEntrada2 := '';
               For j:= 1 To 140 Do
                  Begin
                     Read(VSal,Caracter);
                     LineaEntrada2 := LineaEntrada2 + Caracter;
                  End;      {Pos(Chr(13)+Chr(10)+'09',LineaEntrada2);}
               Posicion := Pos(Chr(10)+'09',LineaEntrada2);
               Tamanio := Tamanio - 140;
            Until (Posicion <> 0) Or (Tamanio <= 140);
            Close(VSal);
            If Tamanio <= 140 Then
               Begin
               {
                  MensajeEnRecuadro2('El archivo no tiene sumario, o no se copi¢ completo',
                              'no se tomar  en cuenta para la dispersi¢n', '');
                  Halt;
               }
               End
            Else If (Posicion <> 0) And (Tamanio >= 200) Then
               Begin
                  Assign(Entrada,MasterDir+'BACKUP\COBRO1');
                  Reset(Entrada);
               End
            Else
               Begin
                {
                  MensajeEnRecuadro2('Este archivo parece no tener informaci¢n',
                              'no se tomar  en cuenta para la dispersi¢n', '');
                  Halt;
                }
               End;
         End
      Else
         Begin
         {
            MensajeEnRecuadro('Disco da□ado, o no estaba lista la unidad de cinta');
            Halt;
         }
         End;
   End; {AbreArchivo}



   Procedure AbreRoe;
   Begin
```

```
   Assign(Errores,MasterDir+'REPORTES\ROE'+BancoRoe);
   If ExisteArchivo(MasterDir+'REPORTES\ROE'+BancoRoe) Then
      Append(Errores)
   Else
     Begin
      Rewrite(Errores);
     End;
End;



Procedure CierraArchivo;
Begin
   Close(Entrada);
   Close(Salida);
   Close(Errores);
End;


Procedure CierraRoed;
Begin
   Close(Errores2);
   Close(ArchivoRoed);
End;



Procedure STD011(Cuenta : String; Peso : PesoGeneralNumerico; Indice : Integer);
Var
   i, j, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta : Integer;

Begin
   Suma:=0;
   Modulo := 11;
   j := 13;
   For i := 13 DownTo 1 Do
      If i <> Indice Then
         Begin
            Val(Cuenta[i],Cuentai,Code);
            Suma := Suma + (Cuentai * Peso[j]);
            j := j - 1;
         End
      Else
         Val(Cuenta[i],DigCuenta,Code);
   Residuo := Suma Mod Modulo;
   Digito := (Modulo - Residuo);
   If Residuo = 0 Then
      Digito := 0;
   If Residuo = 1 Then
      InvalidaTemp := -1
```

```
   Else If (Digito = DigCuenta)  Then
      Invalida := 0;
End;


{ Caso especial banrural en que el resultado del m¢dulo es el d¡gito }
Procedure STD11D(Cuenta : String; Peso : PesoGeneralNumerico; Indice : Integer);
Var
   i, j, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta : Integer;

Begin
   Suma:=0;
   Modulo := 11;
   j := 13;
   For i := 13 DownTo 1 Do
      If i <> Indice Then
         Begin
            Val(Cuenta[i],Cuentai,Code);
            Suma := Suma + (Cuentai * Peso[j]);
            j := j - 1;
         End
      Else
         Val(Cuenta[i],DigCuenta,Code);
   Residuo := Suma Mod Modulo;
   Digito :=  Residuo;
   If Digito = DigCuenta Then
      Invalida := 0
   Else If Residuo = 10 Then
      InvalidaTemp := -1;
End;

{ Caso especial santander en que el m¢dulo es 10 en vez de 11 }
Procedure STD11S(Cuenta : String; Peso : PesoGeneralNumerico; Indice : Integer);
Var
   i, j, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta : Integer;

Begin
   Suma:=0;
   Modulo := 10;
   j := 13;
   For i := 13 DownTo 1 Do
      If i <> Indice Then
         Begin
            Val(Cuenta[i],Cuentai,Code);
            Suma := Suma + (Cuentai * Peso[j]);
            j := j - 1;
         End
      Else
         Val(Cuenta[i],DigCuenta,Code);
```

```
   Residuo := Suma Mod Modulo;
   Digito := (Modulo - Residuo);
   If Residuo = 0 Then
      Digito := 0;
   If Digito = DigCuenta Then
      Invalida := 0
   Else If Residuo = 1 Then
      InvalidaTemp := -1;
End;


Procedure STD010(Cuenta : String; Peso : PesoGeneralNumerico; Indice : Integer);
Var
   i, j, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta : Integer;

Begin
   Suma:=0;
   Modulo := 10;
   j := 13;
   For i := 13 DownTo 1 Do
      If i <> Indice Then
         Begin
            Val(Cuenta[i],Cuentai,Code);
            Suma := Suma + (Cuentai * Peso[j]);
            j := j - 1;
         End
      Else
         Val(Cuenta[i],DigCuenta,Code);
   Residuo := Suma Mod Modulo;
   Digito := (Modulo - Residuo);
   If Residuo = 0 Then
      Digito := 0;
   If Digito = DigCuenta Then
      Invalida := 0
End;


Procedure SUMDIG(Cuenta : String; Peso : PesoGeneralNumerico; Indice : Integer);
Var
   i, j, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta : Integer;
   Unidad, Decena : Integer;

Begin
   Suma:=0;
   Modulo := 10;
```

```
   j := 13;
   For i := 13 DownTo 1 Do
      If i <> Indice Then
         Begin
            Val(Cuenta[i],Cuentai,Code);
            Unidad := (Cuentai * Peso[j]) Mod 10;
            Decena := (Cuentai * Peso[j]) Div 10;
            Suma := Suma + Unidad + Decena;
            j := j - 1;
         End
      Else
         Val(Cuenta[i],DigCuenta,Code);
   Residuo := Suma Mod Modulo;
   Digito := (Modulo - Residuo);
   If Residuo = 0 Then
      Digito := 0;
   If Digito = DigCuenta Then
      Invalida := 0
End;



{ Algoritmo para Bancomer plazas NO ABM }
Procedure SUMDNA(Cuenta : String;
                 Peso   : PesoGeneralNumerico;
                 Indice : Integer;
                 Plaza  : Integer);

Var
   i, j, k, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta : Integer;
   Unidad, Decena : Integer;
   PlazaBancomer : String[03];
   CuentaCompuesta : String[13];

Begin
   k := 1;
   Val(Cuenta[13],DigCuenta,Code);
   While (Invalida = 1) And (k < (NUM_PLAZAS_NO_ABM + 1)) Do
      If Plaza <> PlazasNoABM[k][1] Then
         k := k + 1
      Else
         Begin
            Suma:=0;
            Modulo := 10;
            j := 13;
            Str(PlazasNoABM[k][2]:3,PlazaBancomer);
            If PlazaBancomer[1] = ' ' Then
               PlazaBancomer[1] := '0';
            If PlazaBancomer[2] = ' ' Then
               PlazaBancomer[2] := '0';
```

```
            CuentaCompuesta := ''+PlazaBancomer + '1'+ Copy(Cuenta,6,8);
            While Length(CuentaCompuesta) < 13 Do
               CuentaCompuesta := '0' + CuentaCompuesta;
            k := k + 1;
            For i := 13 DownTo 1 Do
               If i <> Indice Then
                  Begin
                     Val(CuentaCompuesta[i],Cuentai,Code);
                     Unidad := (Cuentai * Peso[j]) Mod 10;
                     Decena := (Cuentai * Peso[j]) Div 10;
                     Suma := Suma + Unidad + Decena;
                     j := j - 1;
                  End
               Else
                    Val(CuentaCompuesta[i],DigCuenta,Code);
            Residuo := Suma Mod Modulo;
            Digito := (Modulo - Residuo);
            If Residuo = 0 Then
               Digito := 0;
            If Digito = DigCuenta Then
               Invalida := 0
         End; {Else}
End;



{ Algoritmo para Bancomer plazas ABM }
Procedure SUMDAB(Cuenta : String;
                 Peso   : PesoGeneralNumerico;
                 Indice : Integer;
                 Plaza  : Integer);

Var
   i, j, k, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta : Integer;
   Unidad, Decena, Constante : Integer;
   PlazaBancomer : String[05];
   CuentaCompuesta : String[13];

Begin
   k := 1;
   Val(Cuenta[13],DigCuenta,Code);
   While (Invalida = 1) And (k < (NUM_PLAZAS_ABM + 1)) Do
      If Plaza <> PlazasABM[k] Then
         k := k + 1
      Else
         Begin
            Suma:=0;
            Modulo := 10;
```

```
            Constante := 6;
            Str(PlazasABM[k]:5,PlazaBancomer);
            If PlazaBancomer[1] = ' ' Then
               PlazaBancomer[1] := '0';
            CuentaCompuesta := ''+PlazaBancomer + '1'+ Copy(Cuenta,6,7);
            While Length(CuentaCompuesta) < 13 Do
               CuentaCompuesta := '0' + CuentaCompuesta;
            k := k + 1;
            For i := 13 DownTo 1 Do
               Begin
                  Val(CuentaCompuesta[i],Cuentai,Code);
                  Unidad := (Cuentai * Peso[i]) Mod 10;
                  Decena := (Cuentai * Peso[i]) Div 10;
                  Suma := Suma + Unidad + Decena;
                  End;
            Suma := Suma + Constante;
            Residuo := Suma Mod Modulo;
            Digito := (Modulo - Residuo);
            If Residuo = 0 Then
               Digito := 0;
            If Digito = DigCuenta Then
               Invalida := 0
         End; {Else}
End;



Procedure CNST06(Cuenta : String; Peso : PesoGeneralNumerico; Indice : Integer);
Var
   i, j, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta, Constante : Integer;

Begin
   Suma:=0;
   Modulo := 10;
   Constante := 6;
   j := 13;
   For i := 13 DownTo 1 Do
      If i <> Indice Then
         Begin
            Val(Cuenta[i],Cuentai,Code);
            Suma := Suma + ((Cuentai * Peso[j]) Mod 10);
            j := j - 1;
         End
      Else
         Val(Cuenta[i],DigCuenta,Code);
   Suma := Suma + Constante;
   Residuo := Suma Mod Modulo;
   Digito := (Modulo - Residuo);
```

```
   If Residuo = 0 Then
      Digito := 0;
   If Digito = DigCuenta Then
      Invalida := 0
End;



Procedure CNST07(Cuenta : String; Peso : PesoGeneralNumerico; Indice : Integer);
Var
   i, j, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta, Constante : Integer;

Begin
   Suma:=0;
   Modulo := 11;
   Constante := 7;
   j := 13;
   For i := 13 DownTo 1 Do
      If i <> Indice Then
         Begin
            Val(Cuenta[i],Cuentai,Code);
            Suma := Suma + (Cuentai * Peso[j]);
            j := j - 1;
         End
      Else
         Val(Cuenta[i],DigCuenta,Code);
   Suma := Suma + Constante;
   Residuo := Suma Mod Modulo;
   Digito := (Modulo - Residuo);
   If Residuo = 0 Then
      Digito := 0;
   If Digito = DigCuenta Then
      Invalida := 0
   Else If Residuo = 1 Then
      InvalidaTemp := -1;
End;



Procedure CNST27(Cuenta : String; Peso : PesoGeneralNumerico; Indice : Integer);
Var
   i, j, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta, Constante : Integer;

Begin
   Suma:=0;
   Modulo := 11;
   Constante := 27;
```

```
   j := 13;
   For i := 13 DownTo 1 Do
      If i <> Indice Then
         Begin
            Val(Cuenta[i],Cuentai,Code);
            Suma := Suma + (Cuentai * Peso[j]);
            j := j - 1;
         End
      Else
         Val(Cuenta[i],DigCuenta,Code);
   Suma := Suma + Constante;
   Residuo := Suma Mod Modulo;
   Digito := (Modulo - Residuo);
   If Residuo = 0 Then
      Digito := 0;
   If Digito = DigCuenta Then
      Invalida := 0
   Else If Residuo = 1 Then
      InvalidaTemp := -1;
End;



Procedure RES001(Cuenta : String; Peso : PesoGeneralNumerico; Indice : Integer);
Var
   i, j, Suma, Cuentai, Modulo : Integer;
   Residuo, Digito, DigCuenta : Integer;

Begin
   Suma:=0;
   Modulo := 11;
   j := 13;
   For i := 13 DownTo 1 Do
      If i <> Indice Then
         Begin
            Val(Cuenta[i],Cuentai,Code);
            Suma := Suma + (Cuentai * Peso[j]);
            j := j - 1;
         End
      Else
         Val(Cuenta[i],DigCuenta,Code);
   Residuo := Suma Mod Modulo;
   Digito := Modulo - Residuo;
   If Residuo = 0 Then
      Begin
         If (NumBancoGir = 58) Or (NumBancoGir = 17) Then  {excepci¢n banco BANREGIO y BBV}
            Digito := 1
         Else
            Digito := 0
      End
```

```
   Else If Residuo = 1 Then
      Digito := 0;
   If Digito = DigCuenta Then
      Invalida := 0
End;


Procedure ObtenNombreBanco(i : Integer);
Var
   j : Integer;
Begin
   j:=1;
   Linea4C := '';
   Repeat
      Linea4C := Concat(Linea4C,
                        ArregloBancosCuentas.Buffer[i].Banco.NombreBanco[j]);
      j := j + 1;
   Until (ArregloBancosCuentas.Buffer[i].Banco.NombreBanco[j] = ' ') And
         (ArregloBancosCuentas.Buffer[i].Banco.NombreBanco[j+1] = ' ') And
         (ArregloBancosCuentas.Buffer[i].Banco.NombreBanco[j+2] = ' ');
   While Length(Linea4C) < 60 Do
      Linea4C := Linea4C + ' ';
End;


Procedure ObtenHora;
Begin
  Hora := TimeToStr(Time);
End;



Procedure ImprimeEncabezado(Parametro : Integer);
Begin
   ObtenHora;
               Linea1[115] := 'H';
               Linea1[116] := 'O';
               Linea1[117] := 'R';
               Linea1[118] := 'A';
               Linea1[119] := ' ';
               Linea1[120] := ':';
               Linea1[121] := ' ';
               Linea1[122] := Hora[1];
               Linea1[123] := Hora[2];
               Linea1[124] := Hora[3];
               Linea1[125] := Hora[4];
               Linea1[126] := Hora[5];
   WriteLn(Errores,Linea1);
   Linea2D:=Copy(CFecha,1,4)+'/'+Copy(CFecha,5,2)+'/'+Copy(CFecha,7,2);
```

```
    Str(NumPagina,Linea2G);
    While Length(Linea2G) < 4 Do
       Linea2G := '0'+Linea2G;
    LineaReporte:=Linea2A+Linea2B+Linea2C+Linea2D+Linea2E+Linea2F+Linea2G;
    WriteLn(Errores,LineaReporte);
    WriteLn(Errores,LineaG);
    Linea3B := NombrePlaza;
    While Length(NombrePlaza) < 25 Do
       NombrePlaza := NombrePlaza + ' ';
    LineaReporte:=Linea3A+Linea3B+Linea3C;
    WriteLn(Errores,LineaReporte);
    {ObtenNombreBanco(EndPointCedHeader); esto cambio}
    ObtenNombreBanco(EndPointBancoRoe);
    LineaReporte:=Linea4A+Linea4B+Linea4C+Linea4D;
    WriteLn(Errores,LineaReporte);
    Linea5B:=''+IdentificacionTrans;
    LineaReporte:=Linea5A+Linea5B+Linea5C;
    WriteLn(Errores,LineaReporte);
    Linea6B:=Linea2D;
    LineaReporte:=Linea6A+Linea6B+Linea6C;
    WriteLn(Errores,LineaReporte);
    Linea7 := Linea7 + LetreroMoneda;
    While Length(Linea7) < 136 do
      Linea7 := Linea7 + ' ';
    WriteLn(Errores,Linea7);
    WriteLn(Errores,LineaG);
    If Parametro = 1 Then
       Begin
          WriteLn(Errores,Linea8);
          WriteLn(Errores,Linea9);
       End
    Else If Parametro = 2 Then
       Begin
          WriteLn(Errores,Linea11);
          WriteLn(Errores,Linea12);
       End
    Else If Parametro = 3 Then
       Begin
          WriteLn(Errores,Linea13);
          WriteLn(Errores,Linea14);
       End;
    WriteLn(Errores,LineaG);
End;


Procedure ImprimeEncabezadoRoed(Parametro : Integer);
Begin
{  ObtenHora;
                Linea1[93]  := 'H';
                Linea1[94]  := 'O';
                Linea1[95]  := 'R';
```

```
                Linea1[96]  := 'A';
                Linea1[97]  := ' ';
                Linea1[98]  := ':';
                Linea1[99]  := ' ';
                Linea1[100] := Hora[1];
                Linea1[101] := Hora[2];
                Linea1[102] := Hora[3];
                Linea1[103] := Hora[4];
                Linea1[104] := Hora[5];
}
   WriteLn(Errores3,Linea1);
   Linea2D:=Copy(CFecha,1,4)+'/'+Copy(CFecha,5,2)+'/'+Copy(CFecha,7,2);
   Str(NumPagina,Linea2G);
   While Length(Linea2G) < 4 Do
      Linea2G := '0'+Linea2G;
   LineaReporte:=Linea2A+Linea2B+Linea2C+Linea2D+Linea2E+Linea2F+Linea2G;
   WriteLn(Errores3,LineaReporte);
   WriteLn(Errores3,LineaG);
   Linea3B := NombrePlaza;
   While Length(NombrePlaza) < 25 Do
      NombrePlaza := NombrePlaza + ' ';
   LineaReporte:=Linea3A+Linea3B+Linea3C;
   WriteLn(Errores3,LineaReporte);
   ObtenNombreBanco(EndPointGirado);
   Linea4B := 'DESTINO  :       ';
   LineaReporte:=Linea4A+Linea4B+Linea4C+Linea4D;
   WriteLn(Errores3,LineaReporte);
   Linea4B := 'CEDENTE  :       ';
   Linea5B:=''+'0000000';
   LineaReporte:=Linea5A+Linea5B+Linea5C;
   WriteLn(Errores3,LineaReporte);
   Linea6B:=Linea2D;
   LineaReporte:=Linea6A+Linea6B+Linea6C;
   WriteLn(Errores3,LineaReporte);
   WriteLn(Errores3,Linea7);
   WriteLn(Errores3,LineaG);
   If Parametro = 1 Then
      Begin
         WriteLn(Errores3,Linea8);
         WriteLn(Errores3,Linea9);
      End
   Else If Parametro = 2 Then
      Begin
         WriteLn(Errores3,Linea11);
         WriteLn(Errores3,Linea12);
      End
   Else If Parametro = 3 Then
      Begin
         WriteLn(Errores3,Linea13);
         WriteLn(Errores3,Linea14);
      End;
```

```
   WriteLn(Errores3,LineaG);
End;



Procedure EscribeError(Mensaje : String);
Begin
   While Length(Mensaje) < 136 Do
      Mensaje := Mensaje + ' ';
   WriteLn(Errores,Mensaje);
End;


Procedure EscribeErrorRoed(Mensaje : String);
Begin
   While Length(Mensaje) < 136 Do
      Mensaje := Mensaje + ' ';
   WriteLn(Errores2,Mensaje);
End;



Procedure FormateaParteEntera(Var ParteEntera : String);
{ Formatea la cadena ParteEntera, con comas }
Var
   i, j : Integer;
   ParteEnteraFormato : String;
Begin
   i:=Length(ParteEntera);
   j:=0;
   ParteEnteraFormato:='';
   If i > 3 Then
      Begin
         While (i > 0)  Do
            Begin
               If j = 3 Then
                  Begin
                     ParteEnteraFormato := ',' + ParteEnteraFormato;
                     j:=0;
                  End
               Else
                  Begin
                     ParteEnteraFormato := ParteEntera[i] + ParteEnteraFormato;
                     i:=i-1;
                     j:=j+1;
                  End;
            End;
         ParteEntera:=''+ParteEnteraFormato;
      End;
End;
```