UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA DE CONTENEDORES PARA EMPAQUE Y FLUJOS EN EL BANCO CENTRAL

T E S I S

QUE PARA OBTENER EL TÍTULO DE
INGENIERO EN COMPUTACIÓN

PRESENTAN
ARMANDO PADILLA TRUJILLO
JOSÉ ADOLFO IBARRECHE MATA
JOSÉ ALEJANDRO ALFARO SOTO
OSCAR GERARDO GUTIERREZ GÓMEZ
SONIA MARTÍNEZ RUIZ

DIRECTOR DE TESIS M I LAURO SANTIAGO CRUZ

CIUDAD UNIVERSITARIA, MEXICO D.F. 2001

A mis padres:
Maria Luisa Trujillo Rangel
Julio Padilla Sánchez.
Por darnos la fuerza de esquivar cualquier obstáculo que se nos presente en la vida y por su gran confianza y amor. GRACIAS.

A mis hermanos:
Jorge, Margarita, Lorena y Carlos.
Por ayudarme siempre que lo necesite y
Por estar siempre unidos. GRACIAS.

ARMANDO PADILLA TRUJILLO

A mis padres y hermanas que me brindaron su apoyo Incondicional para poder realizar esta meta.

JOSÉ ADOLFO IBARRECHE MATA

Este trabajo se lo dedico como muestra de mi agradecimiento a todas aquellas personas que participaron en mi formación profesional.

A mi padres Santiago y María de los Angeles por su amor y cariño, a mis hermanos Angel, Irma, Norma, Luis y Araceli por su apoyo y comprensión.

Particularmente a todas aquellas personas que me ofrecieron sus palabras de aliento en el momento que más lo necesite.

Al Señor mi Dios por haberme permitido valorar cada instante... compartido con ustedes.

JOSÉ ALEJANDRO ALFARO SOTO

Maricarmen (Cal):
or ser el aliento y la esperanza de mis días, el apoyo
ue me mantiene en pie. Por ser la fuerza de mi corazón
lo más importante de mi vida. Gracias.

A mi papá:
Por su gran amor, comprensión y cariño. Gracias por sus sabios consejos que en su momento no entendí, pero con el tiempo y las experiencias comprendí Cuanta razón tenías.

A mi mamá.
Por su amor incondicional a prueba de todo. Por su disciplina y carácter para guiar mis pasos y ayudarme a lograr mis metas. Gracias por ser única, siempre te querré.

A mi hermana Mónica
Con quien compartí mi niñez, tan feliz. Gracias por dejarme tan buenos recuerdos y sobre todo, por defenderme de los niños que me pegaban y darme dos hermosas sobrinas

A mi hermano Isaac:
Espero que este trabajo, fruto de un gran esfuerzo, te sirva de motivación y no desfallezcas para lograr tus propósitos. Gracias por tu admiración y nobleza.

A Marcos.
Gracias, ante todo por tu amistad Por ser mi ejemplo, mi guru y por lograr a tu manera, sacar siempre lo mejor de mi. De verdad gracias.

OSCAR GERARDO GUTIERREZ GÓMEZ

mis padres:
r el apoyo y cariño que siempre nos han dado, y que
s siguen dando a pesar de todo. Gracias.

A ti papi, por estar siempre conmigo. Nunca olvides que eres la persona más importante en mi vida. Te quiero mucho, y por fin... ya acabamos!!! Esto es de los dos

A mi mami, por ser un ejemplo para nosotras.
No olvides que a pesar de que pensamos de manera diferente, siempre tendremos la educación que nos diste. Gracias por alentarnos a ser siempre mejores. Te quiero.

A mis hermanas, Norma y Edna por ser las mejores hermanas, bueno, ¿que puedo decir?, son las únicas hermanas que tengo ¿no?, (aunque ahora también tengo un hermano. .). Gracias a los tres por su apoyo y cariño

También creo oportuno mencionar a todas esas personitas que han sido, y siguen siendo, parte de mi vida y con quienes he compartido alegrías y tristezas. Cada uno de ustedes sabe quienes son .. Gracias por su amistad.

SONIA MARTÍNEZ RUIZ

Tabla de Contenido

NTRODUCCIÓN ..... *V*
Historia y funciones del Banco Central V
Problemática VI
Propuesta de solución VII
Objetivo VII

APÍTULO I *1*

CONCEPTOS BÁSICOS 1
1 1. Metodologías para el desarrollo de sistemas 1
1 1.1 Metodologías orientadas al proceso 2
1.1.2. Metodologías orientadas a los datos 2
1.1.3. Metodologías orientadas a objetos 3
1.1.4 No metodológicas 3
1.2. Metodología orientada a objetos 3
1 3. Lenguaje de Modelado Unificado 6
1.4 Tecnología de marcos de trabajo 9
1.5. Tecnología de códigos de barras 11
1.6. Administración de proyectos 15

APÍTULO II *23*

MODELO DEL NEGOCIO 23
2.1 Descripción del negocio 23
2 2. Autorización de emisión y órdenes de fabricación 25
2 3 Empaque de piezas 26
2 3.1. Corte de hojas 26
2.3.2. Área de empaque 28
2 3.3. Empaque de piezas de forma automática 29
2.4. Armado y desarmado de envases 30
2 5. Remesas 32
2 6 Traspasos 33
2.7. Depósitos 34
2.8. Retiros 35

APÍTULO III *37*

INGENIERÍA DE REQUERIMIENTOS 37
3 1 Análisis del problema 37
3.2 Control de puntos de evento 40
3.3. Impresión de codigos de barras 42
3.4 Armado y desarmado de contenedores 43
3 5 Cambio de estado físico 44
3.6. Autorizaciones de emision y órdenes de fabricación 45
3.7. Empaque de efectivo 46
3 8 Entrega de efectivo 49
3.9. Traspasos 50
3.10 Depósitos 52
3 11. Retiros 53
3 12 Remesas 55

APÍTULO IV *59*

ANÁLISIS DEL SISTEMA 59

7.5.1. Métricas técnicas del *software* 167
7.5.2 Calidad del *software* 167
7.5.3. El reto de las métricas técnicas 168
7.5.4. FURPS 168
7.6. Pruebas al sistema de contenedores. 170
7 6.1. Pruebas de Interfaz Gráfica 170
7.6.2. Pruebas de Unidad al sistema 171
7.6.3. Pruebas de Integración 172
7.6.4. Pruebas de Validación 172

PÍTULO VIII *177*

ONCLUSIONES 177

éndice A *1*

ASOS DE USO 1
A.1. Depósito. 1
A.2. Retiros 5
A.3. Remesas. 12

éndice B *1*

IAGRAMAS DE ENTIDAD-RELACIÓN 1
B.1. Tablas de Operación. 2
B.2. Tablas de Impresión de Códigos de barras. 3
B.3. Tablas de Remesas. 4
B.4. Tablas de Autorizaciones y Órdenes. 5
B.5. Tablas de Depósitos y Retiros 6
B 6. Tablas de Traspasos. 7

éndice C *1*

ÓDIGO DEL SISTEMA 1
C 1. Código para el Servidor. 1
C.2. Código para el Cliente 24
C.3 Código para la Palm 43

éndice D *1*

NFRAESTRUCTURA DEL SISTEMA 1
D.1 Red de comunicaciones 1
D.2 Características del equipo de Código de Barras 2
D.2.1. Terminal portatil para lectura de código de barras 3
D.2.2 Cuna de transmisión de la terminal portátil a la computadora 5
D.2.3 *Software* de desarrollo para la terminal portátil 6
D 2.4. Terminales locales de código de barras 7
D 2.5. Impresora de código de barras 9
D.2 6 Rebobinadoras para impresoras de código de barras 11
D.2 7 Despachador de etiquetas 12
D.3. Características de las etiquetas y cintas de impresión. 12
D.4 Proceso de adquisición. 14
D.5 Características del equipo elegido. 24
D 5 1. Lectora portátil 24
D.5 2 Lectora local 25
D 5 3 Impresora de código de barras. 26

OSARIO *29*

BLIOGRAFIA *31*

INTRODUCCIÓN

istoria y funciones del Banco Central

Banco Central es un organismo autónomo en el ejercicio de sus ciones y en su administración, que se desempeña como banco del ado (de la misma manera que los bancos comerciales hacen las es de empresas), así como regulador y controlador del sistema anciero del país.

principal objetivo es procurar la estabilidad del poder uisitivo de la moneda nacional, fortaleciendo con ello la toría del desarrollo nacional que corresponde al Estado. Sin argo, no constituyen monopolios las funciones que el Estado rza de manera exclusiva, a través del Banco Central en las áreas acuñación de moneda y emisión de billetes.

Banco Central, en los términos que establezcan las leyes y con intervención que corresponda a las autoridades competentes, ula los cambios, así como la intermediación y los servicios ancieros, contando con la autoridad necesaria para llevar a cabo ha actividad.

conducción del banco esta a cargo de personas cuya designación á realizada por el Presidente de la República, con la aprobación la Cámara de Senadores o en su caso, de la Comisión Permanente. as personas desempeñan su encargo por periodos cuya duración y alonamiento provean el ejercicio autónomo de sus funciones, no rán tener ningún otro empleo, cargo o comisión, con excepción de ellos en que actúen en representación del banco y de los no nerados en asociaciones docentes, científicas, culturales o de eficencia Las personas encargadas de la conducción del Banco tral, sólo podrán ser removidas por causa grave y podrán ser etos de juicio político.

establecimiento se hizo realidad en 1925, gracias a los esfuerz presupuestarios y de organización del Secretario de Hacienc Alberto J. Pani, y al apoyo recibido del presidente Plutarco Elí Calles, y actualmente es una institución autónoma que se apoya tres fundamentos: su independencia para determinar el volumen crédito primario que puede ser concedido, la independencia que ha otorgado a las personas que integre su junta de gobierno y independencia administrativa de la institución. Por lo tanto tie el status idóneo para lograr su cometido fundamental, que es el procurar la estabilidad de la moneda nacional.

Por todo lo anterior, el banco goza de gran prestigi Independientemente de sus facultades legales, nadie le disputa rectoría de los sistemas en materia monetaria y las funcion reguladoras que le son propias. De ahí su indiscutible y añe autoridad intelectual y moral.

Es importante destacar que el Banco Central está organizado distintas Direcciones, pero la que nos interesa es la Dirección Emisión, ya que es para la que se desarrollará el presente trabajo

La dirección de Emisión es la encargada de fabricar los billetes circulación en el país y distribuirlos a los bancos usuario Además de recolectarlos y destruirlos si ya están deteriorados o son aptos para circular.

Problemática

El proceso de fabricación y distribución del producto del Banc Central requiere de un máximo nivel de seguridad. El propósi fundamental es evitar la fabricación de piezas falsas en el pa (en adelante se le da el mismo significado al hablar de piezas billetes), así como malos manejos en la distribución del bille auténtico.

El problema se detecta cuando se analiza que en el proceso empaque actual no se tienen unidades estándar para agrupar l piezas de forma organizada y precisa; por el contrario, se agrup en cantidades diferentes de acuerdo al criterio del personal q realizar la operación. De esta forma, es posible encontrar paquet con cinco mil piezas, seis mil piezas, etc.

Lo anterior da como resultado que el control durante el empaque las piezas sea muy vulnerable y el proceso represente un ries importante para el Banco Central.

flujo de piezas en el Banco Central actualmente sólo es ntrolado por las cantidades involucradas, sin embargo no es sible saber que paquetes se han movilizado. Esto implica una ntidad considerable de recursos humanos, materiales y económicos, í como un factor de riesgo en el manejo de las piezas.

ropuesta de solución

mo se ha mencionado, una de las prioridades del Banco Central es funcionamiento óptimo de todos y cada uno de sus procesos, sin vidar el alto grado de seguridad y control requerido durante su ecución, por lo que se ha propuesto realizar un sistema que ntrole, utilizando la tecnología de código de barras, el empaque el flujo de piezas; cubriendo así, la necesidad de identificar de nera única cada envase, saber su contenido y el uso que a éste se dé, con un mínimo de recursos humanos y garantizando un ncionamiento óptimo.

bjetivo

r estas razones, se ha planteado como objetivo principal de la sis, el desarrollo del **Sistema de Contenedores para Empaque y ujos en el Banco Central**, mostrando las ventajas que ofrece a la stitución para que desempeñe mejor sus funciones.

por esto, que el contenido de la tesis comenzará con una breve scripción de conceptos básicos utilizados durante su desarrollo, ra posteriormente describir el negocio, conocer las necesidades los usuarios, analizar, diseñar, implementar y probar el sistema Contenedores para Empaque y Flujos en el Banco Central.

nalmente se presentarán los resultados y conclusiones del sarrollo del sistema, además de la bibliografía consultada y los éndices generados para complementar el presente trabajo.

CAPÍTULO 1

CONCEPTOS BÁSICOS

En este capítulo se trata primeramente el tema de las metodologías para el desarrollo de sistemas, así como una descripción de la Metodología Orientada a Objetos y del Lenguaje de Modelado Unificado (UML: *Unified Modeling Language*) que se utilizan durante el desarrollo del sistema de contenedores. Posteriormente se describen aspectos importantes de Marcos de Trabajo, Administración de Proyectos y de la Tecnología de Códigos de Barras.

1. Metodologías para el desarrollo de sistemas

Ingeniería de *software* ofrece una metodología para producir *ftware* de forma organizada, empleando una colección de técnicas y nvenciones de notación definidas. Los pasos de producción del *ftware* suelen organizarse en un ciclo de vida que consta de rias fases de desarrollo.

ciclo de vida completo de *software* abarca desde la formulación cial del problema, pasando por el análisis, diseño, plementación y pruebas de *software*, hasta una fase operacional nte la cual se llevan a cabo el mantenimiento y las mejoras.

metodologías más comunes pueden clasificarse de la siguiente

- Orientadas a los datos
- Orientadas a objetos
- No metodológicas

Las cuales analizaremos a continuación.

1.1.1. Metodologías orientadas al proceso

Las metodologías orientadas al proceso siguen una aproximació estructurada y de tipo "arriba-abajo", se enfocan en la evaluació del problema que presenta el proceso y los flujos de datos con lo que se conectan. Son ampliamente utilizadas y soportadas en lo actuales entornos CASE de desarrollo. De manera informal podemo decir que han contribuido y contribuyen a la mejora en l comprensión del dominio del problema y de los requisitos de cliente.

Para documentar esta metodología se utilizan diagramas de contexto de flujo de datos, definiciones de almacenes de datos descripciones en lenguaje natural estructurado del proceso.

En un futuro este tipo de metodologías serán integradas dentro d las técnicas que dan soporte a la captación de clientes y a l introducción en la enseñanza de técnicas y metodologías d desarrollo.

1.1.2. Metodologías orientadas a los datos

Por otra parte, las Metodologías orientadas a datos comienzan la actividades de análisis evaluando los datos y sus relaciones, co el objeto de descubrir la arquitectura de datos subyacente; después se aplican los datos de salida de ciertos procesos, a entradas d otros, con el objeto de determinar los requisitos del proceso Para su documentación se utilizan diagramas de entidad relación diagramas jerárquicos, de dependencia, y bases de datos lógicas e tercera forma normal.

Este tipo de metodología está siendo cada vez más difundida utilizada en aplicaciones de gestión relacionadas con la prop estructura del negocio de la empresa, y para un futuro se prevé q será de gran ayuda en la gestión de procesos que involucran u gran cantidad de datos relacionados. Algunas de las metodologí serán difícilmente reemplazables por otras nuevas, debido a dependencia con la organización del propio negocio que presentan.

1.3. Metodologías orientadas a objetos

te tipo de metodologías sigue una aproximación "arriba-abajo" bre los objetos de datos, sus acciones permitidas y los quisitos de comunicación, necesarios para definir la arquitectura l sistema. Manejan diferentes clases de diagramas y se utilizan aplicaciones de todo tipo (en línea y en tiempo real), en el sarrollo de herramientas CASE y nuevos modelos de ciclo de vida, incipalmente. Es previsible que sean las metodologías de sarrollo más utilizadas en los próximos años, y que su uso no se mitará únicamente al desarrollo de modelos de *software*, sino mbién para modificar la organización de la política de negocios.

1.4. No metodológicas

te tipo de metodología se basa en la experiencia personal, y ele ir asociada a desarrollos en los que el ciclo de vida es del po "aprende según avanzas".

cuanto a la documentación, se concreta de manera prácticamente clusiva a la aplicación desarrollada, es escasa. En general, este po de metodología no es muy recomendable ya que tiene muy poco ntrol del proceso de desarrollo, y la introducción de mejoras o paración de fallos es extremadamente costosa. Por estas razones que tiende a desaparecer.

2. Metodología orientada a objetos

spués de analizar las distintas metodologías se decidió utilizar Orientada a Objetos, debido a que proporciona una forma de alizar, diseñar y desarrollar el sistema de una forma más cercana la realidad, además que en el mercado se ofrece una diversidad de rramientas CASE que son de gran ayuda durante el desarrollo del stema. A continuación se presenta una descripción más detallada esta metodología.

la metodología orientada a objetos, el procesamiento nvencional de los datos se centra en los tipos de objetos, cuya ructura de datos sólo pueda controlarse mediante los métodos de clase del objeto. Ocurran eventos que modifican el estado de un jeto. Por lo general, la programación de cada cambio de estado es ual, ya que cada objeto lleva a cabo una función especializada

la razón del envío de éstos, ni las consecuencias de su acció
Como los objetos actúan en forma individual, cada clase se pue
modificar de manera independiente lo que facilita su prueba
modificación.

El mundo orientado a objetos tiene una mayor disciplina que el
las técnicas de estructura convencional. Esto nos lleva a un mun
de clases reutilizables, donde la mayor parte del proceso
construcción de *software* consiste en el ensamblaje de clases
existentes y probadas.

Las Técnicas Orientadas a Objetos (TOO) ligadas a las herramient
CASE con un generador de códigos y un depósito (también orientado
objetos) constituyen el mejor camino conocido para construir u
verdadera ingeniería de *software*.

Las principales características del Análisis y Diseño Orientado
Objetos son las siguientes:

- *Cambian nuestra forma de pensar sobre los sistemas.* Pa
  la mayoría de las personas, la forma de pensar Orienta
  a Objetos es más natural que las técnicas del análisis
  diseño estructurado. Después de todo, el mundo es
  formado por objetos.

- Los usuarios finales y las personas de las empres
  piensan de manera natural en términos de objetos, event
  y mecanismos de activación, por lo que los diagram
  Orientados a Objetos les parecen más familiares que l
  diagramas de relación, tablas de estructura y diagram
  de flujo de datos.

- *Los sistemas suelen construirse a partir de objetos*
  existentes. Esto implica un alto grado de reutilizació
  un ahorro de dinero y tiempo de desarrollo, y una may
  confiabilidad del sistema.

- La complejidad de los objetos que podemos utiliz
  aumenta, debido a que nuevos objetos se construyen
  partir de otros.

- El depósito CASE debe contener una creciente bibliote
  de tipos de objetos, algunos comprados y otr
  construidos. Es muy probable que estos tipos de objet
  sean más poderosos conforme crezca su complejidad y
  mayoría serán diseñados de forma que se adapten a l
  necesidades de los diferentes sistemas.

- La creación de sistemas que funcionen correctamente es más sencilla con las técnicas orientadas a objetos. Esto se debe, en parte, a que las clases orientadas a objetos están diseñadas para reutilizarse, y en parte, a que están autocontenidas y divididas en métodos. Cada método se puede construir, depurar y modificar con relativa facilidad.

- Las técnicas orientadas a objetos se ajustan de manera natural a la tecnología CASE. Existen ciertas herramientas CASE elegantes y poderosas para la implantación orientada a objetos y muchas otras necesitan ciertas mejoras para controlar el análisis y diseño orientado a objetos.

otra parte la orientación a objetos es un paradigma que depende algunos principios fundamentales como son las clases y los etos, en donde una clase es una categoría genérica de objetos tienen los mismos atributos y acciones, y un objeto es una tancia de una clase cuyos atributos serán determinados de erdo a los requerimientos.

herencia es un aspecto importante de la orientación a objetos, ido a que un objeto hereda los atributos y operaciones de su se. De la misma manera una clase también puede heredar atributos cciones de otra.

o aspecto importante es el polimorfismo y se da cuando una ión puede tener el mismo nombre en diferentes clases y cada se la ejecuta de forma distinta.

encapsulamiento permite que los objetos oculten su funcionalidad otros objetos y al mundo exterior. Cada objeto presenta una erfaz para que otros objetos (y personas) puedan aprovechar su cionalidad.

mensajes son peticiones para realizar operaciones y es por io de ellos que los objetos se comunican entre sí.

lo general, los objetos se asocian entre sí y esta asociación de ser de diversos tipos. Un objeto en una clase puede asociarse cualquier cantidad de objetos distintos en otra clase con lo se define su multiplicidad.

agregación es un tipo de asociación. Un tipo agregado consta de conjunto de objetos que lo componen y una composición es un tipo cial de agregación. En un objeto compuesto, los componentes [illegible]

El lenguaje que se utiliza durante el análisis Orientado a Obje es el Lenguaje de Modelado Unificado y del cual se describen aspectos más importantes en la siguiente sección.

1.3. Lenguaje de Modelado Unificado

El Lenguaje de Modelado Unificado (UML) es la creación de Gra Booch, James Rumbaugh e Ivar Jacobson, quienes trabajaban empresas distintas durante la década de los años ochenta principios de los noventa, tiempo durante el cual cada uno dise su propia metodología para el análisis y diseño orientado objetos. Sus metodologías predominaron sobre las de competidores. A mediados de los años noventa empezaron intercambiar ideas entre sí y decidieron desarrollar su trabajo conjunto.

La finalidad de los diagramas es presentar diversas perspectivas un sistema a las cuales se les conoce como modelos. Es importa destacar que un modelo describe lo que supuestamente hará sistema, pero no dice cómo implementarlo.

UML está compuesto por diversos elementos gráficos que se combi para formar diagramas, y debido a que es un lenguaje, cuenta las reglas para combinar tales elementos. En lugar de indic cuáles son los elementos y las reglas, veamos directamente diagramas, ya que se utilizarán para hacer el análisis del siste en cuestión.

Una clase, como se mencionó anteriormente, es una categoría o gr de cosas que tienen atributos y acciones similares, y se represe con un rectángulo dividido en tres áreas, como se muestra en figura 1.1.

El área superior contiene el nombre, el área central contiene atributos, y el área inferior las acciones (o funciones). diagrama de clases está formado por varios rectángulos de este t conectados por líneas que muestran la manera en que las clases relacionan entre sí.

Los diagramas de clases son útiles en la etapa de análisis ya permiten al analista hablarle a los usuarios en su pro terminología, lo que hace posible que indiquen detalles importan de los problemas que requieren ser resueltos.

símbolo que se utiliza para representar un objeto, también es un ángulo, pero el nombre del objeto está subrayado. El nombre de instancia específica se encuentra a la izquierda de los dos tos (:), y el nombre de la clase a la derecha, como se muestra la figura 1.1.

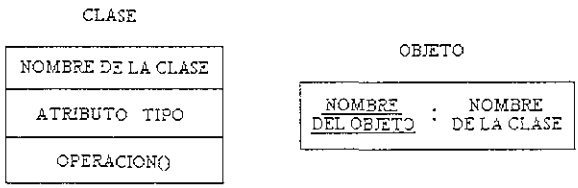


Fig. 1.1. Representación de Clases y Objetos en UML.

caso de uso es una descripción de las acciones de un sistema de el punto de vista del usuario. Para los desarrolladores ésta una herramienta valiosa, ya que es una técnica de aciertos y ores para obtener los requerimientos del sistema por parte del ario. Esto es importante, si la finalidad es crear un sistema pueda ser utilizado por la gente en general (no sólo por ertos en computación).

la siguiente figura se muestra cómo es un diagrama de casos de (Fig. 1.2.).

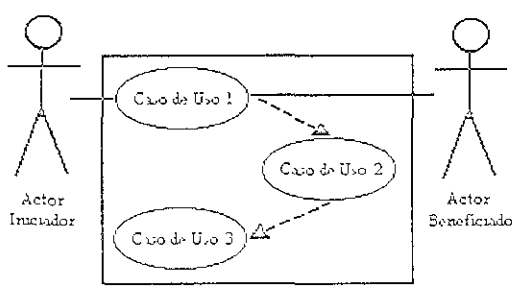


Fig. 1.2. Diagrama de casos de uso UML.

operación completa desarrollada por los actores y por el sistema un diálogo. El conjunto de los casos de uso representa la totali de las operaciones desarrolladas por el sistema. Un actor puede una persona u otro sistema.

Existe también otro tipo de diagrama, denominado Diagrama Actividades (Fig. 1.3.), que es un caso especial de un diagrama estados, en el cual, casi todos los estados son de acc (identifican que acción se ejecuta al estar en él), y casi to las transiciones son enviadas al terminar la acción ejecutada en estado anterior. Puede dar detalle a un caso de uso, un objeto o mensaje en un objeto. Sirven para representar transicio internas, sin hacer mucho énfasis en transiciones o even externos.

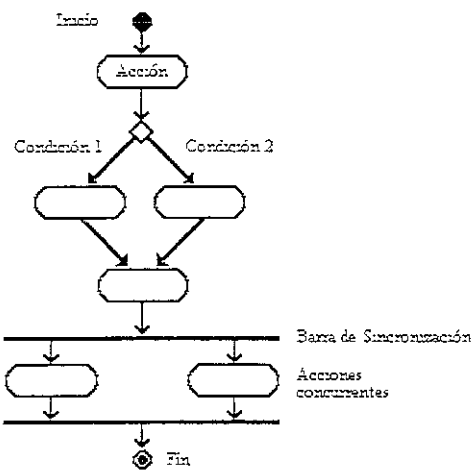


Figura 1.3. Diagrama de actividades UML.

En general, podemos decir que UML es un lenguaje de modelado p la especificación, visualización, construcción y documentación los artefactos de un proceso de sistema intensivo.

Como es un lenguaje, es utilizado para la comunicación, ya que un medio para capturar el conocimiento respecto a un tema sistema en estudio) y expresarlo. Se utiliza para comunicar "q se requiere de un sistema y "cómo" puede ser realizado, por lo es utilizado para guiar su desarrollo y para capturar conocimie a lo largo de todo su ciclo de vida.

UML se enfoca a la comprensión de un tema a través de

mulación de un modelo y su contexto respectivo. El modelo abarca conocimiento del tema, y la aplicación apropiada de este ocimiento constituye la inteligencia.

dando la unificación, UML integra las mejores prácticas de la eniería de la industria tecnológica y sistemas de información, ando por todos los tipos de sistemas, dominios y los procesos de lo de vida.

no es un lenguaje de programación visual, sino un lenguaje de elado visual; no es una herramienta o depósito de ecificación, sino un lenguaje para modelado de especificación; es un proceso, sino que habilita procesos, y finalmente, no es lenguaje cerrado propiedad de alguien, sino más bien, un guaje abierto y totalmente extensible reconocido por la ustria.

damentalmente, UML está relacionado con la captura, comunicación nivelación (disgregación en niveles) de conocimientos. Es un guaje para modelado de propósito general evolutivo, ampliamente icable, soportado por herramientas e industrialmente andarizado que se aplica a diferentes tipos de sistemas, inios, y métodos o procesos.

o lenguaje de propósito general, se enfoca a un conjunto de ceptos para la adquisición y utilización de conocimientos arejados con mecanismos de extensión.

herramientas que usa están disponibles para soportar la icación del lenguaje para especificar, visualizar, construir y umentar sistemas.

posibilita la captura, comunicación y nivelación de ocimiento estratégico, táctico y operacional para facilitar el remento de valor, aumentando la calidad, reduciendo costos y uciendo el tiempo de presentación al mercado; manejando riesgos iendo accesible para el posible aumento de complejidad o cambio.

Tecnología de marcos de trabajo

marcos de trabajo (MT) corresponden a estructuras escritas de idea, y/o conjunto de notas, para facilitar a una organización implementación de las mismas es decir que, mediante los MT se n que todo el personal de una organización se dirija en el

En el área de ingeniería de *software* se puede catalogar a los
por el propósito que cumplen y más de una categoría pue
corresponder a un MT.

La finalidad de los MT es la de mejorar los procesos de *softwar*
brindar pautas para efectuar evaluaciones de la unidad informátic
determinar la potencialidad/capacidad y la forma de realizar s
procesos, y la madurez de la organización. En algunos MT est
aspectos se encuentran más explícitos y por consiguiente ocupan
área central en el mismo. En definitiva los MT buscan:

- Mejorar los procesos de *software*
- Aumentar la productividad y la calidad
- Disminuir los costos

La decisión por parte de una organización de aplicar un MT tr
aparejado muchos beneficios a largo plazo, pero, ¿cuánto es e
disminución de costo?, ¿en qué porcentaje se puede expresar?, ¿có
varía este valor por cada MT?. Son preguntas que todavía no tien
una respuesta directa y formal.

Algunos motivos de esta situación son los siguientes:

- Los MT prácticamente son "nuevos", los más antigu
  tienen un promedio de 7 a 10 años.
  - Se necesita tiempo para ver los resultados
  analizarlos.
- Hay muy pocas organizaciones que publiquen los resultad
  obtenidos al utilizar un MT en particular y que se pres
  para dicho estudio.
  - ¿A quién le gusta mostrar sus defectos y errores?. L
  casos publicados de América Latina son muy escasos.
- La tecnología y herramientas (análisis, diseño, métrica
  etc.) informáticas evolucionan más rápidamente
  comparación con la actualización de los M
  -¿Será una carrera perdida? O ¿Será cierto que los MT s
  totalmente independientes a esta realidad?
- Todavía no se ha masificado el uso de los M
  - La decisión de adoptar un MT implica estudios
  factibilidad. Todavía no hay una exigencia de merca
  importante para inclinarse por un MT en particular.

Sin importar que marco de trabajo se elija, hay tres elementos
considerar para asegurar el éxito de la aplicación de un MT

ticular. Estos son:

- Cambio de cultura en el ámbito organizacional;
- Compromiso pleno por parte del personal con alta jerarquía;
- Educación.

Tecnología de códigos de barras

empresas actuales, que utilizan modernos sistemas de cómputo a realizar sus operaciones, han logrado administrar y distribuir información de una manera eficiente; pero si pensamos un poco re la manera en que esta información es capturada, en su oría, es tecleando la información dentro de la computadora. Los odos actuales de captura deben de ser mucho más rápidos y más uros que los manuales, ya que la dependencia en la información se maneja, puede hacer que los errores de captura se rifiquen, es decir, "si tu metes basura, obtienes basura", y si a basura se utiliza en la toma de decisiones, el resultado puede fatal. Por esta razón, el hacer más eficiente la captura es un o un reto: Códigos de barras, bandas magnéticas, y onocimiento óptico de caracteres, son algunas tecnologías para captura.

código de barras es una simbología consistente en barras curas y claras, diseñadas para la lectura óptica y odificación automática. La simbología de los códigos de barras sido desarrollada para diferentes propósitos y pueden presentar ores numéricos o caracteres alfanuméricos; pueden ser impresos muchos tipos de superficies, incluyendo etiquetas adhesivas, uetes, documentos, etc.; además se pueden colocar en papel, stico, metal, etc. Dependiendo del equipo de lectura, la ología y el tamaño, los códigos pueden ser leídos desde ersas distancias, empezando desde el contacto directo, hasta los pies de distancia.

a reconocer los códigos de barras se utiliza principalmente la tura óptica. El principio básico de esta tecnología, es que, ndo una superficie está iluminada, si ésta es clara, reflejará parte de la luz que recibe, en cambio si la superficie es ura, absorberá una gran cantidad de luz. La diferencia entre la [illegible]

haz de luz, la intensidad de la luz reflejada corresponderá con
anchos de las barras; la mayoría de los lectores de códigos
barras trabajan de esta manera, detectando la luz reflejada con
receptor fotoeléctrico sensitivo. Una vez que el conjunto de bar
y espacios es capturado, puede ser decodificado para conocer
información original contenida en el código de barras.

Con el principio de que un código de barras contiene cie
cantidad de barras claras y obscuras, existen diferentes maneras
decodificar esta información, en algunas simbologías, sólo
anchos de las barras tienen significado mientras que otras utili
también, los espacios entre las barras.

Esta información está codificada siguiendo ciertos patrones
cuanto al tamaño y ubicación de los elementos. Algunas de
simbologías de códigos de barras más comunes están basadas en
sistema numérico binario.

Por ejemplo, el código UPC (Universal Product Code) consta de
dígitos numéricos y fue adoptado por la industria norteameric
para su lectura en las cajas registradoras de los supermerca
(puntos de venta). Para fines didácticos utilizaremos la estruct
que se muestra en la figura 1.4.

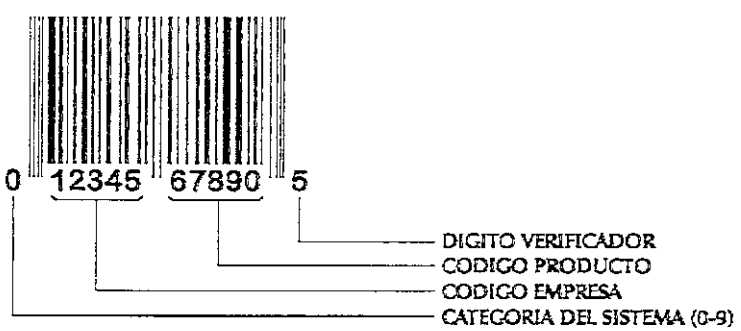


Fig. 1.4. Código de Barras UPC.

El significado de la estructura que se observa en el código
barras UPC se muestra en la siguiente tabla (Tabla 1.1.).

0	Categoría del Sistema. Las categorías que permite e código UPC son las siguientes:

	0,6,7 Códigos regulares UPC, para la mayoría de los productos. 2 Utilizado para productos de peso variable como carne, pollos, que requieren de la marcación de tienda. 3 Productos farmacéuticos o relacionados a la salud. 4 Para uso interno (sin restricción alguna de formato). Para productos marcados por el comerciante y que sean utilizados sólo para venta dentro de sus establecimientos. Por ejemplo: los códigos internos que utilizan los autoservicios nacionales. 5 Para uso de cupones. 1,8,9 Reservados para futuras aplicaciones.
2345	Corresponde al código de empresa, otorgado por la *Uniform Code Council Inc.* en EEUU, Institución encargada de administrar los códigos UPC.
7890	Corresponde al código de artículo.
	Dígito de control, que se calcula completando los 13 dígitos con un cero a la izquierda del código.

Tabla 1.1. Significado del código UPC.

muy común que un dígito verificador, también llamado carácter de trol, sea añadido a la información del código para verificar su idez, este carácter es normalmente impreso como el último de la ie, justamente antes del carácter final. El valor del dígito ificador es calculado con base en los valores de cada uno de los acteres de datos del código. Para ser verificado el carácter, recalcula el mismo después de ser leído, para asegurar que los acteres de datos que fueron leídos dan como resultado el dígito verificación correcto.

código de barras por sí solo no es de gran ayuda, por lo que sitios tener un dispositivo lector. Este dispositivo es una ad que emplea tecnología electro-óptica para determinar la

símbolos del código de barras. Existen diversos tipos de producto entre ellos los lectores de pluma, pistolas láser, ranuras lectura, lectores fijos, etc.

Como ya se mencionó anteriormente, existen diferentes tipos códigos de barras, cada uno creado para satisfacer las necesidad y demandas de cada aplicación en particular. Las simbologías comunes son UPC y sus versiones europeas EAN, Código 39, y 2 de intercalado; sin embargo existen otras que también tienen un amp uso como Codabar, Código 128, Código bidimensional PDF417, Postn etc. En la figura 1.5. podemos ver los diferentes códigos de barr más ampliamente utilizados.

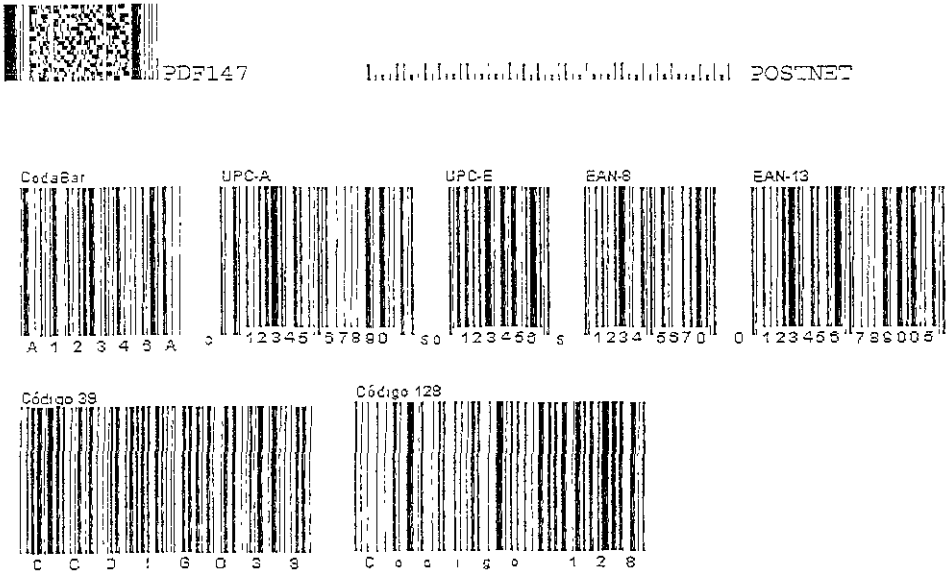


Fig. 1.5. Códigos de Barras ampliamente difundidos.

Cada uno de los códigos mostrados presenta particularidades que hacen útil en diferentes ambientes en los que se desea reconocimiento automático.

Diversos factores son significativos cuando se selecciona dimensiona un código de barras, y esto puede depender del giro la industria, el contenido del código, la velocidad de impres del código de barras, etc.

Esta tecnología tiene muchas ventajas que podemos identifi claramente, sin embargo las principales son la velocidad,

cisión y la confiabilidad.

menciona la velocidad como una ventaja, porque el código de ras puede ser de un 33% a 250% más rápido que una captura hecha medio del teclado; al utilizar códigos de barras, el porcentaje errores es muy bajo (en estudios efectuados se encontró que para captura de 3 millones de caracteres, si era realizada por medio teclado, existían 10,000 errores, mientras que por medio de igo de barras sólo existió 1); y finalmente la confiabilidad en lector láser es muy alta, si la impresión del código es recta.

todas estas razones los códigos de barras están ampliamente undidos en la actualidad, su uso se ha manifestado casi en lquier giro de la industria, enfocándose principalmente en ares donde se manejan grandes volúmenes de información, además requerir una captura ágil y sin errores. Su principal uso se da empaques de productos, credenciales de identificación, catálogos istas de precios, etiquetas para productos, etc.

Administración de proyectos

administración de proyectos es el arte/ciencia de dirigir y rdinar recursos humanos y materiales, a lo largo de la vida de proyecto, usando técnicas determinadas para obtener el costo, mpo, calidad y cantidad predeterminada.

objetivos principales que se persiguen en la administración de yectos son:

- Realizar una adecuada planeación.
- Facilitar la comunicación.
- Verificar el cumplimiento de los objetivos y las especificaciones.
- Optimizar el uso de los recursos y del tiempo.
- Resolver conflictos.
- Coordinar la implementación del proyecto.
- Controlar el proyecto durante su ciclo de vida.

Las estructuras de las empresas están orientadas a facilitar circulación de información para la toma de decisiones relativa la actividad regular de las mismas, y por consiguiente, son p efectivas para realizar otro tipo de actividades como proyecto programas. Estas actividades se caracterizan por la combinación recursos, humanos y no humanos, agrupados con el fin de lograr propósito determinado para la organización. Las relacio existentes entre la estructura organizativa y la generada der del grupo, establece conceptos sobre las dependencias autoridades, llegándose a la organización matricial.

La organización matricial es un esquema de trabajo para desarrollo del sistema de contenedores que está dividido de acue a dos criterios: La plataforma de desarrollo y la funcionalida procesos operativos.

Por plataforma se divide en:

- Servidor
- Cliente
- Lectoras portátiles de código de barras

Y por funcionalidad se organiza en:

- Empaque
- Autorizaciones y Entregas de órdenes de emisión
- Depósitos y Retiros
- Traspasos y Remesas
- Operaciones generales

El objetivo es que cada uno de los integrantes del equipo desarrollo, sea asignado a una de las intersecciones entre es dos criterios, de modo que cada persona trabaje en una s plataforma (Sybase, Power Builder o Codewarrior) y en un s aspecto de funcionalidad del sistema. La tabla 1.6 muestra asignación de recursos humanos propuesta.

De esta forma se asignan actividades de manera lógica, y la ca de trabajo se divide equitativamente en la mayoría de actividades.

IONALIDAD	PLATAFORMA		
Procesos	Servidor	Cliente	Lectoras
	Líder de plataforma Sybase	Líder de plataforma PowerBuilder	Líder de plataforma CodeWarrior
Líder del roceso de Empaque	**Programadores**	**Programadores**	**Programadores**
Líder del roceso de orización y Entrega	**Programadores**	**Programadores**	**Programadores**
Líder de epósitos y Retiros	**Programadores**	**Programadores**	
r de Remesas Traspasos	**Programadores**	**Programadores**	**Programadores**

Tabla 1.6. Propuesta de asignación de recursos humanos.

actividades que se realizan son: revisión de procesos rativos, definición de requerimientos del usuario, codificación jecución de pruebas, además que se tiene un mejor desempeño; rdinación y control en el desarrollo del sistema.

embargo, hay actividades en las que debido a la gran ndencia entre las diferentes intersecciones antes expuestas, rán realizarse en conjunto por todos los miembros del equipo, es el caso del diseño, la planeación de pruebas integrales y la antación en producción del sistema.

personas asignadas en las diferentes plataformas de un mismo lo funcional, formarán a su vez un equipo de desarrollo.

endrá un coordinador de cada módulo funcional, uno por cada aforma, y un líder de proyecto.

endo todo esto en cuenta se definen roles y funciones cíficas para el líder de proyecto, el coordinador de plataforma desarrollo, el coordinador de modulo funcional y para cada grante de los módulos.

líder del proyecto es el responsable de planear, coordinar y rolar todas las actividades de desarrollo del sistema. Está al nte del desarrollo de cada una de las actividades,

verificando que éstas se realicen de manera eficiente; es decir, el tiempo planeado y con la calidad adecuada, y en caso de requerido, tomará las medidas preventivas o correctivas que s convenientes o necesarias, a fin de evitar problemas previsible corregir los que se susciten, lo antes posible. Para ello, ten la facultad de hacer cambios en la asignación de actividades recursos, negociar los requerimientos con los usuarios y reduci aumentar el tiempo asignado a las actividades, y si la situación requiere, incluso omitir aquellas que sean prescindibles para desarrollo del sistema o para que éste cumpla los requerimien indispensables.

También tendrá la responsabilidad de coordinar y supervisar desarrollo del sistema con el fin de verificar que éste cumpla requerimientos generales y que haya uniformidad en la funcionali de los diferentes módulos y plataformas, así como en la interacc entre ellos.

Deberá mantener una buena comunicación con los coordinadores de diferentes plataformas, así como los de los diferentes módu funcionales para prevenir o corregir problemas generales en desarrollo del sistema.

Por otra parte, el líder de proyecto también tendrá la facultad tomar decisiones y hacer los ajustes que sean convenientes p enfrentar cualquier situación no prevista en este plan.

Los coordinadores de cada plataforma de desarrollo tendrán responsabilidad de verificar la uniformidad e integración de diferentes módulos funcionales en su plataforma; esto significa situaciones similares de los diferentes módulos funcionales, de ser resueltos del mismo modo, y buscarán evitar la duplici mediante la reutilización de código (objetos o procedimient comunes, cuando sea posible.

Promoverán que se sigan estándares tanto en la documentación de etapas de análisis y diseño, como en la escritura de código, modo que éstos sean claros y entendibles para el resto de integrantes del equipo de desarrollo. Deberán buscar y propo soluciones a los diferentes problemas o situaciones que presenten en su plataforma y apoyarán en la definición de planes pruebas integrales e implantación de todo el sistema.

Para realizar estas funciones, deberán estar en estre comunicación con los coordinadores de las otras plataformas módulos funcionales, así como con el líder del proyecto.

Por otra parte, los coordinadores de cada módulo funcional tend

responsabilidad de controlar la uniformidad, calidad e egración del desarrollo de su módulo en las diferentes taformas. De igual manera, deberán coordinar y supervisar la formidad en la funcionalidad y la calidad de su módulo con ación a los otros módulos, así como en la interacción con éstos, lo que deberán involucrarse con profundidad en el conocimiento la operación y funcionalidad de todo el sistema, de los uerimientos y diseño generales tanto del sistema, como de cada ulo, y en general de todo aquello que les permita tener una prensión clara de la relación de su módulo con los demás.

erán asegurarse de obtener todos los requerimientos relativos a módulo y verificar que estos sean cubiertos y asignarán y rdinarán las actividades de desarrollo de su módulo, entre los mbros de su equipo, por lo que deberán tener control sobre as, vigilando la eficiencia en su ejecución (calidad y tiempo neado); en caso de detectar desviaciones, podrán tomar isiones que les permitan corregirlas, siempre y cuando los blemas o las soluciones a los mismos no afecten a otros módulos cionales, a toda una plataforma o incluso a todo el proyecto, de el caso, deben comunicar la situación al líder del proyecto, el fin de buscar una solución integral.

earán, coordinarán y controlarán pruebas que permitan asegurar calidad de su módulo, observando que se obtengan resultados cuados, que se realicen todas las afectaciones correspondientes ue se cubran todos los requerimientos además del apoyo que deben porcionar en la definición de planes de pruebas integrales e lantación de todo el sistema.

almente, es importante mencionar que todos los integrantes del ipo de desarrollo deberán apoyar en la definición de los uerimientos específicos de su módulo, y vigilarán que éstos sean ertos satisfactoriamente, por lo que serán responsables de todo que desarrollen, y deberán probar cada uno de los objetos, tanas, procedimientos, triggers, etc., verificando que el tema responda de manera apropiada en todos los posibles enarios de operación; esto es, tanto con casos ideales, cuando entradas del sistema son correctas, como con datos incorrectos anejo inapropiado por parte de los usuarios.

bién serán responsables de vigilar el cumplimiento de los mpos planeados en las actividades que les sean asignadas, en o de enfrentar algún problema que no pueda ser resuelto de era independiente, deberán comunicarlo a su coordinador de ulo funcional, con el fin de buscar una solución.

os los desarrolladores deberán apoyar ideas y

propuestas tanto para mejorar la calidad del sistema, como p buscar soluciones a los problemas que se presenten.

Teniendo todo esto en cuenta, el esquema de trabajo se realizará tal manera que los integrantes de cada módulo funcional se responsables de revisar los procesos operativos, definir requerimientos y elaborar los documentos correspondientes, ju con el o los usuarios involucrados.

Tanto los procesos operativos, como los requerimientos definid serán expuestos al resto de los integrantes del proyecto para análisis, con el fin de que todos estén informados de funcionalidad total que deberá cubrir el sistema y encont aspectos comunes entre los diferentes módulos. Esto último ayud a evitar duplicidad en la creación de diferentes obje (variables, objetos de control, ventanas, *datawindows*, tabl *stored procedures*, etc.) y permitirá establecer los enlaces comunicación o de interacción entre los mismos, además de permitirá a todos los miembros del equipo, tener una visión clara de cómo afecta a los demás, lo que cada uno e desarrollando.

Se plantea que las exposiciones se hagan semanalmente y deberán claras y breves, para poder revisar todos los módulos funcional Esto significa que no se esperará a que se haya concluido con análisis completo de un módulo para presentar sus resultados, s que cada semana, cada uno de los equipos expondrá a los demás avance para que éste sea analizado por todos, con el fin encontrar aspectos comunes, divergencias, etc.

Cada equipo será responsable de integrar las sugerencias y acla con los usuarios las dudas u observaciones que hayan surgido c resultado de la exposición y análisis de su avance, por lo algún integrante del equipo deberá tomar nota de las mismas, incorporarlas a una bitácora que se creará con el fin de dar seguimiento.

En el diseño es necesario tomar en cuenta algunos aspec importantes, ya que esta actividad se realizará con participación de todos los analistas miembros del proyecto, por que, para hacer más dinámica esta actividad, primero cada equ elaborará de manera general una propuesta de diseño del mód funcional que le corresponde y su integración con el resto sistema. El objetivo de este ejercicio será el de encontrar analizar diferentes alternativas de diseño, por lo que deb considerarse la manera en que su módulo se integraría al sistema general y a otros módulos con los que tenga relación.

analizará en grupo ventajas y desventajas de las diferentes puestas con el fin de producir una propuesta de diseño integral ivel general para que, con esta base, cada equipo continúe con diseño específico de su módulo.

eriormente se llevará a cabo la codificación, cuyas actividades n planeadas y controladas a detalle por los coordinadores de módulo funcional. El avance, resultados y problemática serán sados de manera general, en reuniones semanales entre el líder proyecto y los coordinadores de los diferentes módulos ionales, y si es conveniente, otros integrantes del equipo de rrollo participarán en estas reuniones.

analista y cada programador probarán de manera individual la e que le corresponde y las pruebas integrales de cada módulo n planeadas por el equipo correspondiente, y serán controladas su coordinador, mientras que las pruebas integrales de todo el ema serán planeadas y controladas por el líder del proyecto y coordinadores de los diferentes módulos funcionales, y de las intas plataformas de desarrollo.

lmente, el líder del proyecto en conjunto con los diferentes dinadores hará el plan de implantación, y coordinarán y larán su ejecución por parte de todos los miembros del equipo Desarrollo.

este capítulo se trataron conceptos básicos para el sarrollo del proyecto y que durante el transcurso del esente trabajo serán complementados.

el siguiente capítulo se describirán y analizarán los ocesos que realiza el Banco Central durante el empaque y los ujos de efectivo.

CAPÍTULO II

MODELO DEL NEGOCIO

n este capítulo se describe la forma como funciona el Banco entral y se identifican los procesos más importantes que se levan a cabo para la realización de sus funciones.

1. Descripción del negocio

función principal del Banco Central, como se mencionó eriormente, es procurar la estabilidad del poder adquisitivo de moneda nacional, y para poder cumplir con esta responsabilidad va a cabo diferentes procesos que tienen que ver con la emisión a distribución de piezas.

la Figura 2.1 se muestra la forma en que está organizado el co Central.

tro de la organización presentada, la Dirección de Emisión es la tiene como responsabilidad la función de emitir y distribuir zas, y sobre la que se basa este trabajo

ha Dirección tiene la facultad de autorizar emisiones de piezas a su circulación, y lo hace con base en el análisis de estudios entados a conocer los requerimientos de la población, de esta era, planifica las emisiones, de tal forma que se garantice el ecimiento en un balance tal que no haya escasez o exceso de guna de las denominaciones que están en circulación

Además, tiene la responsabilidad de la fabricación de las piez el cual es un proceso complejo donde intervienen distintas área factores para lograr el producto deseado. En este aspecto, desempeña como una fábrica normal, pero debido a que el produ que obtiene es de gran valor, cuenta con grandes medidas seguridad que van desde tener personal específicamente prepara hasta la más avanzada tecnología en esta materia.

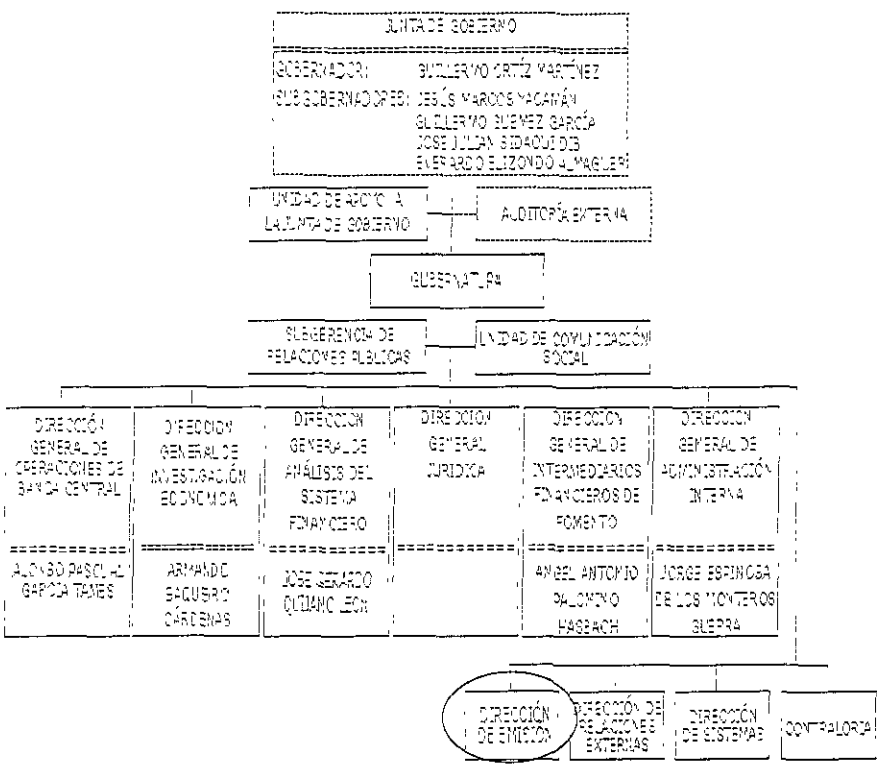


Figura 2.1. Organigrama del Banco Central.

El Banco Central también se desempeña como el Banco de los Banc A su vez, delega a la Dirección de Emisión, la responsabilidad salvaguardar en sus bóvedas, los depósitos que éstos realizan. como también de atender todos los retiros que los bancos demanden, siempre y cuando se apeguen a las reglas determinadas el Banco Central para tener un control de la cantidad de circula en el país.

a de las funciones de la Dirección, es hacer llegar las piezas a os los lugares del país. Esta es una misión difícil que necesita una gran cantidad de recursos humanos y materiales, todos bajando en perfecta coordinación para que en cualquier rincón de República Mexicana la población cuente con los medios para uirir los bienes necesarios para su supervivencia. La forma de rar esto es a través de envíos a Sucursales, localizadas en los tos más estratégicos del país. Cada sucursal distribuye y olecta el efectivo de los bancos de la región por medio de responsales, los cuales son sucursales de los bancos usuarios la facultad de realizar operaciones con el Banco Central. Una ursal del Banco Central, es entonces, una instancia que tiene o objetivo principal la distribución y recolección de piezas en región a la que corresponde, para después hacerlo llegar a la cina Central ubicada en la capital del país.

un ciclo constante en el que se involucran las funciones antes critas el Banco Central, a través de la Dirección de Emisión, cura su objetivo primordial.

ontinuación describiremos a detalle los procesos más importantes se llevan a cabo y que nos interesan para el desarrollo del sente trabajo.

1) Autorización de Emisión y Órdenes de Fabricación
2) Empaque de piezas
3) Armado y Desarmado de envases
4) Remesas
5) Traspasos
6) Depósitos
7) Retiros

Autorización de emisión y órdenes de fabricación

la Autorización de Emisión se hace un estudio de las rantes variables económicas del país, a través del cual se ma un estimado de las piezas, que deben fabricarse en el año iente. A partir de este estimado, se hace una propuesta formal Dirección [illegible]

de Gobierno del Banco Central, autoriza la emisión de efectivo.

Una vez autorizada la emisión, se establecen las caracterísi
que deben imprimirse en cada denominación y se planean órdenes
fabricación, que serán atendidas en los tiempos establecidos.
cada orden de fabricación se especifica el monto por denominac
que se debe tener para determinada fecha, se compran los insu
necesarios y se establece un programa de producción. Debido a
no es fácil tener todos estos recursos en perfecta sincronía,
órdenes deben realizarse, mínimo, con un año de anticipación y
deben cumplir al pie de la letra.

Cuando una orden de fabricación es atendida, se hacen entre
parciales de la orden. Las piezas entran a las bóvedas ya con
valor e inmediatamente forman parte de las existencias del Ba
Central, por lo que dejan de ser piezas sin valor y puede empeza
circular por el país.

De esta forma, la Autorización de Emisión queda satisfecha ha
que todas las órdenes de fabricación planeadas para ésta,
atendidas y las piezas, producto de la fabricación, son entrega
para su circulación.

2.3. Empaque de piezas

El empaque es una actividad que consiste en agrupar las piezas
unidades de empaque.

Para llevar a cabo el empaque se realizan distintos procesos:

- El corte de las hojas en piezas, a través de una Lí
semimanual, y su agrupamiento en unidades de empaque
un área de Empaque.
- El corte y empacado automático a través de una máquina.

A continuación describiremos ambas alternativas.

2.3.1. Corte de hojas

El corte de hojas se hace a través de una línea semimanual,
involucra a una guillotina para cortar las hojas de papel moneda
piezas, y al personal capacitado para hacer una revisión de
calidad de éstas.

a que las hojas pasen por este proceso, es necesario tener un grama de producción, que se toma como referencia para verificar los lotes de hojas que se van a procesar, correspondan con los ecificados en dicho plan.

vez que se tiene el plan de producción, los lotes de hojas son vados al área de la línea semimanual donde se van procesando uno no. Durante esta actividad se toma cada resma del lote que se á trabajando, y se pasa por la guillotina donde cada hoja es tada en piezas.

forme se obtienen las piezas éstas se integran en mazos, para pués pasar por una máquina recontadora, que verifica que por las tro esquinas del mazo haya mil piezas. Esta actividad se realiza el fin de detectar faltantes.

ndo se tiene el mazo recontado, éste pasa por una línea de streo conformada por distintas personas, cuya función es ectar errores en la impresión y calidad de las piezas, cada una ellas enfocándose en ciertas características. Esto es con el pósito de asegurar la calidad del producto.

durante la revisión de calidad se da el caso de que un mazo ase la cantidad máxima de errores permitidos, es necesario lizar una compensación, que consiste en sustituir el mazo con ores por otro que no tiene errores, y el cual se toma de un lote nominado serie especial) previamente escogido para estos casos y es cortado antes que todos.

mazos sin errores son agrupados como se muestra en la Figura ., colocando dos cintas de plástico en el mazo, para que con e agrupamiento los mazos se mantengan unidos hasta ser acados.

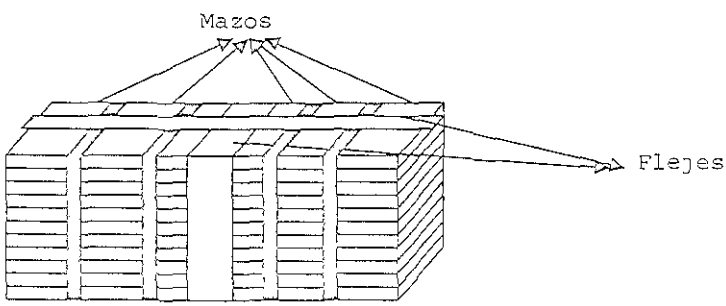


Figura 2 2. Mazos con fleje cruzado para denominación alta.

correspondiente a su denominación, para que posteriormente és sean llevadas al área de empaque, de la cual hablaremos continuación.

La figura 2.3. muestra una plataforma, vista desde arriba, con l mazos flejados que contiene.

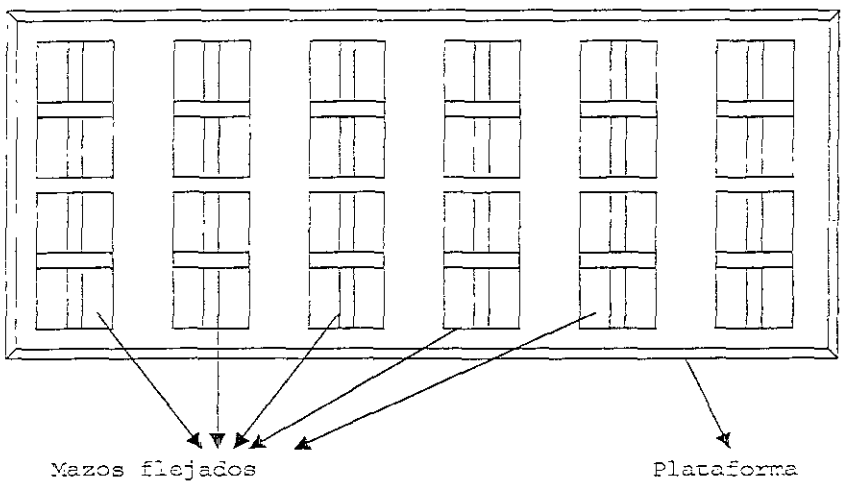


Figura 2.3. Plataforma con mazos flejados.

2.3.2. Área de empaque

Esta área se encarga de empacar las piezas en paquetes o bolsas.

El área de empaque recibe plataformas, con mazos flejados. De es forma el efectivo se va empacando de acuerdo a un plan producción.

Con las plataformas en el área, se procede a tomar cada uno de l mazos flejados y se les coloca una manga de plástico, para despu pasar por una máquina selladora que envuelve los mazos en plástico y lo sella. Una vez que se tiene a los mazos envueltos sellados, pasan por un horno termoencogible, el cual hace que plástico se acople a la forma que tienen los mazos flejados y l mantenga unidos, formándose en ese momento paquetes de efectivo.

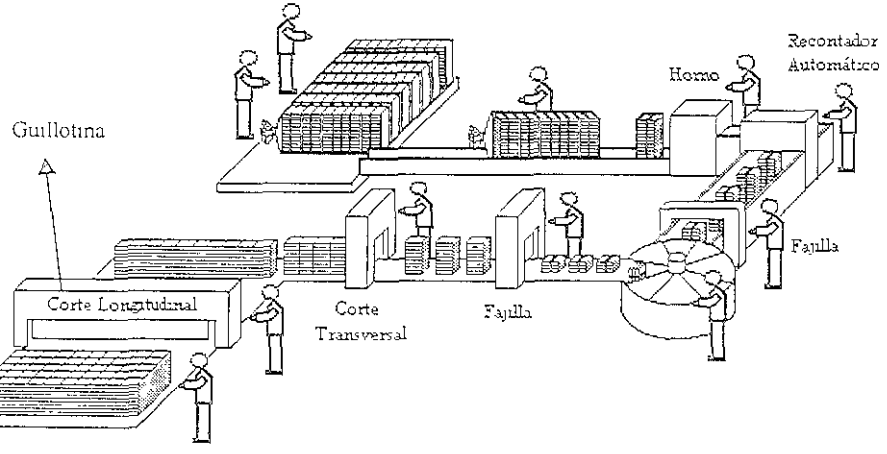
Una vez que se tienen los paquetes, son puestos en bolsas plástico, que son cerradas, con una cinta de plomo, especialmer fabricadas para la Dirección, y que se denomina machimbre.

almente, las bolsas, se colocan en plataformas para que cada una ellas se transporte a las bóvedas correspondientes para su acenamiento.

.3. Empaque de piezas de forma automática

empaque en esta área es un proceso que se lleva a cabo, icamente, en una máquina(que llamaremos de corte y empaque), se encarga de cortar y empacar el efectivo de forma automática. e procedimiento únicamente se utiliza para empacar billetes de a denominación, debido a que es muy complicado y tardado ificar la configuración de la máquina para realizar el corte de es de hojas de bajas denominaciones, ya que el tamaño de la hoja distinto a la de las denominaciones altas.

La figura 2.4. se muestra la línea de producción de la máquina corte y empaque.



ig. 2.4. Línea de producción de la máquina de corte y empaque.

os los lotes de hojas que entran en el área cumplen con un plan producción y son procesados uno por uno. Para procesar un lote hojas, se toman cada una de las resmas que lo componen y son stas en la línea de producción.

esta manera, tenemos que el proceso inicia cuando se toma la mera resma del lote, y para por una guillotina que realiza un t long itudinal. Después, cada hija de la resma es cortada en forma

transversal, para que queden grupos de quinientas piezas. A ca grupo se le coloca una cintilla de papel, también llamada fajil con lo que se obtienen fajillas con quinientas piezas. Cada fajil se coloca en un carrusel donde existe una persona que se encarga revisar la calidad de las piezas enfajilladas, si una fajil contiene un número de piezas con error mayor al permitido, entonc se sustituye por otra fajilla, que como se explicó en la sección línea semimanual, es de un lote de serie especial.

Después que las fajillas son revisadas, se juntan dos de ellas y les coloca otra cintilla de papel, para que así ambas conformen mazo.

Cada mazo pasa por una máquina recontadora, en la cual se verifi que por las cuatro esquinas del mazo se tengan mil piezas.

Posteriormente se les coloca el fleje cruzado, formando paquete los cuales pasan por un horno termoencogible que los envuelve en plástico.

Tras formar los paquetes, se colocan en una bolsa y ésta se cier con un machimbre. Por último se acomodan las bolsas sob plataformas que posteriormente se transportan a las bóvedas para almacenamiento.

Al final del turno se hace un resumen de todo el material que f procesado en el área y si hay lotes de hojas pendientes procesar, se programan para el otro día dándoles la más al prioridad.

De esta forma, se han descrito las dos rutas que se siguen pa empacar el efectivo, siempre y cuando este provenga de una línea producción determinada. Sin embargo, existen procesos en lo que necesario empacar y desempacar efectivo, que no tenga éste mis origen, y el cual describimos a continuación.

2.4. Armado y desarmado de envases

En la Dirección de Emisión existen procesos en que las piezas, ya están empacadas, tienen que ser vueltas a empacar para cump algún requerimiento; como es en el caso de los retiros y remesas, mayoritariamente. Entonces, es necesario prim desempacar las piezas y después empacarlo en un nuevo envase. Es dos pasos son denominados como desarmado y armado, respectivament

ero veamos el proceso de desarmado. Consiste en desarmar un se(paquete o bolsa) con el fin de dejar su contenido libre para pueda ser usado en procesos posteriores.

- En el caso de que se desarme una bolsa, se quita el remache que la mantiene cerrada, y sólo se puede hacer bajo la supervisión del personal encargado de controlar los flujos. Esto con el fin de que no se abran bolsas por personas que no estén autorizadas a hacerlo.

- Al abrir las bolsas quedan paquetes libres que mantienen envuelto el efectivo en una película plástica, y para desarmarlos se utiliza una navaja para rasgar el plástico, cuidando de no dañar los mazos que lo componen. La apertura de paquetes también se hace bajo la supervisión del personal encargado del control de los flujos.

- El desarmado de mazos es un caso excepcional, y se realiza prácticamente cuando las piezas contenidas van a ser destruidas. En este caso no es necesaria la supervisión del personal encargado del control, ya que el desarmado se refleja en las existencias de la bóveda de donde salieron para destruirse y además se elabora un reporte diario, con la cantidad de piezas destruidas. Por el contrario en el caso de los paquetes y bolsas no se reporta cuantos fueron desarmados y es por eso que es necesaria la supervisión durante estas actividades.

- Por otra parte, el armado es un proceso que de forma natural sigue al desarmado, ya que se utilizan los envases que quedaron libres para agruparlos en un nuevo envase. Así, por ejemplo, se puede armar una nueva bolsa con paquetes de otras bolsas, e incluso se puede armar una nueva plataforma colocando en ella bolsas de otras plataformas.

- Para que se realice el proceso de armado, es necesario que exista un informe donde se detalle el motivo por el cual se va a armar el nuevo envase, especificando cual es el proceso en el que se usará; como puede ser un traspaso, un retiro, etc.

- El armado y desarmado de envases se lleva a cabo en un área de preparación, especialmente dedicada para este propósito

- Una vez que ya se tienen las piezas empacadas, es disponibles para empezar a circular y entregarse a bancos usuarios.

2.5. Remesas

Una remesa es una transferencia entre dos sitios que no encuentran en el mismo lugar físico.

Una remesa puede ser enviada a:

- Bancos Locales
- Sucursales
- Corresponsales

El término de remesa también se utiliza cuando se envía efectivo Sucursal a Sucursal, de Sucursal a Corresponsal o de Corresponsa Corresponsal, mientras que se denomina Concentración al proceso envío de efectivo de Sucursal o Corresponsal a la Oficina Centr o también de Corresponsal a Corresponsal.

Este procedimiento comienza con la solicitud de envío de reme programada de acuerdo a las necesidades de efectivo de la reg destino.

Posteriormente se prepara la remesa a través de operaciones Armado y Desarmado. Durante la preparación se tiene un control cada una de las remesas para no mezclarlas y se envíen a destinatarios incorrectos.

Una vez que se tiene la remesa lista, se entrega a la empr encargada del transporte, junto con un pase de salida del efecti

Al sacar las remesas de la bóveda se debe revisar de nuevo que la remesa correcta, si es así, entonces se confirma el envío de remesa y se le da salida. Después se manda el detalle de la rem al Centro de Captura para que se efectúe el registro y se afec así, las existencias de la bóveda de que sale el efectivo.

Al momento que la remesa llega a su destino, se llena el form donde se detalla la recepción y se registra este movimiento través del Centro de Captura de donde se recibe.

esta forma, se distribuye el efectivo a todas las sucursales del co Central a lo largo de la República Mexicana, y éstas a su vez as sucursales de los bancos.

6. Traspasos

traspaso se define como un procedimiento en el que se moviliza ctivo entre dos bóvedas distintas.

traspaso se puede iniciar por dos vías:

- Solicitud de efectivo por parte de una bóveda giradora a una bóveda pagadora
- Envío de efectivo de una bóveda pagadora a una bóveda giradora sin una solicitud previa

el caso de que el traspaso se origine por una solicitud, se ecifica en ella la cantidad de efectivo que necesita la oficina adora.

a solicitud es atendida por la oficina pagadora, la cual tiene responsabilidad de preparar el efectivo que sea necesario e icar también, si es el caso, el número de envase que lo tiene. Para la preparación de la solicitud se realiza el proceso Armado y Desarmado que se describió anteriormente.

ndo está listo el efectivo, entonces se transporta de la bóveda adora a la bóveda giradora. El traslado es supervisado por el sonal encargado del control de flujos a fin de validar la ración.

el momento de recibir el efectivo, la oficina giradora realiza revisión física de los envases que lo contienen, a fin de que resen a la bóveda sin presentar anomalías durante su transporte. vez revisados los envases, se realiza la documentación que la la operación de recepción, confirmando su contenido, ominación y estado físico, e informando al centro de captura de na operación para que se afecten las existencias de las bóvedas olucradas.

el caso de que el traspaso sea por el envío de efectivo de la eda pagadora a la bóveda giradora sin previo aviso, solo se ilada el efectivo, se recibe y se informa al centro de captura a vés de un formato que así lo especifica.

2.7. Depósitos

Un depósito es el ingreso de efectivo, proveniente de los banc usuarios, a las bóvedas de la Dirección a través de una ventanill

La ventanilla es un compartimento el cual sirve para realizar intercambio de efectivo entre el representante del banco usuario el de la Dirección. Esta ventanilla tiene una puerta exter destinada para el acceso del representante del banco usuario y u puerta interior para el personal de la Dirección, en el interior depositan los paquetes con el efectivo. La figura 2.5. muestra vista lateral de una ventanilla.

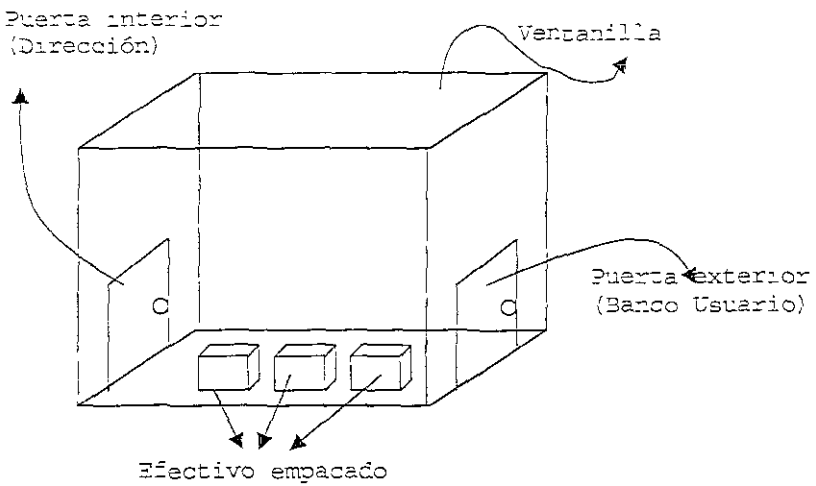


Figura 2.5. Vista lateral de la Ventanilla.

El proceso inicia cuando el representante del banco usuario llega las instalaciones de la Dirección con el efectivo a deposita Elabora una ficha de depósito y espera a ser atendido por u ventanilla.

Una vez que una ventanilla lo puede atender, se abre la puer exterior de ésta para que el representante del banco pueda ingres el efectivo con la ficha de depósito correspondiente. Cua termina de introducir el efectivo, sale de la ventanilla y puerta exterior se cierra.

Inmediatamente después se abre la esclusa interior, e ingresa personal de la Dirección; revisa que los empaques se encuentren

eraciones y verifica que el número de empaques recibidos cuerde con el detallado en la ficha de depósito, en caso de que uentre algún empaque alterado (rasgadura, mal cerrado, etc.) sa al representante del banco para que lo cambie. El personal de Dirección sale de la ventanilla e ingresa al representante del co para cambiar el empaque y actualizar la ficha de depósito.

pasos anteriores se repiten hasta que el personal de la ección determina que los empaques están en buen estado.

paso siguiente es realizar un muestreo del efectivo, eligiendo azar uno o varios paquetes, dependiendo de la cantidad del ósito, dejando los demás dentro de la ventanilla. Cada paquete abre, y se revisan cuidadosamente las piezas que contiene. Si encuentra un número de piezas con error mayor al permitido, el ósito es rechazado por completo. Se llena entonces, una forma las observaciones realizadas para justificar el rechazo y se nite de nuevo el ingreso del representante del banco, a la tanilla, para que retire todos los empaques.

caso de que no haya piezas con error o el número de éstas no ase el límite permitido, se acepta el depósito y los paquetes se sladan a las bóvedas de la Dirección.

último, para afectar las existencias de las bóvedas, se manda ficha de depósito al Centro de Captura donde la operación es istrada.

Retiros

retiro, es el proceso mediante el cual los distintos bancos que ten en el país disponen del efectivo almacenado en las bóvedas a Dirección.

banco tiene derecho a realizar un retiro preavisado y/o un ro urgente al día dentro del horario que la Dirección blezca.

el caso de los retiros preavisados, el banco usuario envía la citud un día antes al que se desea hacer el retiro.

el caso de los retiros urgentes, la solicitud se envía el mismo en que se pretenda hacer el retiro.

por efectivo a las instalaciones de la Dirección, éste po retirarlo. En el caso que la Dirección no pueda atender solicitud, entonces se le hace saber al banco usuario anticipación.

Todos los retiros son preparados previamente, lo que implica que lleven a cabo operaciones de Armado y Desarmado, para que el ba usuario pueda disponer del efectivo.

Cuando el representante del banco usuario llega a las instalacio por el efectivo, entrega la boleta de retiro al personal de Dirección, para posteriormente colocar en la ventanilla[1], paquetes previamente preparados para ese banco usuario. representante del banco usuario ingresa por los valores y se lleva.

Finalmente, la boleta de retiro se envía al Centro de Captura p que se registre la salida del efectivo y las existencias en bóve se afecten.

Si por algún motivo el banco no recoge el retiro que solicitó, sancionado de acuerdo a las normas que el Banco Central, a tra de la Dirección establezca.

Ahora que tenemos la información de cada uno de los procesos qu intervienen, en el siguiente capítulo analizaremos la necesidades de los usuarios para definir los alcances de sistema.

[1] La ventanilla que se hace referencia, es la misma que se usa en depósitos.

CAPÍTULO III

INGENIERÍA DE REQUERIMIENTOS

este capítulo se definen los requerimientos de información
ra los procedimientos implicados en el funcionamiento del
nco Central.

Análisis del problema

el desarrollo del sistema de contenedores es indispensable la
niería de requerimientos, ya que a través de esta actividad se
enen las necesidades del usuario, una vez que éste solicita la
matización de los procesos que lleva a cabo.

ngeniería de requerimientos, para el sistema de contenedores,
rende la conducción de una serie de entrevistas para determinar
usuarios estarán involucrados en el proyecto, así como la
tificación de los procesos que serán automatizados, para que de
forma, se obtenga una lista narrativa que contenga las
iones existentes que deben reimplantarse, las nuevas que
sitan añadirse y los criterios de desempeño del nuevo sistema.

productos que se obtienen de esta actividad son documentos en
que se describen los requerimientos del usuario y que sirven
un contrato entre la parte usuaria y la parte de sistemas, ya
todos los documentos son firmados por cada una de las partes y
cuo acuerdo. Si durante alguno de los procesos posteriores en
arrollo del sistema, el usuario cambia sus requerimientos, se

necesario evaluarlos con respecto a los iniciales, y si indispensable, entonces llevarlos a cabo.

A continuación se describe de manera general el sistema contenedores, su contexto, su frontera, los elementos que componen, etc. Así como la lista de los requerimientos general que se involucran en todos los procesos del sistema.

Cabe mencionar que este capítulo no busca ser un tratado de temas que analiza, sino una ayuda para el entendimiento general sistema así como el cumplimiento de la primer fase del ciclo vida del sistema que es la ingeniería de requerimientos.

Por principio, debemos recordar que el Banco Central tiene en sus funciones sustantivas la fabricación y distribución efectivo, cuya responsabilidad recae en la Dirección de Emisión.

Actualmente, la unidad primordial de fabricación, distribució control, entre otras, es la pieza (un billete). Todas actividades, tanto las órdenes de fabricación, como la producci distribución giran en torno a las piezas. No obstante, se consi conveniente la alternativa de llevar el control tanto fabricación como de distribución (entre otras actividades), varias unidades además de la pieza: como mazos, paquetes, bolsa contenedores.

De esta forma, la Dirección de Emisión del Banco Central pla almacenar y distribuir el efectivo mediante el uso de enva (contenedores, bolsas, paquetes y mazos), mismos que s controlados por códigos de barras; apoyando estas tareas sistemas de cómputo.

El Banco Central busca incrementar la eficiencia, la seguridad control del manejo del billete, desde que se produce hasta llega al usuario final, por medio de la modificación de su empa este empaque es más seguro y resistente que el actual, el cual mayoritariamente en bolsas comunes y corrientes.

Con la operación del sistema de contenedores el Banco Cen pretende obtener mayor seguridad y mejorar el control de procesos debido a que:

- Las entregas de piezas se harán en unidades selladas (contenedores)
- El control de los flujos se hará utilizando códigos de barras

- Se manejan contenedores en lugar de piezas, reduciéndose con ello el número de operaciones y controles en el manejo del efectivo
- El control se hará por medio de contenedores, en lugar de las cantidades de efectivo por valor declarado
- Se opera con menor número de empleados
- Se disminuyen los tiempos de operación
- Se disminuye el consumo de papel, ya que no se generan los formatos necesarios para los movimientos implicados.
- Se disminuyen los tiempos empleados en algunos procesos y la duplicidad de funciones

ionalmente, con el uso de códigos de barras se tienen las ientes ventajas:

- Conocimiento exacto de la ubicación de los envases
- Mayor eficiencia y flexibilidad en la administración de inventarios
- Mayor control y eficiencia en el manejo y movimiento de las piezas
- Asociación de información específica a cada envase

diagrama del contexto general de los proceso involucrados se tra en la figura 3.1., en la cual se representan las relaciones e las distintas áreas de la Dirección.

Gerencia de Caja elabora propuestas de órdenes de emisión y nes de fabricación que son aprobadas por la Dirección. En rización de emisión se establecen las características intivas entre las áreas de Caja, Fábrica y Manufactura Cada rización se divide en varias órdenes de fabricación, las cuales hacen llegar a la Fábrica para que establezca un plan de ucción que cumpla con los requerimientos establecidos por la . Una vez que la orden de fabricación es aceptada, se le rma a Manufactura para que inicie el proceso de producción, lo cual solicita los insumos necesarios al Almacén. Cuando factura termina la producción de piezas éstas son empacadas en área de Procesos Finales, donde además, se lleva a cabo un eso de revisión de las piezas fabricadas para asegurar su dad Después, las piezas empacadas se entregan a la Caja, es ces cuando esta área se encarga de distribuir y recolectar el tivo de los bancos usuarios, y de esta forma poner las piezas rculación

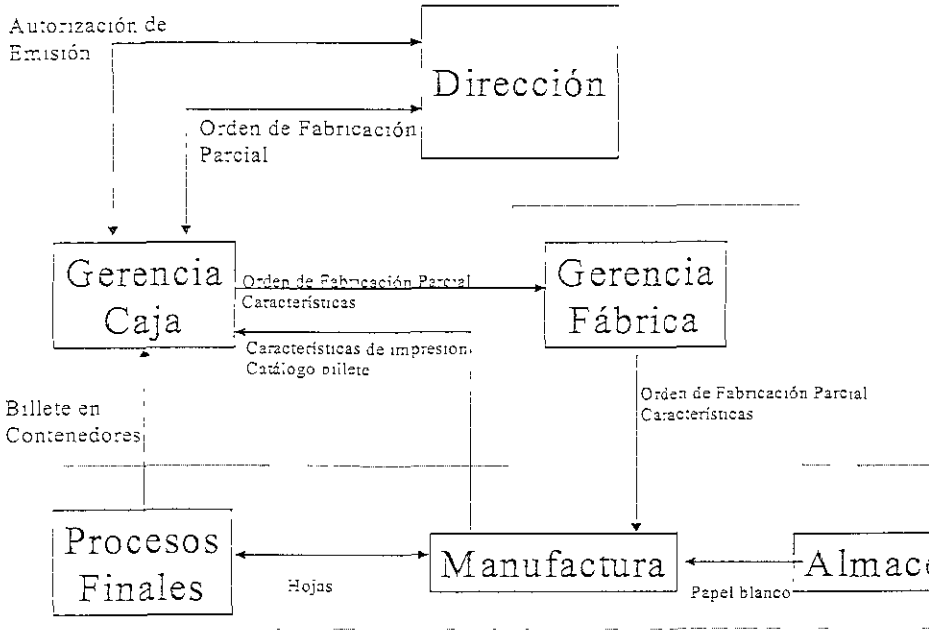


Fig. 3.1. Relación entre áreas del Banco Central.

A continuación, se describen los requerimientos específicos p
cada uno de los procesos.

3.2. Control de puntos de evento

El Banco Central requiere llevar el registro de las operaciones
lugar y momento en que éstas suceden, para que de esta forma pu
contarse con información oportuna y en línea. Esta manera de ope
es llamada Punto de Evento.

De manera formal, un punto de evento es un lugar físico, que pu
corresponder a un área operativa, en donde se realizan operacio
(Traspasos, Remesas, Depósitos y Retiros, entre otros), y que pu
ser responsable del manejo de una o más bóvedas o ventanillas.

El usuario necesita que un punto de evento sea una entidad lóg
donde se registren y consulten operaciones, y que pertenezca a
sola Oficina donde se puedan agrupar varios usuarios y adminis

as bóvedas o ventanillas, con la condición de que una Oficina a tener varios puntos de evento.

un mismo punto de evento se pueden registrar operaciones de rentes tipos, por varios usuarios y desde más de una terminal, ue en su conjunto pueden afectar a una o más bóvedas o anillas. En la figura 3.2. se muestra una representación de concepto.

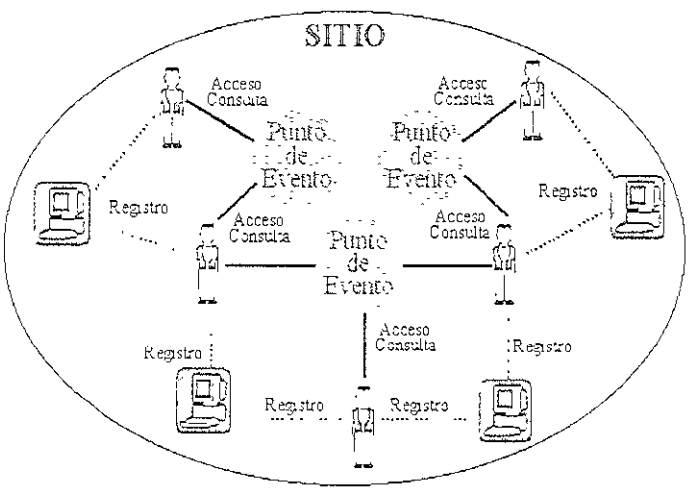


Fig. 3.2. Punto de Evento.

ejemplo, si el usuario necesita verificar las existencias por bóveda, el responsable consultará en el sistema el reporte de tencias y si detecta alguna diferencia procederá a consultar movimientos que afectaron su bóveda o ventanilla durante el para verificarlos contra los documentos que amparan las aciones realizadas, a fin de detectar errores en la captura o alta de registro de alguna operación y hacer las correcciones sean necesarias.

vez que las existencias estén correctas, y no se tengan rencias en ninguna de las bóvedas o ventanillas que le esponden, el responsable del punto de evento procederá a arlo, ejecutando en el sistema el cambio de estado del mismo a ado. Previendo casos de contingencia, el sistema también itirá al responsable de la Oficina Central y al Administrador [illegible] Punto de Venta.

Los puntos de evento pueden ser reabiertos en caso de necesario, ya sea por los responsables de los mismos, por responsable de su Oficina o por el Administrador del Siste siempre y cuando el estado de la Oficina correspondiente indi que está abierta. Si el punto de evento es reabierto, este deb ser cerrado nuevamente para permitir el cierre de la Oficina.

Una vez que todos los puntos de evento hayan cerrado, d permitirse el cierre de la Oficina a la que pertenecen. Ader debe permitirse que las Oficinas sean reabiertas por el responsa de ésta o por el Administrador del Sistema si lo requieren, siem y cuando no se haya ejecutado el fin de día del sistema. Si alg Oficina es reabierta, ésta deberá ser cerrada nuevamente a fin permitir la ejecución del fin de día del sistema.

Al término de las operaciones, una vez que todas las Oficinas ha cerrado, el área responsable de la administración del sistema d realizar una confronta general, si hay alguna diferencia, d proceder a realizar las correcciones necesarias hasta desaparezca. Cuando la confronta no arroje diferencias, ejecutarán los procesos de fin de día (respaldo en históric inicialización de tablas, y cierre del sistema) con el fin de no se pueda registrar ninguna otra operación que afecte existencias.

3.3. Impresión de códigos de barras

Como se mencionó anteriormente, se utilizarán códigos de bar impresos en etiquetas o cintas de papel, para identificar diferentes envases que se utilizan en los procesos. La informac que el usuario requiere que esté impresa en la etiqueta, d estar relacionada con la institución que empacó el billete, indi la denominación y el tipo de envase que se está identificando.

Uno de los puntos que se deben considerar es que el usua utilizará etiquetas de 5 cm de ancho y 10 cm de largo, y tam cintas de 6 cm de ancho y 60 cm de largo, esta última para mazos de la denominación más alta ($500).

Por otro lado, es necesario llevar el control de los códigos barras impresos, de tal forma que no se dupliquen; al mismo tier se debe registrar el área y persona que solicita la impresión dichos códigos.

Un requisito indispensable es que, durante la generación

ıetas, se ofrezcan al usuario dos maneras distintas de hacerlo:

- La primera a través de un plan de producción, en el cual el usuario especifica la denominación y el número de hojas que va a trabajar. Esto significa que cuando el usuario indica la cantidad de piezas que va a empacar, el sistema debe generar el número y tipo de etiquetas correspondientes. De esta forma, si el usuario determina que va a empacar piezas de la denominación de $500, el sistema debe generar etiquetas para esta denominación y para cada uno de los envases que contendrá el efectivo.
- La segunda, bajo demanda. Esto quiere decir que el usuario indica el número y tipo de etiquetas que quiere generar, de acuerdo a las necesidades de su operación. Así, el usuario puede mandar que se genere sólo una etiqueta para un contenedor que tiene empacada la denominación de $200.

último, es necesario que el código que se utilice para ficar la información sea estándar, sencillo y de uso popular en ercado de código de barras.

Armado y desarmado de contenedores

rmado tiene como objetivo agrupar unidades de empaque dentro de unidad. Durante esta operación se leen las etiquetas de las ades de empaque (bolsa, paquete y mazo) para asociarlas a una a unidad de empaque denominada contenedor. Dichas unidades en provenir de otros contenedores, o puede ser la primera vez van a conformar un contenedor.

otra parte el desarmado consiste en abrir una unidad de empaque itar algunas o todas las unidades que contenía. Al momento de r un contenedor, éste deja de existir, quedando liberadas las ades que contenía. Estas unidades de empaque sueltas pueden ser izadas para una operación de armado y conformar una nueva ad de empaque.

suario lleva a cabo, el armado y desarmado de unidades de que en los siguiente procesos:

- Preparación de traspasos
- Armado de depósitos
- [illegible]

- Preparación de retiros

3.5. Cambio de estado físico

La operación Cambio de Estado Físico consiste en pasar el estad el que se encuentra un contenedor a otro estado, y por consecue su contenido. Por ejemplo, cuando un contenedor tiene pi nuevas, y se pretende ponerlas en circulación, es necesario cam su estado físico de nuevas a aptas para circular, ya contablemente ambos adquieren un valor distinto (las piezas nu no tiene valor alguno).

Para poder realizar el cambio de estado físico es necesario t registro de las unidades de empaque a las que se les realizará cambio de estado físico y que contienen el efectivo involucrado.

En cuanto a las transacciones permitidas, el sistema debe perm el registro, la cancelación, la modificación y la consulta cambios de estado físico del efectivo, además debe permitir cambio de estado físico para cualquier tipo de unidad de empaqu aceptar un cambio de estado físico solamente, si éste cumple las validaciones correspondientes.

Transacciones no permitidas. El sistema no debe permitir modificación o cancelación de cambios de estado físico que h sido enviados a otro sistema de información contable o que se h registrado en días anteriores al actual, tampoco debe permitir cambio de estado físico por importes.

Para la modificación de un estado físico ya registrado, el sis deberá presentar una lista con los cambios de estado físico qu han sido procesados contablemente, permitir al usuario modifi los cambios de estado físico que el mismo originó y que selecc algún cambio de la lista, y mostrándole el detalle correspondie También desea que el sistema le solicite la confirmación de modificación al estado físico.

Para la cancelación de un cambio de estado físico, el sis deberá presentar una lista de los cambios de estado físico qu han sido procesados contablemente, además de permitir al usuari cancelación de los cambios de estado físico que el mismo orig para que seleccione algún cambio de la lista y le sea mostrad detalle correspondiente.

El sistema mostrará una lista de los cambios de estado físico pueden consultarse, incluyendo aquellos que han sido cancelad

ficados.

: Autorizaciones de emisión y órdenes de fabricación

utorización de Emisión de efectivo, es un documento oficial e se asienta la cantidad de piezas a fabricar, desglosadas por minación. El Banco Central necesita llevar el control de las rizaciones de Emisión a través del sistema, para llevar un stro de las piezas que se ha mandado fabricar a lo largo de un odo determinado.

poder llevar el control de estas autorizaciones, el usuario sita registrarlas a través del sistema e identificarlas de ra única. Durante el registro es necesario indicar la cantidad iezas por denominación, la fecha en que se autorizó la emisión r último la fecha de emisión, que es la que aparece impresa en pieza.

usuario debe tener acceso a la información de las rizaciones registradas, también debe poder modificar o cancelar na autorización de emisión, siempre que la Junta de Gobierno Banco Central haya aprobado alguno de estos movimientos. Lo rior puede ser por distintas razones, pero básicamente por las as de política monetaria que el Banco Central decida aplicar y no se detallan en esta tesis.

vez que se tiene la autorización de emisión se generan, nces, diversas órdenes de fabricación. Para cada orden se ora un documento denominado Orden de Fabricación Parcial, el va dirigido a la Fábrica de Billetes. En este documento se lla el total de piezas a ordenar por denominación, así como la a de autorización y la fecha de emisión de dicha autorización.

ecesario capturar en el sistema cada orden de fabricación, y r generar fichas contables que avalen la fabricación de piezas, ás de un documento donde se detalle una relación de cterísticas correspondientes, que especifiquen para cada minación series, prefijos, folios, fechas y firmas que estarán esas.

nformación registrada en la Orden de Fabricación debe poder ser altada por cada una de las áreas involucradas en el proceso de ducción.

solicitaron, pero en ocasiones no es posible, lo que obliga usuario a que cancele la fabricación de esas piezas que son remanente, y para lo cual se elabora un documento de Cancelación remanentes, donde se detalla la cantidad de piezas a cancelar denominación. Esta cancelación debe ser registrada en el sistema

De esta forma se tiene que la suma de todas las órdenes fabricación deben satisfacer una autorización de emisión, y usuario necesita llevar el control de las piezas ordenadas, de manera que vaya monitoreando el hecho de no pedir la fabricación un número de piezas que rebase la cantidad autorizada.

3.7. Empaque de efectivo

El usuario conceptualiza el proceso de empaque de efectivo a par del momento en que se inicia su producción, y termina cuando envasado y queda listo para su traspaso o retiro.

Al inicio de la producción, el usuario requiere tener conocimie de los lotes con los que va a trabajar, para generar las etiqu con las que se llevará a cabo el control en los proce subsecuentes. Por esta razón, es necesario que disponga de manera eficaz y precisa, de la información de lotes nuevos, o una lista de lotes pendientes por procesar, debido a que no fue concluidos durante la jornada en que se iniciaron, o porque fue pospuestos por alguna razón especial.

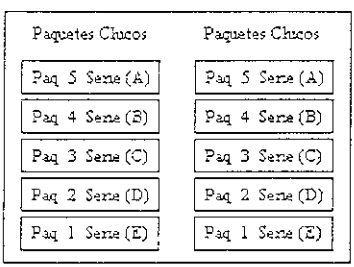
También es indispensable, que en cualquier momento pueda regis los lotes, y la cantidad de hojas con las que trabajará, ya que posible que a media jornada, termine con un lote y tenga comenzar con uno nuevo.

Para mejorar este proceso, el usuario debe tener datos específi del lote, y el orden en que deben tomarse, dependiendo del área la que se van a procesar. En Corte y Empaque automático, se de procesar en orden ascendente, es decir, del millar menor al may mientras que en el área de empaque se deben procesar en fo descendente (del millar mayor al menor).

Una vez que es autorizada la producción, se debe generar secuencia de salida de cada uno de los empaques. Nos referimos secuencia de salida, a toda la información que nos proporcionar cada una de las etiquetas que serán utilizadas en empaques. Esta información debe contener la "historia" de billete, es decir, debe proporcionarnos datos como denominac

, número de millón, millar, serie, prefijo y fecha a la que tenece. Para la denominación más alta ($500), también debe ntificar la posición del mazo.

Corte y Empaque, los mazos de $500 salen en un orden tal que el mismo millar cumplen con la secuencia con la que serán cados, mientras que en el área de empaque, a la salida del o se tienen dos líneas: una para procesar los paquetes chicos y para procesar los paquetes grandes, y quedan ordenados como se tra en la figura 3.3. Es en este mismo orden como les será iada la información correspondiente.



g. 3.3. Orden de los paquetes procesados en el área de empaque.

se mencionó anteriormente, cada paquete (sin importar su minación), debe contener cinco mazos. Y como existen dos tipos paquetes (chicos y grandes), el usuario debe tener los medios identificarlos.

suario considera importante tener un mayor control del efectivo a más alta denominación, por lo que es necesario que cada uno los mazos de $500 pueda ser identificado, y que al momento de ar un paquete se tenga el registro de qué mazos son los que lo gran. Mientras que para paquetes de otras denominaciones, sólo ecesario registrar la secuencia de salida correspondiente.

suario también solicita, que cada vez que sea registrado un ete, le sea mostrada la información que le corresponde

ués de que se han formado paquetes bien identificados, el rio requiere registrar también, que paquetes están contenidos na bolsa, y que dicha bolsa sea identificada con un código o.

otra parte, una bolsa contiene cinco paquetes de alta minación, o bien paquetes de baja denominación, y no hay [illegible]

que puede tener de los dos tipos.

Si durante el transcurso del empaquetamiento se daña una etiq que ya tiene información asociada, se debe poder cambiar etiqueta dañada por una nueva, asignándole a ésta, toda información que tenía relacionada la etiqueta dañada. A esto se denomina reasignación y se debe llevar un control de todas y c una de las reasignaciones, registrando la fecha en que se llev cabo.

Otro punto importante que se debe considerar durante este proce es lo que el usuario llama compensación. La compensación se cuando se tienen que sustituir hojas o mazos, y siempre tiene ser verificada por dos personas de las áreas involucradas (área control y área de procesos finales).

La compensación de hojas puede hacerse cuando, al inicio de producción, se sustituye una cantidad de hojas de cierto millar la misma cantidad de hojas pero de un millar distinto. Las ho compensadas se procesan en un área en la cual el material que s es considerado como no progresivo, es decir, que los paquetes son formados ahí, no tienen una secuencia en los folios aún cua pertenecen al mismo millón.

La compensación de mazos puede darse después de que los bille son cortados, enfajillados y pasados al carrusel, donde como mencionó en el capítulo anterior, los operadores revisan los ma y si encuentran defectos en ellos, los sustituyen con mazos bue del mismo millón. Si un paquete contiene mas de tres ma compensados, también es considerado como no progresivo.

Como ya se mencionó, existe un área de recuperación en la que trabaja material que es no progresivo, con el cual, consecuencia, se forman paquetes y bolsas no progresivos, y requiere que cada uno de estos envases debe tener un código que identifique como tal.

La información que el usuario requiere para identificar mazo paquetes no progresivos debe contener denominación, tiro, mil (si el usuario lo sabe), serie y prefijo.

Otra observación importante por parte de los usuarios, es que tienen que controlar hojas, mazos o paquetes maculatura.

Las hojas maculatura se regresan al área de manufactura para sean destruidas y los mazos maculatura deben ser controlados importar su denominación, por lo que se solicitó por parte de usuarios que sean etiquetados, y como no se forman bol

latura, los paquetes o mazos se colocan directamente en enedores maculatura.

mportante llevar el control de esto, porque al final del tiro eben cuadrar las cifras. Es decir, el número de piezas de la n de fabricación debe ser igual a la suma de las piezas buenas adas, las piezas maculaturas cortadas y las piezas contenidas as hojas maculatura.

stas cifras no cuadran, se requiere hacer un acta de descargo dada por las áreas involucradas, en la que se autorice el ante, explicando los motivos de esto.

camente es de esta manera como se desea llevar el control en el eso de empaque.

Entrega de efectivo

el proceso de entrega de efectivo están involucradas cipalmente dos áreas, el área de Fábrica y el área de Caja. El de fábrica tiene la responsabilidad de entregar efectivo cado en contenedores al área de Caja, para que ésta eriormente se encargue de distribuirlo hacia otras áreas del o Central.

entrega de efectivo está relacionada con una Orden de icación previa. Sin embargo, pueden existir varias entregas espondientes a una sola orden, debido a que el efectivo que se roduciendo no puede estar almacenado hasta que una orden sea lida, por esta razón se hacen distintas entregas que van iendo parcialmente una orden. Cada entrega es por una o más minaciones, esto es, que pueden entregarse en una misma ación piezas de $500, $100, etc.

realizar la operación, el área de Fábrica elige los enedores que serán entregados y registra su salida de la da donde se encuentren, la cual debe ser autorizada por el onal que está a cargo de supervisar los movimientos del tivo y el control de existencias. En este momento las tencias de la bóveda disminuyen y los contenedores se ponen en sito hacia el área de Caja, que aunque está en el mismo sitio na área a la que se llega con permisos de seguridad exclusivos.

vez que los contenedores que tienen el efectivo llegan al área [illegible]

así como que venga con las medidas de seguridad establecidas por Banco Central. Después de verificar que todo está correcto, ace el efectivo, le da entrada en su bóveda y genera el acta de entr correspondiente. El acta de entrega contiene la fecha en que realizó la entrega, un número único, el importe total entregad las características del efectivo entregado. Por cada denominac que se entrega, se genera un acta, con el fin de llevar el cont de las entregas por denominación y no sólo por la cantidad piezas involucradas.

En cada entrega debe incluirse la cantidad de piezas que por alg razón tuvieron defectos durante el proceso de fabricación (pie maculatura), que fueron destruidas por el área de Fábrica y que se están entregando, pero que deben informarse al área de Caja.

Para finalizar la entrega de efectivo ésta debe ser autorizada el personal designado para supervisar el flujo de efectivo en las distintas áreas del Banco Central.

Una vez que el proceso de entrega ha finalizado, éste se d reflejar en las existencias de las bóvedas involucradas, en piezas que faltan por entregar de la orden, en las pie autorizadas que faltan por fabricarse y las que ya se fabricado. En este momento, las piezas adquieren valor conta para el Banco Central, y pueden ser puestas en circulación.

3.9. Traspasos

El traspaso es una operación de transferencia de efectivo, determinadas unidades de empaque, ya sea por solicitud del envío contenedores hacia alguna oficina receptora, o por el aviso entrega de contenedores por parte de alguna oficina pagadora.

Durante el proceso el usuario, requiere identificar cada solici como registrada atendida o cancelada, conforme se desarrolle operación de traspaso.

La información que el usuario requiere identificar en una solici de transferencia de efectivo incluye un número de fo consecutivo, que puede ser generado automáticamente por el siste sucursal que genera la solicitud, oficina que solicita el efecti bóveda o ventanilla que recibe el efectivo, oficina que entrega efectivo, estado físico con el que llegará a la oficina recepto denominación del billete o moneda metálica que se solicita, impo por denominación que se solicita y el importe total de

citud.

suario desea que el sistema le permita cancelar la solicitud ualquier momento, excepto en el caso que el traspaso ya se haya ado a cabo, porque entonces la solicitud pierde su vigencia. realizar la cancelación, la oficina que generó la solicitud la tificará por el número que se le asignó al darla de alta, niendo la información correspondiente, y cuando el usuario lo que cancelará la solicitud.

e cancela una solicitud antes de que el efectivo salga de la ina pagadora, el sistema no debe permitir que se realice el miento de traspaso.

usuario desea conocer las solicitudes de efectivo que le esponden a fin de saber el estado de las mismas durante el eso, y en su caso también poder imprimirlas.

istema debe permitir que la oficina pagadora indique que se n preparando los contenedores para cubrir la solicitud de tivo. El usuario la identificará como en preparación y podrá izar esta información para complementarla e incluirla en el stro de un traspaso.

do se hayan terminado de preparar los contenedores, el usuario cará al sistema que la solicitud ha sido procesada y por lo o, ésta deberá ser identificada como atendida.

aso de que el traspaso se genere por medio de una solicitud de tivo, la información de ésta se utilizará para dar de alta un o traspaso y se complementará con la lista de contenedores espondiente. El sistema debe permitir que cada traspaso sea rizado electrónicamente por el personale responsabel de ficar el flujo de piezas. Al dar de alta el traspaso, el ema debe disminuir las existencias de la bóveda de la oficina dora y poner en tránsito la ubicación de los envases.

suario desea poder cancelar un traspaso sólo si éste aún no ha recibido por la oficina receptora. La oficina pagadora o la ptora pueden cancelar los traspasos en los que estén icipando, lo que representa la cancelación completa del paso, para lo que deberán indicar el motivo de la cancelación.

istema debe mostrar una lista con los folios de los traspasos y stado en que se encuentran, para que el usuario elija el que va ancelar, también se debe permitir que cada cancelación de paso sea autorizada electrónicamente.

Una vez autorizada la cancelación, el traspaso será identifi como cancelado, repondrá las existencias de la oficina pagador restablecerá la ubicación (bóveda origen) de los envases que fo el traspaso.

Un traspaso sólo puede ser recibido si está en tránsito, es d que los envases se encuentran listos a ser ingresados, por lo el sistema debe permitir que se lean los códigos de barras de envases que se están recibiendo.

La lectora portátil de código de barras deberá validar que no lea más de una vez un código de barras y también en el moment realizar la lectura de los códigos, el sistema debe compararlos los del traspaso en tránsito para verificar que sean los mismos caso de que alguno(s) no coincida(n), el sistema debe indica usuario las diferencias y no procederá con la recepción traspaso.

Al recibir el efectivo debe permitir que el interventor de oficina de control y el custodio de la oficina receptora autori En ese momento el traspaso se identifica como recibido y registran las claves de quienes autorizaron la recepción.

Una vez autorizada, el sistema debe aumentar las existencias de oficina receptora en las cantidades establecidas en el trasp cambiar la ubicación de los contenedores que lo forman y, en caso de que haya sido generado a través de una solicitud, ind que ésta ya no tiene vigencia y sólo puede ser consultada.

3.10. Depósitos

Un depósito es la recepción de efectivo en plazas del banco cen por parte de los bancos usuarios, el cual puede ser apto y no para circular.

Al realizar este proceso, el usuario revisa los valores contra ficha de depósito y verifica las características del empa verificando que esté correctamente sellado, y si alguno estuv alterado, el cajero debe salir y entrar el representante del b para cambiar el empaque y de ser necesario, actualizar la fich depósito.

El usuario realizará un muestreo de los empaques que s depositados para identificar cuales serán rechazados o aceptad posteriormente registrará el resultado correspondiente.

l depósito fue rechazado completamente, el usuario requiere
: un acta con esta información, mientras que en caso de un
izo parcial, debe generar una nueva ficha de depósito y un acta
echazo.

erminar el muestreo, para tener un mejor control del depósito,
suario quiere que se permita el registro y almacenamiento de la
iente información contenida en la ficha de depósito.

- Folio del documento
- Cuenta del banco que deposita
- Por omisión (Plaza de la sucursal de captura)
- Fecha y hora de registro del depósito (asignada de forma automática por el sistema)
- Estado físico del billete que se deposita
- Importe total de los empaques aceptados

vez registrado el depósito el usuario debe poder etiquetar cada
de los envases recibidos para identificarlos de manera única y
es acceso a las bóvedas del Banco Central afectando sus
encias.

Retiros

se mencionó en el capítulo anterior, un retiro es la
osición de efectivo realizada por los Bancos usuarios, con el
espondiente cargo a su cuenta única en el Banco Central o
esponsales. Recordemos que estas solicitudes pueden ser de dos
s, preavisadas (enviadas un día antes del retiro), o urgentes
iadas el mismo día en que se pretende realizar el retiro).

endo esto en cuenta, podemos empezar a definir los
erimientos de los usuarios, ya que consideran necesario
strar cada uno de los retiros para controlarlos de una manera
lla y en el momento preciso.

cada retiro, es importante que los usuarios sepan que oficina
olicitó, a que tipo de retiro pertenece, cuál es su importe
l, cuando y dónde será llevada a cabo la operación, que tipo de
da es la que se manejará (por ejemplo: dólares, pesos, etc.),
de el estado físico del efectivo, que importes por
nación contiene, y quien debe identificar a las personas

También es importante para el usuario verificar que el importe denominación del retiro es múltiplo del importe correspondiente empaque mínimo, para manejar envases completos.

Por omisión, el estado físico del efectivo que se retira es a pero si durante la operación no es así, el usuario debe p realizar un cambio de estado físico.

Otro punto importante para el usuario, es que se puedan real consultas de la información que se involucra en este proceso. ejemplo, debe poder consultar los retiros que están aún preparar y revisar la información detallada de cada uno de ellos

También le es útil, que para cada retiro, se calcule la cantid tipo de unidades de empaque por denominación que se requie obteniendo también el total de unidades de empaque, y consiguiente, el total de contenedores por denominación que s utilizados para preparar el total de retiros solicitados.

Además de esto, el usuario considera necesario tener la opción imprimir la consulta que realizó, y sería ideal para él, que sistema le indicara los estados físicos y denominaciones a util en la preparación del retiro, de acuerdo con las existencias.

Por otra parte, se requiere que se pueda cancelar una solicitud retiro, siempre y cuando dicha solicitud no haya sido proce (entregada). Para hacer esta operación, el usuario debe verifica detalle la información que contiene la solicitud, a fin de cancelar otra solicitud por error. Pero, por seguridad, el usu necesita que al realizar una cancelación, el sistema no la eli completamente, por si es necesario aclarar algo posteriorme pero debe quedar claro que una solicitud cancelada no podrá atendida.

Finalmente, antes de realizar el retiro, es requisito indispens que los contenedores estén preparados en el lugar donde efectuará la operación, ya que deben ser "desarmados" para que entregadas las unidades de empaque que contienen y es en momento cuando se debe registrar como procesada.

Para hacer esto, el usuario requiere poder seleccionar, a parti una lista de solicitudes pendientes, la que va a ser atendida, mostrar a detalle su información y poder compararla con unidades de empaque que tiene físicamente, y que serán valid con lo registrado en la ficha de retiro entregada por representante del banco usuario que va a realizar el retiro. todo coincide, se hace la entrega y el usuario registra que retiro ha sido procesado.

2. Remesas

remesa es una transferencia de efectivo entre dos sitios, ndiéndose por sitio a una oficina central, un corresponsal o sucursal.

remesa de la oficina central hacia una sucursal o corresponsal denomina envío, considerando aquí que el estado físico del tivo puede ser nuevo o apto para circular; mientras que a una sa de un corresponsal o sucursal hacia la oficina central se le mina concentración con la característica de que el estado co del billete puede ser apto y no apto para circular. También posible realizar envíos entre sucursales observando que el do físico del billete pudiera ser nuevo, apto o no apto para ular; para el tipo de remesa entre corresponsales se le mina traspaso, en donde encontramos que el estado físico del ete es considerado apto para circular.

nformación que el usuario requiere identificar en una orden de sa incluye un número de folio del documento que pueda ser rado automáticamente por el sistema y tipo de operación que ra, y dependiendo de los sitios que intervienen en la remesa: de efectivo, institución que origina la remesa, bóveda o anilla de donde saldrá la remesa, detalle de importes por minación y estado físico, transporte y personal del área rgada de la programación de remesas que autoriza la operación.

el usuario requiere que los importes por denominación de la sa sean múltiplos del importe correspondiente al empaque mínimo e coincidan con el detalle por denominación y estado físico urados.

suario debe poder cancelar una orden de remesa a excepción de llas que ya han sido movilizadas, por lo que requiere que el ema le ofrezca una lista de aquellas órdenes que no han sido lizadas con el fin de seleccionar alguna para que, con una e de autorización, se le permita cancelar la orden.

l proceso de modificación de una orden de remesa el usuario conocer una lista con las órdenes que aún no han sido lizadas para que pueda modificarlas si él mismo la originó. ello deberá contar con autorización que le permita modificar orden de remesa, en caso de que el usuario confirme la ficación, se deberá dar un registro de folio, fecha de la [illegible]

El usuario requiere consultar los distintos tipos de remes mostrar toda la información que se haya registrado. Tam necesita consultar por el estado físico del billete enviado, algún periodo que defina; se debe poder consultar e impr indistintamente las órdenes de remesa canceladas y/o modific según el usuario lo prefiera.

El sistema deberá generar un reporte por cada uno de los tipo remesa, en el que se indicará al usuario los importes por tip denominación de las remesas que se van a atender, además incluirá la siguiente información: la ubicación de contenedores que se van a movilizar, la cantidad de contened por denominación que se utilizarán para la preparación de remesa, la cantidad y tipo de unidades de empaque por denomina que formarán la remesa, los estados físicos y denominacione utilizar de acuerdo a las existencias.

En el momento de entregar la remesa al transportista, contenedores que la componen ya deben estar listos y preparado el lugar donde se efectuará la entrega de valores, mientras qu sistema ya debe tener el registro de las unidades de empaque se van a utilizar en la remesa, debido a que el usuario requier disponibilidad de esta información para confrontarla con información de las unidades de empaque que entregará transportista. Con esto se validará que dichas unidades son las están autorizadas para cada remesa en particular. Confirmado anterior se solicitará la autorización respectiva para avala operación de salida de contenedores de la ventanilla.

Para que se afecten las existencias en tránsito y las de la bó ó ventanilla de donde sale el efectivo, es necesario registra confirmación de envío de la remesa. Para ello, el sistema mostrarle al usuario una lista de remesas por confirmar, para indique en cuál de ellas va a ratificar su envío. Para lo anter el usuario necesita la información a detalle de la re seleccionada, y que se le permita registrar el destino de la re y la Plaza en donde es recibida.

Una vez hecha la confirmación de envío de la remesa, el sis deberá disminuir las existencias de la ventanilla de donde sal efectivo, e incrementar las existencias en tránsito.

En el momento que se confirme la recepción de la remesa, necesario registrar en el sistema esta operación, con el fi afectar las existencias en tránsito y de las bóvedas o ventani que reciben el efectivo.

El sistema deberá mostrar una lista de remesas por reci

ificadas por tipo, permitiendo al usuario indicar cual va a irmar como recibida y entonces registrar la operación, folio documento, fecha de recepción, número de la remesa, bóveda o anilla en donde se realizará la recepción, importe total de la sa y el estado físico del billete; para que con esto, el rio dé como confirmada la recepción de la remesa.

a remesa es entre oficinas centrales, el usuario deberá leer códigos de barra de las unidades de empaque que se reciben, dar que los códigos de los contenedores recibidos sean los os que se registraron en la confirmación del envío y en caso de oincidir se solicitará una autorización para que se acepte la sa.

.a remesa es de un corresponsal a una oficina central, el rio requiere que el sistema permita capturar los importes por minación del efectivo que recibe y validar el detalle capturado a confirmación de envío de la remesa. El usuario requiere que istema detecte diferencias en los importes por denominaciones o s de empaque y en caso de haberlas, el sistema debe dar la ón de aceptar la confirmación de recepción o permitir la ección del error.

sistema debe realizar la conversión de los importes por minación a unidades de empaque, a excepción de los empaques que ndiquen como denominación general.

sistema debe incluir una opción para imprimir las etiquetas espondientes a los empaques recibidos, y permitirle al usuario strar los códigos de las etiquetas ya pegadas para que el ema le permita ingresar los empaques y validar sus importes.

ás debe permitir que al confirmar la recepción de la remesa, se ice el incremento correspondiente de las existencias de la da que recibe el efectivo y disminuya las existencias en sito.

esto terminamos la descripción de los requerimientos del ario para el Sistema de Contenedores en cada uno de los cesos involucrados, por lo que en el siguiente capítulo se tará el análisis detallado de las especificaciones ya blecidas.

CAPÍTULO IV

ANÁLISIS DEL SISTEMA

el presente capítulo se realiza el análisis del sistema
lizando principalmente Casos de Uso para describir las
nciones del sistema.

propósito de la fase de análisis es estudiar, especificar y
nir el sistema a construir En esta fase se obtienen diagramas
os que se representa qué va a hacer el sistema. La base de
s diagramas es la descripción de los requerimientos del
ema, que en esta fase son refinados y estructurados.

a etapa anterior los requerimientos del sistema están descritos
n lenguaje natural. En esta fase son refinados y estructurados,
al forma que se logren entender mejor. Esto se hace a través de
descripción que es fácil de mantener y que ayuda a tener una
nización completa del sistema

a etapa de análisis se construyen modelos que ayudan a entender
r el sistema.

modelos que se desarrollan durante esta fase son totalmente
ntados a la aplicación, y no se toma en consideración el
ente real en el que el sistema será desarrollado, como puede
el lenguaje de programación o el manejador de la base de datos.
ropósito es formular el problema y construir los modelos que
la capacidad de resolverlo bajo las condiciones ideales. De
forma los modelos están orientados al problema, lo cual da la
ntaja de diferirlos con [illegible]

términos propios de la implementación. La gran ventaja de
manera de hacer el análisis es que los modelos permanecen intac
aún cuando las condiciones de la implementación cambien.

Dos modelos son desarrollados durante el análisis, el modelo
requerimientos y el modelo de análisis. El modelo de requerimie
representa los límites del sistema y define su funcionalidad.
sistema se describe en un número de casos de uso que son utiliza
por un número de actores. Los actores constituyen el ambiente
sistema, y los casos de usos son las actividades que suceden de
del sistema.

El modelo de análisis da una configuración conceptual del siste
que consiste de objetos de control, objetos de entidad y objetos
interfaz. El propósito de este modelo es desarrollar una estruc
robusta que sirva como base en la construcción del sistema.
tipo de objeto tiene su propia contribución para lograr la robus
deseada, y juntos ofrecen la funcionalidad total especificada e
modelo de requerimientos.

El modelo de análisis comprende una especificación de toda
funcionalidad del sistema, sin hacer referencia al ambiente
implementación. Al construir el sistema se toma como base el mo
de análisis. Es entonces cuando se hacen las adaptaci
necesarias acordes al lenguaje de programación, el manejador de
base de datos, el hardware de la computadora, etc.

La razón de no hacer el análisis al mismo tiempo que se diseñ
implementa el sistema, es porque el diseño e implementac
necesitan más detalle que el análisis, así que la separación
necesaria. Además que, antes de iniciar el diseño e implementac
se debe tener un entendimiento preciso y detallado de
requerimientos, en un nivel tal que el usuario no pueda argume
que el sistema que se obtiene como producto no es como lo ha
pedido. Más aún, si se logra tener una estructura que pueda
usada para diseñar el sistema, el proceso resulta ser más valios

Al realizar el análisis y separarlo del diseño se simplifican
actividades siguientes en esta última, delimitando los proble
que necesitan ser resueltos y los que necesitan tomarse en e
actividades. Así, al separar estos conceptos, el personal
desarrolla el sistema puede hacer más fácil el esfuerzo "cue
arriba" que se presenta cada vez que se inicia la implementació
un sistema, y como consecuencia, se evita la parálisis que s
ocurrir cuando se tratan de resolver varios problemas al m
tiempo, incluyendo aquellos que no pueden resolverse totalm
porque los requerimientos son vagos o no se entendi
correctamente.

análisis proporciona una especificación más precisa de los erimientos que los que se tienen durante su captura, ya que se izan diagramas de casos de uso para representar la estructura sistema.

modelos que se obtienen son descritos en el lenguaje de los rrolladores, y se introduce un nivel mayor de formalismo que es o para el funcionamiento interno del sistema.

requerimientos se estructuran en una manera que facilita el ndimiento, la preparación, el control de cambios, y en general, antenimiento de éstos.

productos que se obtienen pueden ser interpretados como un er corte del modelo de diseño (aunque son modelos por sí os), y son la entrada principal cuando el sistema es formado en diseño y la implementación. Ésto es, porque el sistema se ibe como un todo y no sólo como la descripción de erimientos.

ntinuación detallaremos la forma en que los componente de cada de los modelos de la fase de análisis se utilizan en este ajo.

Conceptos del Modelo de Requerimientos

iniciar con los conceptos que utilizaremos definiremos emente los casos de uso.

aso de uso es, en esencia, una interacción típica entre un rio y un sistema de cómputo y tiene las siguientes cterísticas:

- Capta alguna función visible para el usuario
- Puede ser pequeño o grande
- Logra un objetivo discreto para el usuario

epresentación gráfica de los casos de uso es el Diagrama de s de uso, que también forma parte del UML.

sos diagramas identificamos actores y casos de uso como datos

Actor es un término empleado para llamar así al usuario d sistema, y es conveniente pensar en un actor como un representan de un rol o papel, no como en las personas ni en los títulos de s puestos.

También es conveniente aclarar que un actor puede desempeñar uno varios casos de uso, y que un mismo caso de uso puede ser realiza por varios actores. Los actores pueden ser humanos u otros sistem externos que necesiten cierta información del sistema en estudio, aunque esta definición no es la definitiva, para nuestr necesidades es la más completa.

Lo más importante de los casos de uso es comprenderlos y defin los objetivos del usuario que satisfacen.

Además de los vínculos entre actores y casos de uso, hay otros d tipos de vínculos. Éstos representan las relaciones de *uses* (usa) *extends* (extiende) entre los casos de uso.

La relación *extend* se usa cuando se tiene un caso de uso que simular a otro pero que hace un poco más, es una variación conducta normal.

Las relaciones *uses* ocurren cuando se tiene una porción comportamiento que es similar en más de un caso de uso y no quiere copiar la descripción de tal conducta.

Los casos de uso, como ya se sabe, se representan con óvalos, l actores son las figuras estilizadas y las flechas indican el flu de información o el estímulo, es decir, son las líneas comunicación entre los casos y los actores.

La realización de estos diagramas es un paso importante generalizado en los métodos del análisis y diseño orientado objetos, a pesar de que algunos autores no lo consideran como u actividad propia del análisis orientado a objetos.

Los casos de uso podemos clasificarlos de la siguiente manera:

- Casos primarios de uso. Representan procesos comunes m importantes como *Hacer retiros*.
- Casos secundarios de uso. Representan procesos menores raros. Por ejemplo: *Solicitud de traspaso a una bóveda sucursal*.
- Casos opcionales de uso. Representan procesos que pued no abordarse (este tipo de casos de uso no l utilizaremos).

- Casos esenciales de uso. Casos expandidos que se expresan en forma teórica de una manera breve y abstracta.

- Casos reales de uso. Describen el proceso de una manera concreta y bastante detallada.

iderando estas definiciones, decidimos utilizar para el rrollo de este capítulo, los Casos Reales de Uso.

desarrollo de los casos de uso se llevará a cabo con el iente formato:

de Uso : NOMBRE DEL CASO DE USO.

res : Actores que participan en el caso de Uso.

men : Breve descripción del caso de uso.

un caso de uso puede contener puntos de decisión, y ralmente una de las opciones es más representativa, ya que las s raramente ocurren, lo más adecuado es escribir el caso típico el Flujo Normal de Eventos y las otras opciones deberán ibirse en una sección titulada Flujos Alternos.

o Normal de Eventos: Tabla que contiene dos columnas, una iene las acciones de los actores y la otra las respuestas del ema a las acciones realizadas por los actores.

os Alternos: Menciona las acciones que deben llevarse a cabo fo ocurren acciones no contempladas en el flujo normal, y que an dicho flujo.

eriormente, si es necesario, se mostrarán también los modelos ominio de los casos de uso, ya que a veces es difícil definir rea del sistema o sus límites.

onsidera necesario mostrar este tipo de modelos cuando la cificación de requerimientos es vaga. Entonces, es necesario rrollar la vista lógica del sistema utilizando objetos del nio del problema, es decir, objetos que tienen un homólogo cto en el ambiente de la aplicación y que deben ser conocidos el sistema.

tipo de modelo ayuda a desarrollar una lista que será una base especificar los casos de uso y a partir de él es posible ir los conceptos con los que el sistema debe trabajar. Su ipal beneficio es que es una herramienta útil para comunicarse el sistema, ya que los usuarios reconocen los conceptos

Por estas razones se considera que sirve como base para el dise
la implementación. Estos modelos no tienen que definir el pro
de una manera exhaustiva porque probablemente durante su dise
desarrollo tendrían que ser modificados.

El modelo en cuestión lo único que debe contener son los nombre
los objetos y posiblemente los atributos lógicos y
asociaciones, que ayudan a especificar de una manera más comp
al sistema.

4.2. Conceptos del Modelo de Análisis

Cuando el modelo de requerimientos ha sido desarrollado,
análisis se enfoca a la estructura del sistema. Esto se l
desarrollando el modelo de análisis, que complementa
comportamiento del sistema especificado en la descripción de
casos de uso, con los objetos involucrados en el modelo
análisis.

En el modelo de requerimientos se especifica que es lo que se
llevar a cabo en el sistema, mientras que en el modelo de anál
(o de dominio) se crea una buena plataforma para el diseño
sistema.

En el modelo de dominio describimos al sistema utilizando
tipos de objetos: de interfaz, de control y de entidad,
representación se muestra en la siguiente figura (Fig. 4.1.).
uno de estos objetos tiene su propio propósito y modelan un asp
específico del sistema.

Fig. 4.1. Tipos de objetos utilizados en el modelo de dominio

A través de los objetos de interfaz se comunican los actores co
sistema. La tarea principal de dichos objetos es traducir
entradas del actor a eventos del sistema y traducir también
eventos generados por el sistema para que el actor los entienda
decir, describen la comunicación entre el sistema y los usuario
manera bidireccional.

Objetos Entidad corresponden a conceptos en la vida real y son que contienen la información que el sistema manejará durante un o periodo de tiempo, es decir, son depósitos de información que en atributos que pueden ser descritos como una asociación con ombre, una cardinalidad y el tipo de atributo.

objetos de control contienen el comportamiento que no puede ser rito por los otros dos tipos de objetos, ya que no pertenece a nterfaz ni tampoco a la manera en como va a ser manejada la rmación. Generalmente actúa como la unión de otros objetos para formen un solo caso de uso, aunque cada uno de estos objetos tener tareas limitadas para que sean fáciles de entender y ribir.

camente con estos tres tipos de objetos se crearon los modelos se presentan a lo largo de este capítulo.

tipo de diagramas que utilizamos fueron los diagramas de vidades, que son particularmente útiles en conexión con el o de trabajo y para la descripción del comportamiento que tiene gran cantidad de procesos en paralelo. El símbolo clave de s diagramas obviamente es la actividad y su interpretación nde de la perspectiva desde la cual se dibuja el diagrama. eptualmente, una actividad es cierta tarea que debe ser llevada bo, ya sea por un humano o por una computadora pero, desde el o de vista de un colaborador en el desarrollo del sistema, una vidad es un método sobre una clase.

ás de las actividades normales, se utilizan símbolos minados disparadores que básicamente son actividades de sión en las cuales se tiene que optar por un camino u otro, y ién se tienen las barras de sincronización, que nos indican que rentes actividades pueden llevarse a cabo en paralelo, lo cual ifica en esencia que su orden no es significativo.

la figura 4.2. se muestra la notación de un diagrama de vidades. En estos diagramas, las actividades son representadas medio de rectángulos con esquinas redondeadas. Las líneas zontales a donde se concentran, o de donde parten dos o más os (flechas), son las barras de sincronización y el rombo esenta un disparador o expresión lógica a evaluar cuyo ltado (denominado guardia) puede ser verdadero o falso.

camente, un diagrama de actividades es para seleccionar el n en que se realizarán las cosas, indica las reglas esenciales ecuencia que se deben seguir. Probablemente algunas personas en que son iguales a los diagramas de flujo y en parte es así,

se pueden tener procesos simultáneos y en los diagramas de f no. Este punto es importante porque se puede mejorar la eficie y capacidad de respuesta de los procesos del negocio al identif cuales son los procesos que pueden llevarse a cabo de ma simultánea, pero también a veces, la utilización de estos diagr puede no dejar muy claros los vínculos entre acciones y objetos.

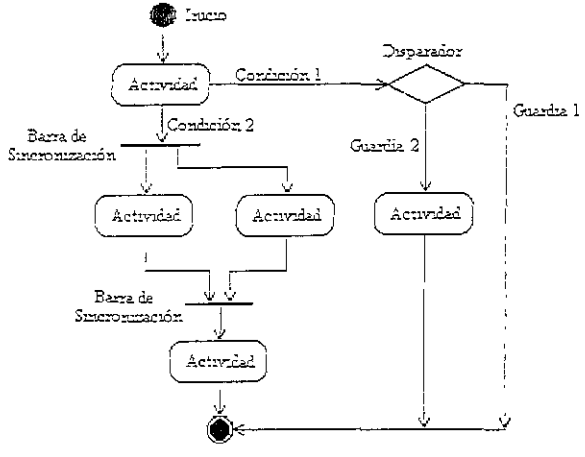


Fig. 4.2. Diagrama de actividades.

A través de la discusión de este capítulo se mostrarán diferentes conceptos que son usados en la práctica, por medio proceso de traspasos. El proceso de traspasos se eligió como caso de estudio para ser descrito en este capítulo, debido a que uno de los procesos más completos. Aunque sólo se describa proceso de traspasos, los elementos de análisis de los proce restantes se encuentran en el Anexo A.

4.3. Modelo de Requerimientos

El modelo de requerimientos, como se mencionó, tiene como propós delimitar el sistema y definir la funcionalidad que se ofrece. modelo funciona como un contrato entre el equipo de desarroll los usuarios del sistema, y proporciona al equipo de desarro desde su perspectiva, lo que el usuario quiere. Sin embargo, fundamental que las personas que no están familiarizadas con

arrollo de sistemas orientados a objetos, puedan leer y entender e modelo.

modelo de requerimientos es la guía para el desarrollo de otros elos, así que este modelo es el núcleo del desarrollo entero del tema. El modelo de requerimientos será estructurado en la etapa análisis, realizado en la etapa del diseño, utilizado en la lementación y probado en el modelo de pruebas. El modelo de uerimientos es la base de los manuales de usuario, debido a que lquier cosa que deba hacer el sistema está descrita aquí desde punto de vista del usuario.

o el desarrollo del sistema inicia conociendo lo que el usuario ere hacer con el sistema, éste es construido desde el punto de ta del usuario, por lo tanto es posible discutir el modelo de uerimientos con los usuarios y hacer los cambios que llegarán a icitar.

modelo de requerimientos se basa, fundamentalmente, en el modelo los casos de uso. Este modelo especifica la funcionalidad que el tema debe ofrecer al usuario, desde su perspectiva y el equipo desarrollo que es lo que pasa dentro del sistema. Este modelo liza actores para representar los papeles que el usuario juega, asos de uso para representar lo que el usuario puede hacer con sistema. Cada caso de uso es visto como un flujo completo de ividades que se realizan hasta el fin del caso de uso. Estas ividades se describen a detalle, en la especificación de los os de uso, y se representan en un diagrama de actividades.

esta forma el modelo de requerimientos sirve como una base común a toda la gente que se involucra en el desarrollo del sistema ipo de desarrollo) y en la especificación de los requerimientos uarios).

1. Actores

a identificar los casos de uso que el sistema lleva a cabo, es esario identificar primero a los usuarios del sistema que ominamos actores. Los actores dan un modelo en el que el usuario identifica; el actor es un tipo de usuario o categoría de urios, y cuando un usuario hace algo con el sistema, el o ella lan como una ocurrencia de algún tipo de actor. De esta forma, persona puede desempeñar el papel de distintos actores. Así, actores definen los papeles que el usuario puede desempeñar.

actor es todo aquello que necesita intercambiar información con

también otros sistemas o equipos, como pueden ser impresoras.

En el caso de traspasos encontramos que existen tres act nombrados como pagador, receptor e interventor. Estos son lo ún que pueden usar el sistema para realizar un traspaso y lo hace distintas maneras:

El pagador tiene como función principal el dar salida a las pi de la bóveda(origen) que es responsable. Lo que implica que se que inicie el traspaso.

El receptor da entrada, en la bóveda que está bajo responsabilidad, a las piezas que se enviaron desde otra bóve que tienen a la suya como destino.

El interventor tiene la responsabilidad de verificar que las pi que salieron de la bóveda origen sean las mismas que llegan bóveda destino, y autorizar la operación de traspaso.

Para aclarar aún más, la diferencia entre usuarios y acto definimos a un actor como una clase, que es, la descripción de comportamiento. Por otra parte, un usuario puede actuar como u más actores. Por ejemplo, un usuario del sistema puede ser J quien tiene a su cargo una bóveda y cuando necesita enviar pieza otra bóveda desempeña el papel de pagador, por el contrario cu recibe piezas provenientes de otra bóveda, desempeña el papel receptor.

4.3.2. Casos de Uso

Una vez que se ha definido que es lo que hay afuera del sist entonces es posible definir la funcionalidad que hay dentro través de los casos de uso. Cada caso de uso es un flujo comp de actividades iniciadas por un actor y en el cual se especifica interacción con el sistema. De esta forma, el caso es una esp de diálogo entre actor y sistema, en el cual intercam información y secuencias de acciones uno con el otro. Todos casos de uso representan todas las formas que hay de usar sistema.

Para identificar los casos de uso se analizan los requerimie desde la perspectiva del usuario, lo que lleva a discusiones ac de los papeles que desempeñan y se plantean algunas preguntas ayudan a vislumbrar el caso de uso, tales como:

- ¿Cuáles son las tareas principales de cada actor?

- ¿El actor tiene que leer, escribir o cambiar información en el sistema?

l caso de traspasos podemos identificar un número determinado casos de uso. Partiendo de acciones de los actores podemos tificar los casos de uso. Para que sea posible un traspaso es sario tener una bóveda origen y una bóveda destino, así que zaremos con el actor responsable de la bóveda origen, que es el dor.

pagador da la salida de las piezas, en contenedores, de su da. En ese momento debe dar de alta el traspaso, en donde ura toda la información necesaria y especifica los contenedores están saliendo, así como la bóveda de destino. En el caso de el traspaso se cancele, debe registrar la cancelación en el ema y regresa los contenedores a su bóveda origen. Si existe solicitud en la que se pida el traspaso de contenedores de su da a otra, es necesario que cuando atienda la solicitud la ue como atendida para indicar a quien originó la solicitud, que preparando los contenedores a enviar. Cada vez que de alta o ele un traspaso, debe introducir una autorización junto con el rventor.

nterventor supervisa que se no se presenten malos manejos en flujos de las piezas, por lo que debe autorizar cada miento de contenedores que se dé, introduciendo una clave y raseña en el sistema. Operativamente, acompaña los contenedores a que llegan a la bóveda destino y los recibe el receptor.

eceptor tiene que registrar en el sistema la entrada de los enedores que está recibiendo. Además tiene que autorizar, junto el interventor, la operación de recepción. Además puede generar citudes de traspaso y cancelarlas. En la figura 4.3. se muestra iagrama de casos de uso para los traspasos.

l diagrama se observa que algunos de los casos de uso están ados con un asterisco, esto significa que son casos de uso es o deseables, y no son necesarios para que el sistema one.

casos de uso que se identifican en la figura son:

- Alta de la solicitud del traspaso
- Recepción de la solicitud del traspaso

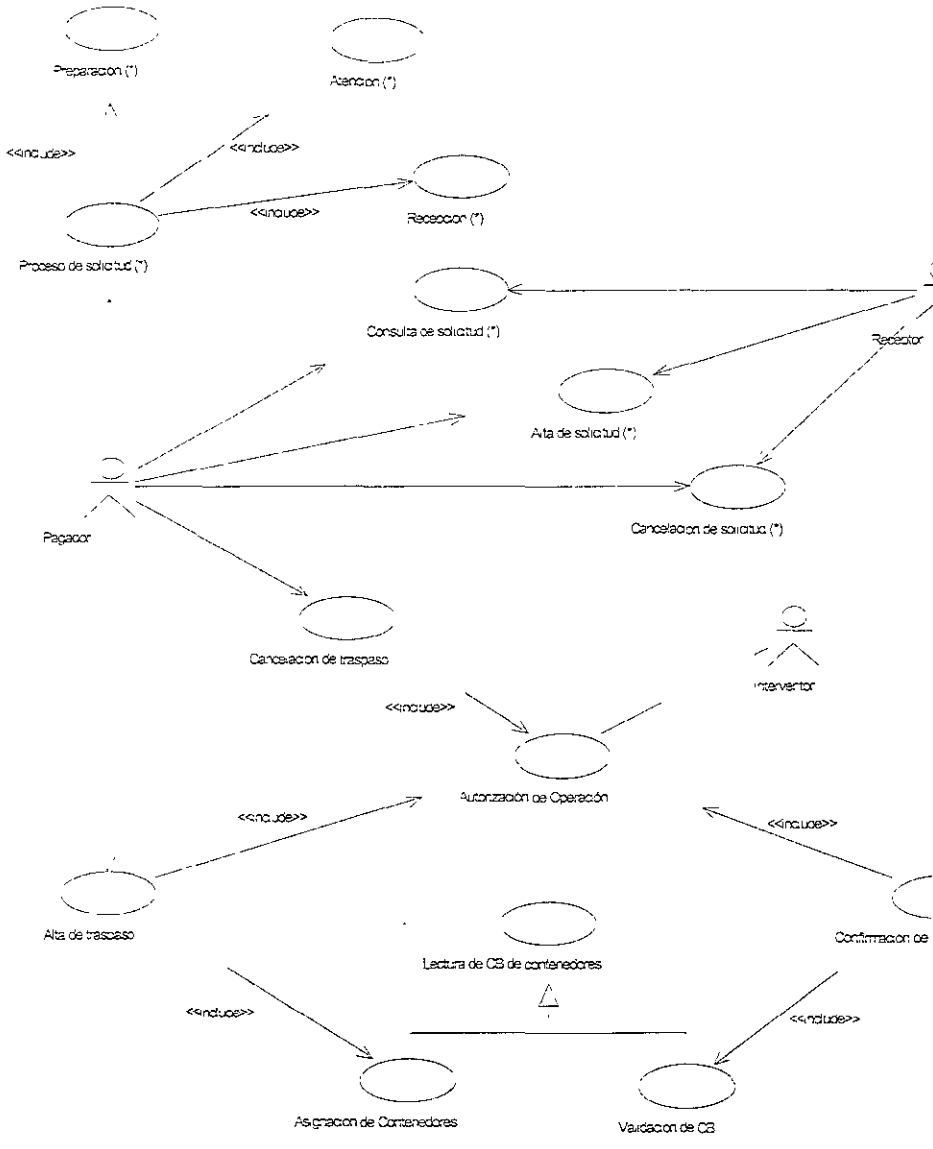


Fig. 4.3. Casos de uso y actores del proceso de Traspasos.

- Cancelación de la solicitud del traspaso
- Alta del traspaso
- Recepción del traspaso

l diagrama anterior encontramos que existen líneas asociativas representan la comunicación entre el actor y el caso de uso, la a que se representa como discontinua indica que la comunicación e dos casos de uso es del tipo "incluir", lo que significa que proceso en particular se incluye en otro y seguramente en otros es decir que es un caso de uso que puede aprovecharse en otros esos dentro del sistema y que no se inician por un usuario ctamente. Los casos de uso marcado con astericos son ideales, ue significa que el usuario no necesita que estén contenidos en istema.

ontinuación se describen algunos de los casos de uso que rvienen en el sistema.

zaremos por el caso de uso para dar de alta un traspaso.

de Uso . Alta de Traspasos.

res : Pagador e interventor.

nen : El Alta de Traspaso requiere que el pagador lemente la información propia del traspaso con la relación de enedores definida para cada traspaso, además de que esta ación deberá ser autorizada electrónicamente.

o normal de los eventos.

ón de los actores	Respuesta del Sistema
l caso de uso inicia cuando l pagador selecciona la pción de capturar un raspaso.	2. El sistema deberá mostrar la pantalla de captura correspondiente, con la clave y descripción de la sucursal donde se realiza el traspaso.

bla 4.1. Caso normal para el alta de un traspaso. (Continúa)

3. El pagador captura los datos correspondientes al traspaso: • Clave de la operación correspondiente al traspaso. • Clave de la divisa. • Clave de la oficina pagadora. • Clave de la oficina receptora. • Clave del estado físico del efectivo que se envía. • Clave del estado físico del efectivo que se recibe. • Clave(s) de la(s) denominación(es). • Importe por denominación.	4. El sistema mostrará descripciones referentes a claves, con ello el usu identificará plenamente información del traspaso. Por cada denominación captu el sistema debe calcular número de piezas equivalente importe de tal denominación. El sistema calculará el imp total del traspaso.
5. El pagador utiliza la lectora de códigos de barras portátil para leer cada uno de los contenedores que forman el traspaso.	
6. Incorpora la lista de contenedores de la lectora portátil a la información del traspaso.	
7. El pagador selecciona la opción aceptar el alta del traspaso.	8. Se despliega la ventana autorización.
9. El interventor y el pagador introducen las claves respectivas.	10.Valida que las cl proporcionadas tengan dere necesarios para autorizar operación.

Tabla 4.1. Caso normal para el alta de un traspaso. (Continúa

	11.Agrega a la información del retiro proporcionada por el Administrador, la clave del usuario que registró la solicitud de retiro.
	12. El sistema agrega los siguientes datos a la información del traspaso: ◦ Clave de la persona que registro la aceptación del traspaso. ◦ Claves de las personas que autorizaron la recepción del traspaso. ◦ Fecha y hora de registro del alta del traspaso. ◦ Login del usuario que captura el alta.
	13. El sistema valida que el importe de los empaques corresponda con el importe por denominación capturados y el estado físico del contenido corresponda con el estado físico origen capturado.
	14. Almacena la información del traspaso en la base de datos del servidor y lo identifica como "dado de alta", y le informa al usuario cual es el número de folio de documento generado para el traspaso.
	15. Disminuye las existencias de la bóveda o ventanilla origen y pone en tránsito los contenedores incluidos en el traspaso.
	16. Está listo para registrar los datos de un nuevo traspaso.

Tabla 4.1. Caso normal para el alta de un traspaso.

Flujos Alternos

Si el sistema detecta que algún importe de los empaques denominación no es múltiplo del importe correspondiente traspaso, deberá mostrar un mensaje de error y le permitirá usuario continuar en la Captura de datos para hacer correcciones necesarias.

Si el sistema rechaza un traspaso, deberá almacenar en la bas datos local la información capturada, y para identificarlo rechazado le asignará un número de folio temporal y le permi hacer al usuario las correcciones necesarias.

En caso de que las claves solicitadas no sean las autorizadas e punto 10, el sistema no deberá permitir que se realice la opera del traspaso mandando un aviso en el que se indique el error.

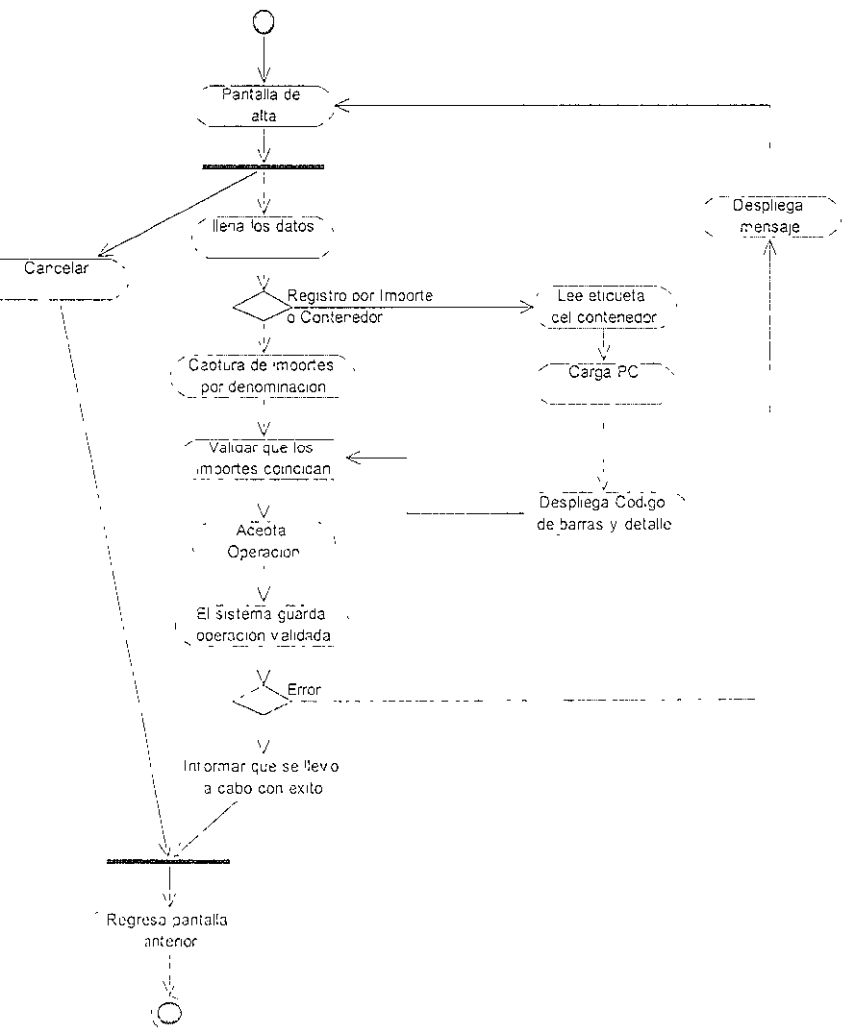
En la figura 4.4 se muestra la utilidad de los diagramas actividades, en el se muestra la secuencia de pasos, proce puntos de decisión y bifurcaciones que indican lo que oc durante el proceso.

El diagrama de actividades, es una representación gráfica de descripción de los casos de uso.

A continuación se muestra el diagrama de actividades del alta traspaso, donde se indica como se van desarrollando las activid para el alta de un traspaso a partir de un punto inicial en d se presenta la pantalla con información de los traspasos hasta se almacena la información.

De esta forma se puede ver en el diagrama que la secuencia actividades inicia cuando se abre la pantalla correspondiente a traspasos, el usuario llena todos los campos de la pantalla co información respectiva. En caso de que el usuario indique que traspaso involucre contenedores, tendrá que leer cada uno de códigos de barras de las etiquetas de los contenedores. De la m forma si se van a traspasar piezas sueltas, esto es que no está contenedores, el usuario debe capturar los importes por denominación. La actividad siguiente es que el sistema almacen operación en las base de datos y debe informar al usuario resultado obtenido. En caso de que no se haya podido almacenar operación exitosamente, el sistema debe informar el error suce para que el usuario corrija la información y se intente almc una vez más el traspaso. En caso contrario a que la operació haya almacenado exitosamente, el sistema debe regresar a pantalla principal e informar al usuario que la operación almacenada.

sistema debe dar la opción para que el uaurio cancele la ación que esta realizando. Así, si hay un error el usuario e decidir no corregirlo y el sistema no almcenará información na del traspaso.



Otro caso de uso muy sencillo es el del proceso de Impresión código de barras que describiremos a continuación.

En el caso de uso de impresión de código de barras se identifica Supervisor como único actor y que es el encargado de imprimir códigos.

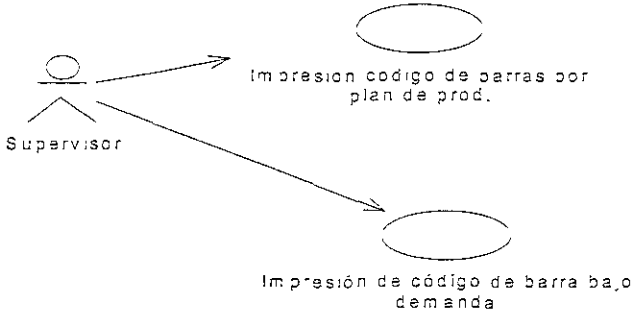


Fig. 4.5. Diagrama de Casos de Uso del proceso de impres de Código de Barras.

Caso de Uso : Impresión de código de barras bajo demanda.

Actores : Supervisor.

Resumen : Este caso de uso tiene la función de calcul generar, imprimir y registrar las etiquetas con código de bar necesarias para un número determinado de envases.

Flujo Normal de los Eventos

Acción de los actores	Respuesta del Sistema
1. El usuario solicita la impresión de etiquetas bajo demanda.	2. Controla los códigos barras impresos con el obj de no repetir su impresión

Tabla 4.2. Flujo normal para la Impresión de Códigos de Barras (Continúa)

ndica la cantidad de hojas tiquetas que requiere imir, proporcionando los ientes datos: ◦ Cantidad de etiquetas ◦ Envase ◦ Denominación ◦ Área que realiza el empaque	4. Registra la clave del usuario que imprimió las etiquetas y área a la que pertenece.
	5. Actualiza el número de la última etiqueta impresa para la institución, denominación y tipo de envase solicitado.
	6. El sistema generará números de etiquetas a partir de las últimas registradas

abla 4.2. Flujo normal para la Impresión de Códigos de Barra.

l caso de uso de impresión de etiquetas por plan de producción suario hace uso de la misma opción en el sistema, la única rencia es que indica la cantidad de hojas que se van a cortar. sistema hace un cálculo basado en la cantidad dada para rminar el número de piezas que se van a empacar y genera las uetas correspondientes a las unidades de empaque calculadas.

aso de uso de impresión por plan de producción es ideal, debido e es más flexible para el usuario indicar el número y el tipo tiquetas que requiere imprimir a través del caso de uso de esión bajo demanda.

cualquier forma, el sistema debe llevar el control de las uetas que se han impreso, sin importar el método, para evitar se tengan códigos de barras duplicados y con el tiempo se an dos ó más unidades de empaque con el mismo código de tificación.

ntinuación se describe el caso de uso denominado Cambio de do Físico que en capítulos anteriores ya ha sido explicado y en igura 4.6. se muestra el diagrama de casos de uso respectivo.

de Uso : Cambio de Estado Físico

Resumen : El cambio de estado físico comprende el alta modificación y la cancelación de cambios de estado físico.

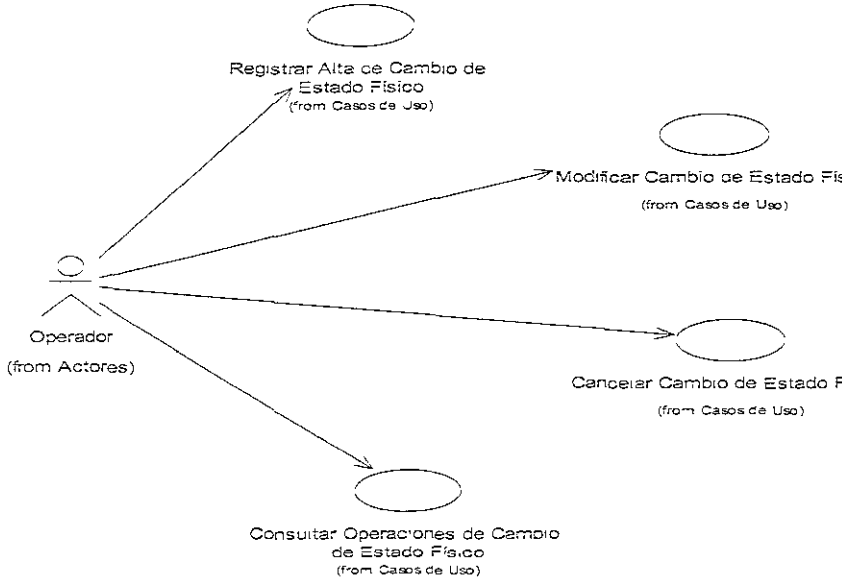


Fig. 4.6. Diagrama de Casos de Uso del Cambio de Estado Físico.

Flujo normal de los eventos.

Acción de los actores	Respuesta del Sistema
1. El operador elige la opción Alta de Cambio de Estado Físico.	2. El sistema muestra la pant de registro con los siguie campos: • Fecha de Registro (inhabilitado: asigna por el sistema) • Folio (inhabilitado: asignado por el sistema) • Tipo de efectivo • Comentario • Institución • Divisa

Tabla 4.3. Caso normal para el Cambio de Estado Físico. (Conti

	◦ Bóveda ◦ Plaza ◦ Estado físico destino ◦ Estado físico origen Se presentará además un campo para indicar si se captura la información en unidades de empaque o en monto, un botón para aceptar y uno para cancelar.
aptura los datos generales de peración.	4. El sistema desplegará los siguientes campos para cada código leído: Espécimen (inhabilitado: asignado por el sistema) En caso de que sea por monto, el sistema desplegará además los siguientes campos: Folio de sistema (asignado por el sistema) Monto total al que se le hará el cambio de estado físico.
El operador captura el lle.	7. El sistema almacena los datos de la operación en la base de datos.

Tabla 4.3. Caso normal para el Cambio de Estado Físico.

ntinuación describiremos una parte del proceso de empaque. do a que es complicado y requiere una atención especial para su ementación, decidimos dividir este proceso en los siguientes s de uso:

- Registro de inicio de producción
- Compensación de mazos
- Asignación de información a mazos o paquetes
- Empaque de mazos en paquetes

- Empaque de paquetes en bolsas
- Empaque de bolsas en contenedores
- Reasignación

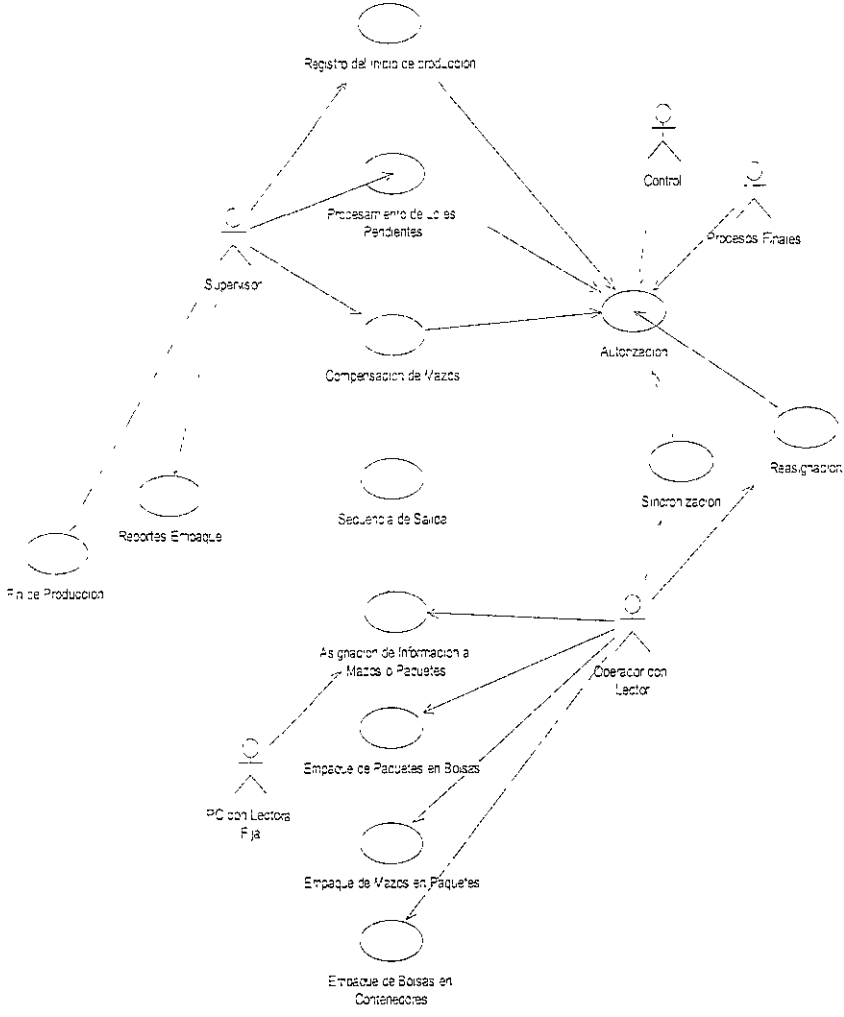


Fig. 4.7. Diagrama de Casos de Uso del proceso de Empaque.

a Figura 4.7. se puede observar que los actores en estos casos so son el supervisor del área de empaque, una persona que lleva ontrol del proceso y los operadores con lectores de códigos de as.

de Uso : Registro de Inicio de Producción.

res : Supervisor del área de empaque, Responsable del de control y del área de Procesos Finales.

nen : Describe la secuencia de actividades realizadas el registro de inicio de producción en las áreas de empaque y *ack* (Corte y Empaque Automático).

o normal de los eventos.

ón de los actores:	Respuesta del Sistema:
icia cuando el supervisor e la opción del sistema para strar el inicio de cción.	2.El sistema le muestra la pantalla correspondiente.
gistra la siguiente rmación: • Clave del supervisor. • Área de trabajo. • Denominación. • Tiro. • Número de millón. s generales de cada uno de otes a procesar como son: • Número de lote • Número de millar de inicio. • Indicar si está completo o no	4. Asigna la fecha de registro (fecha actual) y al tener la denominación y el tiro muestra los siguientes datos: • Fecha de emisión (no modificable). • Serie (no modificable). • Prefijo (no modificable). • Además, debe poner por omisión: • El número de millar de fin: • En *CutPack* se calcula restando nueve al millar de inicio. • En el área de empaque se le suma nueve al millar de inicio.

Tabla 4. Caso normal para el Registro de Inicio de Producción. (continuación)

	• Cantidad de hojas que contiene (de acuerdo millar de inicio y de fin).
	5.Valida que se produzcan millares por lote.
	6.Valida que el millar de fin menor que el millar de inici *CutPack* o que el millar de sea mayor que el de inicio e área de empaque.
	7.Presenta los millares de lote que están pendientes procesar indicando la fecha registro de la producción.
8.El supervisor decide procesar o no los lotes pendientes.	9.Ordena los lotes de cada por el millar en forma ascend o descendente según sea le c y les asigna un número que in el orden en el cual deben procesados.
10. La persona responsable del control y la persona responsable de procesos finales realizan la autorización del registro de producción ingresando sus claves de usuario.	11. Valida las claves ingresa
	12. Almacena la informa correspondiente al inicio producción.

Tabla 4.4. Caso normal para el Registro de Inicio de Producció

Flujos Alternos:

- Si en el punto 5 el sistema detecta que un lote t menos de 10 millares, debe permitir que el usuario señale como lote incompleto, que registre los mill del lote y la secuencia en la que se trabajarán y final, el sistema debe mostrar al usuario su informa asociada.

olvidar que para la realización de este caso de uso, es sario que previamente exista la autorización de emisión del tivo a producir.

de Uso : Empaque de mazos en paquetes.

res : Operador del área de empaque con la lectora de go de barras.

men : Describe el proceso para formar paquetes con cinco s de denominación alta o con seis mazos de denominación baja.

o normal de los eventos.

ón de los actores:	Respuesta del Sistema:
Inicia cuando el operador cciona la opción de Empaque azos en Paquetes.	2. Presenta la pantalla correspondiente.
ee con la lectora de códigos barras la información de la ueta del mazo que formará e del paquete.	4. Despliega en pantalla los siguientes datos del mazo leído: ◦ Denominación ◦ Tiro ◦ Lote ◦ Millón ◦ Millar ◦ Serie ◦ Prefijo ◦ Posición del mazo para denominaciones de $500. ◦ Porcentaje de avance de producción con respecto a los datos capturados al inicio de la misma.
	5. Valida que la lectura realizada corresponda con la esperada.
ontinúa con la lectura de los s mazos hasta haber leído o	7. Indica formación de un paquete.

Tabla 4.5. Caso normal para el Empaque de Mazos en Paquetes (continúa)

8. Realiza la lectura del código de barras que contiene la etiqueta pegada al paquete formado.	9. Registra el paquete y asigna los siguientes datos acuerdo a la secuencia de sal • Códigos de los mazos contiene. • Tipo de paquete (chic grande).
	10. Está listo para continuar la lectura de mazos para fo otro paquete.

Tabla 4.5. Caso normal para el Empaque de Mazos en Paquetes.

Flujos Alternos

- Si en el paso 4, el sistema no encuentra la informa asociada a la etiqueta del mazo leído, debe mandar aviso de error al usuario y quien debe corregir la fal

4.4. Modelo de Análisis

Una vez que está el modelo de requerimientos, el análisis se en a la estructura del sistema. Esto es a través del desarrollo modelo de análisis

En el modelo de requerimientos se especificó aquello que su dentro del sistema. En el modelo de análisis se establecen bases que se utilizarán en la fase de diseño. Así, el modelo requerimientos queda estructurado por medio del modelo de anális

Durante el desarrollo del modelo de análisis se distribuye comportamiento de cada caso de uso, especificado en el descrip de cada uno, en objetos. Un objeto puede ser común a difere casos de uso. Sin embargo, debe establecerse la funcionalidad cada objeto para cada comportamiento del caso de uso donde utiliza. En esta fase no se describen las operaciones o funci específicas de cada objeto. Por el contrario, se escribe una b descripción de la funcionalidad identificada.

En la figura 4.8. se muestran las perspectivas que se obtienen el modelo de requerimientos y el modelo de análisis. Donde casos de uso y los objetos son diferentes vistas del mismo sist

a base se muestran los casos de uso y en la parte lateral se tran los objetos.

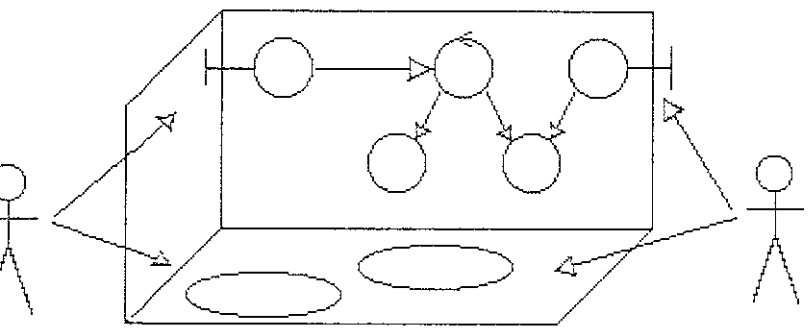


Figura 4.8. Vistas de los casos de uso y objetos.[2]

. Diagrama de Objetos

se mencionó, el modelo de análisis es la representación de n tos. En la figura 4.9. se muestra el diagrama de objetos para lta de traspasos.

objetos de interfaz son aquellos con los que el usuario rcambia información con el sistema, en este caso son.

antalla de lectura de CB. Donde se capturan los códigos de as de los contenedores que se movilizan.

antalla de Autorización. Donde se introducen las claves de rización que validan la operación.

antalla de registro de traspaso. A través de la cual se stra la información para dar de alta un traspaso.

a fase de diseño, las clases de interfaz deben apegarse a icas de diseño de interfaces gráficas para el usuario.

objetos de entidad que se identifican almacenan toda la rmación que el usuario captura en el alta del traspaso. En

obson, "Object Oriented Software Engineering", Addison-Wesley, pp. 1.

ellos se almacena la operación, el detalle del traspaso y envases involucrados. En la fase de diseño se describen relaciones entre ellos y los datos que contiene cada uno.

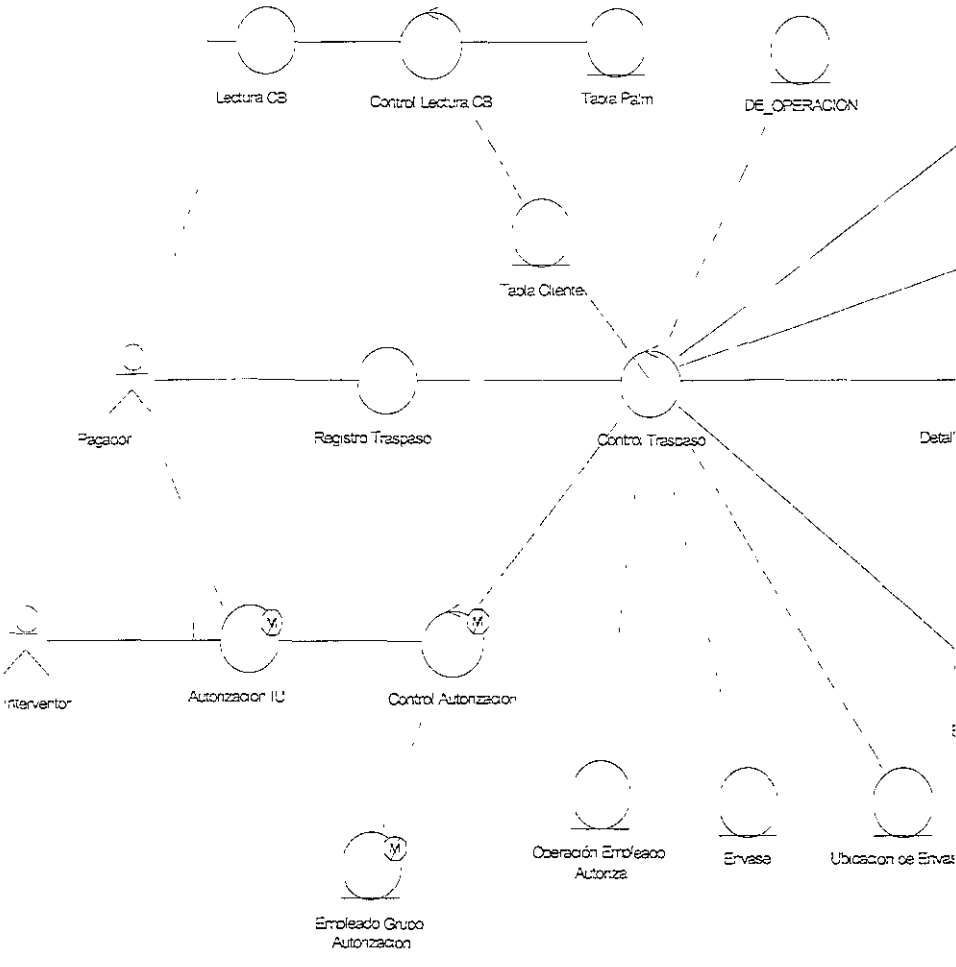


Fig. 4.9. Diagrama de objetos del alta de traspaso.

En algunos casos la funcionalidad del caso de uso puede esta estos dos objetos, sin embargo, en el caso de traspasos necesaria una funcionalidad más compleja. Tal funcionalidad puesta en los objetos de control. La razón del porque es dif

r esa funcionalidad en cualquiera de los dos objetos antes
ritos, es porque realmente no pertenece a las pantallas del
ema o la forma en que la información es almacenada. De esta
a el objeto de control se encarga de resolver este problema y
e como unión entre los objetos de entidad y de interfaz.

el caso de traspasos se tienen tres objetos de control:

ontrol lectura CB. Se encarga de toda la funcionalidad para
códigos de barras y ponerlos en pantalla o almacenarlos.

ontrol autorización. Se encarga de validar que las claves de
rización de una operación sean correctas.

ontrol de traspaso. Hace cálculos con el número de piezas que
ovilizan y valida la información del traspaso, para después
cenarla en los objetos de entidad.

a figura 4.10. que es el diagrama de objetos del proceso de
lo de estado físico, podemos observar que los objetos de
rfaz son:

pantalla de Cambio de Estado Físico. Donde se captura la
rmación necesaria para hacer el cambio de estado físico
rita en el caso de uso correspondiente.

pantalla de la *Palm*. Donde se introduce la información de
ses que van a ser cambiados de estado físico.

este caso se tienen dos objetos de control:

ontrol *PALM*. Se encarga de toda la funcionalidad para ingresar
nformación que se lee con la *PALM*.

ontrol cambio de estado físico. Se encarga de realizar los
ulos para asignar la numeración a las etiquetas.

objetos de entidad que se observan, almacenan toda la
rmación que el usuario captura en este proceso. En ellos se
cena la operación, el detalle del traspaso, los códigos de
as de cada uno de los contenedores involucrados en la
ación, las existencias en piezas, la información necesaria que
a las afectaciones contables a aplicar, etc.

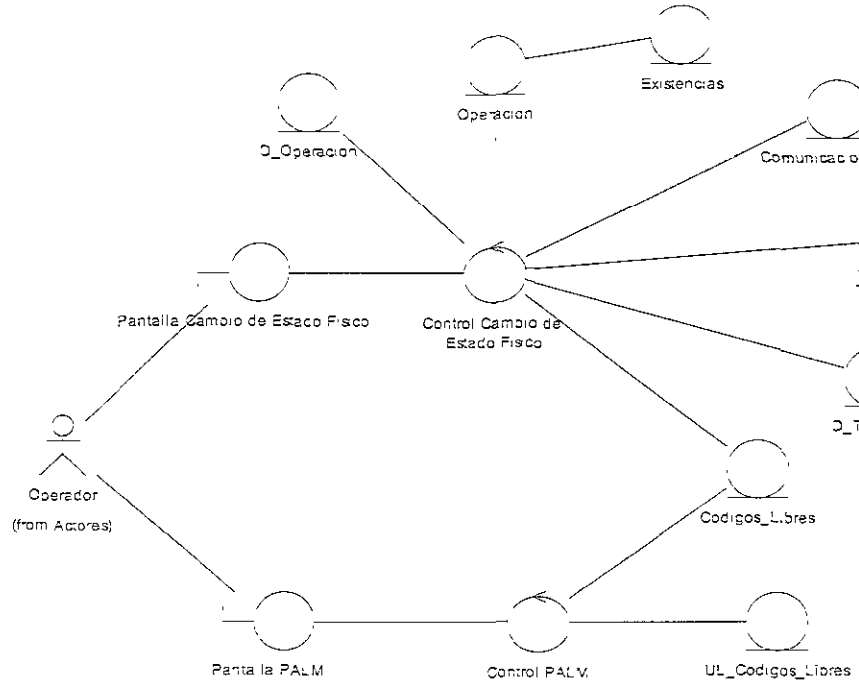


Fig. 4.10. Diagrama de objetos del Cambio de Estado Físico.

4.4.2. Diagrama de Colaboración

En la figura 4.11. se muestra funcionalidad de cada objeto través de una breve descripción de ésta, la dirección en intercambio de información y la secuencia en que se da. A este de diagrama se le denomina de colaboración.

Al trabajar en el modelo de análisis como en este capítulo, trabaja con un caso de uso a la vez. Así, para un caso de específico, se identifican los objetos de interfaz, de contro entidad antes de continuar con el siguiente caso de uso. Aunqu objeto pueda ser usado en otro caso de uso, este proceso iterativo. Esto significa que, aún cuando ya exista un conjunto objetos, éstos pueden ser modificados para acoplarse a otros c de uso.

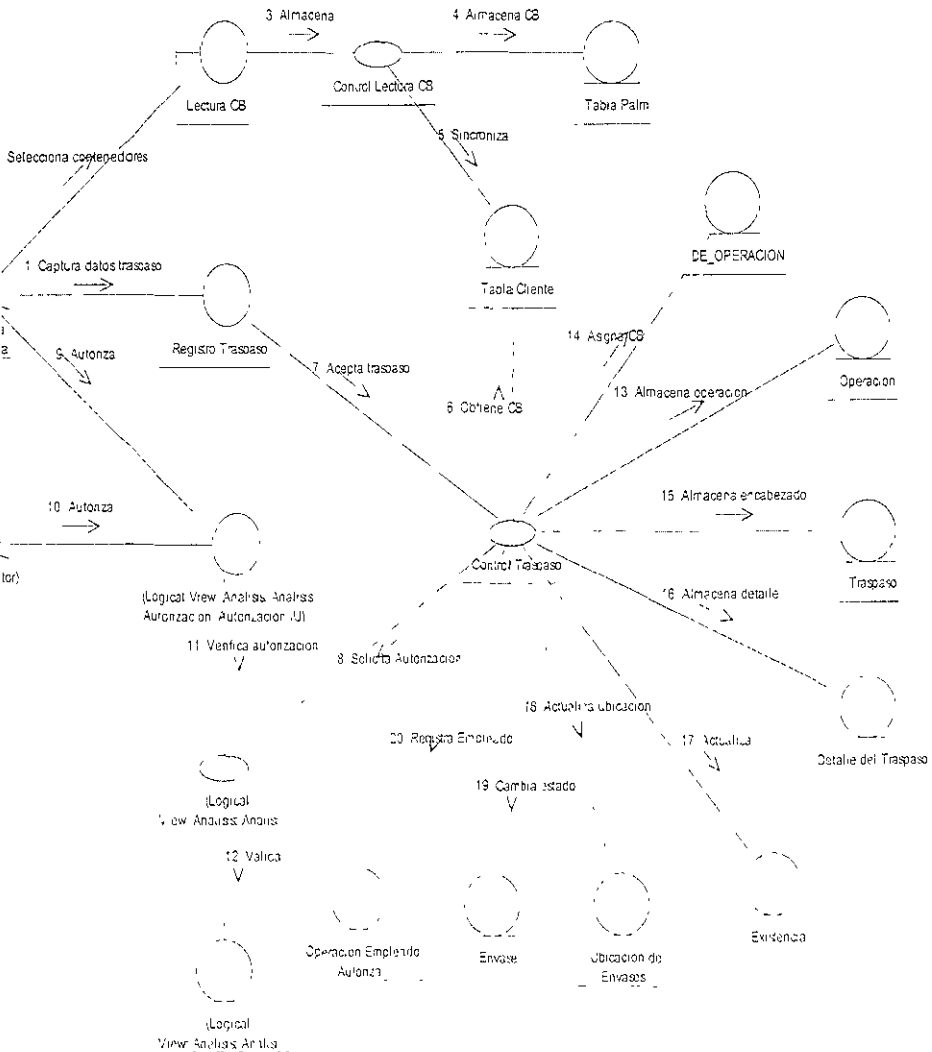


Fig. 1.11 Diagrama de colaboración de objetos.

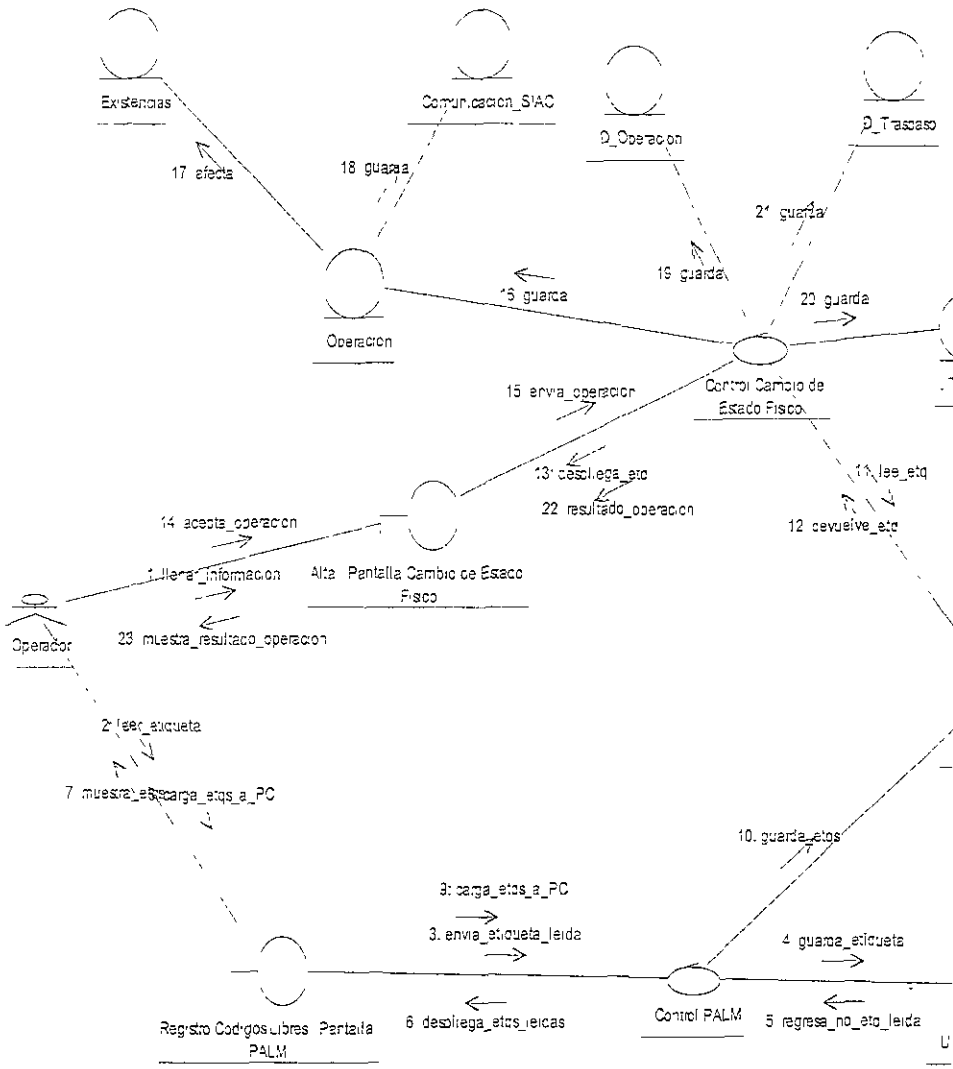


Fig. 4.12. Diagrama de colaboración de objetos para el alta d
cambio de estado físico.

En este diagrama (Fig.4.12.), se muestran las relaciones que ti
los objetos entre sí para modificar el estado físico del efectiv

ejemplo, el usuario llena la información solicitada por el
to de interfaz y el cual responde mostrando el resultado del
eso al operador. También se puede distinguir como el control de
io de estado físico recibe una operación y le despliega el
ltado de la operación a la pantalla de cambio de estado físico,
ás de que el mismo objeto de control almacena la información en
diferentes objetos entidad.

camente es así como se manejaron los diferentes procesos que
rvienen en el sistema y debido a que es muy extenso el
isis, decidimos mostrar únicamente algunos de los procesos,
en el Anexo A se encuentran más casos de uso y diagramas.

este capítulo se describió el proceso de análisis de los
querimientos del usuario, se ejemplificó con el traspaso, el
paque de mazos en paquetes y con otros procesos, la forma como
obtuvieron el modelo de requerimientos y el modelo de
álisis.

el siguiente capítulo se describirá la fase de diseño del
stema, tomando como base los elementos obtenidos de la fase de
álisis.

CAPÍTULO V

DISEÑO DEL SISTEMA

este capítulo se describen los conceptos utilizados para la lización del diseño del Sistema de Contenedores, así como te del diseño elaborado.

Modelo de Diseño

onstrucción del sistema se hace utilizando los modelos de sis y de requerimientos que se obtuvieron en la fase de sis. En primer lugar, se crea el modelo de diseño que es un o más refinado y formalizado del modelo de análisis. Al crear odelo de diseño se debe tener en cuenta el ambiente de ementación. El modelo de análisis fue desarrollado bajo ciones ideales, y a través del modelo de diseño se adapta a la dad. Existen dos razones por las cuales el ambiente de ementación no se involucra en la fase de análisis. La primera orque no se pretende afectar la estructura del sistema con eptos de la implementación. La segunda razón es no perder de los problemas que se plantean en la fase de análisis al ucrarlos con la complejidad que surge durante la codificación solución. En este sentido, no se pierde de vista la esencia fase que es crear la estructura básica del sistema.

sta forma podemos decir que el modelo de diseño es una lización del modelo de análisis para adecuarlo al ambiente de [illegible]

dimensiones, como se muestra en la figura 5.1. Las dimensi correspondientes a la información, comportamiento y presentac son heredadas de la fase de análisis. Donde la dimensión información corresponde a la información que se registra en sistema, la dimensión del comportamiento especifica la forma en se comporta el sistema, y la dimensión de la presentación es forma en que el sistema se muestra al exterior, principalmen los usuarios. La dimensión del ambiente de implementación compr la introducción de conceptos a los cuales se debe adaptar el mo de análisis.

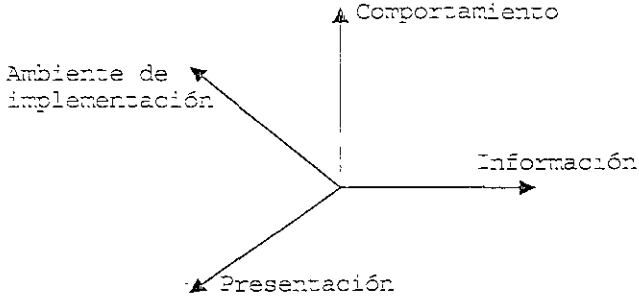


Fig. 5.1. Dimensiones del Diseño.[3]

Al introducir la dimensión del ambiente de implementación en diseño, se tiene como propósito adecuar el modelo análisis ambiente de implementación, al mismo tiempo que refinarlo h dejarlo listo, de tal forma que la codificación sea más senci Debido a que el modelo de análisis tiene todas las propiedades sistema, es la base del modelo de diseño. Sin embargo, se cambios a éste cuando se introducen conceptos tales como, diseñ bases de datos, requerimientos de rendimiento, el lenguaje programación, etc. Es por eso que es necesario desarrollar un n modelo, llamado modelo de diseño.

En el modelo de diseño se refina el modelo de análisis hasta punto de incluir las operaciones en los objetos y los parámetro estas funciones.

En esta fase se utiliza la técnica orientada a objetos lla diseño conducido por responsabilidades. Donde, al crear objeto, se le asigna la responsabilidad de realizar acci específicas, es decir, que se espera un comportamiento determi del objeto ante las reglas establecidas en él. La responsabil

[3] Jacobson, "*Object Oriented Software Engineering*", Addison-Wesl 1996, pp. 144.

ca un grado de independencia o de no interferencia. Por lo, si a un niño se le indica que limpie su cuarto, lmente no se le vigila todo el tiempo en el que realiza la idad - esa no es la naturaleza de la responsabilidad. De forma ar, en el diseño de sistemas se incurre en la dependencia de partes del código con respecto a otras, dentro del sistema. El o conducido por responsabilidades intenta cortar éstas dencias, o al menos reducirlas al mayor nivel posible. De esta se pretende tener objetos que tengan la funcionalidad aria para cumplir con los requerimientos del sistema y puedan sados en distintas partes del sistema de forma independiente e actuando con otros objetos a través de las funciones que son sponsabilidad.

seño de los objetos se representa a través de los diagramas de s y la comunicación entre ellos por medio de diagramas de ncia. A continuación se describen estos conceptos.

Diagramas de Clases

lase es la abstracción de un conjunto de objetos que contienen utos, operaciones y métodos, que son comunes a todo el nto de objetos, es decir, describe un grupo de objetos que n características y comportamiento similar dentro del dominio roblema. Libro, Persona son ejemplos de clases que describen rma genérica diversos grupos de objetos.

método de abstracción es la identificación de las terísticas comunes a un conjunto de elementos, hacia la ipción condensada, de estas características surge la clase. El o de abstracción es arbitrario, ya que depende del punto de , es decir, un objeto del mundo real puede ser visto a través ostracciones diferentes, lo cual implica que es importante minar cual es el contexto de la aplicación.

clase describe las generalidades y el objeto las cularidades. Los objetos informáticos se construyen a partir clase por medio de la instanciación. De esta forma, todo o es una instancia de una clase.

transacción bancaria es una abstracción de una operación erial, refleja una interacción entre un cliente y un banco. talles de [illegible] transacciones habituales, como [illegible] por ejemplo, no son conocidos por el [illegible]

cuestión. La cuenta es otra abstracción del ámbito bancario.

La abstracción disimula la complejidad de la gestión de cuentas, de modo que las abstracciones pueden ser realiz simplemente por el propio cliente, desde cualquier sucursa oficina autorizada. La figura 5.2 ejemplifica lo anterior descrito.

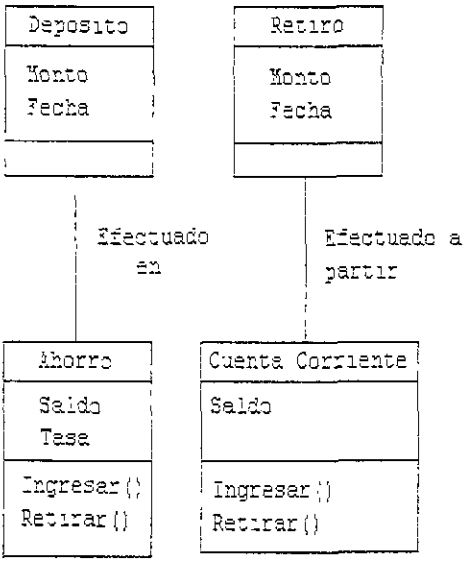


Fig. 5.2. Representación de una parte del ámbito bancario.

Todos los tipos de datos abstractos manipulados en el ám informático son, como su nombre lo indica, abstracciones descr en términos de operaciones aplicables sobre valores. Este géner abstracción pertenece típicamente a la fase de Diseño del sist y no aparece jamás en el análisis, donde el término colecció suficiente para designar las agrupaciones de objetos.

5.2.1. Identificación de Clases

En la figura 5.3. se muestran las distintas representaciones de clases en un Diagrama de Clases.

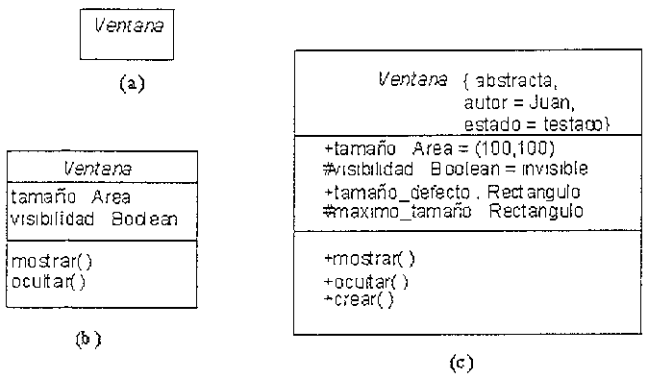


Fig. 5.3. Representación de clases.
(a) Sin detalle. (b) Detalle a nivel de análisis.
(c) Detalle a nivel de implementación.

obtener una clase es necesario identificar los objetos que son vantes para el dominio de nuestro problema y abstraer sus cterísticas.

ntinuación se mencionan las pautas que se pueden seguir para tificar las clases:

- Identificar otros sistemas, elementos o dispositivos. Considerar todos los dispositivos con los que el problema pueda estar relacionado. En nuestro problema sería: Lectora de código de barras, Base de datos, *CutPack (Corte y empaque automático)*, etc

- Identificar acontecimientos que deben ser recordados. Por ejemplo, la firma de un contrato de compra de un departamento, las operaciones realizadas por una cuenta corriente, etc.

- Identificar lugares. Más adelante nos servirá para determinar el contexto en que se produce una determinada asociación entre objetos.

- Identificar unidades organizativas. Por ejemplo: los departamentos en que se divide una empresa, divisiones, etc. En nuestro problema podemos encontrar entre otros Departamento de Contabilidad, Departamento de Dirección, etc.

- Como último, se debe identificar todo aquello que parte inherente de nuestro problema. Por ejemplo Ba Cuentas corriente, etc.

5.2.2. Atributos y operaciones

Un atributo es un dato que caracteriza a cierto objeto de clase, por ejemplo el color de los coches o el título de un li La sintaxis seguida para la descripción de atributos es siguiente:

Nombre_atributo: Tipo_atributo = Valor inicial

Los atributos pueden ser:

- Atributos base. Son los que se obtienen nada más obse las características del objeto a modelar.

- Atributos derivados. Son aquellos cuyo valor se obti en función del resto de atributos del objeto o inclus función de los atributos de otros objetos del sist Estos atributos derivados se representan por una b inclinada al principio del nombre y se transforman operaciones tal y como se muestran en la figura 5 la operación *superficie():integer* se obtiene a partir atributo derivado */superficie:integer*, mismo que obtiene de los atributos longitud y anchura.

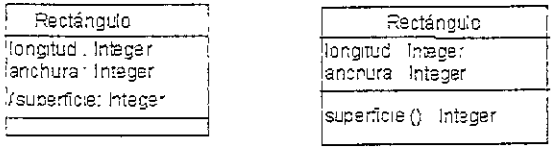
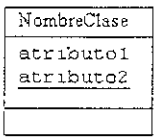


Fig. 5.4. Transformación de un atributo derivado a operación

- Atributos de clase. Estos atributos no caracterizan a objetos individualmente sino al conjunto de todos objetos integrantes de la clase. Se representa por nombre subrayado, tal y como se muestra en la fi 5.5.



g. 5.5. Representación gráfica de los atributos de una clase.

operación es un mensaje a través del cual una clase se comunica otra. Una clase tiene, comúnmente, una o más operaciones; las es determinan el comportamiento de la clase.

intaxis seguida para la descripción de las operaciones es la iente:

re_Operación(Nombre_Argumento:Tipo_Argumento=Valor_Predetermina .):Tipo_Devuelto

specificación de los argumentos se pueden suprimir en los icos, debido a su longitud.

Diagramas de Interacción

diagramas de interacción constituyen una de las herramientas importantes que se generan en el análisis y diseño orientados a os. Son modelos que describen la manera en que colaboran os de objetos para cierto comportamiento, ya que muestran de manera gráfica, la interacción entre objetos ocurrida en un uario o parte del sistema y ayudan a identificar el ortamiento que puede ser ejecutado en paralelo.

s diagramas son realizados en la fase de diseño y no se pueden arar si antes no se generan un modelo conceptual y los casos de A partir del modelo conceptual el diseñador podrá definir las es involucradas en el sistema, ya que los objetos de las clases icipan en las interacciones que se describen gráficamente en diagramas.

a etapa de diseño refinamos la descripción de los casos de uso rando en los diagramas de interacción como se comportan los os. Para cada caso de uso concreto se debe dibujar un diagrama nteracción. La interacción toma lugar cuando los bloques se n estímulos unos a otros por lo que, cuando se dibujan los ramas de interacción se deben definir todos los estímulos

El trabajo de identificar los bloques en el modelo de diseño realizado rápidamente; lo mejor es hacerlo automáticam utilizando herramientas específicas. Por otro lado, el diseño casos de uso implica una gran cantidad de trabajo. Aquí deb definir exactamente como participan los objetos en la comunicac Las descripciones hechas en el análisis pueden ser cambiadas y lo tanto, se afecta la estructura del sistema. Esto puede ocasi propuestas de cambios en el modelo de análisis o el modelo requerimientos.

En la etapa de diseño existen dos tipos de diagramas interacción: los diagramas de secuencia y los diagramas colaboración.

5.3.1. Diagramas de Secuencia

Cuando se diseña un caso de uso se deben identificar de los blo u objetos que participan en el caso de uso. Se debe comenzar el inicio del flujo del caso de uso, y después seguir ese f paso a paso, decidiendo que objetos del diseño y que interacci de instancias de actores son necesarias para realizarlos.

En los diagramas de secuencia, un objeto se muestra como una en la parte superior de una línea vertical. Esta línea vertica llama línea de vida del objeto y representa, como su nombre dice, la línea (tiempo) de vida del objeto durante su interacc El esquelto de un diagrama de interacción se muestra en la fi 5.6.

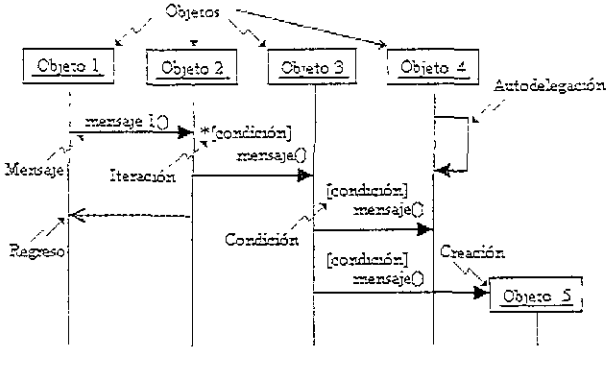


Fig. 5.6. Esqueleto de un Diagrama de Secuencias.

ya se ha mencionado con anterioridad, los objetos se comunican
e sí por medio de mensajes: el objeto origen solicita (llama a)
operación del objeto destino. Cada mensaje se representa en los
ramas de secuencia mediante una flecha entre las líneas de vida
los objetos (Fig. 5.6.). El orden en que se mandan estos
ajes transcurre de arriba hacia abajo debido a que el eje del
po del diagrama de secuencia es descendente. Es necesario
rar que la distancia entre dos eventos en el eje del tiempo del
rama no tiene relación con el tiempo real.

mensaje es etiquetado por lo menos, con el nombre del mensaje;
en incluir también los argumentos y alguna información de
rol, y se puede mostrar la autodelegación, que es el mensaje
un objeto se envía a sí mismo, regresando la flecha de mensaje
uelta a la misma línea de vida como se muestra en la figura

partes de la información de control son valiosas. Primero, hay
condición, que indica cuándo se envía un mensaje. El mensaje se
a sólo si la condición es verdadera. El segundo marcador de
rol útil es el marcador de iteración, que muestra que un
aje se envía muchas veces a varios objetos receptores, como
dería cuando se itera sobre una colección. La base de la
ación se puede mostrar entre corchetes.

ten distintos tipos de mensajes, según cómo se producen en el
po: simples, síncronos, y asíncronos. Los mensajes simples son
que no tiene que realizarse dependiendo de otras tareas. Los
ajes síncronos son aquellos que se llevan a cabo cuando una
a debe esperar a que otra haya terminado para continuar,
tras que en los mensajes asíncronos, son los que cuando una
l es mandada, quien la envió continua su ejecución sin esperar
eceptor, mientras que el receptor puede estar activo cuando la
l llegue por lo que deberá ser encolada hasta que el receptor
rocese.

ién se utiliza una línea flecha punteada para indicar un
eso, que es simplemente el regreso de un mensaje, no un nuevo
aje (fig. 5.6.). Algunos autores consideran que los regresos
ran el diagrama y tienden a complicar el flujo. Todos los
esos están implícitos por el modo como se secuencian los
ajes y sólo se deben utilizar si esto aclara el diagrama.

ros resumir entonces que los diagramas de secuencia muestran
interacciones entre un conjunto de objetos, ordenadas según el
po en que tienen lugar. En los diagramas de este tipo
rvienen objetos, que son instancias concretas de una clase que
[illegible]

la ejecución de la interacción, se puede crear o puede destruido durante la ejecución de la interacción. Un diagram secuencia representa una forma de indicar el período durante el un objeto está desarrollando una acción directamente o a través un procedimiento.

A medida que se van detallando los diagramas de interacción es probable que se encuentren caminos alternativos de los casos uso, y al añadir esta información, se descubrirán nu excepciones no consideradas durante la captura o el análisi requisitos.

5.3.2. Diagrama de Colaboración

El otro tipo de diagramas de interacción es el diagrama colaboración. Estos diagramas muestran la interacción entre va objetos y los enlaces que existen entre ellos. A diferencia d diagrama de secuencias, un diagrama de colaboraciones muestra relaciones entre los objetos, no la secuencia en el tiempo en se producen los mensajes. Los diagramas de secuencias y diagramas de colaboraciones expresan información similar, per una forma diferente.

Formando parte de los diagramas de colaboración nos encontramos objetos, enlaces y mensajes. Un objeto es activo si posee un th o hilo de control y es capaz de iniciar la actividad de cont mientras que un objeto es pasivo si mantiene datos pero no in la actividad.

En estos diagramas los objetos ejemplo se muestran como iconos. flechas indican los mensajes enviados dentro de un caso de dado. Sin embargo, la secuencia se indica numerando los mensajes

Este tipo de diagramas puede utilizar distintas formas del esq de nombrado de objetos en UML. Tal esquema adopta la forma *NombreObjeto : NombreClase*, donde se puede omitir uno u otro. S omite el nombre del objeto, es necesario conservar los dos pu (:), para que quede claro que es el nombre de la clase y no objeto.

Un enlace es una instancia de una asociación que conecta objetos de un diagrama de colaboración. El enlace puede reflexivo si conecta a un elemento consigo mismo. La existenci un enlace entre dos objetos indica que puede existir un interca de mensajes entre los objetos conectados.

Los diagramas de interacción indican el flujo de mensajes e

entos del modelo, el flujo de mensajes representa el envío de ensaje desde un objeto a otro si entre ellos existe un enlace. mensajes que se envían entre objetos pueden ser de distintos s, también según como se producen en el tiempo; existen ajes simples, sincrónicos, *balking*, *timeout* y asíncronos. nte la ejecución de un diagrama de colaboración se crean y ruyen objetos y enlaces.

s estos conceptos se pueden aclarar viendo la siguiente figura 5.7.), en la cual se observa como se representan los objetos, enlaces y los métodos.

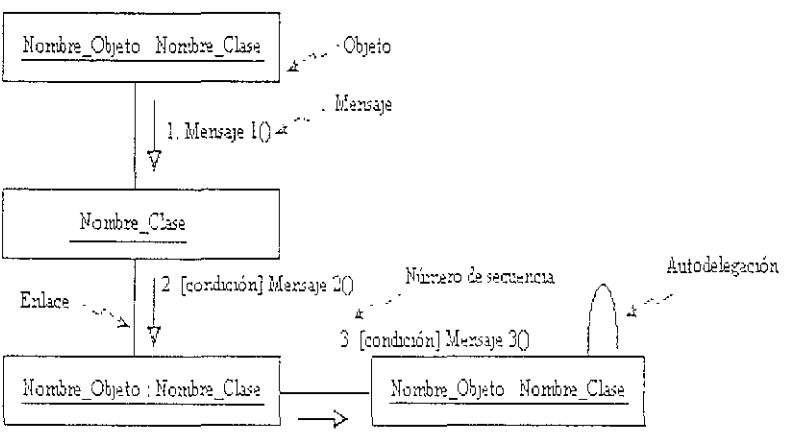


Fig. 5 7. Notación de un Diagrama de Colaboración.

diagramas de interacción son muy útiles cuando el rtamiento que describen es simple y se deben utilizar cuando sea ver el comportamiento de varios objetos en un caso de uso. o se tiene que mostrar un comportamiento condicional lo más endable es utilizar diagramas separados para cada situación en de emplear condiciones en los mensajes, ya que esto los icaría.

do en cuenta las características de los dos tipos de diagramas nteracción descritos anteriormente, decidimos utilizar los amas de secuencia porque consideramos que con ellos es más apreciar el orden en que ocurren los casos debido a [illegible] para [illegible]

5.4. Arquitectura de Desarrollo

La arquitectura utilizada en el Sistema de Contenedores es arquitectura cliente/servidor. Esta arquitectura es un modelo y el desarrollo de sistemas de información en el que, transacciones se dividen en procesos independientes que coope entre sí para intercambiar información, servicios o recursos. denomina cliente al proceso que inicia el diálogo o solicita recursos, y servidor al proceso que responde a las solicitudes.

Este modelo es el modelo de interacción más común en aplicaciones en una red. No forma parte de los conceptos de Internet como los protocolos IP, TCP o UDP, sin embargo todos servicios estándares de alto nivel propuestos en Internet funci según este modelo.

Los principales componentes del esquema cliente/servidor entonces los Clientes, los Servidores y la infraestructura comunicaciones.

En este modelo, las aplicaciones se dividen de forma que servidor contiene la parte que debe ser compartida por va usuarios, y en el cliente permanece sólo lo particular de usuario.

Los Clientes interactúan con el usuario, usualmente en fo gráfica. Frecuentemente se comunican con procesos auxiliares que encargan de establecer conexión con el servidor, enviar el ped recibir la respuesta, manejar las fallas y realizar actividades sincronización y de seguridad.

Los clientes realizan generalmente funciones como:

- Manejo de la interfaz del usuario.
- Captura y validación de los datos de entrada.
- Generación de consultas e informes sobre las bases datos.

Los Servidores proporcionan un servicio al cliente y devuelven resultados. En algunos casos existen procesos auxiliares que encargan de recibir las solicitudes del cliente, verificar protección, activar un proceso servidor para satisfacer el ped recibir su respuesta y enviarla al cliente. Además, deben man

bloqueos, la recuperación ante fallas, y otros aspectos afines.
las razones anteriores, la plataforma asociada con los
idores es más poderosa que la de los clientes. Además deben
jar servicios como administración de la red, mensajes, control
dministración de la entrada al sistema ("login"), auditoría y
peración y contabilidad. En el caso del Banco Central se tienen
idores existe dedicados a dar servicio a las bases de datos.

de las principales aportaciones de esta arquitectura, es la
rfaz gráfica de usuario. Gracias a ella se dispone de un manejo
fácil e intuitivo de la aplicación mediante el uso de un
ositivo tipo ratón. En esta arquitectura los datos se
entan, editan y validan en la parte de la aplicación del
nte.

cuanto a los datos, cabe señalar que en la arquitectura
nte/servidor se evitan las duplicidades (copias y comparaciones
datos), teniendo siempre una imagen única y correcta de los
os, disponible en línea para su uso inmediato.

esto tiene como fin que el usuario trabaje desde su estación
rabajo con distintos datos y aplicaciones, sin importarle dónde
n o dónde se ejecuta cada uno de ellos.

ia figura 5.8 se muestra los componentes principales de ésta
itectura

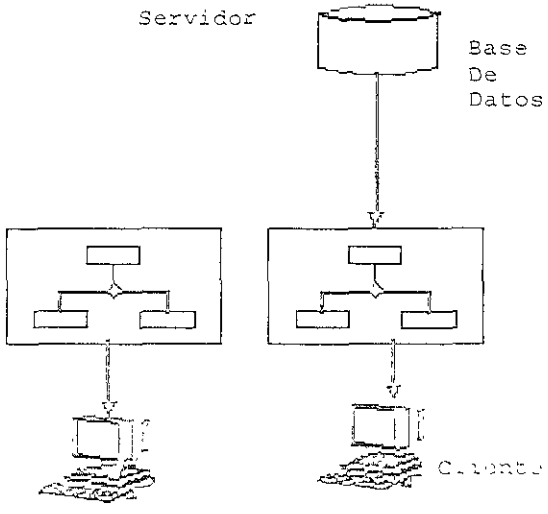


Fig. 5.8 Arquitectura Cliente/Servidor.

5.4.1. Plataformas

Para el Cliente se utiliza la herramienta de desarrol PowerBuilder versión 7.0.. Es una plataforma para el desarrollo sistemas orientados a objetos, funciona en el sistema operat Windows 95 o posterior y se implanta en Computadoras Personales.

Para el Servidor se utiliza la herramienta de desarrollo *ADVANTAGE* que soporta la codificación en sentencias SQL (*Structu Query Language*), la base de datos funciona sobre un servidor 4000 con sistema operativo SOLARIS 2.5.1.

Las herramientas antes mencionadas son las herramientas a ni institucional que se manejan en el Banco Central, por eso es que sistema debe desarrollarse en ellas y en la arquitect cliente/servidor descrita.

5.5. Base de Programación

En el desarrollo del sistema se utiliza la tecnología de Marcos Trabajo, tomando como base las PFC's (*PowerBuilder Foundat Classes*), las cuales consisten de un primer grupo de cua librerías que son el centro del producto(Nivel PFC), y un segu grupo que forma una capa de extensión(Nivel PFE) para el pri grupo. La capa de extensión es provista para aislar aplicaciones de futuros cambios en la capa central. Si se de mejorar las PFC, entonces la capa de extensión es donde se d realizar el cambio; la capa central no debe ser tocada. En Figura 5.9. se muestran la capa central y la capa de extensión.

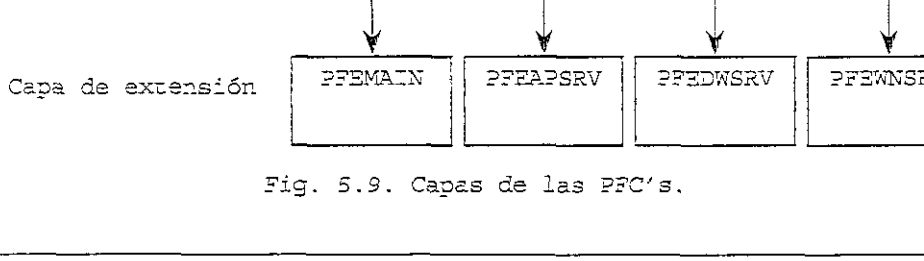


Fig. 5.9. Capas de las PFC's.

ilizar las PFC's está claro que muchas facilidades pueden ser s u omitidas de acuerdo a como sea necesario. Esto es un ficio y a la vez un peligro. Puede ser un signo de pereza mal ida, una oposición para explorar y aprender lo que las PFC's n que ofrecer, llevando esto a un esfuerzo excesivo de amación. Por otro lado, si una característica es muy ticada o muy compleja para ser usada en alguna situación, ser por lo general omitida sin problema. Esto ayuda también te el periodo de aprendizaje, porque existen demasiadas cosas producto como para ser atacadas todas a la vez.

avés de la PFC's se tiene una gama de funcionalidad ya icada y en la que el desarrollo del sistema no se enfoca en mentarla, sino por el contrario, solamente en usarla. De esta el desarrollo del sistema se hace de forma más inteligente, e no tiene que volverse a implementar la funcionalidad que es al.

a figura 5.10. se muestra a las PFC's como la base de amación sobre la cual será construido el sistema de nedores.

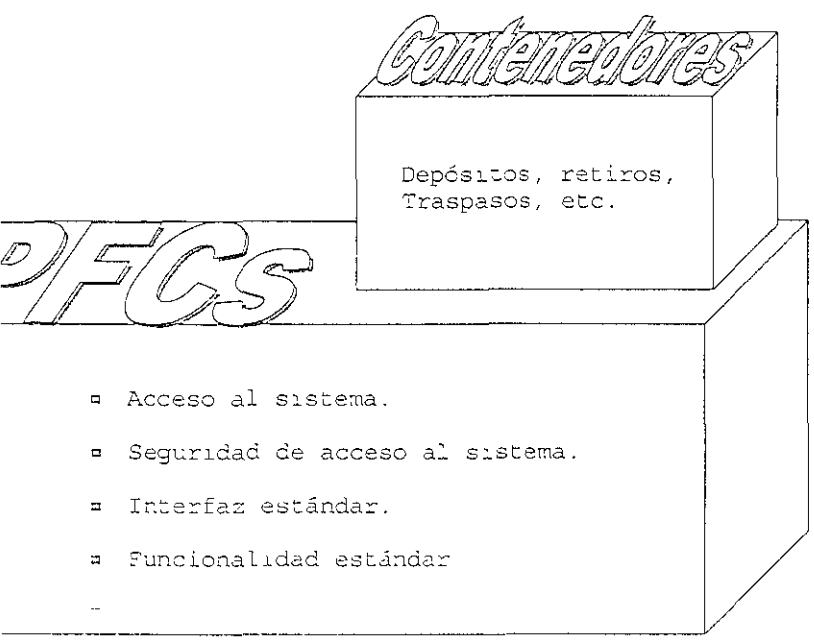


Fig. 5.10. Las PFC's como base de programación.

5.6. Base de datos

La necesidad de utilizar una base de datos surge por la capac limitada de memoria primaria o por la necesidad de gua información más tiempo de lo que dura la ejecución de un prog (persistencia), proporcionando así, maneras eficientes de acced los objetos. La persistencia significa que los objetos son copi de una rápida memoria volátil primaria a una "lenta" y const memoria secundaria. Esto se representa de una manera gráfica e figura 5.11.

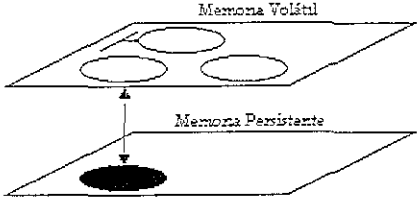


Fig. 5.11. Un objeto es copiado desde/hacia la memoria volát hacia/desde la memoria persistente.

Para el usuario de la base de datos no es imprescindible saber está almacenada la información, por lo que es necesario tene sistema manejador de Base de datos (DBMS: *DataBase Manage System*) que lo haga. Este DBMS debe estar integrado al sistem una manera transparente, a esto se le conoce como especializa de la BD.

Un DBMS tiene características principales tales como:

- Concurrencia: permite a varios usuarios trabajar con base de datos común simultáneamente.
- Recuperación: Si ocurre una falla en el hardwar *software*, el DBMS debe ser capaz de regresar la base datos a un estado consistente de los datos.
- Facilidad de Consulta: El DBMS debe tener una ma sencilla de accesar a la información en la base de da

A menudo es obvio que un producto DBMS debe ser utilizado en desarrollo y generalmente esto está implícito en la especifica de requerimientos del sistema. Si esto no existe, entonces necesidad se descubre al estudiar los requerimientos del mod

propiedades típicas que indican la necesidad de utilizar un son:

- La información debe ser persistente.
- Más de una aplicación comparte la información.
- Estructuras de información con un gran número de instancias.
- Búsquedas complejas en la estructura de la información.
- Generación avanzada de reportes a partir de la estructura de la información.
- Manejo de las transacciones del usuario.

embargo, estos requerimientos y modelos de análisis pueden ser diados para decidir que objetos son persistentes. Como los tos entidad generalmente tienen información que perdura a los s de uso, son los principales candidatos a ser almacenados. caso es cuando el número de instancias (y/o el tamaño de cada es muy grande comparado con el espacio de memoria primaria, lo cual se utilizan archivos del sistema o un DBMS.

rincipal trabajo de integración se hace en el proceso de rucción, sin embargo en esta etapa de diseño es donde debemos ir cómo debe ser incorporado el DBMS. Además debemos obtener rmación de cómo optimizar el almacenamiento del estado del o, por ejemplo, cómo debe ser indexada una estructura de tabla realizar búsquedas más rápidas.

manejadores de bases de datos se han desarrollado a través de rentes generaciones incluyendo la jerárquica, de red, cional y ahora las bases de datos orientadas a objetos. lmente el tipo predominante en las aplicaciones es el modelo cional y también debido a la infraestructura del banco es el de base de datos que utilizaremos en el desarrollo del nte trabajo.

na base de datos relacional, la información es guardada en s. Los tipos de datos a ser utilizados en las tablas son los más primitivos tales como caracteres, enteros y otros. Esto ca algunos problemas en nuestro diseño por la necesidad de enar objetos, ya que solo datos de tipos primitivos pueden guardados y no el comportamiento del objeto o su compleja ctura. Sin embargo muchos vendedores añaden estas capacidades manejadores de bases de datos relacionales (RDBMS: *Relational Base Management System*).

Cuando un lenguaje de programación orientado a objetos es conec
a un DBMS se tienen ciertos problemas. El primer problema es qu
nuestro sistema toda la información está guardada en los obje
por lo que necesitamos transformar nuestra estructura
información de objetos en una estructura orientada a tablas. Es
menudo se denomina problema de impedancia. El problema es que
programas tienen demasiados tipos de datos, incluyendo
definidos por el usuario, pero esto se tiene que reso
convirtiendo todos los tipos de datos a tipos primitivos.

Esto nos lleva a otro problema, que es que se crea un fuerte en
entre la aplicación y el manejador de base de datos. Para hacer
el diseño esté mínimamente afectado por el DBMS, debemos hace
posible para que sólo unas partes de nuestro sistema conozcan
relacionen con la interfaz del DBMS.

Un tercer problema es como expresar la herencia en la base
datos. Para resolver esto lo primero que se tiene que hacer
decidir que clases y que variables de las clases deben
almacenadas en la base de datos. Cada una de estas clases s
representadas por lo menos por una tabla en la base de datos.
lo tanto tenemos que una clase es mapeada en tablas de la sigui
manera:

Se asigna una tabla para la clase.

- Cada atributo primitivo se convertirá en una columna
  la tabla. Si el atributo es complejo (compuesto por o
  tipos de DBMS), se tiene que añadir una tabla adici
  para los atributos o dividir el atributo en va
  columnas en la tabla de la clase.

- La columna de la llave primaria debe ser el ú
  identificador de la instancia. El identifica
  preferiblemente debe ser invisible al usuario, porque
  cambios de las llaves por razones administrativas
  deben afectar al usuario; lo más recomendable es util
  llaves generadas por la máquina.

- Cada instancia de la clase se representa por un reng
  en la tabla.

- Cada asociación con una cardinalidad mayor a 1
  convertirá en una nueva tabla. Esta nueva tabla conec
  las tablas representando los objetos que van a
  asociados. Las llaves primarias de estas tablas pue
  ser utilizadas en esta tabla de adquisiciones. En alg

casos podemos tener la relación representada sólo por una columna (atributo) en la tabla de objetos.

ás de mapear las clases diseñadas es necesario reflejar en la de datos el concepto de herencia. Cuando se tiene una clase ereda a otra, se tiene que hacer lo siguiente:

- Los atributos heredados son copiados a todas las tablas que representan las clases descendientes. Ninguna tabla representará las clases abstractas.

- La clase abstracta está en una tabla de su propietario, a la cual las tablas de las clases descendientes se refieren.

oncepto importante de las bases de datos es la normalización, se refiere a la eliminación de redundancia y a evitar ciertas alías al hacer actualizaciones. Existen varias reglas de alización de las bases de datos, pero generalmente sólo se izan las tres primeras. Estas reglas son:

- En la Primera Forma Normal la tabla debe tener un solo valor para cada renglón, es decir, no puede contener grupos repetitivos.

- En la Segunda Forma Normal la tabla debe estar en la primera forma normal (1FN) y cada columna que no es llave debe ser dependiente de la llave primaria completa.

- La Tercera Forma Normal indica que la tabla debe estar en la segunda forma normal (2FN) y que una columna que no es llave primaria no debe depender de otra columna no llave.

quí se desprenden algunos conceptos que se tienen que recordar:

- Una llave primaria es una columna o grupo de columnas que identifican de una manera única a cada renglón de la tabla, no puede ser nula y cuando está compuesta de varias columnas se denomina llave primaria compuesta.

- Una llave foránea o secundaria, es una columna o combinación de columnas en una tabla que se refieren a una llave primaria en la misma o en otra tabla, y son utilizadas para hacer uniones entre tablas.

a mayoría de los casos es suficiente alcanzar la tercera forma al para el diseño de la base de datos. Actualmente, un diseño a base de datos basado en un modelo de objetos terminará con la ra forma normal. Esto es que para todo el tiempo cada uno de

los renglones consiste de un único identificador junto con número de valores de atributos mutuamente independientes. Por tanto, la tabla estará en la tercera forma normal si tenemos modelo de objetos donde cada objeto tiene un único identificad los atributos son mutuamente independientes (es decir que nin de la funcionalidad de los atributos depende de otra).

Mas intuitivamente, desde que los modelos de objetos son a me modelos de la realidad, tendemos a identificar objetos únic también a asignar los atributos a los objetos donde perten naturalmente. Por lo tanto, desde que la realidad está normali como tal, un buen modelo de objetos también debe ser normalizado

Antes de continuar consideramos importante mencionar alg conceptos básicos para entender los diagramas de entidad relació

Los componentes del modelo E-R son las entidades y las relacio Las entidades representan a los objetos de los cuales se nece conocer información y que tienen atributos, y se representan cuadrados o rectángulos con las esquinas redondeadas, dentro cada uno de los cuales se escribe el nombre de la entidad mayúsculas y en singular, y los atributos en minúsculas (ver 5.12.).

En esta misma figura se observa la representación de una entid la del otro componente del modelo que es la relación, la cual e evento bidireccional que representa la asociación entre entidades, o entre una entidad consigo misma. Cada relación t un nombre, una opción y un grado o cardinalidad que puede se uno a uno, de uno a muchos o de muchos a muchos.

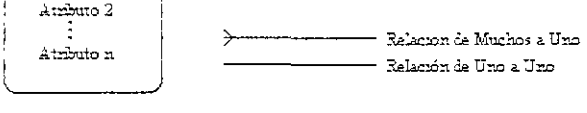
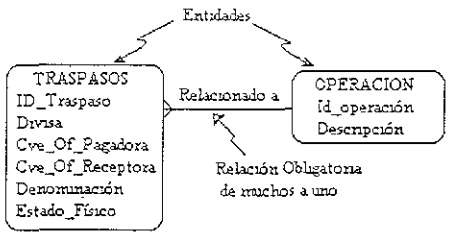


Fig. 5.12. Representación gráfica de entidades y relaciones.

Estos conceptos se ejemplifican en la siguiente figura(5.1 donde se puede ver dos entidades relacionadas con una cardinal de muchos a uno, lo que quiere decir que un Traspaso t relacionada una sola operación de la entidad del mismo nombre, un renglón de la tabla Operación puede estar relacionado a u

s registros de la tabla de Traspasos.



ig. 5.13. Ejemplo Sencillo de un Diagrama Entidad-Relación.

do en cuenta los conceptos mencionados hasta el momento en capítulo a continuación se presenta una parte del diseño del ma de contenedores.

Diseño del sistema de Contenedores

el diseño del sistema se toman en cuenta todos los conceptos definidos y se diseñan aquellos procesos que, de acuerdo al sis de los requerimientos establecidos, deben implementarse en stema.

iseño del sistema abarca los módulos de empaque de piezas, izaciones de emisión de piezas, entregas de piezas fabricadas, asos, remesas, depósitos y retiros. Cada uno de éstos módulos iseñado de acuerdo a los lineamientos de UML y del diseño de de datos relacionales, sin embargo, debido a que no es le, por razones de espacio, poner cada uno de los diagramas de o de cada proceso, a continuación se describen solamente, los spondientes al diseño de traspasos.

se mencionó en los capítulos de Procesos, Requerimientos y sis, el traspaso está compuesto por la salida de piezas de su de origen y la entrada de las mismas al lugar destino.

vés de un trabajo de equipo, en el que se tienen como base el o de requerimientos y el modelo de análisis, se obtuvieron los amas de clases, diagramas de secuencia y diagramas de entidad los traspasos, que se muestran a continuación

5.7.1. Diagrama de clases de Traspasos

El proceso de traspaso inicia cuando las piezas salen de la bó
origen y son puestas en tránsito mientras llegan a la bó
destino.

El usuario intercambia información con el sistema a través de m
y ventanas que están representadas por las cl
M_REGISTRO_TRASPASO para el menú y por W_REGISTRO_TRASPASO para
ventana, tanto para el registro de la salida y la recepción
traspaso. La figura 5.14. muestra las clases involucradas.

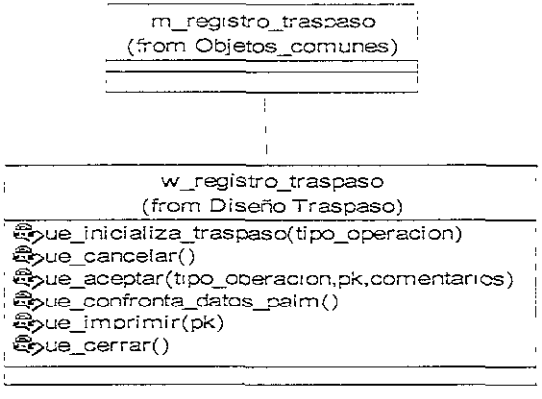


Fig. 5.14. Diagrama de clases de interfaz.

Existe una clase que se encarga de controlar la información que
desde la interfaz hacia la base de datos, esta clase es
denominada como N_CST_CONTROL_TRASPASO y puede realizar, en
otras funciones, el procesamiento del traspaso. Al procesar
información, la clase de control distribuye la información en
las clases de entidad correspondientes. De esta forma se crea
registro en la clase OPERACION. A su vez, la información especí
del traspaso y su detalle se almacena en las clases TRASPAS
D_TRASPASO. Posteriormente se registran en la clase DE_OPERACI
los envases involucrados en el traspaso, y son puestos en tráns
reflejando el cambio en la clase ENVASE_UBICACIÓN, las existenc
de la bóveda origen se disminuyen en la clase EXISTENCIA y
último se registra en la clase OPERACIÓN_EMPLEADO_AUTORIZA
claves del personal que autoriza la salida de las piezas. En
figura 5.15 se muestran las clases descritas anteriormente.

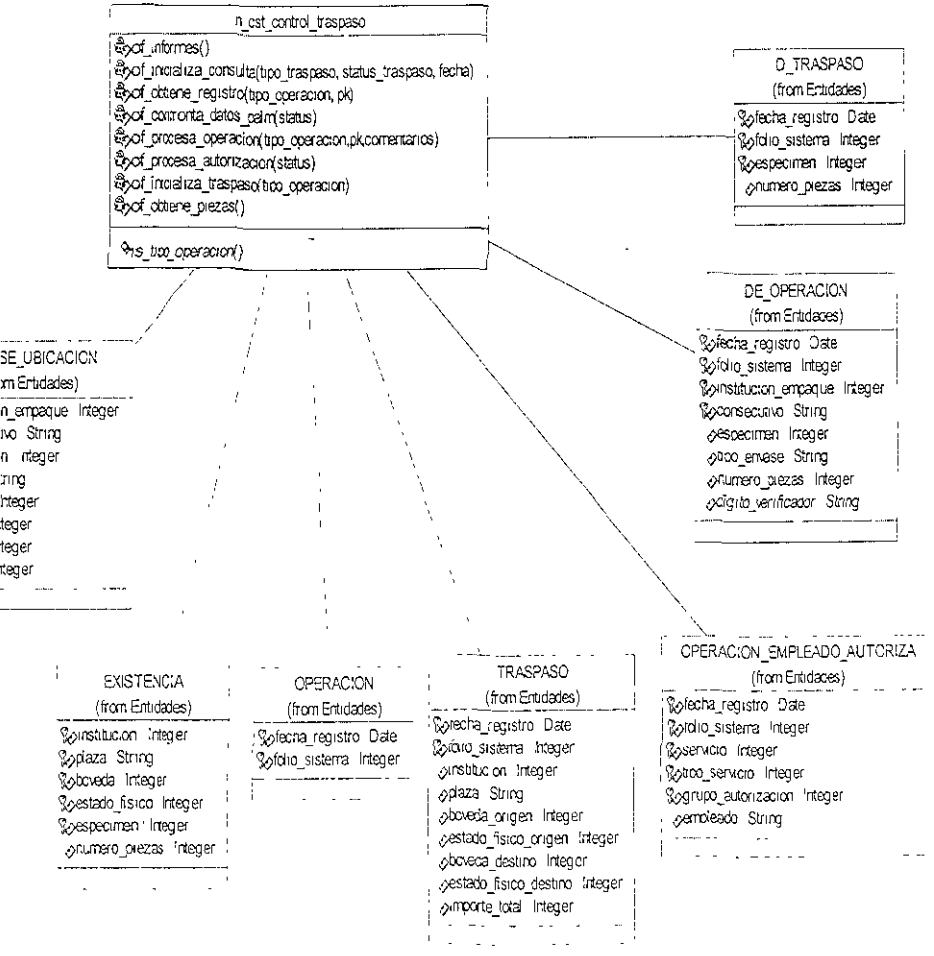


Figura 5.15. Clase de control y de entidad.

la lectura de los códigos de barras de los envases que tienen las piezas involucradas en el traspaso, se tiene la clase CTURA_PALM, que es la interfaz para el usuario con el lector de gos de barras. Una vez que se leen los códigos, son enviados a lase de entidad TABLA_CLIENTE, a través del objeto de control minado N_CST_LECTURA_PALM. La figura 5.16 muestra las clases rias anteriormente

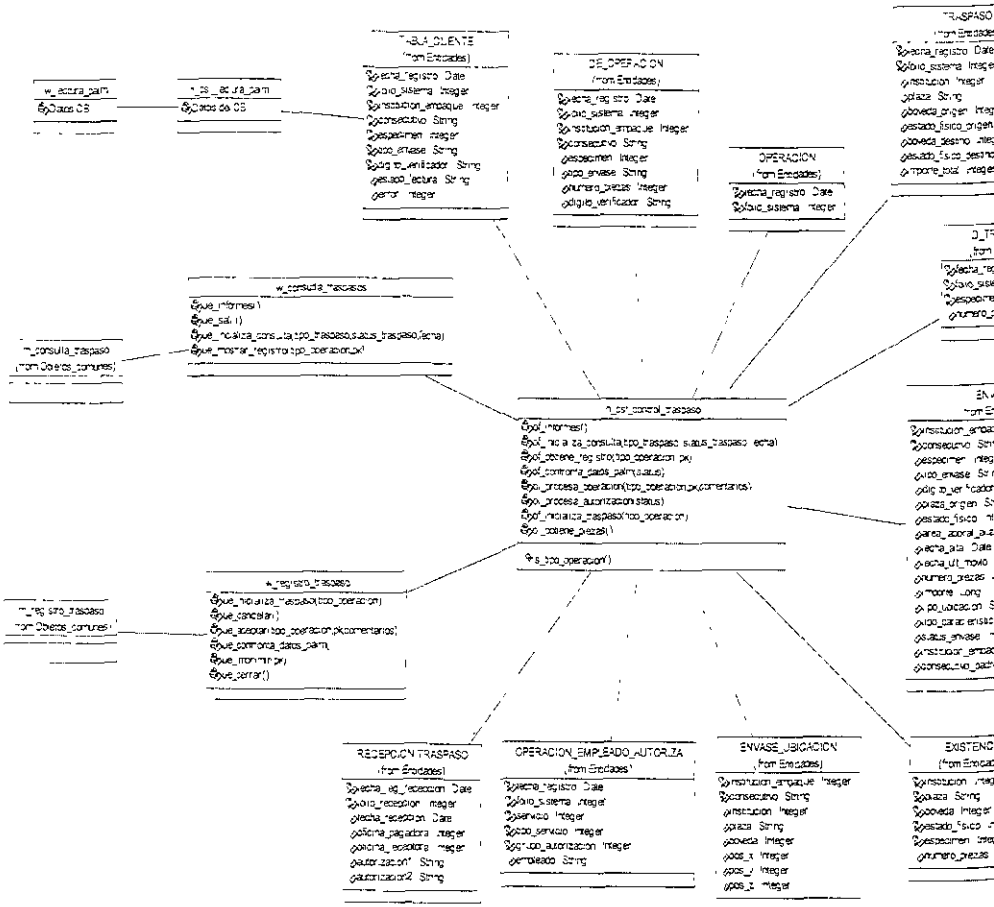


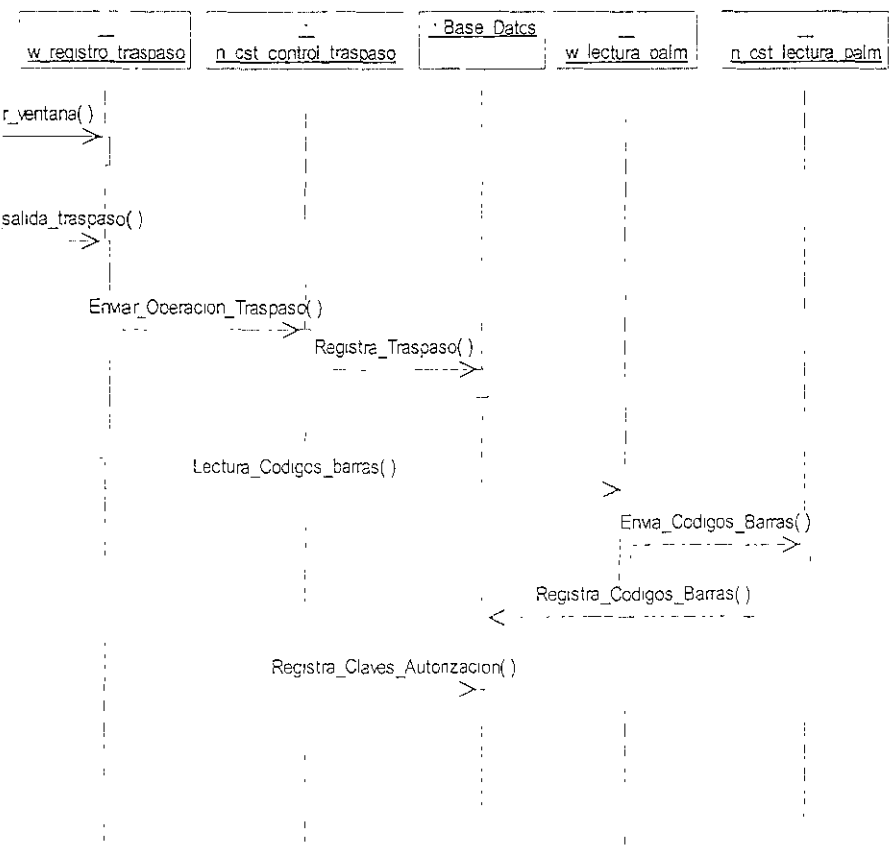
Figura 5.19. Diagrama de clases de traspasos.

5.7.2. Diagramas de secuencia de Traspasos

El intercambio de mensajes entre las clases durante la salida traspasos se muestra en la figura 5.20.

Para llevar a cabo el proceso de salida de envases el usuario a la ventana para registrar la salida. La ventana llama a un mét de la clase de control para enviar el traspaso y a su vez la cl de control guarda la información en cada una de las tablas entidad, que por razones de espacio se representan por la cl

)ATOS, pero que se especifican en los diagramas de clases .ores. Una vez que se almacena la información general, se leen ódigos de barras a través del método lectura_codigos_barras de ase W_LECTURA_PALM, la cual los envía a la clase de control _LECTURA_PALM y se almacenan en la base de datos.



g. 5.20. Diagrama de secuencia para la salida de traspasos.

. figura 5.21. se muestra la secuencia de acciones que se zan en el sistema para la recepción del traspaso. En primer el usuario abre la ventana donde consulta la lista de sos que está por recibir, de la cual elige uno. Al hacer la ón, el sistema consulta la base de datos la información y uestra al usuario. Después le corrobora que la información,

recibir, el usuario acepta la recepción. La clase de control encarga de almacenar la operación de recepción en la base de da que como se mencionó representa de forma general las clases entidad que se ven afectadas.

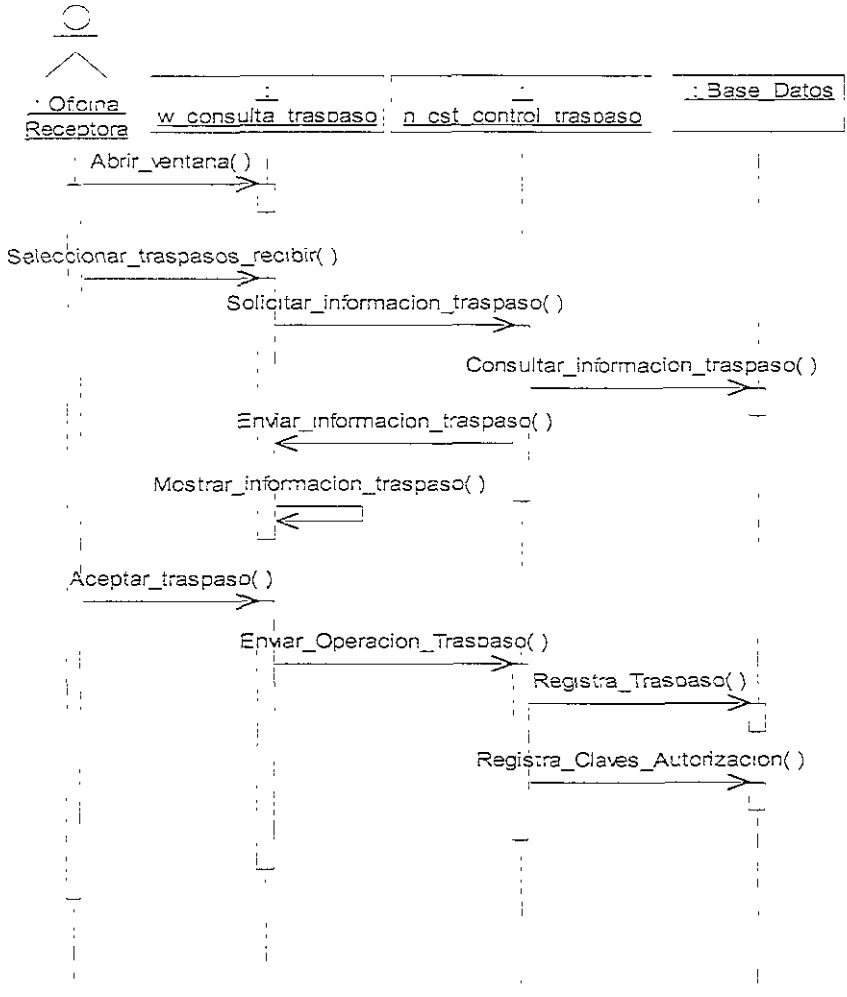


Figura 5.21. Diagrama de secuencia de la recepción de traspaso

Diagrama entidad relación

figura 5.22. se muestra un concepto general de cómo están
:turadas las operaciones. Se tiene la información general de
peración almacenada en la tabla OPERACIÓN y el detalle
spondiente a cada una de las operaciones permitidas en el
na se encuentra en las tablas correspondientes. De esta forma,
ıformación detallada del traspaso se encuentra en la tabla
ASO, y así para cada una de las tablas mostradas en la figura
Las tablas mostradas corresponden a las clases de entidad del
ama de clases, pero como se mencionó anteriormente, es a
s del diagrama entidad relación donde se representan a
le.

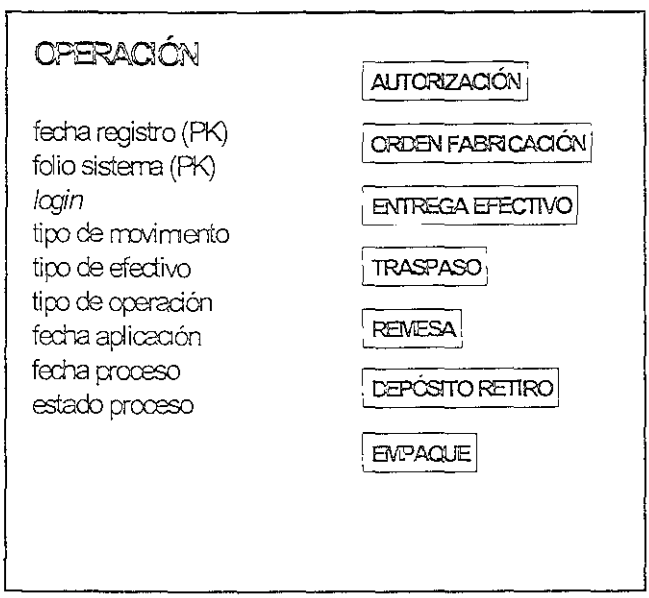


Figura 5.22. Diagrama conceptual de las operaciones.

o a que los diagramas entidad relación son varios y
ejos, los principales se muestran en el Anexo B.

En este capítulo se describió el proceso y los elementos
diseño del sistema. En el siguiente capítulo se tratará la fa
de implementación del sistema.

CAPÍTULO VI

IMPLEMENTACIÓN

fase de implementación se basa en los modelos realizados en etapas anteriores de análisis y diseño. Es en esta fase de se realiza la codificación de la aplicación ente/servidor y la interfaz con el usuario para construir el tema integral.

bjetivo principal de la implementación es integrar la tectura y el sistema. Esto implica la codificación de las s definidas durante el diseño en componentes. Cada uno de componentes contiene código fuente que debe ser revisado y do individualmente, asegurándo así que cumple con su objetivo, posteriormente integrarlo, compilarlo y enlazarlo con el resto s componentes para generar uno o más ejecutables que formarán licación final.

Normas de Codificación

ormas para un buen estilo de codificación se establecen con el e que, el código sea legible tanto para el que lo implementa para aquellas otras personas que tengan que actualizar el o. De estas normas se obtienen muchos consejos que sirven para nier tipo de lenguaje de programación en los que se rollan las distintas aplicaciones que se consideren en esta

proyecto de *software*, éstas incluyen identación apropi comentarios, prefijos, sintaxis en la definición de las funcio entre otras.

Una de las razones de estas normas, es porque permite a diseñadores de *software* inexpertos construir aplicaciones conf a una previa planificación, consistencia, o documentación establecida, con el fin de evitar código de sea difícil de le entender, e imposible de actualizar.

Una buena parte del mantenimiento del código es cómo formatearse y cómo debe comentarse. Todo lo que se codifique largo de un proyecto dado debe estructurarse de la misma manera.

A continuación se describen algunas normas para una b codificación:

6.1.1. Identación

Todo el código debe identarse apropiadamente. Ésta es la cosa fundamental que se puede hacer para mejorar legibilidad de programas. Aun cuando no se comente el código, pero si se ide será una gran ayuda a cualquiera que tiene que leer el có después de quien lo escribe propiamente, por ejemplo:

```
while ($x < $z) {
    if ($a == 1) {
        echo 'A fue igual a 1';
    } else {
        if ($b == 2) {
            //Haz algo
        } else {
            //Si no haz esto
        }
    }
}
```

En el código anterior se aprecia claramente el alcance de sentencia programada.

Es recomendable usar un cierto número de espacios o tabuladore el código, con el fin de familiarizarse con los cód desarrollados por todo el grupo que participe en el desarroll cada clase o función de un mismo proyecto.

. Estructuras de control

e utilizan expresiones condicionales (p.e. *if then else*) sin
parentesis, no sólo resultará un código menos entendible, sino
también pueden introducirse alteraciones cuando alguien
lice tal código.

ejemplo:

```
$a == 1) echo 'A es igual a 1';
```

tipo de codificación no es la más apropiada, quizá de forma
onal nos parezca bien, pero seguramente causará confusión para
personas que deban leer el código.

te otro ejemplo:

```
a == 1)
 echo 'A es igual a 1';
```

esto es por lo menos leíble, pero todavía no requiere de
n mantenimiento. ¿Qué pasaría si se requiere de una acción
onal cuando ocurra $a==1?, seguramente se necesitará agregar
tesis. Finalmente una forma adecuada para expresar la
ctura de control anterior es:

```
$a == 1) && ($b==2)) {
 echo 'a es igual a 1';
 //Es fácil de agregar más código
eif (($a == 1) && ($b==3)) {
 //Haz algo
```

observamos que el espacio después del *if* y *elseif* nos ayuda
nguir la condicional de las llamadas de la sentencia *if*.

. Funciones

funciones se delimitan claramente para identificar todo el
o que éstas contienen. Por ejemplo:

```
CION miFuncion($var1, $var2 = '')

 //Indentar todo el código aquí
 return $result;
```

observamos como no hay ningún espacio entre el nombre de la

función y el paréntesis, y que los parámetros se espacían muy bi
Todo el código dentro de la función será identado por lo menc
espacios.

Otro principio importante cuando codificamos funciones, es
siempre deben regresar el control directamente al lugar dc
fueron solicitadas.

6.1.4. Comentarios

Los comentarios son de suma importancia, ya que detallan el códi
principalmente resulta útil para establecer condiciones para
buen uso de la función, tal y como se muestra en el siguie
código:

```
/**
 * Breve decripción de la función
 *
 * Opcionalmente se añade una descripción más detallada.
 *
 * @param $paramName - type - proósito
 * @param ...
 * ...
 * @return tipo y descripción
 */
```

6.1.5. Prefijos / nomenclatura

El uso de prefijos es una buena forma de identificar los elemer
que desempeñan una labor particular, de hecho nos permite leer b
cierta norma, las acciones que se esperan de un proceso;
ejemplo, los procedimientos de uso externo se nombraran con
prefijo **spsc_**, seguido por algún nombre que dé idea de
finalidad, y de manera similar los de uso interno llevarán
prefijo **spsci_**. Los triggers deberán nombrarse empleando el pref
ti_, **tu_** o **td_**, según su finalidad (*insert*, *update*, *delete*).
también, los nombres de tablas, *scripts*, tipos de datos de
corresponder con la nomenclatura asignada a cada uno de ellos,
y como se ejemplificará durante este capítulo.

6.2. Lineamientos para la codificación del servidor

En la codificación del servidor básicamente tenemos sto

dures, triggers, tablas y *scripts*. A continuación se describen ineamientos para cada uno de ellos.

. Stored Procedures

stored procedures (SP'S) permiten almacenar funciones y dimientos compuestos por varias instrucciones, introducir s, ciclos, etc., y desde el punto de vista de ejecución, se n dos tipos: los que pueden ser ejecutados por el usuario (uso rno) y los que sólo pueden ser ejecutados por otros SP'S o *ers* (uso interno).

de uso externo pueden ser ejecutados por el usuario para ractuar con el sistema de forma directa o indirecta, es decir, suario podría ejecutar directamente el SP mediante el cliente *ybase*, o bien, puede ejecutarlo por medio de un cliente rollado en Power Builder. Con este tipo de SP'S se regresan resultantes de una consulta y se envía información de alguna ción al servidor.

de uso interno son aquellos que no deben ser ejecutados de a independiente y sólo pueden ser llamados por otros SP'S o ers, debido a que sus efectos son parciales sobre la base de (BD), y de ejecutarse alguno de ellos sin que se ejecute otra de eventos relacionados, puede ocasionar problemas de sistencia en la BD. Con este tipo de SP'S se realizan por lo: validaciones, afectaciones sobre algún dato en particular, otienen datos para ser empleados o transformados por otro SP de ser entregados al usuario, etc.

distinguir claramente entre estos dos tipos de SP'S, se leció la norma de que los de uso externo deberán nombrarse con refijo **spsc_**, seguido por algún nombre que dé idea de su idad, y de manera similar los de uso interno deberán nombrarse ando el prefijo **spsci_**. Ambos prefijos deberán escribirse re con minúsculas.

s, los archivos de texto con el código de creación de los deberán incluir todo el código necesario para su creación, es , el código para borrar el SP en caso de que exista amente, el código de creación de tablas temporales que deben ir para poder crear el SP y el código para asignar derechos de ción sobre ellos a los diferentes grupos *sybase*, en caso de rse de SP'S de uso externo ya que sobre SP'S de uso interno no asignarse derechos de ejecución

```
/** Selecciona la BD que utiliza el SP **/
use bd_contenedores
go

/** Crea las tables temporales que utiliza el SP **/
create table #elem_caract (elemento varchar(13), reg numeric(3,0) identity)
go
create table #elem_folio (elemento1 varchar(13), reg numeric(3,0) identity)
go
create table #d_caract (tiro char(6), serie varchar(3), prefijo char(1))
go
create table #d_folio (tiro char(6), folio_inicio int, folio_fin int, numero_piezas int)
go

/** Si existe borra el SP de uso interno que inserta producción **/
drop proc spsici_inserta_produccion
go

/** Crea el SP de uso interno que inserta producción **/
create proc spsici_inserta_produccion
    @terminal       int,        /* Parametros de OPERACION */
    @fol_term_orig  int,
    @folio_term     int,
      .
            .
            .
return 0                        /* Regresa cero si el SP se ejecutó correctamente */
go

/** Se borran las tablas temporales **/
drop table #elem_caract
go
drop table #d_caract
go
drop table #elem_folio
go
drop table #d_folio
go
```

En el caso de un *stored procedure* de uso externo, debe incluirse final del archivo de texto realizado para su creación, el có que asigna los derechos de ejecución a los grupos creados en la de *sybase* (manejador de la base de datos). Por ejemplo, como en sistema de contenedores se definieron tres grupos de usuari administrador, monitor_contenedores y operador_contenedores, texto que debe incluirse al final del archivo para la creación un SP de uso externo que envía el registro de inicio de producci es el siguiente:

```
/** Asigna derechos a los grupos de usuarios **/
grant execute on spsc_envia_REG_INIC_PROD to administrador
```

```
t execute on spsc_envia_REG_INIC_PROD to monitor_contenedores
t execute on spsc_envia_REG_INIC_PROD to operador_contenedores
```

mencionar que el administrador, como su nombre lo dice, se
ga de administrar el sistema. El grupo monitor_contenedores se
ga de las operaciones comunes del sistema y el grupo de
dor_contenedores es el que utiliza el sistema y registra las
entes operaciones.

estandarizar la forma de escribir código fácilmente
dible, se llevaron a cabo las siguientes recomendaciones:

o se utilizan instrucciones agrupadas por *begin* y *end*, se debe
uso de sangrías con tres espacios (no tabuladores), alineando
las instrucciones (incluso el *begin* y el *end*) de manera que
claro, de que instrucción o condición dependen. Además se
ienda comentar los *end's* para identificar con facilidad a que
corresponden. Esto se ejemplifica a continuación con una
de código:

```
e (@reasig_f_proc is not NULL)
gin
ect  @reasig_f_proc = MIN(fecha_proceso)
om REGISTRO_REASIGNACION
ere fecha_registro   = @fecha_vmd
and estado_proceso = @reasig_proc
and fecha_proceso  > @reasig_f_proc

(@reasig_f_proc is not NULL)
egin
elect @inst_reasig = institucion_empaque,
      @cons_reasig = consecutivo_anterior,
      @cons_nvo    = consecutivo_nuevo
 from REGISTRO_REASIGNACION
where fecha_registro   = @fecha_vmd
  and estado_proceso = @reasig_proc
  and fecha_proceso  = @reasig_f_proc

update #mazo
   set consecutivo = @cons_nvo
where institucion_empaque = @inst_reasig
  and consecutivo         = @cons_reasig

end  /* if */

d    /* while */
```

alinear las palabras reservadas a la derecha y los cam condiciones, etc. a la izquierda, así como los signos de relac ya sea en las asignaciones o en las comparaciones, como se p observar en el ejemplo anterior.

6.2.2. *Triggers*

Los *Triggers* se comportan en gran manera como los procedimie almacenados, con la diferencia de que se ponen en marcha cuand ejecuta una inserción, borrado o actualización en una tabla.

Los *triggers* deberán nombrarse empleando el prefijo **ti_**, **tu_** o según su finalidad (*insert*, *update*, *delete*), y el nombre de tabla sobre la cual son creados. Por ejemplo, el *trigger* de in para la tabla TIRO deberá nombrarse ti_TIRO.

Si el nombre de la tabla es muy largo y al unirlo al prefijo re los 30 caracteres que *sybase* pone como límite para nombrar obj en la BD, el nombre de la tabla se abrevia de manera que siendo fácil de identificar. Por ejemplo: el *trigger* para inse en la tabla REGISTRO_EMPAQUE_MZO_PAQ_NVO podría nombra ti_REGISTRO_EMPAQUE_MZO_PAQ_NV.

Cuando un mismo *trigger* funciona para más de un evento, se aj el prefijo para indicar su uso. Por ejemplo, si un mismo tri sirve para insertar y actualizar, se usará el prefijo **tiu_**.

Todos los *triggers* deberán dar un *rollback* (acción de deshacer afectaciones hechas a la base de datos) en caso de error a transacción, este *rollback* se nombrará con el prefijo **tr** y nombre de la tabla a la que pertenece el *trigger*. Por ejemplo:

```
if (@status !=0)
  begin
  rollback trREG_INICIO_PRODUC
  raiserror @status "Error al buscar valor NO INICIADO"
  return
  end
```

Al incluir una instrucción *insert*, *update* o *delete* en cualquier o *trigger*, debe incluirse con anterioridad un *save point* (punto salvamento), para poner en firme las afectaciones que se han h hasta ese momento. El *save point* debe ser nombrado con el pre **tr** y el nombre de la tabla, por ejemplo:

```
' tran trREG_INICIO_PRODUC

rt into REG_INICIO_PRODUC values (@fecha_registro, @id_produccion, NULL, @terminal,
                                  @login, @area_laboral, @supervisor, @fecha_emision,
                                  @especimen, @tiro, @millon, NULL, @estado_proceso)
ct @status = @@error
status != 0)
gin
llback trREG_INICIO_PRODUC
turn @status
d
```

sta parte de código se observa como se define un *save point* y aso de que ocurra un error en la inserción de un registro, se el *rollback* a la transacción que se haya abierto, con el re que haya sido empleado (en este caso, trREG_INICIO_PRODUC).

Tablas

lineamientos referentes a las tablas son: que los nombres de tablas se definen siempre con mayúsculas y los de los campos o utos con minúsculas. Además por claridad, se deben utilizar oras completas para formar los nombres, y únicamente se izan abreviaciones cuando el nombre es demasiado largo.

Scripts

Scripts, son archivos de texto que contienen un conjunto de ucciones que realizan una tarea específica y son útiles cuando ecesario repetir los pasos que se siguieron para hacer un o, para incoporar algo nuevo o para realizar una tarea ífica en el servidor, tal como preparar o limpiar la BD para as.

scripts generalmente incluyen instrucciones para crear tablas, ers, SP's; instrucciones para insertar datos a alguna tabla, tar nuevos errores, etc. Por ejemplo, el siguiente *script* ene las instrucciones para insertar mensajes de error, tar registros, crear tablas, *triggers* y un SP, cuyo código no ecesario incluir ya que se encuentra en otro archivo de texto.

```
rt into ERROR values (40381,'No fue posible obtener hora de apertura del sistema')
rt into ERROR values (40382,'No fue posible obtener hora de cierre del sistema')

t into REGLA values ('EDO_PROC_SINCRONIZAC','ANULADO','A','char(1)')
```

insert into REGLA values ('EDO_PROC_SINCRONIZAC','CANCELADO','A','char(1)')

create proc spsif_sincroniza_mazos

create table RSP_REG_INIC_PROD_ESTADO
create table RSP_LOTE_ESTADO
create table RSP_REG_SINCRO_ESTADO

create trigger tu_REG_INICIO_PRODUC on REG_INICIO_PRODUC
create trigger tu_REG_INICIO_PRODUC_LOTE on REG_INICIO_PRODUC_LOTE
create trigger tu_REG_INICIO_PRODUC_LOTE_SEC on REG_INICIO_PRODUC_LOTE_SEC

6.3. Lineamientos para la codificación del cliente

Los estándares establecidos para el desarrollo de la aplicación cliente del Sistema de Contenedores en PowerBuilder abarcan nomenclatura de componentes tales, como: objetos, funcion eventos, sentencias de control y variables, además de la estruct y organización de los mismos.

La utilización de estos estándares nos permite desarrollar aplicación consistente, además de facilitar el trabajo en equip la legibilidad del código escrito por otras personas, para que mantenimiento se lleve a cabo con mayor facilidad.

Comenzaremos explicando la nomenclatura de los componentes.

El nombre de un componente se divide en tres partes: el prefijo debe estar formado por los primeros caracteres del nombre del t de componente, el separador y el resto que debe describir función en la aplicación, evitando las abreviaturas. El nom puede tener hasta 40 caracteres, debe estar en minúsculas, y tiene más de una palabra, se debe utilizar el guión bajo c separador. La lista de prefijos utilizados se encuentra en la ta 6.1. que se presenta a continuación.

PRIVATE Tipo de Control	Prefijo
CheckBox	Cbx
CommandButton	Cb
DataWindow	dw
DropDownListBox	Ddlb
DropDownPictureListBox	ddplb
EditMask	em

Tabla 6.1. Prefijos para controles. (Continúa)

Graph	gr_
GroupBox	gb_
HScrollBar	hsb_
Line	ln_
ListBox	lb_
ListView	lv_
MultiLineEdit	mle_
OLE	ole_
Oval	oval_
Picture	p_
PictureButton	pb_
PictureListBox	plb_
RadioButton	rb_
Rectangle	r_
RichTextEdit	rte_
RoundRectangle	rr_
SingleLineEdit	sle_
StaticText	st_
Tab	tab_
TreeView	tv_
UserObject	uo_
VScrollBar	vsb_

Tabla 6.1. Prefijos para controles.

mbre de las variables debe consistir de un prefijo que indique cance, el tipo de dato, y el nombre que describa su función, ados éstos últimos por un guión bajo (_). La forma general nombrar a una variable es: {alcance}{tipo de dato}_{nombre}.

tinuación se presentan las tablas de los prefijos de acuerdo cance de las variables y los tipos de datos que utilizamos as 6.2. y 6.3.).

Alcance de la variable	Prefijo
Global	g
Compartida (*shared*)	s
De instancia	i
Local	l
Parámetro (*argument*)	a

Tabla 6.2. Prefijos para el alcance de las variables.

{PRIVATE}Categoría	Tipo de Dato	Prefijo
Tipos de dato estándar	Any	A
	Blob	Bb
	Boolean	B
	Character	Ch
	Date	D
	DateTime	Dt
	Decimal	C
	Double	Db
	Integer	I
	Long	L
	Real	R
	String	S
	Time	T
	UnsignedInteger	Ui
	UnsignedLong	Ul
Tipos de dato para objetos del sistema	DataWindow	Dw
	DataWindowChild	Dwc
	MailSession	Ms
	Menu	M
	Structure	Str
	Transaction	Tr
	User object	Nv
	Window	W

Tabla 6.3. Prefijos para tipos de datos.

Por ejemplo, basándonos en estos lineamientos si necesitamos variable local de tipo entero cuya función es obtener respuesta, su nombre debe ser *li_respuesta* y si tenemos que def un parámetro de tipo *long* que contiene el folio de un documento variable se nombraría *al_folio_documento*.

En cuanto a la nomenclatura para las funciones, definimos que nombre debe consistir de un prefijo que indique el alcance de función, un guión bajo (_), y un nombre que describa el proceso realiza. Si la función es para un evento, el nombre debe consis del prefijo ue (*user event*), un guión bajo (_), y un nombre describa el proceso que se realiza con el código del *script* ese evento.

Los prefijos a utilizar son: f para funciones globales y of funciones declaradas en un componente. Por ejemplo, el nom *of_obtiene_datos_servidor* nos indica que es una función de componente que obtiene ciertos datos del servidor.

s de las convenciones para los nombres de variables, funciones ros, se definieron también algunas recomendaciones para bir el código, como por ejemplo:

- Dejar un espacio antes y después del símbolo de asignación (=), de los operadores (+, -, *, >, etc.), y entre parámetros de una función.
- Utilizar una línea por instrucción.
- Usar sangrías para mejorar la legibilidad de las instrucciones.
- Cuando se hace referencia a un objeto o control, deben utilizarse, si es posible, los pronombres de tales como *Parent*, *This* y *ParentWindow*, en lugar de utilizar explícitamente el nombre del objeto o control.
- Se recomienda utilizar mayúsculas para instrucciones de *PowerScript* y SQL.

anto a los objetos DataWindow:

- Se debe procurar que las consultas a la base de datos sean mediante *stored procedures*.
- El código de error del *stored procedure* en un objeto *datawindow* debe capturarse en la variable *sqldbcode* del evento *dberror*. Para ello debe utilizarse en el *stored procedure* la instrucción *raiserror*, con el código de error, antes de terminar su ejecución con un *return*.

s se deben documentar todos los *scripts* de funciones o os. Cada uno debe comenzar con un encabezado en forma de tario que indique su propósito, así como características y sitos relevantes tales como parámetros y valor de retorno. Se omitir acentos en este encabezado. Un ejemplo de la ctura del encabezado y los datos que debe incluir se muestra a nuación:

```
////////////////////////////////////////////////////////////////////////////////
ion:    of_nombreEspecimen
jeto:   nv_especimen
ceso.   public
gumentos: ref integer ai_especimen - clave del especimen del que se quiere obtener la descripción
gresa:  String  - nombre del especimen
                "" (cadena vacia) si no existe la clave de especimen
scripcion: Obtiene el nombre del especimen de la base de datos.
////////////////////////////////////////////////////////////////////////////////
```

También se deben agregar comentarios más específicos a lo largo
script para explicar algún proceso. Estos comentarios pueden
por bloque de instrucciones o por una sola instrucción y de
colocarse en la línea inmediata anterior a la(s) instrucción(es
la(s) que hace referencia. Por ejemplo:

```
////////////////////////////////////////////////////////////////////////////
// Obtiene y valida la autorizacion para la compensacion de mazos
////////////////////////////////////////////////////////////////////////////
ln_constante = CREATE n_constante
ll_error = ln_constante.of_getConstante('SERV_COMPENSA_MAZO', ls_cte)
DESTROY n_constante

lb_resultado = lnv_envase.of_validaAutorizacion( gnv_constantes.of_getServExeComp(), &
                                                 ls_claveAutorizacion[], &ls_passwordAutorizacion[], s
                                                 gnv_constantes.of_getTipoServExeComp() )
destroy luo_envase
...
```

o si sólo hace referencia a una instrucción:

```
// Si la autorizacion es valida...
IF lb_resultado THEN
    SetPointer(HourGlass!)
    ...
```

Otro tipo de comentario es el utilizado para identificar bloques
de instrucciones correspondientes a una sentencia de cont
específica, que se agrega después de la instrucción que indica
fin de bloque en una sentencia de control (en la misma línea).
ejemplo:

```
IF lb_resultado THEN
    SetPointer(HourGlass!)
    ...
    IF ai_numeroOrden <> li_numeroOrden THEN
        MessageBox ("Error","El número de orden no correponde a ...")
        Return
    END IF  // IF ai_numeroOrden ...
END IF      // IF lb_resultado
```

En general, estos son los lineamientos que se aplicaron en
codificación de la aplicación para el cliente.

Estándares para la interfaz con el usuario

terfaz tiene como propósito permitir que los usuarios lleven a acciones o actividades de procesamiento de manera eficiente, ndoles posibles confusiones que los llevan a cometer errores ocesamiento, o la interrupción de operaciones esperadas por el ma, por lo que en esta sección se describen los puntos que os en cuenta para realizar las pantallas correspondientes a la faz con el usuario. Algunos de estos puntos son:

- Los tipos de letra, *command buttons*, *radio buttons*, y *check boxes* deben ser consistentes de una ventana a otra dentro de la aplicación.
- La navegación dentro de la ventana debe ser de izquierda a derecha y de arriba hacia abajo.
- El uso de teclas de función debe ser definido de manera consistente para toda la aplicación.
- Se deben utilizar colores asignados por omisión siempre que sea posible.
- Debe evitarse el uso de ventanas de tipo modal (de respuesta), a menos que la aplicación requiera de una respuesta inmediata por parte del usuario.
- Se debe incluir la posibilidad de que el usuario cancele una acción.
- Se debe dar al usuario diversas formas de realizar una acción, por ejemplo: el cierre de una ventana o de un archivo.
- Los componentes de la pantalla que estén relacionados deberán aparecer uno junto al otro, quizá con un marco que los borde.
- Se recomienda reducir los colores en la pantalla debido a que demasiados colores distraen al usuario de la tarea por realizar y tampoco se deben utilizar fuentes cursivas y ornamentales
- Los componentes (como los botones y cuadros de lista) deben ser del mismo tamaño siempre que sea posible.

de todas estas recomendaciones, también se consideraron los

estándares básicos establecidos por Microsoft Windows, tales
la localización de la barra de menús, de cuadros de minimizac
restablecimiento y maximización de las ventanas y la utilizació
barras de desplazamiento vertical y horizontal además de
colores y tipos de letras.

Basándonos en lo explicado anteriormente, realizamos las inte
gráfica para el sistema, y de la cuál, a continuación se mues
algunas pantallas que corresponden al módulo de Traspasos
sistema de contenedores.

La pantalla de la fig. 6.1. corresponde a la interfaz general
sistema de contenedores que contiene un menú principal con
submenús correspondientes a las operaciones principales que
Remesas, Retiros y Traspasos. Cada uno de estos submenús cont
a otros. Por ejemplo, el menú de Traspasos contiene las opcione
Solicitud, Salida y Recepción de un traspaso (como puede observ
en la figura).

El formato de todas las pantallas es el mismo: la barra de tí
es azul con letras blancas, el fondo de la pantalla es gris y
general es igual al estándar de Windows.

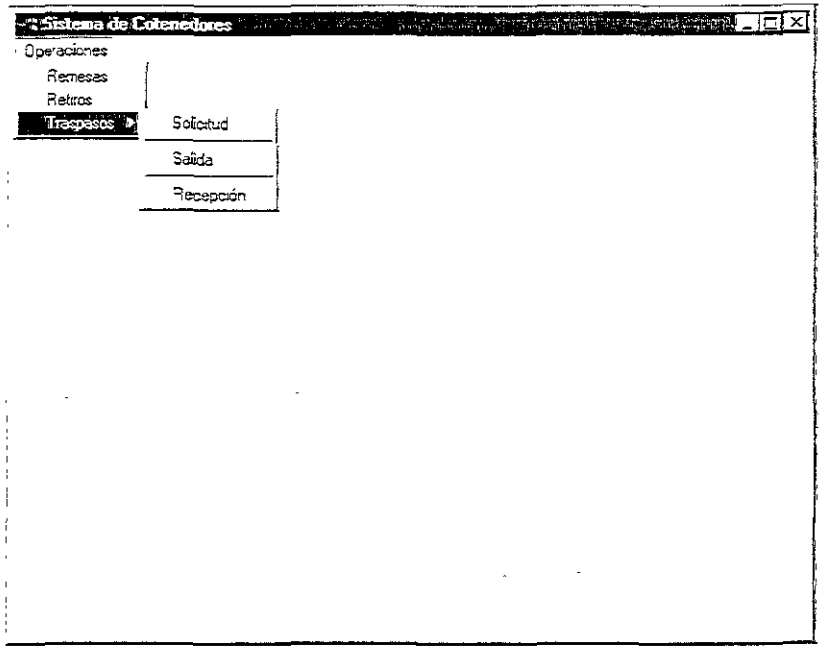
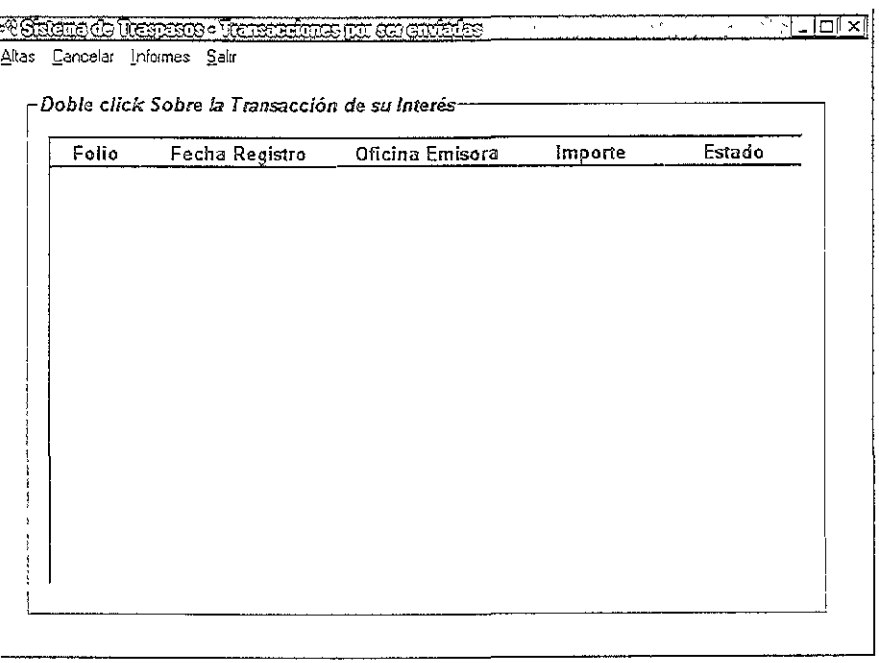


Fig. 6.1. Interfaz general del sistema de contenedores.

ntinuación, en la figura 6.2., se muestra la pantalla del lo de Traspasos, correspondiente a los traspasos que están ientes de ser enviados. En la pantalla se observa que en la a de menús están las opciones de Alta, Cancelación y Generación nformes de Traspasos, además, por supuesto, de la opción de da de este módulo.

puede observarse, al centro de la pantalla se despliegan datos el folio, la fecha de registro, la oficina emisora, el importe estado de cada uno de los Traspasos que estan pendientes por ar.



6.2. Pantalla del proceso de Traspasos pendientes por enviar.

vez que se ha seleccionado una de las transacciones a ser adas en el proceso de Traspasos, se muestra una pantalla como e la figura 6.3., que corresponde a la consulta de un Traspaso cífico, en donde pueden observarse sus datos más relevantes En rimer *frame* (marco) se observa información correspondiente al paso, como: la clave de la operación, fecha, hora, número de o y bóveda (clave y descripción).

l siguiente frame, cuya información más detallada, como: la

descripción del tipo de divisa, de la oficina receptora, del est
físico de origen y del estado físico destino. A la derecha de
información se muestra de qué tipo es el efectivo y cuál es
importe total.

En la parte inferior de la pantalla se tienen dos pestañas con
títulos "Por Denominación" y "Por Contenedores". En la pestaña
Denominación" se muestra información desglosada del traspaso
datos, tales como: la denominación, clave, importe y número
piezas de efectivo que conforman el traspaso, además de mostra
importe total de la denominación seleccionada.

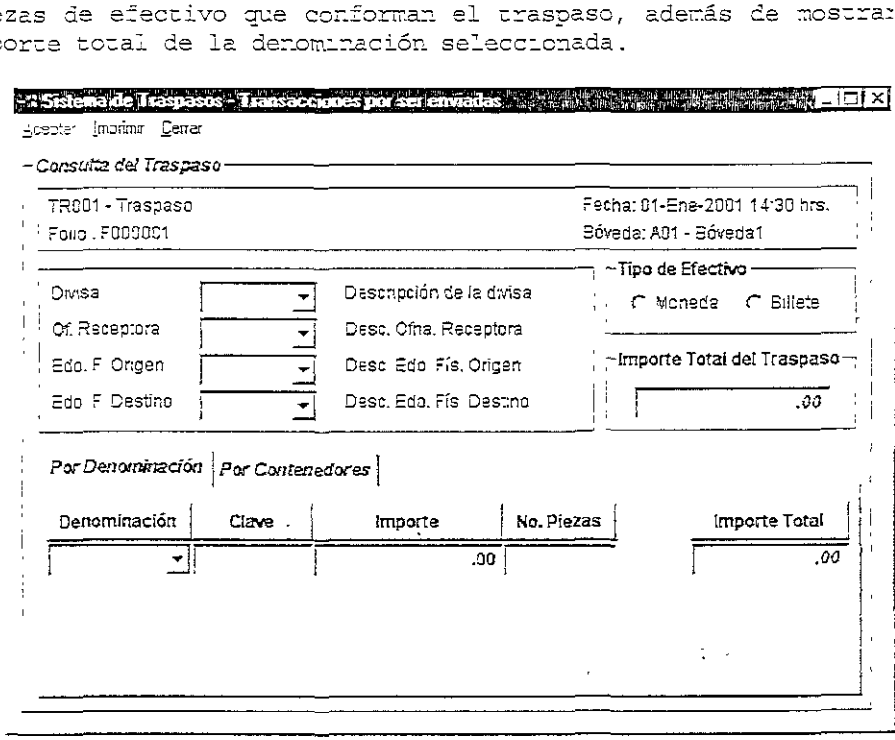


Fig. 6.3. Pantalla de consulta de un Traspaso.

En la siguiente figura se muestra la pantalla de Alta de
Traspaso, que básicamente es igual a la de consulta descr
anteriormente, pero en esta ocasión se muestran los datos que
requeridos cuando se ingresa la información desglosada
Contenedores" en lugar de hacerlo por denominación. La informac
es la misma, unicamente con la diferencia de que en esta pestaña
tiene un botón que al darle un "*click*" (o *enter*) se obtienen
datos leídos por la Palm.

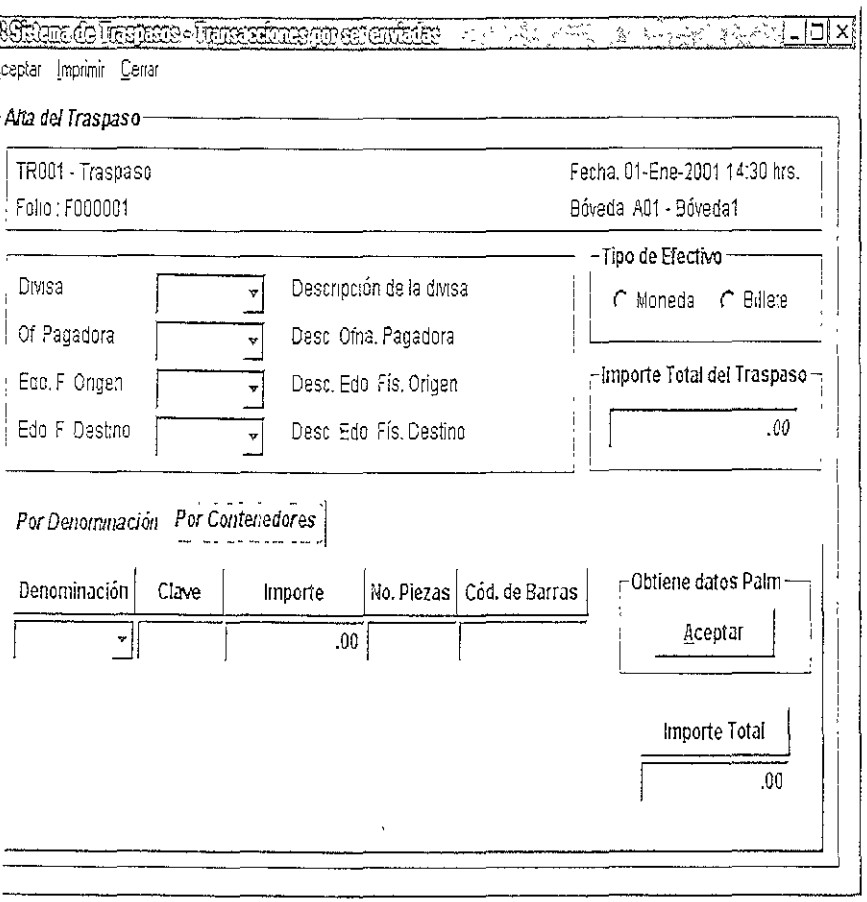


Fig. 6.4. Pantalla de Altas de Traspasos.

gura 6.5. presenta una pantalla similar a las anteriores, e corresponde a la Cancelación de un Traspaso, con la encia que en la parte inferior tiene una pestaña más con el "Motivo de la Cancelación", en la que se tiene la opción de ibir las razones por las cuales se cancela transacción.

[illegible] pantallas se tienen además, los menús de Aceptar [illegible], Imprimir y Cerrar [illegible]

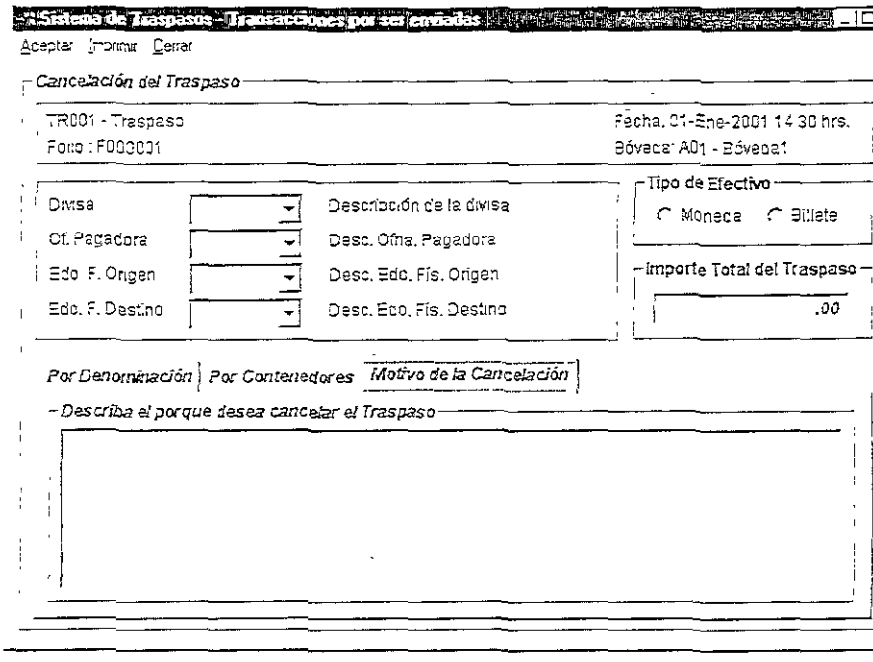


Fig. 6.5. Pantalla de cancelación de un traspaso.

Otra de las pantallas importantes del sistema de Contenedores pantalla de autorización. Esta pantalla se presenta cada vez q necesario autorizar alguna de las transacciones llevadas a ca el proceso de Traspasos, por parte de los responsables. primer *frame* se tiene que ingresar la clave y *password* del cus autorizado de la oficina pagadora y en el otro se ingresa la y *password* del interventor.

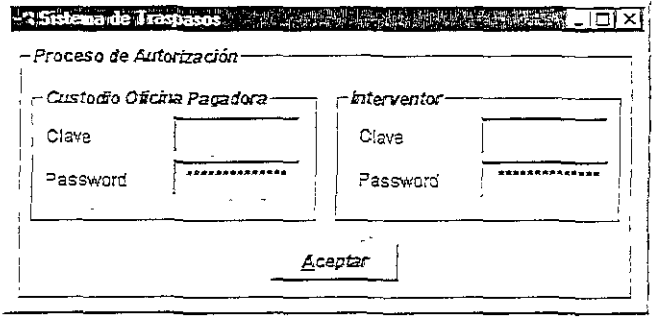
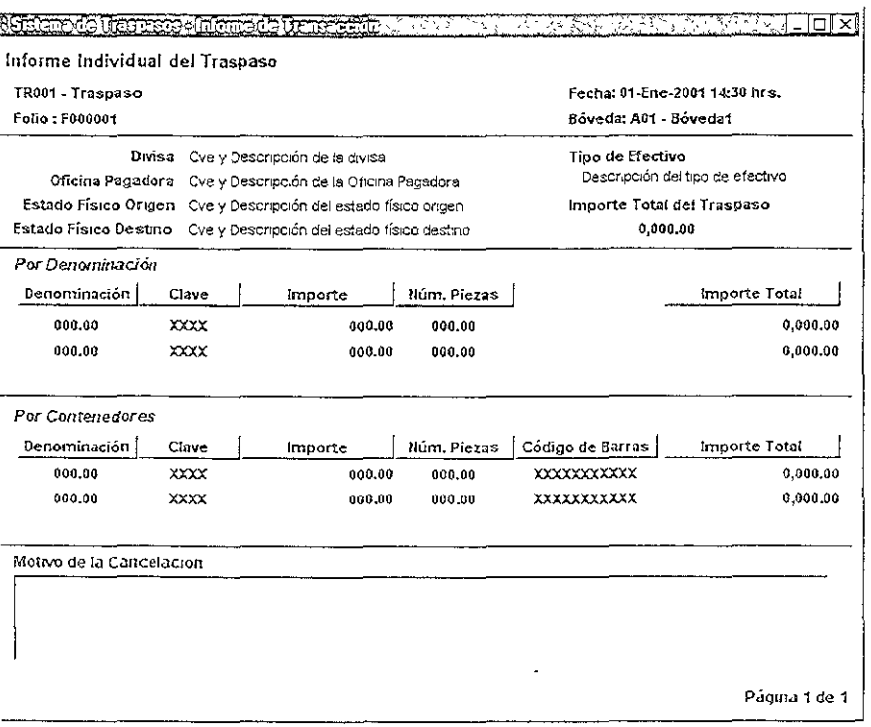


Fig. 6.6. Pantalla de Autorización en el sistema de Traspas

pantalla correspondiente a la opción de Informe de Traspasos, uestra toda la información que fue ingresada en el alta y/o lación correspondiente del traspaso seleccionado.


Sistema de Traspasos - Informe de Transacción

Informe Individual del Traspaso

TR001 - Traspaso — Fecha: 01-Ene-2001 14:30 hrs.

Folio : F000001 — Bóveda: A01 - Bóveda1

Divisa: Cve y Descripción de la divisa
Oficina Pagadora: Cve y Descripción de la Oficina Pagadora
Estado Físico Origen: Cve y Descripción del estado físico origen
Estado Físico Destino: Cve y Descripción del estado físico destino

Tipo de Efectivo: Descripción del tipo de efectivo
Importe Total del Traspaso: 0,000.00

Por Denominación

Denominación	Clave	Importe	Núm. Piezas	Importe Total
000.00	XXXX	000.00	000.00	0,000.00
000.00	XXXX	000.00	000.00	0,000.00

Por Contenedores

Denominación	Clave	Importe	Núm. Piezas	Código de Barras	Importe Total
000.00	XXXX	000.00	000.00	XXXXXXXXXX	0,000.00
000.00	XXXX	000.00	000.00	XXXXXXXXXX	0,000.00

Motivo de la Cancelacion

Página 1 de 1

g. 6.7. Pantalla correspondiente al Informe de un Traspaso.

Codificación del Sistema de Contenedores

ez que se han definido los estándares de la codificación, a nuación se muestran pequeños fragmentos del código utilizado codificación del servidor, del cliente y de la *palm*.

apéndice C se presenta más partes del código.

6.6.1. Código para el Servidor

En esta sección presentamos parte del código realizado pa
Servidor.

```
create proc < Parámetros >
as
declare @status int, @gpo_valida tinyint, @login_usuario  varchar(30), @t_afectacion char(3),
        @gpo_operacion smallint, @num_autoriza   varchar(15), @restric_valida char(1),
        @afecta_elem_1 int, @afecta_elem_2 int,  @afecta_elem_3 int, @fecha_entrega datetim
        @num_pzs_no_emi  int


select @login_usuario   =suser_name()

/** Obtiene la fecha de operación actual del sistema **/
select @fecha_recepcion=fecha_vmd
  from FECHA_REGISTRO

/** Verifica si el documento tiene alguna autorización en AUTORIZACION **/
exec @status =spsiif_verifica_autorizacion  @folio_docto,  @tipo_operacion, @importe, 'A',
                                            @num_autoriza output
    if (@status not in (0,40173))    return @status

/** Prepara renglon de existencia **/
exec @status =spsiif_prep_renglon_EXISTENCIA  < Parámetros >
  if (@status !=0)   return @status

/* ----------------------------------
** Comienzan las actualizaciones
** ----------------------------------
** Asignamos el folio del sistema a la operación **/
save tran trFOLIO_SISTEMA

update FOLIO_SISTEMA
  set folio_sistema =folio_sistema +1

select @status =@@error
  if (@status !=0)
     begin
     rollback inserta_ent_bill
     return @status
     end

select @folio_sistema =folio_sistema
  from FOLIO_SISTEMA

if (@folio_sistema is NULL)
   begin
   rollback inserta_ent_bill
   return 30226
   end
```

```
Se inserta la información general de la operación y el detalle **/
tran trOPERACION

 into OPERACION
lues (< Parámetros >)

t @status =@@error
@status !=0)
egin
ollback inserta_ent_bill
eturn @status
nd

Se inserta el comentario de la operación en la tabla COMENTARIO  **/
comentario is not NULL) and (@comentario !=' ')
in
e tran trCOMENTARIO

rt into COMENTARIO values < Parámetros >

ct @status =@@error
(@status !=0)
 begin
 rollback inserta_ent_bill
 return @status
 end
d


nserta detalle de la operación en la tabla DE_OPERACION **/
 into DE_OPERACION
lect < Parámetros >
rom ENVASE e, DE_CONS_ENVASE_ENTREGA d
here d.fecha_consulta = @fecha_consulta
and d.id_consulta = @id_consulta
and e.institucion_empaque = d.institucion_empaque
and e consecutivo = d.consecutivo

 @status =@@error
@status !=0)
egin
llback inserta_ent_bill
turn @status
nd


nserta en la tabla OPERACION_CONSULTA **/
ran trOPERACION_CONSULTA

 into OPERACION_CONSULTA
lues< Parámetros >
```

```
select @status =@@error
  if (@status !=0)
    begin
    rollback inserta_ent_bill
    return @status
    end

/** Genera el número de acta **/
exec @status =spsifl_asigna_num_acta_ent @fecha_recepcion, @num_pzs, @mac_proc_fin,
                                         @mac_manufactura, @numero_acta output
select @status =@@error
  if (@status !=0)
    begin
    rollback inserta_ent_bill
    return @status
    end

/** Insertamos Encabezado y otros **/
save tran trENTREGA_BILLETE

insert into ENTREGA_BILLETE
values < Parámetros >

select @status =@@error
  if (@status !=0)
    begin
    rollback inserta_ent_bill
    return @status
    end

/** ejecuta procedimiento para Actualizar estado fisico y status_envase de la tabla
ENVASE **/
save tran trENVASE

update ENVASE
    set < Parámetros >
    from ENTREGA_BILLETE e, DE_OPERACION d, ENVASE en
    where e.fecha_registro = @fecha_recepcion
      and e.folio_sistema = @folio_sistema
      and d.fecha_registro = e.fecha_registro
      and d.folio_sistema = e.folio_sistema
      and d.institucion_empaque = en.institucion_empaque
      and d.consecutivo = en.consecutivo
      and en.status_envase =20

select @status =@@error
  if (@status !=0)
    begin
    rollback inserta_ent_bill
    return @status
    end
```

```
Actualización de campos en la tabla ENVASE_UBICACION**/
 tran trENVASE_UBICACION

ate ENVASE_UBICACION
et < Parámetros >
om ENTREGA_BILLETE e, DE_OPERACION d, ENVASE_UBICACION u
where e.fecha_registro = @fecha_recepcion
  and e.folio_sistema = @folio_sistema
  and d.fecha_registro = e.fecha_registro
  and d.folio_sistema = e.folio_sistema
  and d.institucion_empaque = u.institucion_empaque
  and d.consecutivo = u.consecutivo

ct @status =@@error
(@status !=0)
begin
ollback inserta_ent_bill
return @status
end

Actualización de campos en la tabla MACULATURA_MANUFACTURA **/
)mac_manufactura != 0)
gin
ve tran trMACULATURA_MANUFACTURA
date MACULATURA_MANUFACTURA
 set < Parámetros >
here especimen = @especimen
 and  fecha_emision = @fecha_emision
 and  tiro = @tiro

lect @status = @@error
(@status != 0)
begin
ollback inserta_ent_bill
return @status
end
d

Inserta los datos de la entrega en la tabla CONTROL_ENTREGA_BILLETE **/
ct @num_pzs_no_emi= @mac_proc_fin + @mac_manufactura

 tran trCONTROL_ENTREGA_BILLETE

t into CONTROL_ENTREGA_BILLETE
es (< Parámetros >)

ct @status =@@error
(@status !=0)
begin
ollback inserta_ent_bill
return @status
end
```

```
/** Obtiene el tipo de afectación correspondiente al tipo de operación. **/
exec @status =spsifi_da_tipo_afectacion < Parámetros >
 if (@status !=0)
    begin
    rollback inserta_ent_bill
    return @status
    end

select @num_pzs_no_emi = @num_pzs_no_emi * @afecta_elem_2
select @num_pzs        = @num_pzs        * @afecta_elem_2

/** Incrementa número de piezas_recibidas y número de piezas_no_entregadas, p
tablas
** CONTROL_FABRICACION_BILLETE y CONTROL_AUTORIZACION_BILLETE de acu
al
** t_afectacion, **/
save tran trCONTROL_FABRICACION_BILLETE

update CONTROL_FABRICACION_BILLETE
    set < Parámetros >
 where numero_orden = @numero_orden
   and especimen = @especimen

select @status =@@error
  if (@status !=0)
     begin
     rollback inserta_ent_bill
     return @status
     end

save tran trD_CTRL_FABRICACION_BILL
update D_CONTROL_FABRICACION_BILLETE
    set < Parámetros >
 where numero_orden = @numero_orden
   and especimen = @especimen
   and fecha_emision = @fecha_emision
   and tiro = @tiro

select @status =@@error
  if (@status !=0)
     begin
     rollback inserta_ent_bill
     return @status
     end

/**Actualiza lo campos número de piezas_recibidas y numero_piezas_no_entregad
** en las tablas CONTROL_FABRICACION_BILLETE y CONTROL_AUTORIZACION_B.
**/
save tran trCONTROL_AUTORIZACION_BILLETE

update CONTROL_AUTORIZACION_BILLETE
    set < Parámetros >
 where fecha_emision = @fecha_emision
```

```
d especimen = @especimen

t @status =@@error
@status !=0)
egin
ollback inserta_ent_bill
eturn @status
nd

Si hay una autorización para el documento se lo asigna **/
num_autoriza is not NULL)
in
c @status =spsifi_asigna_autorizacion  < Parámetros >
if (@status !=0)
   begin
   rollback inserta_ent_bill
   select @folio_sistema =NULL
   return @status
   end
d

n 0
```

```
dContenedores


proc spsif_envio_entrega_billete
```

Código para el Cliente

anto a la codificación del cliente presentamos parte del
> que corresponde a la declaración de objetos y sus
dimientos.

```
Declaración de objetos **/
rd
al type uo_control_entrega_billete from n_base
' type
rward

al type uo_control_entrega_billete from n_base
 type

al uo_control_entrega_billete uo_control_entrega_billete

ariables
trega_billete    uo_entrega_billete
```

```
/** Objetos **/
uo_folio_terminal            iuo_folio_terminal
uo_operacion                 iuo_operacion
uo_comentario                iuo_comentario
uo_validacion_operacion      iuo_validacion_operacion
uo_tipo_operacion    iuo_tipo_operacion
uo_restriccion_existencia    iuo_restriccion_existencia
uo_sitio                     iuo_sitio

/** Declaración de variables con datos **/
string is_tipo_afectacion, is_plaza_origen, is_plaza_destino, is_string_null
long   il_tipo_sitio_origen, il_tipo_sitio_destino, il_institucion_origen, il_institucion_destino,
       il_boveda_origen, il_boveda_destino, il_folio_terminal, il_estado_fisico_origen,
       il_estado_fisico_destino, il_long_null, il_resultado

/*****************************************************************
   Estas variables almacenan los datos que envia el  servidor y que no se capturan
   cuando es una cancelación o una modificación
*****************************************************************
datetime id_fecha_registro
long     il_folio_sistema,     il_terminal
end variables

/** Declaración de funciones **/
forward prototypes

public FUNCION long of_obtiene_datos_servidor(long al_terminal, long al_inst_terminal,
              string as_plaza_terminal, datetime ad_fecha_registro, long al_folio_sistema,
              ref long al_terminal_emisora, ref long al_folio_terminal,
              ref long al_folio_documento, ref long al_inst_movto, ref string as_plaza_movto,
              ref string as_tipo_movto, ref long al_tipo_efectivo, ref long al_tipo_operacion,
              ref string as_divisa, ref decimal adc_importe, ref datetime ad_fecha_aplicacion,
              ref string as_numero_acta, ref datetime ad_fecha_emision,
              ref string as_numero_orden, ref datetime ad_fecha_entrega, ref long al_fabrican
              ref long al_edo_fis_orig, ref long al_boveda_dest, ref long al_edo_fis_dest,
              ref long al_especimen, ref string as_tiro, ref string as_series, ref string as_prefijo
              ref string as_millones, ref long adc_num_piezas, ref long adc_mac_proc_fin,
              ref long adc_mac_manufactura, ref string as_comentario,
              ref datetime ad_fecha_consulta, ref long ai_id_consulta)

public FUNCION boolean of_elimina_en_comunicacion_siac (datetime adt_fecha_registro,
              long al_folio_terminal)

public FUNCION boolean of_existe_operacion (datetime ad_fecha, long al_folio)

public FUNCION boolean of_guarda_en_comunicacion_siac (long al_tipo_operacion, string as_div
              long al_tipo_efectivo, string as_estado,  long al_tipo_sitio_orig,
              long al_tipo_sitio_dest, string as_suc_sitios, long al_edo_fis_orig,
              long al_edo_fis_dest, datetime dt_fecha_registro, long al_folio_terminal,
              long al_folio_documento, long al_folio_sistema, string as_tipo_movimiento)

public FUNCION boolean of_modifica_en_comunicacion_siac(long al_tipo_operacion, string as_di
              long al_tipo_efectivo, string as_estado, long al_tipo_sitio_orig,
```

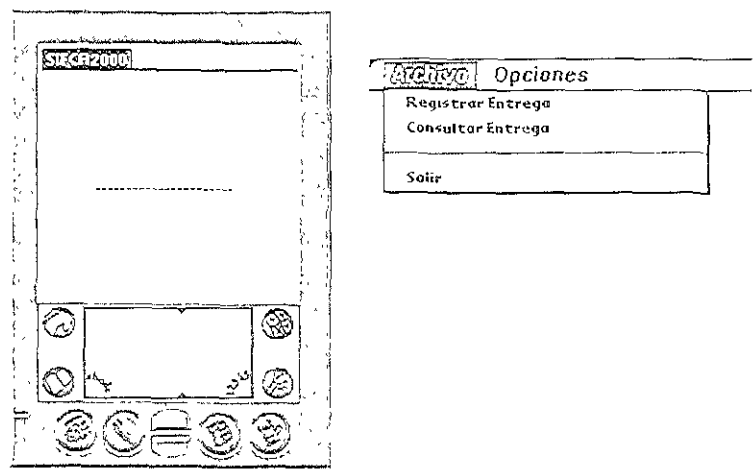
```
    long al_tipo_sitio_dest, string as_suc_sitios, long al_edo_fis_orig,
    long al_edo_fis_dest, datetime adt_fecha_registro, long al_folio_terminal,
    long al_folio_documento, long al_folio_sistema, string as_tipo_movimiento)

c FUNCION boolean of_obtiene_datos_a_desplegar(datetime ad_fecha_registro_param,
    long al_folio_sistema_param, ref long al_folio_documento,
    ref datetime  ad_fecha_aplicacion, ref long al_tipo_efectivo, ref string as_divisa,
    ref string as_numero_acta, ref datetime ad_fecha_emision,
    ref string as_numero_orden, ref datetime ad_fecha_entrega,
    ref long al_fabricante, ref long al_edo_fis_orig, ref long al_inst_dest,
    ref string as_plaza_dest, ref long al_boveda_dest, ref long al_edo_fis_dest,
    ref long al_especimen, ref string as_tiro, ref string as_series,
    ref string as_prefijos, ref string as_millones, ref long ad_num_piezas,
    ref long ad_mac_proc_fin, ref long ad_mac_manufactura, ref decimal ad_importe,
    ref string as_comentario, ref datetime ad_fecha_consulta, ref long al_id_consulta)

c FUNCION boolean of_elimina_operacion (string as_tipo_movimiento)

c FUNCION boolean of_continuar_con_error (string as_mensaje_error)

c FUNCION boolean of_guarda_operacion (long al_folio_documento, string as_tipo_movimiento,
    long al_tipo_efectivo, long al_tipo_operacion, decimal ad_importe,
    datetime ad_fecha_aplicacion, string as_numero_acta, datetime ad_fecha_emision,
    string as_numero_orden, datetime ad_fecha_entrega, long al_fabricante,
    long al_estado_fisico_origen, long al_institucion_dest, string as_plaza_dest,
    long al_boveda_dest, long al_estado_fisico_dest, long al_especimen, string as_tiro,
    string as_series, string as_prefijos, string as_millones, long al_numero_piezas,
    long al_numero_piezas_proc, long al_numero_piezas_manu, string as_divisa,
    string as_comentario, datetime ad_fecha_consulta, long al_id_consulta,
    long al_calculo_mac_proc, string as_claves[], string as_passwords[],
    long al_numero_claves, long argl_grupos_autorizacion[])

c FUNCION boolean of_modifica_operacion (long al_folio, string as_tipo_movimiento,
    long al_tipo_efectivo, long al_tipo_operacion, decimal ad_importe,
    datetime ad_fecha_aplicacion, string as_numero_acta, datetime ad_fecha_emision,
    string as_numero_orden, datetime ad_fecha_entrega, long al_fabricante,
    long al_estado_fisico_orig, long al_institucion_dest, string as_plaza_dest,
    long al_boveda_dest, long al_estado_fisico_dest, long al_especimen, string as_tiro,
    string as_series, string as_prefijos, string as_millones, decimal ad_numero_piezas,
    decimal ad_mac_proc_fin, decimal ad_mac_manufactura,
    datetime ad_fecha_consulta, long al_id_consulta, long al_calculo_mac_proc,
    long al_grupo_aut1, long al_grupo_aut2, string as_empleado1,
    string as_empleado2, string as_password1, string as_password2,
    string as_divisa, string as_comentario)

c FUNCION boolean of_obtiene_datos_base_local(datetime ad_fecha_registro_param,
    long al_folio_terminal_param, ref long al_folio_documento,
    ref datetime ad_fecha_aplicacion, ref long al_tipo_efectivo, ref string as_divisa,
    ref string as_numero_acta, ref datetime ad_fecha_emision,
    ref string as_numero_orden, ref datetime ad_fecha_entrega, ref long al_fabricante,
    ref long al_edo_fis_orig, ref long al_inst_dest, ref string as_plaza_dest,
    ref long al_boveda_dest, ref long al_edo_fis_dest, ref long al_especimen,
    ref string as_tiro, ref string as_series, ref string as_prefijos, ref string as_millones,
```

```
                ref decimal ad_num_piezas, ref decimal ad_mac_proc_fin,
                ref decimal ad_mac_manufactura, ref datetime ad_fecha_consulta,
                ref long al_id_consulta, ref long al_calculo_mac_proc, ref long al_grupo_aut1,
                ref long al_grupo_aut2, ref string as_empleado1, ref string as_empleado2,
                ref string as_password1, ref string as_password2, ref decimal ad_importe,
                ref string as_comentario)

end prototypes

/** Definición de funciones **/
public FUNCION long of_obtiene_datos_servidor (long al_terminal, long al_inst_terminal,
                string as_plaza_terminal, datetime ad_fecha_registro, long al_folio_sistema,
                ref long al_terminal_emisora, ref long al_folio_terminal,
                ref long al_folio_documento, ref long al_inst_movto, ref string as_plaza_movto,
                ref string as_tipo_movto, ref long al_tipo_efectivo, ref long al_tipo_operacion,
                ref string as_divisa, ref decimal adc_importe, ref datetime ad_fecha_aplicacion,
                ref string as_numero_acta, ref datetime ad_fecha_emision,
                ref string as_numero_orden, ref datetime ad_fecha_entrega, ref long al_fabrica
                ref long al_edo_fis_orig, ref long al_boveda_dest, ref long al_edo_fis_dest,
                ref long al_especimen, ref string as_tiro, ref string as_series, ref string as_prefi
                ref string as_millones, ref long adc_num_piezas, ref long adc_mac_proc_fin,
                ref long adc_mac_manufactura, ref string as_comentario,
                ref datetime ad_fecha_consulta, ref long ai_id_consulta);

integer     li_tipo_efectivo
string      ls_importe
long ll_num_piezas, ll_mac_proc_fin, ll_mac_manufactura, ll_estado_fis_orig, ll_importe

as_numero_acta      = space (255)
ls_importe          = space (255)
ad_fecha_entrega    = datetime(date('01/01/1990'))
ad_fecha_aplicacion = datetime(date('01/01/1990'))
ad_fecha_emision    = datetime(date('01/01/1990'))
as_tipo_movto       = space (255)
as_divisa           = space (255)
as_plaza_movto      = space (255)
as_comentario       = space (255)
as_tiro             = space (255)
as_series           = space (255)
as_prefijos         = space (255)
as_millones         = space (255)
as_numero_acta      = space(255)
as_numero_orden     = space(255)
ll_num_piezas       = 0
ll_mac_proc_fin     = 0
ll_mac_manufactura  = 0

il_resultado = gnv_tr_servidor.spsif_conscan_entrega_billete (al_terminal, al_inst_terminal,
                as_plaza_terminal, ad_fecha_registro, al_folio_sistema,
                al_terminal_emisora, al_folio_terminal, al_folio_documento,
                al_inst_movto, as_plaza_movto, as_tipo_movto, al_tipo_efectivo,
                al_tipo_operacion, as_divisa, ls_importe,     ad_fecha_aplicacion,
                as_numero_acta, ad_fecha_emision, as_numero_orden,
```

```
        ad_fecha_entrega, al_fabricante, al_edo_fis_orig, al_boveda_dest,
        al_edo_fis_dest, al_especimen, as_tiro, as_series, as_prefijos,
        as_millones, adc_num_piezas, adc_mac_proc_fin, adc_mac_manufactura,
        as_comentario,ad_fecha_consulta, ai_id_consulta)

'_tr_servidor.SQLCode <> 0 then
ssagebox('of_obtiene_datos_servidor',gnv_tr_servidor.SQLCode)
ssagebox('Error','uo_control_entrega_billete.obtiene_datos_servidor ()')
ssagebox('of_obtiene_datos_servidor',string(il_resultado))
esultado = gnv_tr_servidor.SQLDBCode

_resultado <> 0 then
f gnv_constantes.of_getdesplegarmensajes () then
  uo_obtiene_descripcion_error luo_obtiene_descripcion_error
  luo_obtiene_descripcion_error  = create  uo_obtiene_descripcion_error
  luo_obtiene_descripcion_error.of_obtiene_descripcion_error(il_resultado)
  destroy luo_obtiene_descripcion_error
nd if
1 if
f

importe = Dec (ls_importe)
n il_resultado
n 1
FUNCION
```

Código para la Palm

tinuación se muestra el código que controla los eventos de la lla principal, cuando una opción del menú es seleccionada con la pantalla correspondiente (Fig. 6.8.).



```
#include <Pilot.h>
#include <SysEvtMgr.h>
#include "MainFormRsc.h"
#include "entregadb.h"
#include "utils.h"

/***********************************************************************
* FUNCION:    MainFormDoCommand
* DESCRIPCION: Esta rutina realiza las actividades
*                  correspondientes a la opción del menú seleccionada.
* PARAMETROS:  command  - id del menu
* REGRESA:    nada
* REVISION :
***********************************************************************/
static Boolean MainFormDoCommand(Word command)
{
    Boolean handled = false;
    EventType               newEvent;

    switch (command)
            {
                case ArchivoRegistrarEntrega:
                    FrmGotoForm(IdConsultaForm);
                    handled =true;
                    break;

                case ArchivoConsultarEntrega:
                    FrmGotoForm(ConsultaForm);
                    handled =true;
                    break;

                case ArchivoSalir:
                    MemSet(&newEvent, sizeof(EventType), 0);
                    newEvent.eType = appStopEvent;
                    EvtAddEventToQueue(&newEvent);
                    handled =true;
                    break;

                case MainOptionsSicronizar:
                    SetFieldText(MainActividadField,"Sincronizando...",     18, true);
                    Sincroniza();
                    SetFieldText( MainActividadField, " ", 18,true);
                    FrmAlert(SincronizacionAlert);
                    handled =true;
                    break;

                case MainOptionsPrueba:
                    FrmGotoForm(PruebaForm);
                    handled =true;
                    break;
```

```
                case MainOptionsAcercade:
                        FrmAlert(AcercadeAlert);
                        handled =true;
                        break;
        }
return handled;


*****************************************************************
INCION:   MainFormHandleEvent
ESCRIPCION: Esta rutina es el manejador de eventos de la
            ventana principal.
RAMETROS:  eventP  - apuntador del tipo de evento.
GRESA:    true – Si el evento puede ser manejado aquí
          false- si el evento no puede ser manejado aquí.
VISION:
*****************************************************************/
an MainFormHandleEvent(EventPtr eventP);
an MainFormHandleEvent(EventPtr eventP)

olean handled = false;
rmPtr frmP;

switch (eventP->eType)
        {
        case menuEvent:
                return MainFormDoCommand(eventP->data.menu.itemID);

        case frmOpenEvent:
                frmP = FrmGetActiveForm();
                MainFormInit( frmP);
                FrmDrawForm ( frmP);
                handled = true;
                break;

        default:
                break;
        }
return handled;
```

vez que el usuario elige la opción para registrar los nedores que forman la entrega de piezas, la PALM pide el ificador de la relación en donde se encuentra la lista de nedores a entregar, esto con el fin de que se valide que los nedores que se van a entregar coincidan con los que se aron entregar.

e mencionó anteriormente, otra parte del código para la Palm stará en el apéndice C.

En este capítulo se definieron los estándares que establec
para la codificación del sistema de contenedores, además de
se presentó parte del código que se implementó para el clie
el servidor y la palm.

En el siguiente capítulo se desarrollarán las pruebas
Sistema.

CAPÍTULO VII

PRUEBAS DEL SISTEMA

el presente capítulo se describe la razón por la cual es esario realizar pruebas a los sistemas, además de conceptos y icas para llevarlas a cabo.

Definiciones básicas y objetivos de las pruebas

se de pruebas, por lo general, lleva la mitad del tiempo de rollo del sistema, y el método utilizado para llevarlas a cabo relativamente, independiente de la metodología usada en el rollo. El realizar pruebas al sistema es importante y no puede rse, porque aunque en la fase de análisis y diseño se hayan bien las cosas, se debe verificar que no existen errores. Si l contrario, las cosas se hicieron mal, entonces se sigue un so iterativo donde, en primer lugar, se detecta la presencia rores y después se verifica que los programas corregidos onen adecuadamente.

trategia de la fase de pruebas es ascendente. El enfoque iente se inicia al probar módulos individuales y pequeños, idamente, lo que se denomina pruebas de unidades o de mas. Luego, los módulos individuales se combinan para formar los más grandes que se prueban en masa, esto se conoce como as de subsistemas. Finalmente, todos los componentes del a se combinan para ser probados, esto se conoce como pruebas [illegible]. El [illegible] las pruebas de

aceptación, donde el usuario utiliza sus propios casos de p
para verificar que el sistema está trabajando de forma correct

Las actividades realizadas durante las pruebas se divide
verificación y validación. En la verificación se revisa qu
resultados de las pruebas sean de acuerdo a las especificaci
En la validación se verifica que los resultados sean realment
que el usuario quiere. Estos términos se resumen en dos pregun

- Verificación. ¿Se está construyendo correctament
  sistema?

- Validación. ¿Se está construyendo el sistema correc

Como se mencionó anteriormente, las pruebas se llevan una
parte de tiempo de trabajo y por consecuencia deben planearse
y como se hace con el análisis y el diseño.

Es importante en la fase de pruebas definir algunos conc
fundamentales como son falla, defecto y error. Una **falla** o
cuando un programa tiene un comportamiento distinto al esperad
esta forma una falla es una propiedad del sistema cuando es
ejecución. Un **defecto** existe en el código del programa. S
código está mal, éste puede arreglarse cambiando el código
defecto, si lo hay, puede causar una falla. Como consecuenci
hay defectos si el sistema no falla. Un **error** es una acción h
que resulta en una falla del sistema. Así, un error puede lle
la inclusión de una falla en el sistema, haciendo que el si
falle.

En este sentido es importante decir que una prueba resulta ex
cuando se encuentran fallas, por el contrario, una prueba fr
si no se encuentran fallas. El propósito de las prueba
encontrar fallas en el sistema.

Por otra parte, durante la fase de pruebas es común que al cor
las fallas detectadas, se introduzcan nuevas fallas. Así, cuan
sistema ha sido corregido, éste debe probarse una vez más. A
técnica se le denomina prueba de regresión y su propósit
verificar que la antigua funcionalidad permanece.

Para obtener un producto de calidad, no basta solamente c
realización de las pruebas. Además de validar los dist
procesos durante el desarrollo, se realizan actividades tales
revisiones e inspecciones del código. A través de este méto
obtiene código de alta calidad. En la sección 7.3. y 7.4
discuten conceptos relacionados con las métricas para
aseguramiento de calidad.

Estrategias de prueba

nomina como estrategia de prueba a las técnicas de diseño de de prueba que señalan una serie de pasos definidos que hay levar a cabo, además de que se deben planificar y determinar iempo, esfuerzo y recursos requeridos. Para el sistema de nedores se tienen una primera etapa para probar los módulos de izaciones, órdenes de fabricación, Empaque de Piezas y Entrega ezas y que está organizada como se muestra en la tabla 7.1.

idad	Duración
as internas para Autorizaciones, Orden y Entregas	4 días
as con usuarios para Autorizaciones, Orden y gas	3 días
a integral para Autorizaciones y Orden	2 días
cciones empaque	4 días
as de empaque área de empaque	5 días
a integral empaque en F.B.	5 días
a integral Entrega	2 días

Tabla 7.1. Primera etapa de pruebas.

neral las estrategias coinciden en que se debe comenzar la a a nivel de módulo e incrementarse progresivamente hasta rar todo el sistema.

comendable asignar a alguien responsable de la prueba, y según omento en que se realice, se pueden utilizar diferentes cas.

strategia de prueba de *software* debe contener pruebas de bajo o nivel. Las pruebas de bajo nivel son para verificar que el o se ha implementado correctamente y las de alto nivel son verificar que los requerimientos del usuario son cumplidos por ferentes funciones del sistema.

en dos categorías de diferentes técnicas de diseño de s. las pruebas de caja blanca y las pruebas de caja negra.

ruebas de caja blanca se centran en la estructura de control rograma. Se obtienen casos de prueba que aseguren que durante ueba se han ejecutado por lo menos una vez todas las ías del programa y que se ejercitan todas las condiciones s. Este tipo de pruebas se centran en [illegible]

escala debido a que son típicamente aplicadas a peq componentes de programas (por ejemplo, al módulo de traspasos, retiros).

Por otro lado, la prueba de caja negra amplía el enfoque y se denominar como prueba a gran escala. Este tipo de pruebas diseñadas para validar los requisitos funcionales del sistema fijarse en el funcionamiento interno de un programa. Estas téc se centran en el ámbito de información de un programa, de forma se proporcione la cobertura completa de prueba.

La estrategia de Pruebas Orientadas a Objetos (POO) tiene el concepto, empezar por pruebas de menor a mayor escala.

7.3. Tipos de pruebas

Como se mencionó brevemente en los objetivos de las pru existen tres niveles de pruebas:

Pruebas de Unidad. En las que sólo se prueba una unidad. unidad mínima a probar en el contexto de programación orienta objetos (POO) es la clase u objeto encapsulado, por lo qu prueba se enfoca a las operaciones encapsuladas por la clase comportamiento de la clase misma.

Pruebas de Subsistemas o de Integración. Son realizadas co propósito de que las unidades en conjunto trabajen correctame Este tipo de pruebas para *software* orientado a objetos hacerse probando las clases independientes (que son las qu utilizan otras clases, o utilizan muy pocas) para seguir des con las clases dependientes hasta probar el sistema completo. forma de realizar este nivel de pruebas es utilizando la estra de pruebas basadas en hilos, que integra el conjunto de clases responden a una entrada o evento del sistema y lo prueba.

En el nivel de Pruebas de Validación o del Sistema, la valida se centra en las acciones visibles por el usuario y las sal reconocibles por éste. En este tipo de pruebas el ejecutor basarse en los casos de uso realizados en la etapa de análisis que poseen una alta probabilidad de tener errores encubiertc los requisitos de interacción del cliente. Es a este nivel dond utilizan los métodos de caja negra, ya que se realizan las pr del sistema completo o de la aplicación como tal, sin toma cuenta el funcionamiento interno del sistema. A este nive importante la vista de usuario final del sistema y en los cas

a se realizan las acciones típicas del mismo.

os de los tipos de pruebas que, aunque son independientes sí, pueden utilizarse en combinación, son:

ruebas de Operación, son las más comunes, ya que con ellas se el sistema en su operación normal por un periodo largo de . Esta prueba es para encontrar errores normales, es decir, es que el usuario normal se puede esperar que haga. Si el ma es reconfigurado durante su operación, entonces debe ser lo también.

ueba a Gran Escala significa ejecutar el sistema con todos sus etros en sus valores límite, todos los equipos son conectados, sistema es utilizado por muchos usuarios, cada uno ejecutando asos de uso simultáneamente. Esta prueba es costosa pero la iencia muestra que es útil para detectar nuevas fallas. Un extremo de este tipo de pruebas son las pruebas de estrés en he los límites extremos del sistema son probados.

ruebas de Ejecución o de Capacidad tienen como propósito medir bilidad de procesamiento del sistema con diferentes cargas. Se zan para medir entre otras cosas la asignación de enamiento, la utilización de CPU o la velocidad en un caso de pecífico.

ruebas de Sobrecarga van mas allá que las pruebas a Gran y de Ejecución, su propósito es ver como se comporta el ma cuando está sujeto a sobrecarga. Se debe ejecutar bien y no isminuir su rendimiento.

ueba Negativa es un tipo de prueba de estrés, prevista para r al sistema a tensiones más allá para las que ha sido uido. Se deben probar simultáneamente los casos de uso que mente no son ejecutados al mismo tiempo, como en el caso en gún proceso requiera de condiciones muy particulares para su ión, provocando así que el sistema sea utilizado de forma ecta.

uebas Basadas en Requerimientos son aquellas que se obtienen amente de los requerimientos.

rueba de Aceptación generalmente es planeada por la zación que pidió el sistema y es la revisión final por parte uario en el ambiente para el cual ha sido desarrollado. Este e pruebas a veces es llamado pruebas alfa. Una vez que se ha ado este tipo de pruebas, se toma la decisión de aceptación o [illegible]

que el producto es probado por usuarios especialmente seleccio quienes reportan las fallas que detectan. La prueba beta e forma de pre-liberación.

Otro tipo de pruebas son las **Ergonómicas**, que son import cuando hay una interfaz hombre-máquina en el sistema.

También existen métodos de pruebas especializadas que t directrices y enfoques únicos para ciertas aplicaciones. En tipo de pruebas podemos distinguir las siguientes:

- Pruebas de interfaces gráficas de usuario.
- Pruebas de arquitectura cliente/servidor.
- Pruebas de sistemas en tiempo real.

De estas últimas, las pruebas de interfaces gráficas de us (IGU) son un tipo especial de interfaz ergonómica, en el senti que a partir de ellas se reflejará una mayor productividad pa usuario.

Se dice que las pruebas del sistema nunca terminan, simplemen transfieren del ingeniero de al cliente, porque cada vez q cliente usa el programa, lleva a cabo una prueba.

Durante la realización de las pruebas es importante obs métricas para evaluar la calidad del código producid continuación se describen los conceptos más importantes acerc aseguramiento en la calidad del sistema.

7.4. CMM (*Capability Maturity Model*)

Durante todo el desarrollo del sistema es ideal que el equi desarrollo tenga la madurez para resolver los problemas que su sin embargo, es en la etapa de pruebas donde la madurez del pr de desarrollo del sistema se pone a prueba, debido, por una p a que los usuarios tienen la mira fija en el producto, y por parte, a que la fecha de entrega está próxima y se tiene un periodo para realizar las correcciones necesarias. Es por es en esta sección se describen los distintos niveles de madure existen y la mejor manera de atacar los problemas durant pruebas.

Dada la importancia de una filosofía para el proceso de desar

oftware en una organización, se han definido, por parte de tutos de ingeniería del *software*, ciertas clasificaciones, que ste momento ya son ampliamente aceptadas y aplicadas por idad de organizaciones que lo consideran de gran utilidad para gro de sus proyectos.

clasificación es denominada **CMM** (*Capability Maturity Model*) y ende los niveles 1 a 5, los cuales describimos a continuación:

- Nivel inicial. Comprende la utilización de un método de desarrollo no documentado, y cada desarrollador de *software* aplica su propia forma de trabajo.

- Nivel repetible. Cuenta con algún método de desarrollo, pero aún no ha sido formalizado, se consideran los resultados que se obtienen para que se continúe aplicando en el desarrollo de sistemas.

- Nivel definido. Utiliza un método formal bien documentado, mismo que es refinado por el grupo de desarrollo.

- Nivel controlado. Realiza evaluaciones formales de las diferentes características de los procesos y productos que se obtienen. Estas evaluaciones se hacen con periodicidad e incluyen: tiempos, costos, productividad, eficiencia y calidad.

- Nivel óptimo: Las medidas aplicadas en el nivel anterior, son retroalimentadas sistemáticamente para optimizar el proceso de desarrollo.

Organizaciones inmaduras

rocesos de desarrollo de *software* en organizaciones inmaduras generalmente improvisados, los jefes están enfocados en rer crisis inmediatas. No hay bases objetivas para juzgar la id de un producto o para resolver problemas del proceso o del cto. La calidad del producto es difícil de predecir.

beneficios de mejores métodos, herramientas o nuevas ogías no pueden alcanzarse efectivamente en un proyecto o o indisciplinado.

en organizaciones indisciplinadas, algunos proyectos de ro producen excelentes resultados. Esto ocurre generalmente

7.4.2. Organizaciones maduras

Por otro lado, en una organización madura, el proceso de desar de *software* es comunicado con precisión tanto a los n empleados como a los actuales, las actividades de trabajo se l a cabo de acuerdo a un plan. Los roles y responsabilidades claros y están bien definidos tanto en el proyecto como e organización.

Los jefes monitorean la calidad del *software* producido satisfacción del cliente. Los programas y presupuestos se bas históricos del desempeño y son realistas. Los resultados espe para costo, tiempo, funcionalidad y calidad son normal alcanzados.

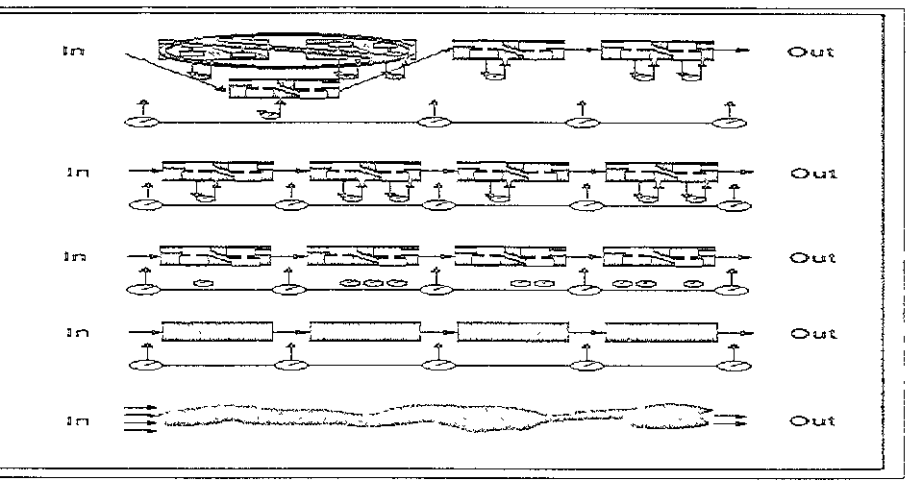
Las mejoras continuas sólo pueden ocurrir mediante un esf enfocado y mantenido dirigido a la construcción de una estru de procesos de prácticas de manejo, control e ingenierí *software*.

7.4.3. Visibilidad en el proceso de *software*

Como se observa en la figura 7.1, en el nivel 1 el proces desarrollo del *software* es considerado como una caja negra que la visibilidad en los procesos del proyecto es limitada. Y la organización de las actividades en etapas está pobre definido, para los líderes o jefes es muy difícil establece progreso del proyecto y de las actividades. Los requerimi fluyen dentro del proceso de *software* de manera no controlada.

En el nivel 2, los requerimientos del cliente y los productos controlados, se establecen prácticas de control y manej proyectos. Estos controles permiten visibilidad en el proyec ocasiones definidas. El proceso de desarrollo de *software* pued visto como una sucesión de cajas negras que le permiten dirección visibilidad en los puntos de transición, mientras qu actividades se suceden dentro de las cajas. Aunque la dire puede no saber los detalles de lo que ocurre dentro de la caja productos del proceso y los puntos para inspección identificados y confirman que el proceso está trabajando dirección reacciona a los problemas en cuanto ocurren.

rme avanzan los niveles 3, 4 y 5, la estructura de las cajas ace más visible y controlada. Los problemas se identifican y gen más prontamente.



a 7.1 Imagen externa de la visibilidad del Proceso de *Software* en cada nivel de madurez.

nivel construye los cimientos para subsecuentes niveles. Sin go, las organizaciones pueden usar exitosamente procesos itos en niveles de madurez superiores al que se encuentran. sos de Ingeniería como el análisis de requerimientos, el o, la codificación y las pruebas no se discuten en el CMM el nivel 3, aún así, organizaciones en el nivel 1 deben r a cabo estas actividades.

mplementar procesos de niveles de madurez mayores debe irse con inteligencia y sólo cuando sea necesario. El saltarse es puede ser contraproducente debido a que cada nivel ituye los cimientos necesarios para alcanzar el siguiente . Los procesos sin las bases necesarias fallan en el punto en ue más se necesitan (bajo estrés), y no proveen bases para as mejoras.

e debe quedar claro es que el esfuerzo para mejorar el proceso enfocarse en las necesidades de la organización en el contexto ambiente. La habilidad de implementar procesos de un nivel de er mayor no implica que se puedan saltar niveles de madurez.

Como se observa cada organización desea y trabaja para mejor proceso de desarrollo y mantenimiento del *software*, sin embarg se tienen actualmente los lineamientos o métodos a seguir qu ayuden a cumplir este objetivo.

El CMM provee el esquema necesario para ayudar a nu organización a mejorar su proceso de desarrollo de *software*, p provee límites para incrementar la madurez exitosamente y de organizada. Sin metas no sabríamos a donde queremos llegar, estas metas son inalcanzables causarán frustración y fracaso intento. El CMM nos ayuda a definir dichas metas y a ev nuestros avances, por lo tanto nuestro objetivo será alcanz siguiente nivel de madurez al que nos encontramos adaptando e a nuestra organización.

Como se ha mencionado, existen diferentes niveles de ma respecto al proceso de desarrollo de *software* que se sigue e organización. Particularmente en el Banco Central encontramos pesar de los buenos resultados en diferentes áreas y d introducción de nuevas metodologías o tecnologías, se encuent el nivel 1 debido principalmente a la falta de métr organización y documentación de los proyectos.

Por lo tanto, el objetivo es alcanzar el nivel 2 del CMM. Para se tiene que estudiar en qué consiste, saber cuáles son características, adaptarlo al Banco Central y crear los elem que se indiquen de acuerdo a los proyectos propios.

Como el CMM menciona, en el nivel 2 no se incluye el ciclo de para los proyectos; sin embargo, se especifica que es obliga para cualquier organización en cualquier nivel de madurez, p que también deberá establecerse.

De esta forma, a pesar de que en México el desarrollo de sis se encuentra en el nivel 1, en el desarrollo del sistem contenedores se aplica un control sobre cada fase y sobre to la fase de pruebas donde se pretenden realizar activi programadas que faciliten esta fase, además de observar mét para asegurar la calidad del sistema.

7.5. Métricas de calidad

Las métricas de la medición del *software* se basan en medi reglas que se deben seguir para medir en forma eficiente la ca del *software*. Debe tomarse en cuenta sin embargo que estas me

asadas en ciertas características definidas para el *software*, pudieran ser rendimiento o funcionalidad entre otros.

Métricas técnicas del *software*

lemento clave de cualquier proceso de ingeniería es la ión. Empleamos medidas para entender mejor los atributos de odelos que creamos.

erencia de otras disciplinas, la ingeniería del *software* no basada en leyes cuantitativas básicas como la Física. En su , intentamos obtener un conjunto de medidas indirectas que dan a métricas que proporcionan una indicación de la calidad de *software*. Como las medidas y métricas del *software* no son utas, están abiertas a debate.

Calidad del *software*

puntos importantes a considerar en la calidad del *software*

- Los requerimientos del sistema son la base de las medidas de la calidad. La falta de concordancia con los requerimientos es una falta de calidad. Es por eso que desde el desarrollo del sistema hasta las pruebas del mismo, los requerimientos obtenidos en la fase de requerimientos son la base de las pruebas. Así, por ejemplo, se vigila que los requerimientos establecidos para la Autorizaciones, entregas de piezas, traspasos, etc. se cumplan durante las pruebas.

- Estándares específicos que definan un conjunto de criterios de desarrollo que guíen la manera en que se hace la ingeniería del *software*. Si no se siguen los criterios, habrá seguramente poca calidad.

- Existe un conjunto de requisitos implícitos que ha menudo no se nombran, estos pueden ser la disponibilidad de recursos en disco, funcionalidad de dispositivos que participen en la operación del *software*, entre otros. Si el *software* cumple con sus requerimientos explícitos pero falla en los implícitos, la calidad del *software* no será fiable. La calidad del *software* es una compleja mezcla de factores que variarán a través de diferentes aplicaciones [illegible]

individualmente para asegurar que su funcionamient
el correcto.

7.5.3. El reto de las métricas técnicas

Durante las pasadas dos décadas, muchos investigadores
intentado desarrollar una sola métrica que proporcione una
completa de la complejidad del *software*.

Aunque se han propuesto docenas de medidas, en las que cad
tiene un punto de vista diferente de lo que se incluiría
evaluar la complejidad del *software*, por analogía, consideremo
métrica para evaluar un coche. Algunos observadores podrían
énfasis en el diseño de la carrocería, otros podrían hacer hi
en las características mecánicas, otros podrían valorar el p
o el rendimiento, o la economía de consumo o la capacid
reutilizarlo cuando se vaya a tirar. Como cualquiera de
características puede competir con las otras, es difícil obte
solo valor del atractivo del coche. Lo mismo ocurre c
software.

De esta forma, se sugiere que las métricas a utilizar para as
la calidad del producto, obtenido durante las diferentes fase
desarrollo del *software*, cumplan al menos con las sigui
características:

- Simple y fácil de calcular.
- Empírica e intuitivamente persuasiva.
- Consistente y objetiva.
- Consistente en el empleo de unidades y tamaños.
- Independiente del lenguaje de programación.
- Que incluya un mecanismo eficaz para la retroaliment
  de calidad.

Como un modelo de métricas que cumplen con lo estableci
parrafos anteriores, presentamos a continuación el conjun
métricas aplicadas en este proyecto.

7.5.4. FURPS

En la fase de pruebas se utiliza un conjunto de factores de c

rollado por *Hewlett Packard* y que se denomina como FURPS *ionality, Usability, Reliability, Performance, Supportability:* onalidad, Facilidad de empleo, Fiabilidad, Rendimiento y idad de soporte). Los factores de calidad FURPS definen los ente atributos principales.

- La ***funcionalidad*** se valora evaluando el conjunto de características y capacidades del programa, la generalidad de las funciones entregadas y la seguridad del sistema global. En el sistema de contenedores se mide constantemente este factor, debido a que las pruebas más importantes del sistema se enfocan a que la funcionalidad sea coherente con los requerimientos del usuario.

- La ***facilidad*** de uso se valora considerando factores humanos, la estética, consistencia y documentación general. Para probar la facilidad de uso se tomó como estándar la interfaz de *Microsoft Windows*, ya que el usuario trabaja en este ambiente. Así, se probó que el usuario tuviera acceso a todas las opciones del sistema de una forma fácil y familiar. Si para el usuario no era sencillo identificar las opciones del sistema, el proponía los cambios y éstos eran llevados a cabo, llegando incluso a detalles en los colores de las pantallas.

- La ***fiabilidad*** se evalúa midiendo la frecuencia y gravedad de los fallos, la exactitud de las salidas, el tiempo medio entre fallos, la capacidad de recuperación de un fallo y la capacidad de predicción del programa. En este aspecto se realizaron pruebas en las cuales, ante una acción incorrecta del usuario (error), el sistema debería responder en una forma determinada. En cuanto a las fallas del sistema, se evaluaron y se estimó el tiempo que llevaría corregirlas de acuerdo a la gravedad. Las fallas más graves, y en las cuales se puso especial atención, fueron aquellas donde se involucra el almacenamiento de datos, debido a que el sistema debe garantizar que los datos se registran confiablemente.

- El ***rendimiento*** se mide por la velocidad de procesamiento, el tiempo de respuesta, consumo de recursos, rendimiento efectivo total y eficacia. El aspecto de rendimiento se probó levemente debido a las características de potencia que brinda el hardware en la actualidad.

- La ***capacidad*** de soporte comprende la capacidad de ampliar [illegible]

hacer pruebas, la compatibilidad, la capacida configuración, la facilidad de instalación del sist la facilidad en la localización de problemas.

Como podemos ver, las métricas del *software* proporcionan una cuantitativa de valorar la calidad de los atributos interno producto, permitiendo por tanto valorar la calidad del proc Además, proporcionan la visión interna necesaria para crear m efectivos de análisis y de diseño, un código sólido y pr minuciosas.

Para que sea útil en el contexto del mundo real, una métric *software* debe ser simple y calculable, persuasiva, consiste objetiva. Debería ser independiente del lenguaje de programac proporcionar una retroalimentación eficaz para el equip desarrollo.

Una vez descritos los conceptos utilizados durante la fas pruebas, para asegurar la calidad del sistema, describiremo pruebas hechas al mismo y los resultados obtenidos.

7.6. Pruebas al sistema de contenedores

Los conceptos antes descritos fueron tomados en cuenta durant pruebas al sistema de contenedores. Se realizan pruebas a dist niveles en el sistema y durante la corrección de falla observaron las métricas para conservar la calidad del códig continuación se describen las pruebas realizadas.

7.6.1. Pruebas de Interfaz Gráfica

En el desarrollo del sistema de contenedores se realizó la p para la Interfaz Gráfica de Usuario, en la que se consideraro siguientes puntos:

Para las ventanas se verificó entre otras cosas, que tuvier nombre correcto, que se abrieran basándose en el teclado o menú; que se pudieran mover, ajustar en tamaño y desplega problemas; que la información contenida en ellas adecuadamente accesible con el ratón, teclas de función, flech dirección y teclado en general; que la ventana activa resa adecuadamente; que sus funciones estuvieran o no operativas revisó también que todos los menús, barras de herramientas, b

antes, cuadros de diálogo, botones, iconos y otros controles antes se desplegaran apropiadamente además de que las as se cerraran de una forma adecuada.

os menús emergentes y operantes con el ratón se verificó que nara el despliegue de los menús, que las funciones de menú eran resaltadas (habilitadas) o difuminadas (deshabilitadas), iendo del contexto de la ventana activa, y que se ejecutaran las funciones como se esperaba; que los nombres de menús y nes fuera evidente y también se revisó el tipo, tamaño y o del texto.

se verificó que los cursores y punteros cambiaran según la ión que se realizara.

nto a los datos introducidos se verificó que se aceptaran los de datos requeridos, y que se detectaran datos inválidos. se revisó que los mensajes fueran inteligibles.

anto a las pruebas de Arquitectura Cliente/Servidor, se có el rendimiento del sistema en la plataforma, la cación de red, y el servicio proporcionado a múltiples es desde la base de datos centralizada.

as pruebas de sistemas en tiempo real se tuvo que probar cada independientemente para descubrir errores de lógica y de namiento, posteriormente se probó el comportamiento de o a los sucesos de la aplicación, examinando individualmente nportamiento del sistema para encontrar errores asociados a cesos. Subsecuentemente se probaron los sucesos en orden y ncia aleatoria para detectar errores de comportamiento y lo nte es comprobar si se producen errores de sincronismo entre que se comunican con otras, con diferentes datos y cargas de os. Además se probaron las tareas que se comunican mediante de mensajes o almacenes de datos, para detectar errores en el de esas zonas de almacenamiento de datos, y finalmente se ó una prueba el sistema completo para intentar descubrir s de interfaz *software-hardware*.

Pruebas de Unidad al sistema

e las pruebas del sistema de contenedores, cada *stored* ure, *trigger* y objeto se probó individualmente durante la implementación, para garantizar que el funcionamiento de no [illegible] correcto antes de probarlo en junto a los demás. [illegible] una prueba a *stored procedure* que se [illegible]

fabricadas. También se probó la ventana a través de la cu usuario introduce los datos de la entrega de piezas. Y se ve que ambas funcionan independientemente.

7.6.3. Pruebas de Integración

Una vez que cada unidad es probada y se garantiza su funcionar individual, ambas se unen para obtener el resultado correcto. caso de las entregas, se prueba que la ventana a través de la se introduce la información, envíe al servidor los datos utili el *stored procedure* correspondiente. Durante la prueba se pr encontrar fallas (si las hay) en la compatibilidad de los dato la ventana envía al servidor. Así, si el *stored procedure* recibir parámetros tales como: Clave de la bóveda que (entero), Número de piezas a entregar(entero), Denominac entregar (Caracter), etc. Entonces, el cliente a través d objetos debe permitir la captura de los datos y enviarl servidor manteniendo la compatibilidad de datos.

7.6.4. Pruebas de Validación

En las pruebas de validación se toman los casos de uso, des durante la fase de análisis, como guía, y en conjunto c usuario se sigue la secuencia de pasos de cada caso de uso realizar las pruebas del sistema.

En este sentido, para la entrega de efectivo, se realizaro pruebas de los siguientes casos de uso:

- **Alta Relación Entrega.** En donde el usuario, que en el efectivo, especifica los códigos de barras d contenedores que se van a entregar, a través d relación que se registra en el sistema.

- **Cancelar Relación Entrega anterior.** En este cas usuario que registró la relación de entrega, verifi sea posible cancelarlas, en caso de que así lo re la operación.

- **Consultar Relación Entrega anterior.** Cada usuario, que el que entrega como el que recibe las piezas, que sea posible consultar las relaciones de entreg ha registrado en el sistema.

- **Alta Operación Entrega.** El usuario que recibe las piezas, prueba que los códigos de barras de los contenedores que está recibiendo se transmitan a la computadora por medio de la lectora de código de barras, que el sistema valide que son los correctos y una vez que se introducen los datos que complementan el registro, éstos sean enviados al servidor y se almacenen.

- **Modificación Operación de Entrega.** Es necesario probar que el sistema permita la modificación de la operación de entrega, para que en caso de que uno o más datos sean incorrectos, el usuario pueda cambiarlos sin necesidad de cancelar la entrega y volverla a dar de alta.

- **Cancelar Operación Entrega.** Si el usuario decide cancelar la entrega, el sistema debe permitirlo y por lo tanto debe ser probada esta funcionalidad.

- **Imprimir Acta de Entrega.** El usuario verifica que la información registrada durante la entrega se pueda imprimir para que las partes involucrada firmen de conformidad.

de las pruebas, fundamentales, antes descritas, el sistema
sto a prueba durante un periodo de dos meses, en el cual se
ó a su máxima capacidad, tanto en la cantidad de usuarios que
esan como en la cantidad de operaciones que se registran
mente. Con el fin de juzgar las pruebas objetivamente, es
rio crear un grupo, que esté constituido en su mayoría, por
uarios, y por gente responsable del desarrollo del sistema.
mente se reportan los resultado de las pruebas en un formato
se especifican los siguientes rubros: Responsables de la
, caso de uso a probar, resultados esperado, resultado
dos y fallas detectadas.

emplo, se tiene que para el día 1, se probó la entrega de
en la que estuvieron involucrados las áreas usuarias y el
e sistemas y se obtuvieron los resultados que se muestran en
la 7.2.

DÍA	OBJETIVO	ACTIVIDADES	OBSERVACIONES	ACCION CORREC
1	Realizar pruebas de la Entrega de Piezas.	Se generó una relación de entrega.	El sistema mostró bolsas y paquetes en la lista de selección, cuando solo debería mostrar contenedores o plataformas.	Si se pretend entregar habr asignarlos a plataformas c entregarlos individualmer De otra forma seguirán apareciendo d disponibles p ser entregado cual es corre
		Se registra la operación de entrega y se imprime.	Se intentó imprimir la consulta y el sistema mandó un error indicando que no realizó la acción.	Se copió al sistema una I que es necesa para que el sistema pueda imprimir.

Tabla 7.2. Actividades y resultados de las pruebas al siste

Con base en los resultados de las pruebas, es posible rea acciones correctivas en el código del sistema para obten producto requerido.

Se probó también la interfaz con el usuario y en los casos era confuso para él interactuar con el sistema, se cambi presentación de las pantallas. En el total de las pruebas, só bajo porcentaje de pantallas se modificaron.

Durante esta fase se trabaja en un servidor de pruebas don tiene una replica del servidor de producción. De esta for servidor de pruebas es un espejo del servidor de producció terminar de ser probado el sistema, se hace un cambio en el ar de inicio del sistema donde se indica que la conexión debe s servidor de producción, y así el sistema entra en operaci forma transparente para el usuario. Los datos de prueba n puestos en el servidor de producción, y el sistema inicia cero. Si alguna información histórica necesita ser preparada que el sistema entre en producción, se analiza junto co usuarios cual es la información y el área de sistema responsable de cargar los datos en el servidor de producción.

este capítulo se mencionaron los conceptos básicos acerca de teoría en el diseño y desarrollo de pruebas, además de los ltados de las pruebas generadas al sistema. Con base en esto el siguiente capítulo presentaremos las conclusiones.

CAPÍTULO VIII

CONCLUSIONES

sarrollo de sistemas es un área que durante las últimas as ha sido muy importante, ya que casi todo lo que nos rodea relacionado con los sistemas de cómputo o con algún tipo de ol.

o a la competencia que existe en esta área, es necesario rollar sistemas flexibles, sencillos de manejar, que tengan nterfaz agradable al usuario y principalmente que sean es y de gran calidad. Estas características son difíciles de ar, pero no imposibles y a lo largo de la ejecución de este to, hemos aprendido que mientras más cuidado se tenga en cada e las etapas de su desarrollo, será menor la cantidad de s y consecuentemente se obtiene un producto de mayor calidad.

n día el diseñador de *software* que intenta representar un a puede elegir entre una gran variedad de notaciones, cada eneralmente integrada en una metodología de análisis y diseño fica. La llegada de UML, con un amplio respaldo de las ías líderes en la industria del *software*, representa uno de esarrollos más importantes dentro de la metodología orientada etos, aportando un lenguaje de modelado universal que puede lizado con cualquier método.

anto en nuestro proyecto lo utilizamos ampliamente ya que el to de UML es visualizar, especificar, construir y documentar as Orientados a Objetos.

mos también que los componentes necesarios en todo proyecto na notación, un proceso, y una herramienta. Podemos disponer notación para [illegible] pero [illegible] proceso), no [illegible]

pero si no podemos comunicarlo (notación), la situación se cualquier caso deficiente. De igual manera, si no es po documentar los elementos del proyecto el éxito no puede completo. En resumen, debemos utilizar los tres componentes especificar, visualizar, documentar y crear de forma incremen iterativa una solución de *software* al problema a resolver.

Con el desarrollo de este trabajo se comprobó que el d orientado a objetos es más fácil de mantener, ya que los objet pueden entender y modificar como entidades autónomas. Esto q decir que si es necesario cambiar la implementación de un o para añadirle servicios o nuevas características, no se afe otros objetos del sistema. Además, como existe una correspond clara entre las entidades del mundo real y los objetos qu controlan en el sistema, se tiene un mejor entendimiento y consecuencia el mantenimiento se facilita.

Con todo esto podemos concluir que con el desarrollo de trabajo aumentamos nuestros conocimientos en diferentes puntos la utilización de UML en las etapas de análisis y d (principalmente), lo relacionado con la metodología y desarrol sistemas orientados a objetos.

Por otra parte, con el sistema de contenedores el Banco Ce obtiene mayor eficiencia en los procesos principales que rea De esta forma, el empaque de las piezas que se produce confiable y la información de cada envase puede consultars través del sistema, con fines de explotar la información obtener estadísticas que ayuden a monitorear la producció piezas en un periodo. En el mismo sentido, el sistema incremen control de los flujos de piezas, debido a que cada oper relacionada con los flujos es registrada en el momento en ocurren. Así las existencias de cada bóveda son verifi continuamente para determinar que no haya diferencias entr número de piezas que tiene físicamente una bóveda y el núme piezas que tiene registrado el sistema.

En resumen, el Banco Central, con el uso del sistem contenedores, incrementa la eficiencia, la seguridad y el co del manejo del billete, desde que se produce hasta que llega bancos usuarios.

En otro sentido, y en relación a la experiencia profes adquirida. Durante la elaboración de esta tesis, aumentamos nu experiencia en el desarrollo de investigaciones para que fuera organizadas y estructuradas. Además, obtuvimos experiencia planificación del trabajo y, asignación de actividades y recu debido a que tuvimos que realizar un plan de trabajo con fech

ga determinadas, lo cual implicó una distribución de recursos poder cumplir con las actividades programadas. Sobre este el trabajo de equipo se fortaleció, sobre todo al momento de ntar ideas y de discutir acerca de temas donde los puntos de diferían de un integrante a otro, sin embargo, siempre se presente el objetivo principal de cada discusión para echar el tiempo dedicado a ella.

tro lado, también se adquirió experiencia en el manejo de la logía de código de barras, la cual cada vez cobra mayor tancia a nivel mundial, debido a que agiliza las operaciones ualquier negocio realiza y que tienen que ver con el manejo de ctos, como en el caso del Banco Central.

mente, es importante mencionar que además de aprender cosas s en cada una de las etapas llevadas a cabo para el desarrollo Sistema, confirmamos que gracias a la formación profesional rida en la máxima casa de estudios y particularmente en la tad de Ingeniería podemos resolver los distintos problemas que esenten en nuestra vida profesional y personal.

Apéndice A

CASOS DE USO

Depósito

ntinuación se muestran el diagrama con los casos de uso ificados para el proceso de depósitos, así como la descripción s casos de uso más importantes.

casos de uso identificados, y que se muestran en la figura para el proceso de depósito son:

- Registro de depósito
- Registro del muestreo
- Generación de etiquetas

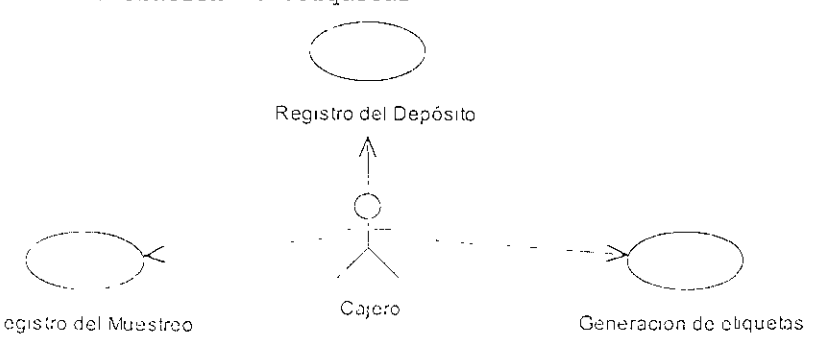


Figura A.1. Casos de uso para los depósitos.

Caso de uso : REGISTRO DE DEPOSITO.

Actores : Cajero, Representante del banco.

Resumen : Describe el proceso que se sigue en el sistema el registro de los depósitos.

Flujo Normal de Eventos

Acción de los actores	Respuesta del Sistema
1.- Se abre la ventanilla para que el representante del banco pueda ingresar sus valores y colocar la ficha de depósito sobre ellos.	2.- El sistema deberá per registrar y almacenar siguiente información cont en la ficha de depósito: • Clave de operación (depósito) • Folio de documento • Cuenta única de banc que deposita • Clave de institución donde se deposita • Clave de la plaza en que se realiza el depósito. • Fecha y hora de regi del depósito • Importe total de los empaques aceptados • Detalle del importe denominación del efe que deposita.
3.- Entrará el representante del banco, para cambiar el empaque y de ser necesario, actualizar la ficha de depósito.	4.-El sistema debe per realizar la actualización ficha de depósito

Tabla A.1. Caso normal para el registro del depósito. (Continúa)

e llena un formato con datos como: fecha, divisa, estado o del billete, importe del ito, importe por inación, hora de ingreso y del cajero.	
El cajero revisa que los ues se encuentren sin aciones y verifica que el o de "empaque" recibido erde con el número de ue detallado en la ficha de ito.	

Tabla A.1. Caso normal para el registro del depósito.

s Alternos

os empaques están alterados (raspadura, plomo mal colocado avisará al representante del banco para que los cambie y lice la ficha de depósito.

de uso : REGISTRO DE MUESTREO.

es : Cajero, personal de distribución.

en : Describe el proceso que se sigue en el sistema para gistro del muestreo.

Normal de Eventos

n de los actores	Respuesta del Sistema
cajero realizará un reo eligiendo al azar uno o s paquetes, dependiendo de ntidad de depósitos	2.- El sistema debe registrar los resultados del muestreo.

la A.2 Caso normal para el registro del muestreo (Continúa)

3.- Se llenará un formato con las observaciones realizadas para justificar el rechazo y se permite de nuevo el ingreso del representante del banco para retirar los empaques de la ventanilla.	4.- Si el muestreo refle rechazo parcial, el sistema generar una nueva fich depósito correspondiente a empaques aceptados y una ac rechazo.
5.- si es aceptado el depósito se traslada a la bóveda de caja.	

Tabla A.2. Caso normal para el registro del muestreo.

Caso de uso : GENERACIÓN DE ETIQUETAS.

Actores : Cajero, personal de distribución.

Resumen : Describe el proceso que se sigue en el si para el Generador de etiquetas.

Flujo Normal de Eventos

Acción de los actores	Respuesta del Sistema
	1.-El Sistema debe hace conversión de los importe denominación a unidades empaques, a excepción de empaques indicados denominación general, con e de generar las etic correspondientes
	2.-El sistema debe tener opción que permita la impr de las etiquetas correspondi a los empaques depositados

Tabla A.3. Caso normal para la generación de Etiquetas. (Continúa)

	3.-El sistema debe mostrar la siguiente información: ◦ La clave y descripción la institución que empaca el efectivo (banco que deposita).
	◦ Clave y descripción de la denominación del billete en el empaque. ◦ Clave y descripción de la unidad de empaque ◦ Consecutivo (formado por tres letras y tres dígitos) ◦ Dígitos de verificación.
cajero pega las etiquetas a mpaques aceptados.	6 -Las etiquetas, ya pegadas, deben leerse para dar de alta en el sistema los empaques recibidos.
El Cajero por medio de la ra de código de barras icara su contenido.	8.-El sistema debe incrementar las etiquetas de la ventanilla de recolección en el numero de pieza equivalentes al deposito.

Tabla A.3. Caso normal para la generación de Etiquetas.

Retiros

uerdo con el análisis de los datos hemos podido determinar tos que tienen que ser implementados, entre los que podemos onar:

- Alta de solicitudes de retiro (captura).
- Procesamiento de una solicitud de retiro
- Modificación de la solicitud de retiro.
- Cancelación de la solicitud de retiro.

- Consulta de las solicitudes de retiro procesadas procesadas.

A continuación se muestra un diagrama de este proceso (Fig.A en el que se identifican los casos de uso que intervienen e captura, procesamiento, modificación, cancelación, consult impresión de la consulta de una solicitud de retiro, y en el los actores son: el vigilante, el administrador y el responsab la preparación del retiro.

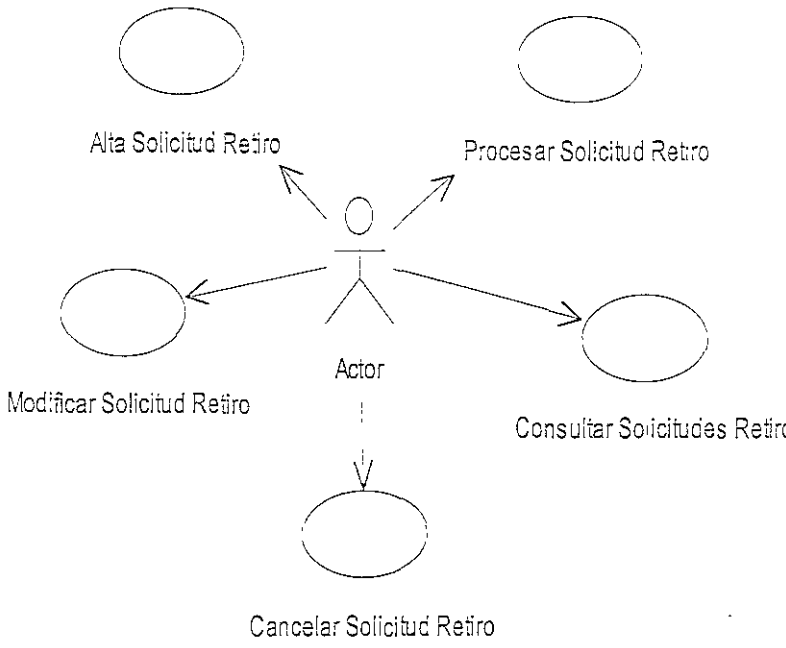


Figura A.2. Diagrama de Casos de Uso para Retiros.

A continuación se describe cada caso de uso.

Caso de Uso : ALTA DE LA SOLICITUD DE RETIRO.

Actores : Vigilante, Administrador y Responsable preparación del retiro.

Resumen : Describe el proceso que se sigue para regist el sistema una solicitud de retiro.

Normal de Eventos

n de los actores	Respuesta del Sistema
l caso de uso Captura de itud de Retiro comienza o el Administrador llega a ntanilla correspondiente.	
Selecciona la opción de rar una nueva solicitud de tro.	3. El sistema le muestra la pantalla de captura correspondiente.
Introduce los datos spondientes al retiro: • Operación (Retiro urgente o preavisado) • Folio del documento • Oficina central en la que se realiza el retiro. • Ventanilla de donde se retira el efectivo. • Banco Usuario o Caja Administrativa que retira. • Divisa. • Estado físico del efectivo. • Importe total del retiro. • Importes por denominación.	5. Despliega la Fecha en que se efectuará el retiro. Si es una solicitud de retiro preavisado la fecha será del día hábil bancario después de la fecha de recepción de la solicitud. Si el retiro es urgente, la fecha será el mismo día en que se recibe la solicitud Por cada denominación debe: ◦ Mostrar la descripción de la denominación. ◦ Calcular y mostrar el número de piezas correspondientes al importe.
lecciona la opción aceptar licitud.	7. Agrega a la información del retiro proporcionada por el Administrador, la clave del usuario que registró la solicitud de retiro.
	8. Almacena los datos de la solicitud de retiro en la base de datos

Tabla A.3 Caso normal para el alta de la Solicitud de retiro (Continúa)

	9. Limpia los campos de capt
	10. Listo para registrar datos de una nueva solicitud

Tabla A.3. Caso normal para el alta de la Solicitud de reti

Flujos Alternos

Si en el almacenamiento de la solicitud (Punto 8) el si detecta que algún importe por denominación no es múltipl importe correspondiente al empaque mínimo a retirar, debe mo un mensaje de error y permitir al usuario continuar en la ca de datos (Punto 4) para hacer las correcciones necesarias.

Si en el almacenamiento de la solicitud (Punto 8) el si detecta que la clave capturada por el estado físico no corres al billete apto, debe mostrar al usuario un mensaje de er permitirle continuar la captura de datos (Punto 4) para correg error.

Caso de Uso :MODIFICACIÓN DE LA SOLICITUD DE RETIRO.

Actores :Vigilante, Administrador y Responsable preparación del retiro.

Resumen :Describe el proceso que se sigue para modi cierta información de la solicitud de retiro antes de procesada.

Flujo Normal de Eventos

Acción de los actores	Respuesta del Sistema
1. Inicia cuando el usuario consulta la solicitud de retiro en proceso de preparación.	2. El sistema le muestr pantalla de cor correspondiente.
3. Selecciona la solicitud que desea modificar.	

Tabla A.3. Caso normal modificar la Solicitud de retiro.(Cont

ge la opción de Modificar.	5. Muestra una pantalla con los datos correspondientes a la solicitud de retiro seleccionada, una opción para aceptar los cambios y una para salir de la pantalla actual sin realizar cambios.
aliza los cambios necesarios s datos de la solicitud y iona la opción de aceptar mbios.	7. Muestra un mensaje indicando que se va a modificar la solicitud de retiro.
ecciona la opción aceptar.	9. Agrega a la información del retiro la clave del usuario que modificó la solicitud.
	10. Almacena en la base de datos la solicitud de retiro modificada.
	11. Regresa a la pantalla con la lista de solicitudes de retiro sin procesar.
	12. Está listo para que el usuario seleccione otra solicitud de retiro.

Tabla A.3. Caso normal modificar la Solicitud de retiro.

Alternos

la captura de datos (Punto 6) el usuario elige la opción ar, el sistema regresa a la ventana con la lista de tudes de retiro sin procesar (Punto 11), y el caso de uso a con el sistema listo para que el usuario seleccione otra tud de retiro (punto 12).

es de Empaque Inválidas. Si en el almacenamiento de la tud (Punto 10) el sistema detecta que algún importe por nación no es múltiplo del importe correspondiente al empaque a retirar, debe mostrar un mensaje de error y permitir al o continuar la captura de datos (Punto 6) para hacer las ciones necesarias.

detecta que la clave capturada por el estado físico no corres al billete apto, debe mostrar al usuario un mensaje de er permitirle continuar la captura de datos (Punto 6) para correg dato.

Caso de Uso :CANCELACIÓN DE LA SOLICITUD DE RETIRO.

Actores :Vigilante y Administrador.

Resumen :Describe el proceso que se sigue para cancela solicitud de retiro antes de ser procesada. No se pueden can solicitudes ya procesadas o canceladas previamente.

Flujo Normal de Eventos

Acción de los actores:	Respuesta del Sistema:
1. Inicia cuando el usuario selecciona la opción de consulta de solicitudes de retiro en proceso de preparación.	2. El sistema le muestr pantalla de cor correspondiente.
3. Selecciona la solicitud que desea cancelar.	
4. Elige la opción de Cancelar.	5. Muestra una pantalla co datos correspondientes a solicitud de retiro seleccio una opción para realiza cancelación y una para sal la pantalla actual sin rea cambios.
6. El usuario selecciona la opción de aceptar la cancelación.	7. Muestra un mensaje indi que se va a cancelar la soli de retiro.
8. Selecciona la opción aceptar.	9. Agrega a la información retiro la clave del usuari canceló la solicitud.

Tabla A.4. Caso normal cancelación Solicitud de reti
(Continúa)

	10. Almacena en la base de datos la solicitud de retiro cancelada.
	11. Regresa a la pantalla con la lista de solicitudes de retiro sin procesar.
	12. Está listo para que el usuario seleccione otra solicitud de retiro.

Tabla A.4. Caso normal cancelación Solicitud de retiro.

s Alternos

el Punto 5 el usuario elige la opción Cancelar (salir de la lla sin realizar cambios), el sistema regresa a la ventana con sta de solicitudes de retiro sin procesar (Punto 11) y el caso o termina con el sistema listo para que el usuario seleccione solicitud de retiro (punto 12).

de Uso :PROCESAMIENTO DE LA SOLICITUD DE RETIRO.

es :Vigilante y Administrador.

en :Describe el proceso en el cual se atiende una itud de retiro para su entrega al banco usuario.

Normal de Eventos

n de los actores:	Respuesta del Sistema:
nicia cuando el usuario ciona la opción de consulta olicitudes de retiro en so de preparación.	2. El sistema le muestra la pantalla de consulta correspondiente con las solicitudes registradas hasta el momento.
lecciona la solicitud que procesar.	4. Muestra una pantalla con los datos correspondientes a la solicitud de retiro seleccionada.

Tabla A.5 Caso normal procesamiento de la solicitud de retiro (Continúa)

5. Inicia la lectura de los códigos de barra de las unidades de empaque que cubren la solicitud de retiro.	
6. Indica al sistema que ha terminado de registrar los empaques que conforman el retiro.	7. Valida que los imp registrados con la lectur códigos de barra coincida los importes por denomir registrados en la ficha retiro.
	8. Si los importes coinc cambia el estado de la soli a procesada para que pueda entregada al banco usuario.
	9. Almacena en la base de toda la información proces completa la solicitud de ret

Tabla A.5. Caso normal procesamiento de la solicitud retiro.

Flujos Alternos

Si en el Punto 7 el sistema detecta que los importes denominación no corresponden, el sistema muestra un mensaj error y solicita que se hagan los cambios pertinentes teniend regresar al punto 5.

Si en el punto 7 el sistema detecta que el código de barr tiene asociado un contenedor, muestra un mensaje de error, sol una nueva lectura (Punto 5) y descarta la anterior.

Cabe aclarar que antes de realizar el procesamiento de solicitud de retiro se deben prepara los contenedores y regi las unidades de empaque que lo conforman.

A.3. Remesas

Los casos de uso más importantes en el proceso de remesas son:

- Registro de órdenes de remesa

- Registro de salida de la remesa
- Registro de la confirmación envío de la remesa
- Registro de la recepción de la remesa

figura A.3. se muestra el diagrama de casos de uso para el o de remesa

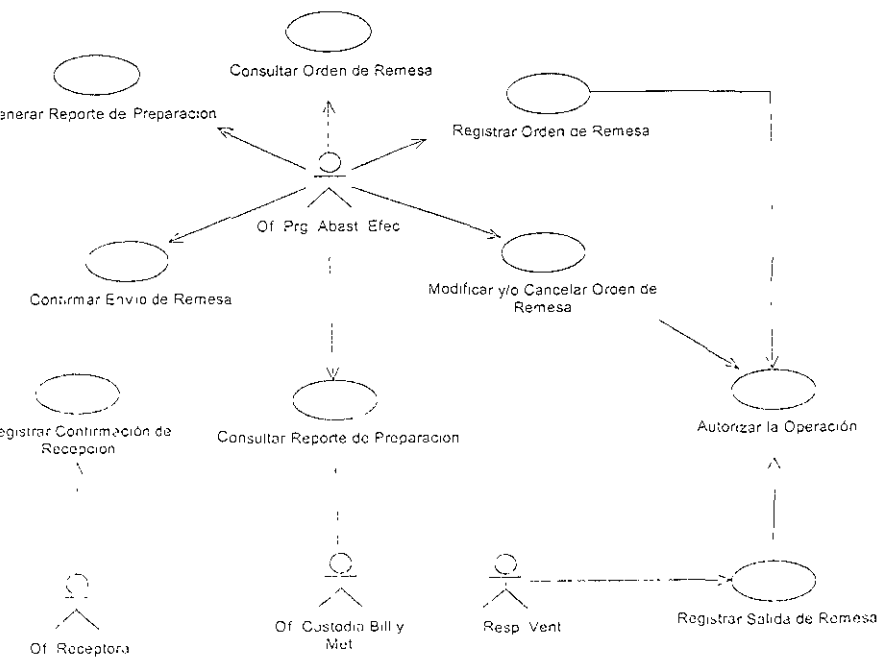


Figura A.3. Diagrama de casos de Uso de las remesas.

inuación se describen los caso de uso de remesas.

e Uso : REGISTRO DE LA ORDEN DE REMESA.

s : Responsable de la oficina que programa y abaste el vo

n A través del caso de uso se registran órdenes de

corresponsal.

Flujo normal de los eventos.

Acción de los actores	Respuesta del Sistema .
1. El caso de uso inicia cuando el usuario elige la opción para dar de alta una orden de remesa.	2. El sistema muestra la pa para el registro del Alta Orden de Remesa.
3. El usuario introduce los datos del alta: • Clave del tipo de Operación relacionada • Clave del tipo de efectivo • Clave de la institución que genera la orden • Clave de la plaza origen • Clave de la bóveda o ventanilla de donde saldrá la remesa • Fecha de embarque • Número de orden • Detalle de los importes por denominación y estado físico • Importe total de la orden • Clave del transporte	4. El sistema debe valida los importes de cada denomi sean múltiplos de la unid empaque mínima.
5. El usuario elige la opción para aceptar el alta.	6. El sistema despliega ventana de autorización.
7. El responsable de la oficina introduce clave y contraseña de autorización y elige la opción para aceptar la autorización.	8. El sistema valida que la tenga derecho de autoriza registro de órdenes.

Tabla A.6. Caso normal de registro de la orden de remesa.
(Continúa)

El sistema agrega los entes datos al Alta: • *Login* del usuario que captura el alta. • Fecha de registro del alta. • Número consecutivo para el folio del documento. • Clave que autorizó el alta.	
	10. El caso de uso termina cuando el sistema esta listo para recibir otra orden de remesa.

Tabla A.6. Caso normal de registro de la orden de remesa.

s Alternos

aso de que el servidor rechace el Alta, el sistema debe enar en la base de datos local la información capturada y r el registro como rechazado.

stema genera un número de folio temporal para el alta de la .

istema debe permitir que el usuario haga las correcciones arias al alta de orden rechazada.

aso de que la clave de autorización no tenga derechos de lo, el sistema no debe permitir que se realice la operación y mandar un aviso indicando el error.

servidor está fuera de servicio y no acepta el Alta, el ma debe almacenar la información en la base de datos local y r el registro como pendiente por enviar.

ez que el sistema detecte que el servidor está activo, deben rse y validarse todas las operaciones pendientes por enviar.

de Uso : MODIFICAR ÓRDENES DE REMESA.

es : Responsable de la oficina que programa y abastece activo, responsable de la oficina de control.

Resumen : El usuario selecciona una remesa de una lis órdenes que no han sido movilizadas, y el sistema le pe modificarla. En el momento que acepte las modificaciones reali se le solicitarán las claves autorizadas para realiza operación.

Flujo normal de los eventos

Acción de los actores	Respuesta del Sistema
1. El caso de uso inicia cuando el usuario elige la opción de órdenes de remesas del menú.	2. El sistema despliega en pantalla la lista de órdene no han sido movilizadas.
2.El usuario selecciona una orden de remesa de la lista de órdenes que no ha sido movilizadas.	
3.El usuario elige la opción para modificar la orden.	4. El sistema despliega pantalla toda la inform registrada en la orden de r y con todos los campos l para ser modificados.
5. El usuario corrige la información de cada uno de los campos necesarios e ingresa el motivo de la modificación.	
6. El usuario elige la opción para aceptar los cambios.	7. El sistema despliega pantalla de autorización.
8. Los usuarios responsables de la operación, introducen sus claves y contraseñas correspondientes.	9. El sistema verifica que autorizaciones sean válidas la operación.
	10. El sistema guarda los ca de la operación e inform usuario el resultado.

Tabla A.7. Caso normal de la modificación de la orden de reme
(Continúa)

	12. El sistema debe agregar al registro modificado el *login* del usuario que realizó los cambios y las claves de autorización correspondientes.
	13. El caso de uso termina cuando el sistema regresa a la pantalla que contiene la lista de órdenes que no han sido enviadas.

a A.7. Caso normal de la modificación de la orden de remesa.

alternos

rvidor está fuera de servicio y el sistema no puede almacenar operaciones, el sistema deberá almacenar localmente la, ión donde se hizo el registro. Una vez que el servidor esté rvicio, el sistema debe enviar la operación para que se e.

tema debe informar el resultado de la operación, y en su caso arle al usuario si la operación fue rechazada y preguntar si corregirla. En caso de que el usuario elija corregir la ión, el sistema debe regresar a la pantalla anterior. En caso el usuario elija no corregir la operación el sistema debe ar a la pantalla que contiene la lista de las órdenes no das.

e Uso : CANCELACIÓN DE LA ORDEN DE REMESA.

s : Oficina que programa y abastece efectivo.

n : En esta operación se realiza la cancelación de una y debe ser autorizada electrónicamente.

normal de los eventos

de los actores	Respuesta del Sistema
usuario elige la opción ancelar la orden.	2. El sistema despliega en pantalla toda la información registrada en la orden de remesa.

A.8. Caso normal de la cancelación de la orden de remesa. (Continúa)

	3. El sistema debe inhab todos los campos de la orde que no puedan ser modificad
4. El usuario corrobora, con base en la información desplegada, que es la orden que quiere cancelar y describe el motivo por el que se cancela la orden.	
5. El usuario elige la opción para cancelar la orden.	6. El sistema desplieg pantalla de autorización.
7. Los responsables de la oficina que programan el efectivo y el de la oficina de control, introducen sus claves y contraseñas correspondientes.	
	8. El sistema verifica q autorizaciones sean válidas la operación.
	9. El sistema cancela la o le informa al usuario resultado.
	10. El sistema debe agreg registro cancelado el *logi* usuario que canceló y las de autorización correspondi
	11. El caso de uso termina el sistema regresa a la pa que contiene la lista de ó que no han sido enviadas.

Tabla A.8. Caso normal de la cancelación de la orden de reme

Caso de Uso : CONFIRMACIÓN DEL ENVÍO DE LA REMESA.

Actores : Responsable de la oficina que programa y ab
el efectivo.

Resumen : El usuario selecciona la remesa la rem

rmar y captura la clave de la institución y la plaza destino, que posteriormente el sistema la ubique en tránsito trando las afectaciones que resulten necesarias.

normal de los eventos

n de los actores	Respuesta del Sistema
caso de uso inicia cuando suario elige la opción del na para confirmar el envío remesa.	2. El sistema despliega en una pantalla la lista de remesas por confirmar.
Usuario selecciona la Remesa firmar.	4. El Sistema muestra el detalle de la Remesa con los siguientes campo para ser capturados: Clave de la Institución destino. Clave de la plaza destino.
l usuario complementa la nación y elige la opción confirmar el envío.	6. El sistema valida los datos de la confirmación.
	7. El sistema registra la confirmación de la Remesa y se pone en tránsito el efectivo.
	8. El caso de uso termina cuando el sistema regresa a la pantalla que contiene la lista de órdenes que no han sido enviadas.

Tabla A.9. Caso normal de la confirmación de envío de la remesa.

de Uso : REGISTRO DE SALIDA DE LA REMESA.

es : Responsable de la ventanilla donde saldrá el vo.

en El usuario selecciona la remesa a entregar y a los códigos de las unidades de empaque con la información de [illegible] información [illegible] sistema validará y solicitará la [illegible]

Flujo normal de los eventos.

Acción de los actores	Respuesta del Sistema
1. El caso de uso inicia cuando el usuario elige la opción para dar de alta una salida de remesa.	2. El sistema muestra la par una lista de las remesas entregar
3. El Usuario elige la Remesa que va a atender	4. El Sistema muestra al Us los detalles de la Remesa de que confirme que seleccionando la adecuada.
5. El Usuario lee las etiquetas con la lectora de códigos de barras a fin de relacionar los códigos de las unidades de empaque con la información de la Remesa.	6. El Sistema valida siguiente: En caso de usar contenedor valida que las unidades empaque estén previa registradas en el sistema Que el importe de la Remes esta saliendo coincida co información de la orden de R seleccionada previamente
	7. El sistema despliega ventana de autorización.
8. El responsable de la ventanilla proporciona su clave y contraseña de autorización y elige la opción "aceptar".	9. El sistema valida que la tenga derecho de autoriza salida de una Remesa.
	10. El sistema Verifica que conexión con el servidor: Si hay comunicación mand información al servidor.

Tabla A.10. Caso normal de la salida de la remesa.(Continúa

	Si no hay comunicación registra localmente la operación.
	11. El caso de uso termina cuando el sistema esta listo para recibir otra salida de remesa.

Tabla A.10. Caso normal de la salida de la remesa.

ste anexo se presentó la descripción de los casos de uso espondientes a los depósitos, retiros y remesas.

Apéndice B

DIAGRAMAS DE ENTIDAD-RELACIÓN

presente anexo se muestran los diagramas de entidad relación clases, complementarios a los presentados en el capítulo V, se trata el diseño del sistema de contenedores.

iagramas de entidad relación que se muestran incluyen las que se utilizan para el control de la impresión de tas, el registro de remesas, autorizaciones de emisión y s de fabricación, traspasos, depósitos y retiros.

contenido del anexo no se describe cada uno de los diagramas e muestran, debido a que la intención es solamente la de ecer el capítulo correspondiente al diseño.

B.1. Tablas de Operación

En la figura B.1. se muestran las tablas donde se registran las operaciones del sistema de contenedores.

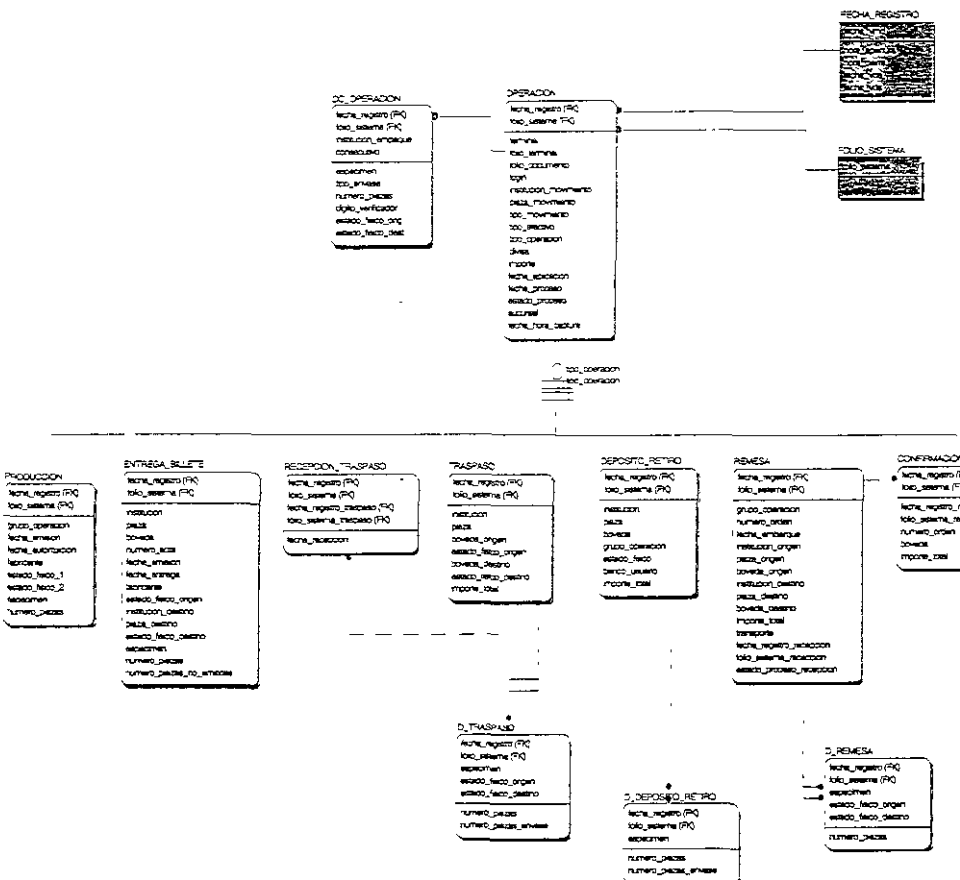
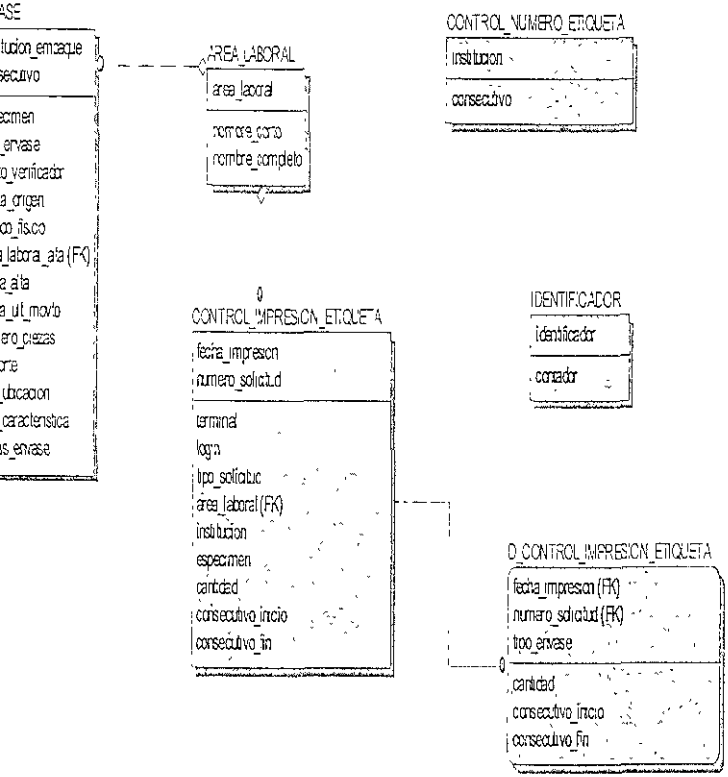


Figura B.1. Diagrama Entidad-Relación de las tablas de operación.

Tablas de Impresión de Códigos de barras

figura B.2. se muestran las tablas para el control en la sión de etiquetas con códigos de barras.



B 2. Diagrama Entidad-Relación de las tablas de impresión de código de barras.

B.3. Tablas de Remesas

En la figura B.3. se muestran las tablas para el contro
registro de remesas.

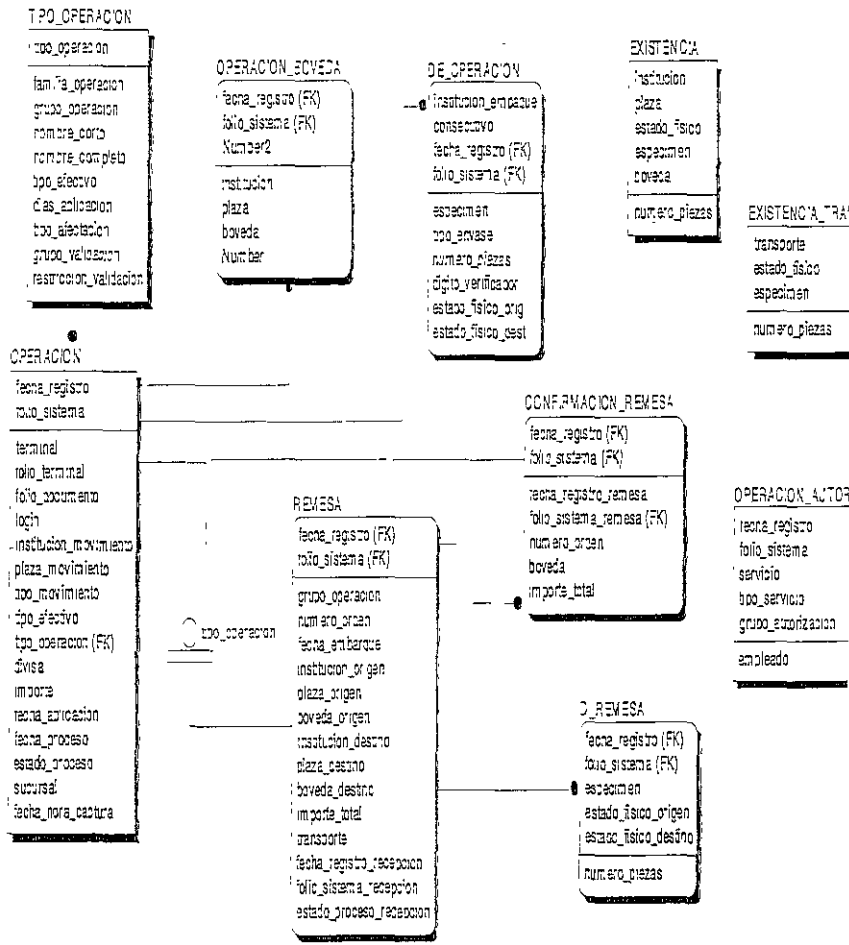


Figura B.3. Diagrama Entidad-Relación de las tablas de remes

Tablas de Autorizaciones y Órdenes

a figura B.4. se muestran las tablas para el control de las izaciones de emisión y órdenes de fabricación.

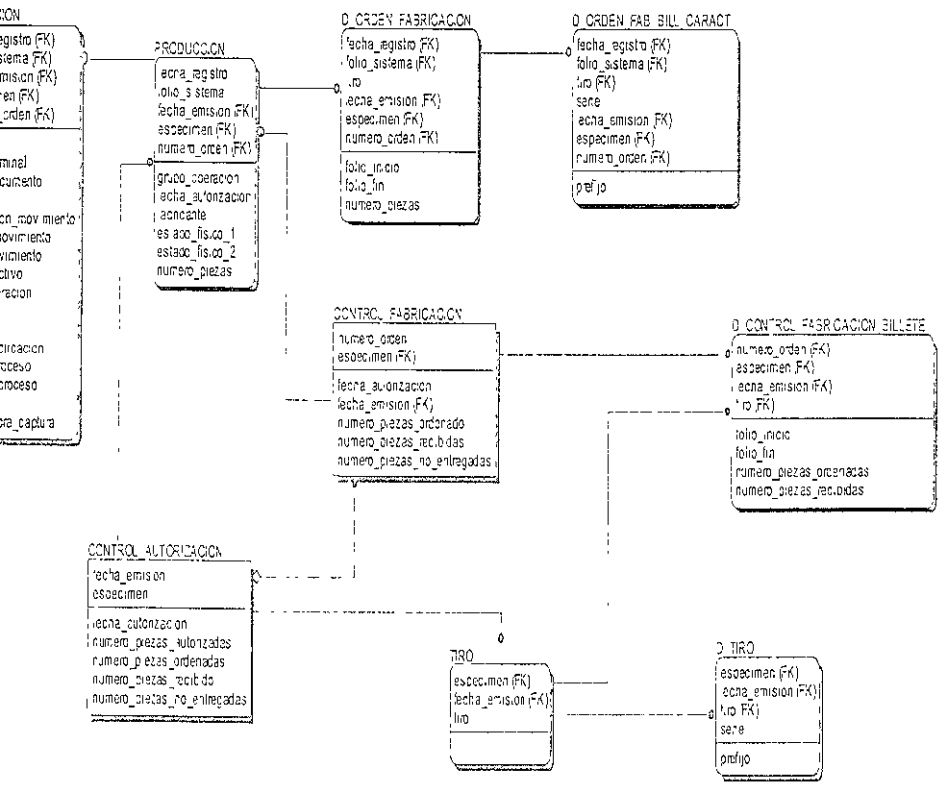


Figura B.4 Diagrama Entidad-Relación de las tablas de Autorizaciones y Órdenes.

B.5. Tablas de Depósitos y Retiros

En la figura B.5. se muestran las tablas para el control d depósitos y retiros.

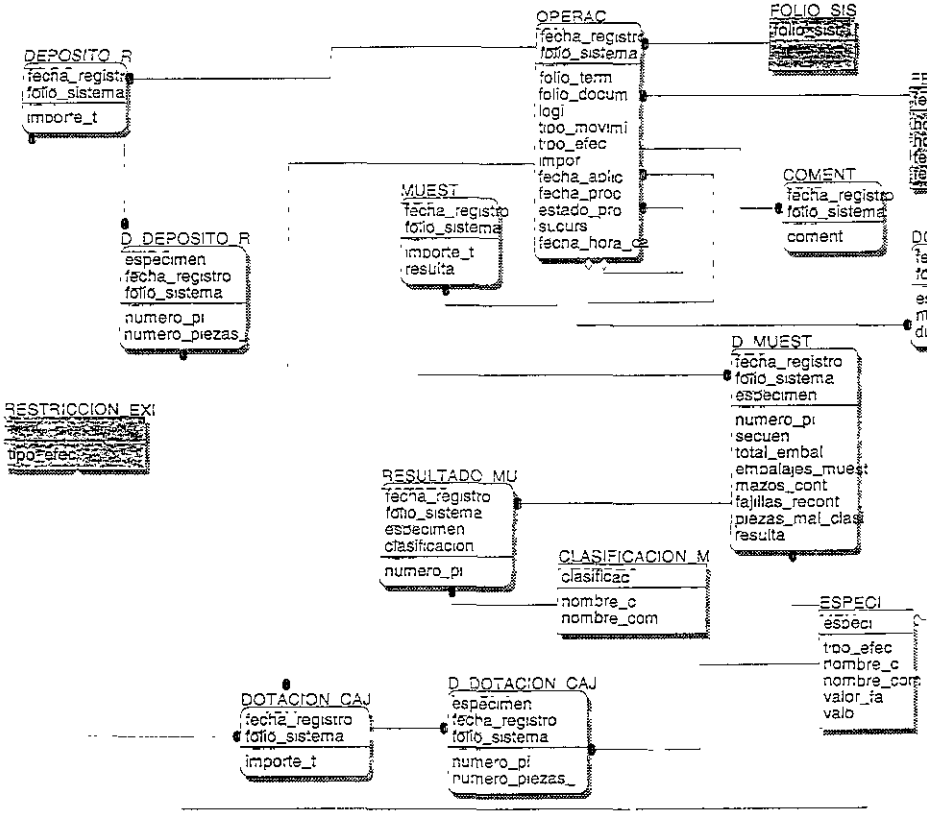
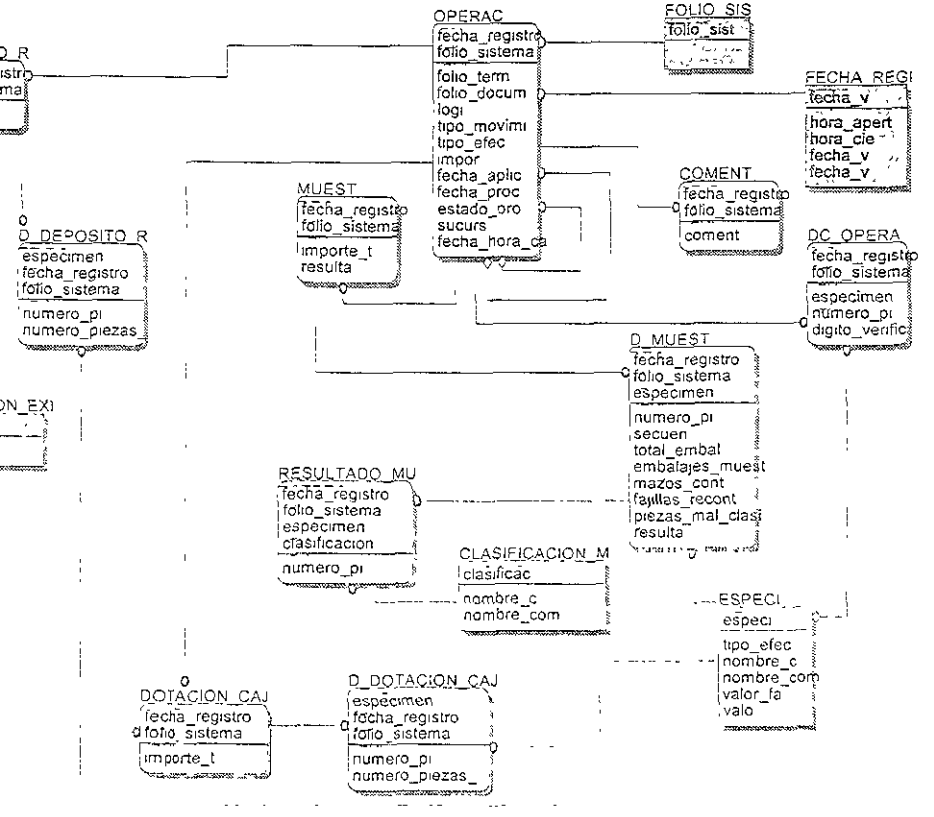


Figura B.5. Diagrama Entidad-Relación de las tablas de Depósit Retiros.

a figura B.6. se muestran las tablas para el control de los pasos.



ura B.6. Diagrama Entidad-Relación de las tablas de Traspasos.

Apéndice C

CÓDIGO DEL SISTEMA

te apéndice presentamos parte del código generado para el
ma de Contenedores que es complemento del capítulo de
ientación.

Código para el Servidor

```
e proc spsif_envio_entrega_billete
     < Parámetros >


Actua de acuerdo al @tipo_movimiento de la siguiente manera:
lta       : solo ejecuta procedimiento de inserción del nuevo registro.
ancelacion : sólo ejecuta procedimiento de cancelación del registro indicado
             en los parámetros output definidos del SP.
odificacion: -Ejecuta procedimiento de cancelación para el registro indicado en los
              parámetros de output
             -Inserta el nuevo registro de alta para la modificación correspondiente,
              con los parámetros recien recibidos por el stored procedure

e @status  int, @folio_sistema_orig int, @fol_term_orig int, @tipo_efectivo tinyint,
                @divisa char(2), @importe money, @fecha_aplicacion datetime,
                 @fecha_emision datetime, @fecha_entrega datetime,
                @boveda_orig smallint, @estado_fisico_orig smallint, @especimen smallint,
                @tiro char(6), @series varchar(50), @prefijos varchar(50),
                @millones_envase varchar(50), @numero_piezas int,
                @num_pzs_no_emi int,@mac_proc_fin int, @mac_manufactura int,
                @fabricante smallint, @afecta_elem_1 int, @afecta_elem_2 int,
                @afecta_elem_3 int, @res int, @res2 int, @numero_orden char(8), @max int,
                @i int,  @millon_cadena  varchar(30), @elemento varchar(30),
                @millon_salida varchar(30)

 @res=count(empleado)from EMPLEADO_GRUPO_AUTORIZACION
 servicio = @tipo_operacion
 (empleado = @empleado1
 empleado  = @empleado2)
```

```
if (@res != 2) return 30079        /** CLAVE DE ACCESO INVALIDA**/
   select @res2=count(contrasena_autorizacion)
     from EMPLEADO
   where empleado=@empleado1
     and contrasena_autorizacion =@contrasena1
       or empleado=@empleado2
     and contrasena_autorizacion =@contrasena2

   if (@res2 != 2) return 40321      /** CONSTRASEÑA INVALIDA **/
   select @folio_sistema_orig = @folio_sistema
   select @fol_term_orig      = @fol_term_alta
   select @fecha_aplicacion=fecha_vmd
    from FECHA_REGISTRO

   select @fabricante = fabricante
    from FABRICANTE
  where nombre_corto = 'FAB. BILLETES'

   /* Verifica que exista el tipo de movimiento indicado para la operación */

  if not exists (select * from REGLA  where regla     = 'TIPO_MOVIMIENTO'
                 and valor_regla = @tipo_movto) return 40014  /* Tipo de Movimiento Invali

/** Crea tablas temporales para obtener el millón y contenedores maculatura **/

create table #I_cont_padre (institucion_empaque int not null, consecutivo char(6) not null,
                            tipo_envase char(2) not null, digito_verificador char(1) not null,
                            institucion_empaque_padre int not null,
                            consecutivo_padre char(6) not null, millones varchar(50)  null,
                            numero_piezas  int not null, status_envase  smallint not null)

create table #millones_envase (millones varchar(50), indice numeric(3,0) identity)

create table #elem_millon (millon  char(6),  indice numeric(3,0) identity)

create table #millon_distinto (millon varchar(20),  indice numeric(3,0) identity)

/** Obtiene el millón contenido en cada envase**/
  insert into #millones_envase
  select millones_envase
    from DE_CONS_ENVASE_ENTREGA
  where fecha_consulta = @fecha_consulta
    and id_consulta = @id_consulta

 select @i=1,   @max = MAX(indice)
  from #millones_envase

/** Pone en la tabla uno por uno lo millones **/
    while (@i <= @max)
      begin
        select @millon_cadena =millones
        from #millones_envase
```

```
 where indice = @i

while (@millon_cadena is not NULL)
 begin
 exec spsifi_desconcatena_x_coma @millon_cadena output, @elemento output
 insert into  #elem_millon (millon)
 select @elemento where @elemento != ''
 end
  select @i=@i+1
nd

nsert into #millon_distinto
elect distinct millon
 from #elem_millon

elect @i=1, @max =MAX(indice), @millon_salida = '' from #millon_distinto

while @i<=@max
begin
   if (@i=1)
      select @millon_salida=millon
       from #millon_distinto
       where indice =@i
  else
      select @millon_salida=@millon_salida+','+millon
      from #millon_distinto
     where indice =@i

  select @i=@i+1

end

se obtienen de la tabla CONS_ENVASE_ENTREGA especimen, fecha_emision, tiro,
millones_solicitados, series, prefijos **/

t  @especimen = especimen, @fecha_emision = fecha_emision, @tiro = tiro,
   @millones_envase = @millon_salida, @series = series, @prefijos = prefijos
n CONS_ENVASE_ENTREGA co, DE_CONS_ENVASE_ENTREGA dc
e @fecha_consulta = co.fecha_consulta
 @id_consulta = co.id_consulta
 co.fecha_consulta = dc.fecha_consulta
 co.id_consulta = dc.id_consulta

se obtiene el numero_orden a la cual corresponde esta entrega contenido en la tabla
ONTROL_FABRICACION_BILLETE **/
ct  @numero_orden =df.numero_orden
n CONTROL_FABRICACION_BILLETE cf, D_CONTROL_FABRICACION_BILLETE df
e cf.especimen = @especimen
 cf.fecha_emision = @fecha_emision
 df.especimen = cf.especimen
 df.fecha_emision = cf.fecha_emision
 df.tiro = @tiro
```

```
/** se obtienen de la tabla ESPECIMEN el tipo_efectivo y la divisa **/
 select @tipo_efectivo = tipo_efectivo, @divisa = divisa
   from ESPECIMEN
  where especimen = @especimen

/** Obtiene la fecha de operación actual del sistema **/
 select @fecha_entrega =fecha_vmd
   from FECHA_REGISTRO

/** Se obtiene de la tabla CONSTANTE los valores para estado_fisico_orig y boveda
**/
 select @estado_fisico_orig =convert(smallint, valor)
   from CONSTANTE
  where constante ='EDO_FIS_BILL_ORDENADO_P/IMP'

if (@estado_fisico_orig is NULL) return 30230  /* CONSTANTE no definida. Comuni
                                                  con la  administración del banco */

 select @boveda_orig =convert(smallint, valor)
   from CONSTANTE
  where constante ='BOVEDA_P/ENTREGA_CAJA'

if (@boveda_orig is NULL) return 30230 /* CONSTANTE no definida. Comuniquese con la
                                          administración del banco */

/** se obtiene  @numero_piezas, @importe, @mac_proc_fin y @mac_manufactura
if not exists (select * from DE_CONS_ENVASE_ENTREGA
               where fecha_consulta = @fecha_consulta
               and id_consulta = @id_consulta) return 40319 /* La consulta indicada no existe

 select @numero_piezas=sum(e.numero_piezas)
   from ENVASE e, DE_CONS_ENVASE_ENTREGA d
  where d.fecha_consulta = @fecha_consulta
    and d.id_consulta = @id_consulta
    and e.institucion_empaque = d.institucion_empaque
    and e.consecutivo = d.consecutivo

if (@numero_piezas is NULL) return 40320 /* No fue posible calcular el número de piezas

 select @importe =@numero_piezas * convert(money, e.valor)
   from ESPECIMEN e
  where @especimen =e.especimen

if (@importe is NULL) return 40317  /* Error en importe de acuerdo al número de piezas
                                       especimen */
/** Si existe MACULATURA se agrega el total contemplando maculatura de manufa
**  de procesos finales **/
if (@ent_maculatura =1) /* Verifica si se incluira la maculatura o no, 1 incluye, 0 excluy
  begin
  exec @status = spsifi_calcula_mac_entrega @fecha_emision, @especimen, @tiro, @mac_proc
                                            output, @mac_manufactura output
  if (@status != 0) return @status
  end
```

```
(@mac_proc_fin is null) select @mac_proc_fin = 0
(@mac_manufactura is null) select @mac_manufactura = 0

gin tran envio_entrega_billete
(@tipo_movto in ('C','M'))  /* verifica si se hará cancelación */
gin

 Bloquea el registro seleccionado en OPERACION y si es el caso, en
  COMUNICACION_SIAC, ya que no todas las operaciones se envian a SIAC. Si es el
  caso, valida que la operación pueda ser cancelada o modificada. */

/e tran trans_bloquea_operacion

ec @status = spsifi_bloquea_operacion @fecha_recepcion, @folio_sistema, @tipo_movto

@status != 0)
begin
ollback envio_entrega_billete
elect @folio_sistema =NULL
eturn @status
nd

/e tran cancela_ent_bill

ec @status =spsifi_cancela_ent_bill  < Parámetros >

@status !=0)
egin
ollback envio_entrega_billete
elect @folio_sistema =NULL
eturn @status
nd

/* fin if (@tipo_movto in ('C','M')) */

ipo_movto in ('A','M'))  /* Si hay alta que insertar agrega nuevo registro */
in
ect @fecha_recepcion =NULL, @folio_sistema = NULL

e tran inserta_ent_bill

c @status =spsifi_inserta_ent_bill  < Parámetros >

@status !=0)
egin
ollback envio_entrega_billete
elect @folio_sistema =NULL
eturn @status
nd
* fin if (@tipo_movto in ('A','M')) */

t into OPERACION_EMPLEADO_AUTORIZA
```

```
    select < Parámetros >
     from EMPLEADO_GRUPO_AUTORIZACION e
    where e.empleado = @empleado1
          or e.empleado = @empleado2
      and e.servicio = @tipo_operacion

  select @status =@@error
    if (@status !=0)
      begin
      roliback envio_entrega_billete
      return @status
      end

commit tran envio_entrega_billete
drop table #l_cont_padre

return 0
go

grant execute on spsif_envio_entrega_billete to monitor_sifca
go
grant execute on spsif_envio_entrega_billete to operador_sifca
go

-------------------------------------------------------------------
/** Crea tablas temporales que deben haberse creado previamente para poder ejecu
     este sp **/
create table #elem_caract (elemento varchar(15), reg numeric(3,0) identity)
go
create table #elem_folio (elemento1 varchar(25), reg numeric(3,0) identity)
go
create table #elem_pzs_canc (elemento2 varchar(25), reg numeric(3,0) identity)
go
create table #d_caract (tiro char(6), serie varchar(3), prefijo char(1))
go
create table #d_folio (tiro char(6), folio_inicio int, folio_fin int, numero_piezas int)
go
create table #d_pzs_canc (numero_orden varchar(8) NULL, tiro char(6), serie varchar(3),
                          prefijo char(1), millon_inicio int, millon_fin  int, piezas_ordenadas int,
                          piezas_canceladas int, indice numeric(3,0) identity)
go
create table #t_orden1 (numero_orden varchar(8),tiro char(6), piezas_canceladas int)
go
create table #t_orden2 (numero_orden varchar(8), piezas_canceladas int)
go

use bdContenedores
go
drop proc spsifi_inserta_produccion
go
create proc spsifi_inserta_produccion <Parametros>
as
```

```
are @status int, @gpo_valida  tinyint, @login_usuario  varchar(30), @t_afectacion char(3),
                  @gpo_operacion smallint, @num_autoriza varchar(15), @restric_valida char(1),
                  @i int, @num_columnas int, @pzs_sin_ord int, @num_pzs_aut int

ct @login_usuario  =suser_name()

ct @fecha_recepcion =fecha_vmd
m FECHA_REGISTRO


 Validación de datos de Operación **/
 @status =spsifi_val_env_operacion  @fecha_recepcion,  @terminal, @folio_term, @folio_docto,
                                    @login_usuario, @institucion, @plaza, @tipo_movto,
                                    @tipo_efectivo, @tipo_operacion,  @divisa, @importe,
                                    @f_aplicacion, @t_afectacion  output,
                                    @gpo_operacion output, @gpo_valida output,
                                    @restric_valida output
status !=0)  return @status
 Verifica si el documento tiene alguna autorización en AUTORIZACION **/
 @status =spsifi_verifica_autorizacion  @folio_docto, @tipo_operacion, @importe, 'A',
                                        @num_autoriza output
f (@status not in (0,40173))  return @status

 Comienzan las actualizaciones **/
 Asignamos el folio del sistema a la operación **/
 tran trFOLIO_SISTEMA

ate FOLIO_SISTEMA
t folio_sistema =folio_sistema +1

ct @status =@@error
(@status !=0)
begin
ollback inserta_produccion
return @status
end

t @folio_sistema =folio_sistema
m FOLIO_SISTEMA

folio_sistema is NULL)
gin
lback inserta_produccion
urn 30226
d

 Se inserta la información general de la operación **/
 tran trOPERACION

t into OPERACION
s (parametros)

t @@error
```

```
    if (@status !=0)
      begin
      rollback inserta_produccion
      select @folio_sistema =NULL
      return @status
      end

  /** Se inserta el comentario de la operación **/
  if (@comentario is not NULL) and (@comentario !=' ')
    begin
    save tran trCOMENTARIO

    insert into COMENTARIO values (@fecha_recepcion,  @folio_sistema, @comentario)

    select @status =@@error
      if (@status !=0)
        begin
        rollback inserta_produccion
        select @folio_sistema =NULL
        return @status
        end
    end

  /** Insertamos el Encabezado **/
  save tran trPRODUCCION

  insert into PRODUCCION
  values (parametros))

  select @status =@@error
    if (@status !=0)
      begin
      rollback inserta_produccion
      select @folio_sistema =NULL
      return @status
      end

  if (@gpo_operacion = 1 and @det_pzs_canc1 =NULL)
    begin

    exec @status =spsifi_obt_detalle_bill  @detserp1,  @detserp2, @detserp3, @detfolio1, @detfoli
      if (@status !=0)  return @status

  /** Insertar la Orden de Fabricación **/
    insert into ORDEN_FAB_BILL_CARACT
    select < Parámetros >
      from #d_folio

    select @status =@@error
      if (@status !=0)
        begin
        rollback inserta_produccion
        select @folio_sistema =NULL
```

```
 return @status
 end

* Insertar las características de Billete **/
sert into D_ORDEN_FAB_BILL_CARACT
lect < Parámetros >
om #d_caract

lect @status =@@error
f (@status !=0)
 begin
 rollback inserta_produccion
 select @folio_sistema =NULL
 return @status
 end

sert into ORDEN_FAB_BILL_FIRMA
lect < Parámteros >
m CATALOGO_FIRMA_BILLETE

lect @status =@@error
f (@status !=0)
 begin
 rollback inserta_produccion
 select @folio_sistema =NULL
 return @status
 end

ve tran trTIRO
ert into TIRO
 select < Parámteros >
   from #d_folio

lect @status =@@error
f (@status !=0)
 begin
 rollback inserta_produccion
 return @status
 end


e tran trD_TIRO
rt into D_TIRO
lect < Parametros >
om #d_caract

lect @status =@@error
f (@status !=0)
 begin
 rollback inserta_produccion
 return @status
 end
```

```
/** Inserta datos para llevar el control de la fabricación **/
 insert into CONTROL_FABRICACION_BILLETE
  select < Parámteros >

  select @status =@@error
    if (@status !=0)
      begin
      rollback inserta_produccion
      return 40390 /* Número de Orden de Fabricación indicado ya existe */
      end

 insert into D_CONTROL_FABRICACION_BILLETE
 select < Parámetros >
  from #d_folio

  select @status =@@error
    if (@status !=0)
      begin
      rollback inserta_produccion
      return @status
      end

 update CONTROL_AUTORIZACION_BILLETE set < Parámtros >
  where fecha_emision = @fecha_emision
    and  especimen     = @especimen

 select @status =@@error
    if (@status !=0)
      begin
      rollback inserta_produccion
      return @status
      end

 end /** (if @gpo_operacion = 1 and @det_pzs_canc1 =NULL) **/

/** Para la cancelación de piezas **/
if (@gpo_operacion = 1 and  @det_pzs_canc1 !=NULL)
   begin

   exec @status =spsifi_obt_detpzs_canceladas  @det_pzs_canc1, @det_pzs_canc2, @det_pzs_
    if (@status !=0)  return @status

     update #d_pzs_canc set numero_orden = d.numero_orden
       from D_CONTROL_FABRICACION_BILLETE d, #d_pzs_canc pz
      where d.especimen = @especimen
        and d.fecha_emision = @fecha_emision
        and d.tiro = pz.tiro

/** seleccionó el mayor índice **/

  select @i = MAX(indice) from #d_pzs_canc
```

```
ile (@i > 0 )
 begin
 if (select piezas_ordenadas from #d_pzs_canc where indice = @i) = 0
   begin
   delete from D_CONTROL_FABRICACION_BILLETE
   from D_CONTROL_FABRICACION_BILLETE cb, #d_pzs_canc pz
   where cb.numero_orden = pz.numero_orden
     and cb.especimen = @especimen
       and cb.fecha_emision = @fecha_emision
       and cb.tiro                = pz.tiro
       and pz.indice              = @i

  select @status =@@error
   if (@status !=0)
     begin
     rollback inserta_produccion
     return @status
     end

al momento de borrar el registro en la tabla TIRO se eliminan  características en
D_TIRO siempre que el número de piezas ordenadas sea =0 **/
  save tran trTIRO
  delete TIRO
    from TIRO t, #d_pzs_canc pz
   where t.especimen =@especimen
     and t.fecha_emision =@fecha_emision
     and t.tiro = pz.tiro
     and pz.indice = @i

 select @status =@@error
   if (@status !=0)
     begin
     rollback inserta_produccion
     return @status
     end
nd /** (if select piezas_ordenadas =0) **/

'select piezas_ordenadas
  from #d_pzs_canc
   where indice = @i) != 0

egin
update D_CONTROL_FABRICACION_BILLETE < Parámteros >
from D_CONTROL_FABRICACION_BILLETE cd, #d_pzs_canc pz
  where cd.numero_orden = pz.numero_orden
     and  cd.especimen = @especimen
     and  cd.fecha_emision = @fecha_emision
     and  cd.tiro = pz.tiro
     and pz.indice = @i

elect @status = @@error
 if (@status !  0)
   begin
```

```
        rollback inserta_produccion
        return @status
        end
    end /* if (select piezas_ordenadas != 0) **/

select @i=@i - 1
end /** (while) **/


insert into #t_orden1
select distinct numero_orden,tiro, piezas_canceladas
          from #d_pzs_canc

insert into #t_orden2
 select numero_orden, piezas_canceladas=sum(piezas_canceladas)
  from #t_orden1
  group by numero_orden

update CONTROL_FABRICACION_BILLETE
    set < Parámetros >
  from CONTROL_FABRICACION_BILLETE a, #t_orden2 t
 where a.especimen = @especimen
   and a.numero_orden = t.numero_orden

select @status =@@error
    if (@status !=0)
      begin
      rollback inserta_produccion
      return @status
      end

update CONTROL_AUTORIZACION_BILLETE < Parámetros >
   from CONTROL_AUTORIZACION_BILLETE
 where  fecha_emision = @fecha_emision
    and especimen = @especimen

select @status =@@error
    if (@status !=0)
      begin
      rollback inserta_produccion
      return @status
      end

select  @num_columnas = columnas_hoja
  from ESPECIMEN_BILLETE
where especimen = @especimen

insert into ORDEN_FAB_BILL_CARACT
  select distinct < Parámetros >
            from #d_pzs_canc pz
           where pz.millon_fin <> 1000000

select @status =@@error
```

```
f (@status !=0)
 begin
 rollback inserta_produccion
 return @status
 end

sert into ORDEN_FAB_BILL_CARACT
select distinct < Parámetros >
from #d_pzs_canc pz
where pz.millon_fin = 1000000

lect @status =@@error
(@status !=0)
 begin
 rollback inserta_produccion
 return @status
 end

sert into D_ORDEN_FAB_BILL_CARACT
elect < Parámetros >
om #d_pzs_canc

lect @status =@@error
f (@status !=0)
 begin
 rollback inserta_produccion
 return @status
 end

ert into ORDEN_FAB_BILL_FIRMA
lect < Parámetros >
m CATALOGO_FIRMA_BILLETE

lect @status =@@error
(@status !=0)
 begin
 rollback inserta_produccion
 return @status
 end
** (if @gpo_operacion = 1 and @det_pzs_canc1 =NULL) **/

po_operacion = 32
in

* Alta cancelación de piezas Autorización **/
xists (select fecha_emision from CONTROL_AUTORIZACION_BILLETE
        where fecha_emision =@fecha_emision
          and especimen =@especimen)
 begin
 select @pzs_sin_ord=(numero_piezas_autorizadas - numero_piezas_ordenadas)
   from CONTROL_AUTORIZACION_BILLETE
  where fecha_emision =@fecha_emision
    and especimen = @especimen
```

```
        if (@num_piezas <= @pzs_sin_ord)
          begin
          select @num_pzs_aut = (numero_piezas_autorizadas - @num_piezas)
            from CONTROL_AUTORIZACION_BILLETE
           where fecha_emision = @fecha_emision
             and especimen = @especimen

          update CONTROL_AUTORIZACION_BILLETE < Parámetros >
           where fecha_emision = @fecha_emision
             and especimen = @especimen

          select @status =@@error
           if (@status !=0)
            begin
            rollback inserta_produccion
            return @status
            end

          end /** if (@num_piezas <= @pzs_sin_ord) **/
        else

          return 40419 /** Número de piezas a cancelar mayor que el número de piezas sin ordenar
      end /** if exists **/
    else
     begin
      /** Alta Autorización **/
     insert into CONTROL_AUTORIZACION_BILLETE
         select < Parámetros >

      select @status =@@error
       if (@status !=0)
        begin
        rollback inserta_produccion
        return @status
        end

      end /** else exists **/
    end /** (if @gpo_operacion =32) **/


    /** Si hay una autorización para el documento se lo asigna **/
      if (@num_autoriza is not NULL)
        begin
        exec @status =spsifi_asigna_autorizacion  @fecha_recepcion, @folio_sistema, @num_autoriz
          if (@status !=0)
            begin
            rollback inserta_produccion
            select @folio_sistema =NULL
            return @status
            end
      end
    return 0
```

```
p table #elem_caract

p table #d_caract

p table #elem_folio

p table #d_folio

p table #elem_pzs_canc

p table #d_pzs_canc

p table #t_orden1

p table #t_orden2


bdcontenedoresgo
p trigger ti_ENTREGA_BILLETE

te trigger ti_ENTREGA_BILLETE on ENTREGA_BILLETE
nsert

n
are @status int, @cte_macula varchar(30),   @nulo tinyint, @renglones int,
                 @cte_emitido  varchar(30),  @nulo_plaza  char(5), @factor_origen money,
                 @valor_cte  varchar(30), @t_sitio  tinyint, @factor_destino  money,
                 @error int, @maculatura smallint, @relleno money,  @texto varchar(80),
                 @emitido smallint, @pzas_afectadas  money,  @nuevo_estado char(1),
                 @nombre_tabla  varchar(30)

are @inst_movto int, @plaza_movto  char(5), @tipo_movto char(1), @tipo_efectivo tinyint,
    @tipo_operacion  int, @divisa char(2), @importe_movto money, @f_aplicacion datetime,
     @estado_proceso char(1)

are @fecha_registro  datetime, @divisa_esp char(2), @t_efec_esp tinyint, @folio_sistema int,
    @importe money, @folio_documento int, @valor_especimen money

are @numero_acta varchar (20), @inst_destino int, @numero_orden  varchar(8),
    @fecha_emision datetime, @plaza_destino  char (5),  @especimen smallint,
    @fecha_entrega datetime, @boveda_destino  smallint, @numero_piezas money,
    @fabricante smallint, @edo_fis_destino smallint, @mac_proc_fin  money,
    @mac_manufactura money, @edo_fis_origen  smallint,  @tiro char(6), @series varchar(50),
    @prefijos varchar(50), @millones varchar(50), @numero_autorizacion varchar(15),
    @pzs_no_emitidas money

Se verifica que solamente se haya escrito un renglon */
t @renglones =@@rowcount
renglones !=1)
gin
ll back TRENTREGA_BILLETE
```

```
   raiserror 30211 "Solamente se permite insertar un renglon a la vez"
   return
   end

select @nulo = NULL,  @cte_macula = 'EF_MACULATURA', @nulo_plaza = NULL,
       @cte_emitido ='EF_B_EMITIDO', @nombre_tabla ='ENTREGA_BILLETE'

/** Se obtienen de la tabla de constantes EF_MACULATURA y EF_B_EMITIDO **/
exec @status =spsci_da_valor_constante  @cte_macula, @valor_cte output
    if (@status !=0)
       begin
       rollback trENTREGA_BILLETE
       raiserror @status "Error al obtener Estado Fisico de Maculatura"
       return
       end

select @maculatura =convert(smallint, @valor_cte)

exec @status =spsci_da_valor_constante  @cte_emitido, @valor_cte output
    if (@status !=0)
       begin
       rollback trENTREGA_BILLETE
       raiserror @status "Error al obtener Estado Fisico de Emitido"
       return
       end

select @emitido =convert(smallint, @valor_cte)

select @fecha_registro = fecha_registro, @folio_sistema = folio_sistema,
       @numero_acta = numero_acta, @fecha_emision = fecha_emision,
       @numero_orden = numero_orden, @fecha_entrega = @fecha_entrega,
       @fabricante = fabricante, @edo_fis_origen = estado_fisico_origen,
       @inst_destino = institucion_destino,  @plaza_destino = plaza_destino,
       @boveda_destino = boveda_destino, @edo_fis_destino = estado_fisico_destino,
       @especimen = especimen, @tiro = tiro, @series = series, @prefijos = prefijos,
       @millones = millones, @numero_piezas = numero_piezas, @mac_proc_fin =mac_proc_fin
       @mac_manufactura =mac_manufactura
 from inserted

select @pzs_no_emitidas=@mac_proc_fin+   @mac_manufactura
select @pzas_afectadas  =@numero_piezas +@pzs_no_emitidas

exec @status =spsci_da_p_procesar_OPERACION  @fecha_registro, @folio_sistema,
                        @inst_movto output,  @plaza_movto output, @tipo_movto output,
                        @tipo_efectivo output, @tipo_operacion  output,  @divisa output,
                        @importe_movto  output,  @f_aplicacion output, @estado_proceso  outpu
                        @folio_documento output
    if (@status !=0)
       begin
       rollback trENTREGA_BILLETE
       raiserror @status "No existe informacion de esta transaccion en OPERACION"
       return
       end
```

```
@estado_proceso !='I')
egin
llback trENTREGA_BILLETE
iserror 30064 "La transaccion ya habia sido procesada"
turn
nd


/** Verifica si el documento tiene alguna autorización en AUTORIZACION **/
t @status = spsci_verifica_autorizacion @folio_documento, @tipo_operacion, @importe_movto,
                                        @tipo_movto, @numero_autorizacion output
f (@status not in (0,40173))
 begin
 rollback trENTREGA_BILLETE
 raiserror @status "Error al verificar si el documento tiene AUTORIZACION"
 return
 end

@inst_movto !=@inst_destino) or (@plaza_movto !=@plaza_destino))
egin
llback trENTREGA_BILLETE
iserror 40104 "Sitio del Movimiento distinto al Sitio de la Operacion"
turn
nd

t @status =spsifi_val_familia_operac  @tipo_operacion,  @nombre_tabla
f (@status !=0)
 begin
 rollback trENTREGA_BILLETE
 raiserror @status "Esta operacion no pertenece a esta Familia"
 return
 end

t @t_efec_esp =tipo_efectivo, @divisa_esp =divisa, @valor_especimen = valor
n ESPECIMEN
e especimen =@especimen

t_efec_esp is NULL)
gin
back trENTREGA_BILLETE
serror 40036 "ESPECIMEN Inexistente"
urn
f

ipo_efectivo !=@t_efec_esp)
in
back trENTREGA_BILLETE
error 40013 "Tipo Operacion Definido para otro Tipo Efectivo"
rn
f

ivisa_esp !=@divisa)
```

```
    begin
    rollback trENTREGA_BILLETE
    raiserror 40054 "Divisas distintas a la divisa de la operacion"
    return
    end

if not exists (select * from FABRICANTE where fabricante =@fabricante)
    begin
    rollback trENTREGA_BILLETE
    raiserror 40100 "FABRICANTE Inexistente"
    return
    end

/** Se obtiene el tipo de sitio del movimiento y el tipo de sitio del destino **/
exec @status =spsci_da_tipo_sitio   @inst_destino, @plaza_destino, @t_sitio  output
    if (@status !=0)
        begin
        rollback trENTREGA_BILLETE
        raiserror @status "Error al obtener Tipo de Sitio"
        return
        end

exec @status =spsci_valido_en_RESTRICCION  @inst_destino,   @plaza_destino, @boveda_destin
                    @edo_fis_destino, @tipo_efectivo, @numero_autorizacion
    if (@status !=0)
        begin
        rollback trENTREGA_BILLETE
        raiserror @status "SITIO-BOVEDA-ESTADO FISICO Destino Inexistente"
        return
        end

/** Se valida que el movimiento sea valido segun el tipo de operación, es decir, se va
** que con el tipo de operación se permita modificar las existencias en producción. *
exec @status =spsci_movimiento_valido  @tipo_operacion, @nulo,  @nulo, @nulo_plaza, @nulo,
                    @edo_fis_origen, @t_sitio, @inst_destino, @plaza_destino, @boveda_destin
                    @edo_fis_destino, @numero_autorizacion
    if (@status !=0)
        begin
        rollback trENTREGA_BILLETE
        raiserror @status "Movimiento invalido por VALIDACION_OPERACION"
        return
        end

exec @status =spsci_da_tipo_afectacion   @tipo_operacion, @tipo_movto, @factor_origen output,
                    @factor_destino output, @relleno output
    if (@status !=0)
        begin
        rollback trENTREGA_BILLETE
        raiserror @status "Error al obtener el tipo de Afectacion"
        return
        end

/** Verifica registros de existencia **/
```

```
ot exists (select * from EXISTENCIA
           where institucion = @inst_destino
             and plaza = @plaza_destino
             and boveda = @boveda_destino
             and estado_fisico = @edo_fis_destino
             and especimen= @especimen)
egin
ave tran trEXISTENCIA

sert into EXISTENCIA
  values (@inst_destino, @plaza_destino, @boveda_destino, @edo_fis_destino, @especimen, 0)

elect @status = @@error
 if (@status !=0)
   begin
   rollback trENTREGA_BILLETE
   raiserror @status "Error al insertar registros nuevos en EXISTENCIA"
   return
   end
nd

* ACTUALIZACION DE LAS EXISTENCIAS **/
* Actualiza piezas entregadas, afecta EXISTENCIA_PRODUCCION **/
e tran trAfecta_EXIS_prod

c @status =spsci_afecta_EXIS_produccion @fecha_emision, @edo_fis_origen, @especimen,
                  @pzas_afectadas, @factor_origen
f (@status !=0)
  begin
  rollback trENTREGA_BILLETE
  raiserror @status "Error al actualizar EXISTENCIA_PRODUCCION. Entregado"
  return
  end

* Actualiza piezas no entregadas, afecta billete emitido **/
e tran trAfecta_EXIS_prod

c @status =spsci_afecta_EXIS_produccion @fecha_emision, @emitido, @especimen,
                @numero_piezas, @factor_destino
f (@status !=0)
  begin
  rollback trENTREGA_BILLETE
  raiserror @status "Error al actualizar EXISTENCIA_PRODUCCION. Emitido"
  return
  end

* Actualiza piezas no entregadas, afecta maculatura **/
e tran trAfecta_EXIS_prod

c @status =spsci_afecta_EXIS_produccion @fecha_emision, @maculatura, @especimen,
                @pzas_no_emitidas, @factor_destino
f (@status !=0)
  begin
```

```
        rollback trENTREGA_BILLETE
        raiserror @status "Error al actualizar EXISTENCIA_PRODUCCION. Maculatura"
        return
        end

/** Actualiza piezas no entregadas, afecta EXISTENCIAS **/
save tran trEXISTENCIA

update EXISTENCIA
     set numero_piezas =numero_piezas + (@numero_piezas * @factor_destino)
 where institucion = @inst_destino
    and plaza = @plaza_destino
    and boveda = @boveda_destino
    and estado_fisico = @edo_fis_destino
    and especimen = @especimen

select @status = @@error
   if (@status !=0)
      begin
      rollback trENTREGA_BILLETE
      raiserror @status "Error al actualizar la tabla de EXISTENCIA"
      return
      end

/** Asigna fecha de proceso y cambia el estado de la operación a (P)rocesada **/
select @nuevo_estado ='P'

save tran mod_estado_OPERACION

exec @status =spsci_mod_estado_OPERACION   @fecha_registro, @folio_sistema, @nuevo_estad
     if (@status !=0)
        begin
        rollback trENTREGA_BILLETE
        raiserror @status "Error al actualizar el estado en OPERACION"
        return
        end

exec @status = spsci_valida_fecha_minima @fecha_emision
     if (@status != 0)
        begin
        rollback trENTREGA_BILLETE
        raiserror @status "Fecha de emision menor a fecha minima aceptada por el sistema"
        return
        end

exec @status = spsci_valida_fecha_minima @fecha_entrega
     if (@status != 0)
        begin
        rollback trENTREGA_BILLETE
        raiserror @status "Fecha de entrega menor a fecha minima aceptada por el sistema"
        return
        end
```

```
rn

-------------------------------------------------------------------

bdcontenedores

 trigger tu_ENTREGA_BILLETE

te trigger tu_ENTREGA_BILLETE on ENTREGA_BILLETE
pdate

n
are @regs_afectados     int

t @regs_afectados = @@rowcount
regs_afectados =0)   return

ack trENTREGA_BILLETE
rror 40058 "No se permite modificar datos en transacciones ya
esadas"
n


-------------------------------------------------------------------

ddcontenedores

 trigger td_ENTREGA_BILLETE

e trigger td_ENTREGA_BILLETE on ENTREGA_BILLETE
elete

clare @regs_afectados  int, @status int

ect @regs_afectados = @@rowcount
@regs_afectados = 0)   return
** VERIFICA QUE YA SE HAYA EFECTUADO EL RESPALDO
xists (CONDICION DE VALIDACION DE RESPALDO)
in
ollback trENTREGA_BILLETE
iserror 40067 "Transaccion aun no respaldada"
eturn
*/

No permite borrar transacciones pendientes de Procesadar */
sts (select * from OPERACION O, deleted d
      where O.fecha_registro = d.fecha_registro
        and O.folio_sistema  = d.folio_sistema
        and O.estado_proceso = 'I')
```

```
    begin
      rollback trENTREGA_BILLETE
      raiserror 40092 "No se puede borrar transacciones pendientes de procesar"
      return
    end

    /* Borra en cascada, si se elimina el encabezado, borra detalles y datos
    correspondientes en OPERACION */

    save tran trOPERACION

    delete OPERACION
      from deleted d, OPERACION Op
    where Op.fecha_registro = d.fecha_registro
      and Op.folio_sistema  = d.folio_sistema

    select @status = @@error
    if (@status !=0)
    begin
      rollback trENTREGA_BILLETE
      raiserror @status "Error al borrar datos en OPERACION"
      return
    end

    save tran trCOMUNICACION_SIAC

    delete COMUNICACION_SIAC
      from deleted d, COMUNICACION_SIAC CS
     where CS.fecha_registro = d.fecha_registro
      and CS.folio_sistema  = d.folio_sistema

    select @status = @@error
    if (@status !=0)
      begin
      rollback trENTREGA_BILLETE
      raiserror @status "Error al borrar datos en COMUNICACION_SIAC"
      return
      end

    save tran trCOMENTARIO

    delete COMENTARIO
      from deleted d, COMENTARIO C
    where C.fecha_registro = d.fecha_registro
      and C.folio_sistema  = d.folio_sistema

    select @status = @@error
    if (@status !=0)
      begin
      rollback trENTREGA_BILLETE
      raiserror @status "Error al borrar datos en COMENTARIO"
      return
    end
```

```
rn

-----------------------------------------------------------------------

bdcontenedores

 proc spsc_alta_EMPLEADO

te proc spsc_alta_EMPLEADO
      @terminal int,
      @inst_terminal int,
      @plaza_terminal char(5),
      @empleado char(6),
      @nombre char(50),
      @tipo_empleado char(1),
      @area_laboral varchar(6),
      @contrasena varchar(20)

are  @status          int,
      @valor_regla    char(10),
      @tipo_servicio  tinyint,
      @nombre_sp      varchar(30),
      @estado         char(1),
      @n_nombre       char(50),
      @t_empleado     char(1)

sts (select * from EMPLEADO where empleado = @empleado)
urn 40116   /* El elemento ya existe en el catálogo */

t @nombre_sp = 'spsc_alta_EMPLEADO'

t @tipo_servicio = tipo_servicio
n SP_SERVICIO
e stored_procedure = @nombre_sp

tipo_servicio is NULL)   return 40126

@status = spsci_val_stored_procedure @terminal, @inst_terminal,  @plaza_terminal,
                  @nombre_sp, @tipo_servicio
@status != 0) return @status

n tran alta_EMPLEADO

 tran trEMPLEADO

t @estado = estado, @n_nombre = nombre, @t_empleado = tipo_empleado
n hisifca. HST_EMPLEADO where empleado =@empleado

estado is NULL)
```

```
    begin
    insert into hisifca..HST_EMPLEADO values (@empleado, @nombre, @tipo_empleado, @area_l
                                              @contrasena, getdate(), '')
    select @status =@@error
    if (@status !=0)
      begin
      rollback alta_EMPLEADO
      return @status
      end
    end
  else
    begin
    update hisifca..HST_EMPLEADO set fecha_cambio = getdate(),
                                     estado = '' where empleado =@empleado
    select @status =@@error
    if (@status !=0)
      begin
      rollback alta_EMPLEADO
      return @status
      end
    select @nombre = @n_nombre, @tipo_empleado = @t_empleado
    end

  insert into EMPLEADO values(@empleado, @nombre, @tipo_empleado, @area_laboral, @contras

  select @status =@@error
    if (@status !=0)
       begin
       rollback alta_EMPLEADO
       return @status
       end

  commit tran alta_EMPLEADO

  return 0
  go
  grant execute on spsif_alta_EMPLEADO to monitor_contenedores
  grant execute on spsif_alta_EMPLEADO to administrador
  go
```

C.2. Código para el Cliente

```
/*********************************************************************
  Esta función elimina la operación si ésta existe en COMUNICACION_SIAC.
  Esto se hace cuando la operación ha sido enviada al servidor o se ha cancelado
  localmente.
*********************************************************************
public FUNCION boolean of_elimina_en_comunicacion_siac(datetime adt_fecha_registro,
                long al_folio_terminal);
```

```
o_comunicacion_siac        iuo_comunicacion_siac
o_comunicacion_siac = create uo_comunicacion_siac

* Si existe la operación:
iuo_comunicacion_siac.existe_operación (adt_fecha_registro, al_folio_terminal) then
    /* La elimina
if not iuo_comunicacion_siac.elimina(adt_fecha_registro, al_folio_terminal)  then
   destroy iuo_comunicacion_siac
   return FALSE
end if
nd if

estroy iuo_comunicacion_siac
turn TRUE
FUNCION

lic FUNCION boolean of_existe_operacion (datetime ad_fecha, long al_folio);
turn iuo_operacion.of_existe (ad_fecha, al_folio)
FUNCION

**********************************************************************
neca si la operación debe enviarse a SIAC (tiene una operación equivalente para este
ema)  y la guarda en la tabla COMUNICACION_SIAC. Regresa TRUE si la operación
almacenada correctamente o si no debía enviarse SIAC, y FALSE en caso de ocurrir
error durante el   proceso.
**********************************************************************/
lic FUNCION boolean of_guarda_en_comunicacion_siac (long al_tipo_operacion, string as_divisa,
            long al_tipo_efectivo, string as_estado,  long al_tipo_sitio_orig,
            long al_tipo_sitio_dest, string as_suc_sitios, long al_edo_fis_orig,
            long al_edo_fis_dest, datetime dt_fecha_registro, long al_folio_terminal,
            long al_folio_documento, long al_folio_sistema, string as_tipo_movimiento);

ng        ll_tipo_operacion_siac, ll_sucursal
olean     lb_resultado

_comunicacion_siac        iuo_comunicacion_siac
)_comunicacion_siac = create uo_comunicacion_siac

as_tipo_movimiento = gnv_constantes.of_gettipomovcancel () or
as_tipo_movimiento = gnv_constantes.of_gettipomovmodif () then
destroy iuo_comunicacion_siac
   return TRUE
1 if

' Primero pregunta si se envía a SIAC  **/
resultado=iuo_comunicacion_siac enviar_operacion_a_siac(al_tipo_operación, as_divisa,
            al_tipo_efectivo,as_estado, al_tipo_sitio_orig, al_tipo_sitio_dest, as_suc_sitios,
            al_edo_fis_orig, al_edo_fis_dest, ll_tipo_operacion_siac)

 Revisa el resultado para ver si no hubo "problemas al obtener" la operación SIAC
  equivalente **/
) resultado then
```

```
    /* Si no tiene operación equivalente en SIAC, no debe guardarse en
       COMUNICACION_SIAC; pero esto no quiere decir que no se pudo obtener la
       operación equivalente; por esta razón regresa TRUE. */
    if IsNull (il_tipo_operacion_siac) then
      destroy iuo_comunicacion_siac
      return TRUE
    end if
    /* Obtiene datos complementarios
    if not iuo_sitio.obtiene_sucursal (gnv_constantes.of_getinstitucion (),
              gnv_constantes.of_getplaza (),il_sucursal ) then
      destroy iuo_comunicacion_siac
      return FALSE
    end if
    /* almacena la operación
    lb_resultado = iuo_comunicacion_siac.inserta (adt_fecha_registro, al_folio_terminal,
              il_sucursal,gnv_constantes.of_getterminal (), al_folio_documento,
              al_folio_sistema, il_long_null, /*folio_siac*/ il_tipo_operacion_siac,
              as_tipo_movimiento, 'SI', /*estado*/ il_long_null  /*error*/)
    if not lb_resultado then
      destroy iuo_comunicacion_siac
      return FALSE
    end if
  end if
  destroy iuo_comunicacion_siac
  return TRUE
end FUNCION

/*****************************************************************
 Modifica_en_comunicacion_siac
 Guarda los cambios hechos en la operación si ésta existe en COMUNICACION_SIAC
*****************************************************************
public FUNCION boolean of_modifica_en_comunicacion_siac(long al_tipo_operacion, string as_divis
              long al_tipo_efectivo, string as_estado, long al_tipo_sitio_orig,
              long al_tipo_sitio_dest, string as_suc_sitios, long al_edo_fis_orig,
              long al_edo_fis_dest, datetime adt_fecha_registro, long al_folio_terminal,
              long al_folio_documento, long al_folio_sistema, string as_tipo_movimiento);

  long      il_tipo_operacion_siac, il_sucursal
  boolean   lb_enviar, lb_existe, lb_resultado

  uo_comunicacion_siac      iuo_comunicacion_siac
  iuo_comunicacion_siac = create uo_comunicacion_siac

  /* Pregunta si ya existe la operación en comunicación siac
  lb_existe = iuo_comunicacion_siac.existe_operacion (adt_fecha_registro, al_folio_terminal)

  /* Obtiene la operación siac
  lb_enviar = iuo_comunicacion_siac.enviar_operacion_a_siac (al_tipo_operacion, as_divisa,
              al_tipo_efectivo, as_estado, al_tipo_sitio_orig, al_tipo_sitio_dest, as_suc_sitios,
              al_edo_fis_orig, al_edo_fis_dest, il_tipo_operacion_siac)

  /* Este resultado es para revisar si hubo error "al obtener" la operación SIAC
     equivalente */
```

```
not lb_enviar then
MessageBox ('Aviso', 'No se puede saber si se envia la operacion a SIAC _ o no.')
return FALSE
nd if

* y este resultado es para checar si tiene operación equivalente
IsNull (il_tipo_operacion_siac) then
lb_enviar = FALSE
se
    lb_enviar = TRUE
nd if

*  No enviar
not lb_enviar then
    /* Si ya existe, borra la operación
 if lb_existe then
   lb_resultado = iuo_comunicacion_siac.elimina(adt_fecha_registro,al_folio_terminal)
 else
   lb_resultado = TRUE
 end if
 destroy iuo_comunicacion_siac
 return lb_resultado
nd if

* Enviar
* Obtiene la sucursal
not iuo_sitio.obtiene_sucursal(gnv_constantes.of_getinstitucion(),
            gnv_constantes.of_getplaza(),il_sucursal) then
destroy iuo_comunicacion_siac
return FALSE
nd if

* Si existe la operación, la modifica
lb_existe then
lb_resultado = iuo_comunicacion_siac.modifica (adt_fecha_registro,
            al_folio_terminal, il_sucursal, gnv_constantes.of_getterminal (),
            al_folio_documento, al_folio_sistema, il_long_null, /*folio_siac*/
            il_tipo_operacion_siac, as_tipo_movimiento, 'SI', /*estado*/ il_long_null
ror*/)
:e
   /* si no, la guarda
resultado = iuo_comunicacion_siac.inserta (adt_fecha_registro, al_folio_terminal, il_sucursal,
            gnv_constantes.of_getterminal(),  al_folio_documento, al_folio_sistema,
            il_long_null, /*folio_siac*/  il_tipo_operacion_siac, as_tipo_movimiento,
            'SI', /*estado*/ il_long_null /*error*/)
d if
stroy iuo_comunicacion_siac
urn lb_resultado
FUNCION


c FUNCION boolean of_obtiene_datos_a_desplegar(datetime ad_fecha_registro_param,
            long al_folio_sistema_param, ref long al_folio_documento,
```

```
                ref datetime  ad_fecha_aplicacion, ref long al_tipo_efectivo, ref string as_divisa,
                ref string as_numero_acta, ref datetime ad_fecha_emision,
                ref string as_numero_orden, ref datetime ad_fecha_entrega,
                ref long al_fabricante, ref long al_edo_fis_orig, ref long al_inst_dest,
                ref string as_plaza_dest, ref long al_boveda_dest, ref long al_edo_fis_dest,
                ref long al_especimen, ref string as_tiro, ref string as_series,
                ref string as_prefijos, ref string as_millones, ref long ad_num_piezas,
                ref long ad_mac_proc_fin, ref long ad_mac_manufactura, ref decimal ad_import
                ref string as_comentario, ref datetime ad_fecha_consulta,
                ref long ai_id_consulta);


  /* Variables para almacenar el detalle que envía el servidor
  string      ls_tipo_movto
  long        ll_retorno, ll_tipo_operacion

  /* Almacena valores de la operación en las variables de ins
  id_fecha_registro= ad_fecha_registro_param
  il_folio_sistema = al_folio_sistema_param
  il_terminal = 0
  il_folio_terminal = 0
  ls_tipo_movto = ''

ll_retorno = this.of_obtiene_datos_servidor (gnv_constantes.of_getterminal(),
                gnv_constantes.of_getinstitucion(), gnv_constantes.of_getplaza (),
                datetime(id_fecha_registro), il_folio_sistema, il_terminal,
                il_folio_terminal, al_folio_documento, al_inst_dest, as_plaza_dest,
                ls_tipo_movto, al_tipo_efectivo, ll_tipo_operacion, as_divisa, ad_importe,
                ad_fecha_aplicacion, as_numero_acta, ad_fecha_emision,
                as_numero_orden, ad_fecha_entrega, al_fabricante, al_edo_fis_orig,
                al_boveda_dest, al_edo_fis_dest, al_especimen, as_tiro, as_series,
                as_prefijos, as_millones, ad_num_piezas, ad_mac_proc_fin,
                ad_mac_manufactura, as_comentario, ad_fecha_consulta,
                ai_id_consulta)

  if ll_retorno = 0 then
       return TRUE
  else
    return FALSE
  end if
end FUNCION


public FUNCION boolean of_elimina_operacion (string as_tipo_movimiento);
  if this.of_elimina_en_comunicacion_siac (id_fecha_registro,il_folio_terminal)  then
    if iuo_operacion.of_actualiza_movimiento_y_estado (id_fecha_registro, il_folio_terminal,
                as_tipo_movimiento, gnv_constantes.of_getprocesada ()) then
      commit using gnv_tr_local;
      return TRUE
    end if
     end if
  rollback using gnv_tr_local;
  return FALSE
```

```
 FUNCION


lic FUNCION boolean of_continuar_con_error (string as_mensaje_error);
teger li_respuesta

_respuesta = MessageBox ('Error en la validación...', as_mensaje_error +
esea almacenar la operación para modificarla posteriormente?' +
a "No", para corregir ahora.',question!, YesNo!, 2)
li_respuesta = 1 then
    return TRUE
se
    return FALSE
nd If
 FUNCION


lic FUNCION boolean of_guarda_operacion (long al_folio_documento, string as_tipo_movimiento,
        long al_tipo_efectivo, long al_tipo_operacion, decimal ad_importe,
        datetime ad_fecha_aplicacion, string as_numero_acta, datetime ad_fecha_emision,
        string as_numero_orden, datetime ad_fecha_entrega, long al_fabricante,
        long al_estado_fisico_origen, long al_institucion_dest, string as_plaza_dest,
        long al_boveda_dest, long al_estado_fisico_dest, long al_especimen, string as_tiro,
        string as_series, string as_prefijos, string as_millones, long al_numero_piezas,
        long al_numero_piezas_proc, long al_numero_piezas_manu, string as_divisa,
        string as_comentario, datetime ad_fecha_consulta, long al_id_consulta,
        long al_calculo_mac_proc, string as_claves[], string as_passwords[],
        long al_numero_claves, long argl_grupos_autorizacion[]);

* Variables
ng    ll_retorno, ll_tipo_sitio, ll_folio_siac, ll_folio_documento
ring  ls_estado
olean lb_ok = true

 Si no fuera una alta, estos datos se habrían asignado en
 obtiene_datos_a_desplegar()'  */
as_tipo_movimiento = gnv_constantes of_gettipomovalta() THEN
/* Variables que son obtenidas por el servidor
l_folio_siac = 0
l_folio_sistema = 0

/* Se inicializa variable de terminal
l_terminal = gnv_constantes of_getterminal()
d If

 Obtiene el folio siguiente
olio_terminal = uo_folio_terminal of_folioterminal()
olio_documento = il_folio_terminal

 Si es cancelación, no hace validación alguna
NOT as_tipo_movimiento = gnv_constantes of_gettipomovcancel() THEN
   /* Valida el sitio destino
```

```
if lb_ok AND not iuo_sitio.existe_sitio (al_institucion_dest,  as_plaza_dest) then
   lb_ok = FALSE
end if

/* Si no hay autorización para esta operación, realiza las validaciones
   correspondientes */
if lb_ok AND not iuo_validacion_operacion.existe_autorizacion(al_folio_documento,
            al_tipo_operacion, ad_importe, as_tipo_movimiento) then
      /* Valida que exista el registro de existencia
   if lb_ok AND not iuo_restriccion_existencia.existe_en_existencia(al_institucion_dest,
               as_plaza_dest, al_boveda_dest, al_estado_fisico_dest, al_tipo_efectivo) then
      lb_ok = FALSE
   end if
   /* Obtiene tipo sitio destino
   if lb_ok AND not iuo_sitio.obtiene_tipo_sitio (al_institucion_dest, as_plaza_dest,
            il_tipo_sitio) then
      lb_ok = FALSE
   end if
   /* Asigna valores para validación
   il_tipo_sitio_origen = il_long_null
   il_institucion_origen = il_long_null
   is_plaza_origen = is_string_null
   il_boveda_origen = il_long_null
   il_estado_fisico_origen = al_estado_fisico_origen
   il_tipo_sitio_destino = il_tipo_sitio
   il_institucion_destino = al_institucion_dest
   is_plaza_destino = as_plaza_dest
   il_boveda_destino = al_boveda_dest
   il_estado_fisico_destino = al_estado_fisico_dest
 end if
end If

if lb_ok then
      ls_estado = gnv_constantes.of_getinsertada ()
else
  if not this.of_continuar_con_error('Se hallaron errores al validar.')then
    rollback using gnv_tr_local;
       return FALSE
      end if
      ls_estado = gnv_constantes.of_getrechazada ()
end if

/* Inserta operación
if not iuo_operacion.of_inserta (id_fecha_registro, il_folio_sistema, il_terminal, il_folio_terminal,
                  il_folio_documento, gnv_constantes.of_getusuario(), al_institucion_dest,
                  as_plaza_dest, as_tipo_movimiento, al_tipo_efectivo, al_tipo_operacion,
                  as_divisa, ad_importe, ad_fecha_aplicacion, id_fecha_registro, ls_estado) the
  rollback using gnv_tr_local;
      return FALSE
end if

/* Inserta entrega_billete
if not iuo_entrega_billete.of_inserta (id_fecha_registro, il_folio_terminal, as_numero_acta,
```

```
        ad_fecha_emision,  as_numero_orden,  ad_fecha_entrega, al_fabricante,
        al_estado_fisico_origen, al_institucion_dest, as_plaza_dest, al_boveda_dest,
        al_estado_fisico_dest, al_especimen, as_tiro, as_series, as_prefijos, as_millones,
        al_numero_piezas, al_numero_piezas_proc, al_numero_piezas_manu,
        ad_fecha_consulta, al_id_consulta, al_calculo_mac_proc, as_claves,
        as_passwords, argl_grupos_autorizacion) then
rollback using gnv_tr_local;
return FALSE
nd if

* Inserta el comentario
as_comentario <> '' then
if not iuo_comentario.of_inserta (id_fecha_registro, il_folio_terminal, as_comentario) then
   rollback using gnv_tr_local;
      return FALSE
end if
nd if

* Operación insertada correctamente
mmit using gnv_tr_local;

* Envía al servidor la operación insertada
lb_ok then
gnv_app.event pfc_idle()
nd if
turn TRUE
FUNCION


ic FUNCION boolean of_modifica_operacion (long al_folio, string as_tipo_movimiento,
        long al_tipo_efectivo, long al_tipo_operacion, decimal ad_importe,
        datetime ad_fecha_aplicacion, string as_numero_acta, datetime ad_fecha_emision,
        string as_numero_orden, datetime ad_fecha_entrega, long al_fabricante,
        long al_estado_fisico_orig, long al_institucion_dest, string as_plaza_dest,
        long al_boveda_dest, long al_estado_fisico_dest, long al_especimen, string as_tiro,
        string as_series, string as_prefijos, string as_millones, decimal ad_numero_piezas,
        decimal ad_mac_proc_fin, decimal ad_mac_manufactura,
        datetime ad_fecha_consulta, long al_id_consulta, long al_calculo_mac_proc,
        long al_grupo_aut1, long al_grupo_aut2, string as_empleado1,
        string as_empleado2, string as_password1, string as_password2,
        string as_divisa, string as_comentario);

Variables
g       retorno, ll_tipo_sitio
lean    lb_ok = TRUE
ng      ls_estado

/* Asigna valores para validación
l_tipo_sitio_origen      = il_long_null
l_institucion_origen     = il_long_null
s_plaza_origen           = is_string_null
l_boveda_origen          = il_long_null
l_estado_fisico_origen   = al_estado_fisico_orig
```

```
        il_tipo_sitio_destino      = il_tipo_sitio
        il_institucion_destino     = al_institucion_dest
        is_plaza_destino = as_plaza_dest
        il_boveda_destino          = al_boveda_dest
        il_estado_fisico_destino   = al_estado_fisico_dest

    if lb_ok then
        ls_estado = gnv_constantes.of_getinserzada ()
    else
        if not this.of_continuar_con_error ('Se hallaron errores al validar.') then
            return FALSE
        end if
        ls_estado = gnv_constantes.of_getrechazada ()
    end if

    /* Modifica operación
    if not iuo_operacion.of_modifica (id_fecha_registro, il_folio_sistema, il_terminal, il_folio_termina
                al_folio, gnv_constantes.of_getusuario(),al_institucion_dest, as_plaza_dest,
                as_tipo_movimiento, al_tipo_efectivo, al_tipo_operacion, as_divisa, ad_importe,
                ad_fecha_aplicacion, id_fecha_registro, ls_estado) then
        rollback using gnv_tr_local;
        return FALSE
    end if

    /* Modifica entrega_billete
    if not iuo_entrega_billete.of_modifica (id_fecha_registro, il_folio_terminal, as_numero_acta,
                ad_fecha_emision, as_numero_orden, ad_fecha_entrega, al_fabricante,
                al_estado_fisico_orig, al_institucion_dest, as_plaza_dest, al_boveda_dest,
                al_estado_fisico_dest, al_especimen, as_tiro, as_series, as_prefijos, as_millones,
                ad_numero_piezas, ad_mac_proc_fin, ad_mac_manufactura, ad_fecha_consulta,
                al_id_consulta, al_calculo_mac_proc, al_grupo_aut1, al_grupo_aut2,
                as_empleado1, as_empleado2, as_password1, as_password2) then
        rollback using gnv_tr_local;
        return FALSE
    end if

    /* Modifica el comentario
    if not iuo_comentario.of_modifica (id_fecha_registro, il_folio_terminal, as_comentario) then
        rollback using gnv_tr_local;
        return FALSE
    end if

    /* Operación insertada correctamente
    commit using gnv_tr_local;

    /* Envía al servidor la operación insertada
    if lb_ok then gnv_app.event pfc_idle()
        return TRUE
    end if
end FUNCION


public FUNCION boolean of_obtiene_datos_base_local(datetime ad_fecha_registro_param,
```

```
        long al_folio_terminal_param, ref long al_folio_documento,
        ref datetime ad_fecha_aplicacion, ref long al_tipo_efectivo, ref string as_divisa,
        ref string as_numero_acta, ref datetime ad_fecha_emision,
        ref string as_numero_orden, ref datetime ad_fecha_entrega, ref long al_fabricante,
        ref long al_edo_fis_orig, ref long al_inst_dest, ref string as_plaza_dest,
        ref long al_boveda_dest, ref long al_edo_fis_dest, ref long al_especimen,
        ref string as_tiro, ref string as_series, ref string as_prefijos, ref string as_millones,
        ref decimal ad_num_piezas, ref decimal ad_mac_proc_fin,
        ref decimal ad_mac_manufactura, ref datetime ad_fecha_consulta,
        ref long al_id_consulta, ref long al_calculo_mac_proc, ref long al_grupo_aut1,
        ref long al_grupo_aut2, ref string as_empleado1, ref string as_empleado2,
        ref string as_password1, ref string as_password2, ref decimal ad_importe,
        ref string as_comentario);

* Variables para almacenar los datos no capturados que envía el servidor
ring      ls_tipo_movto
ng        ll_retorno, ll_tipo_operacion
ring      ls_login, ls_estado_proceso
atetime   ld_fecha_proceso
atastore  lds_detalle

* Almacena valores de la operación en las variables de ins
_fecha_registro = ad_fecha_registro_param
folio_terminal      = al_folio_terminal_param

* Obtiene los datos de la tabla OPERACION
uo_operacion.of_obtiene_datos (id_fecha_registro, il_folio_sistema, il_terminal,
            il_folio_terminal, al_folio_documento, ls_login, al_inst_dest, as_plaza_dest,
            ls_tipo_movto, al_tipo_efectivo, ll_tipo_operacion, as_divisa, ad_importe,
            ad_fecha_aplicacion, ld_fecha_proceso, ls_estado_proceso) <> 0 then
eturn FALSE
d if

uo_entrega_billete.of_obtiene_datos (id_fecha_registro, il_folio_terminal, as_numero_acta,
            ad_fecha_emision, as_numero_orden, ad_fecha_entrega, al_fabricante,
            al_edo_fis_orig, al_inst_dest, as_plaza_dest, al_boveda_dest, al_edo_fis_dest,
            al_espécimen, as_tiro, as_series, as_prefijos, as_millones, ad_num_piezas,
            ad_mac_proc_fin, ad_mac_manufactura, ad_fecha_consulta, al_id_consulta,
            al_calculo_mac_proc, al_grupo_aut1, al_grupo_aut2, as_empleado1,
            as_empleado2, as_password1, as_password2) <> 0 then
eturn FALSE
d if

Obtiene el comentario
comentario      = uo_comentario.of_obtiene_comentario (id_fecha_registro,  il_folio_terminal)
      return    TRUE
UNCION


n_control_entrega_billete.create
super::create
on
```

```
on uo_control_entrega_billete.destroy
  call super::destroy
end on

event constructor;call super::constructor;iuo_entrega_billete = Create uo_entrega_billete
  /* Crea los objetos generales para la insercion de operaciones
iuo_operacion          = create uo_operacion
iuo_comentario         = create uo_comentario
iuo_validacion_operación = create uo_validacion_operacion
iuo_tipo_operacion     = create uo_tipo_operacion
iuo_restriccion_existencia = create uo_restriccion_existencia
iuo_sitio              = create uo_sitio
iuo_folio_terminal     = create uo_folio_terminal
id_fecha_registro      = gnv_constantes.of_getfechahoy ()
SetNull (il_long_null)
SetNull (is_string_null)
end event

event destructor;call super::destructor;Destroy iuo_entrega_billete
end event
```

OBJETO VENTANA

```
/** Se declaran los objetos de la ventana **/
forward
  global type w_captura_entrega from w_sheet_captura  end type
  type dw_captura_entrega from u_captura_relacion_entrega within w_captura_entrega  end type
  type dw_entrega_contenedores from u_entrega_contenedores  within w_captura_entrega
  end type
  type lb_millon from u_lb within w_captura_entrega          end type
  type st_1 from statictext within w_captura_entrega         end type
  type gb_comandos from groupbox within w_captura_entrega     end type
  type cb_consulta from u_cb_aceptar within w_captura_entrega end type
  type cb_imprime from u_cb_aceptar within w_captura_entrega  end type
  type gb_opciones from groupbox within w_captura_entrega     end type
end forward

/** Se declaran las variables globales **/
global type w_captura_entrega from w_sheet_capturainteger
width = 3904
integer height = 2324string
title = "RELACION DE ENTREGA [ALTA]"
boolean toolbarvisible = false
dw_captura_entrega dw_captura_entrega
dw_entrega_contenedores dw_entrega_contenedoreslb_millon lb_millonst_1 st_1 gb_comandos
gb_comandoscb_consulta cb_consultacb_imprime cb_imprime
gb_opciones gb_opcionesend typeglobal w_captura_entrega w_captura_entrega

type variables
long il_num_regsstring is_millonboolean
     ib_consultado  = false
end variables
```

```
* Se declara el prototipo de las funciones **/
ward prototypes
ublic subroutine of_elije_contenedores_entrega ()
ublic subroutine of_limpia_seleccion ()
ublic subroutine of_desglosa_codigo_de_barras (string as_codigo_barra,  ref string as_banco,
 string as_especimen, ref string as_cve_envase,
 string as_consecutivo, ref string as_digito)
d prototypes

* Se especifica cada función **/
blic subroutine of_elije_contenedores_entrega();
* Obtiene los datos para buscar los contenedores en el Servidor
Boolean lb_res
ong         ll_errorn_cst_itemattrib lnv_items[ ]
nteger      li_selected
ong         li_indicestring
s_millon = ""
SetPointer (Hourglass!)

/* Trae los contenedores de los millones seleccionados
_selected = lb_millon.of_GetSelected(lnv_items)
f li_selected = 0 then
  MessageBox("ERROR", "Falta información por proporcionar")
  Return
nd if

_indice = 1
o while li_indice <= li_selected
  if not ls_millon = "" THEN
    ls_millon = ls_millon + "," + lnv_items[li_indice].is_itemtext
  else
    ls_millon = ls_millon + lnv_items[li_indice].is_itemtext
  end if
  li_indice++
oop

ong ll_terminal, ll_institucion, ll_denominacion datetime ld_fecha
tring   ls_plaza, ls_tiros_millon      = ls_millon
_terminal  = gnv_constantes.of_getterminal()
_institución = gnv_constantes.of_getinstitucion()
:_plaza = gnv_constantes.of_getplaza()
_denominación = dw_captura_entrega.GetItemNumber(1,1)
l_fecha = dw_captura_entrega.GetItemDatetime(1,2)
:_tiro = dw_captura_entrega.GetItemString(1,3)

* Obtiene del Servidor, los contenedores disponibles segun denominación,
  fecha emisión, tiro y millón seleccionados */
_num_regs = dw_entrega_contenedores retrieve(ll_terminal, ll_institucion, ls_plaza,
                ll_denominacion, ld_fecha, ls_tiro, ls_millon)
' subroutine

Esta función limpia los controles de la ventana
```

```
public subroutine of_limpia_seleccion();
   lb_millon.event
   pfc_SelectAll()
   lb_millon.event
   pfc_InvertSelection()
end subroutine


/* Esta función desglosa el los datos codificados en el código de barras
public subroutine of_desglosa_codigo_de_barras (string as_codigo_barra,  ref string as_banco,
                  ref string as_especimen, ref string as_cve_envase, ref string as_consecutivo,
                  ref string as_digito);

   as_banco   = mid(as_codigo_barra,1,5)
   as_espécimen = mid(as_codigo_barra,6,3)
   as_cve_envase = mid(as_codigo_barra,9,2)
   as_consecutivo = mid(as_codigo_barra,11,5)
   as_digito = mid(as_codigo_barra,16,1)
   return
end subroutine

on w_captura_entrega.createint iCurrentcall super::create
this.dw_captura_entrega = create dw_captura_entrega
this.dw_entrega_contenedores = create dw_entrega_contenedores
this.lb_millon = create lb_millonthis.st_1=create st_1
this.gb_comandos = create
gb_comandosthis.cb_consulta = create cb_consulta
this.cb_imprime = create cb_imprimethis.gb_opciones = create gb_opciones
iCurrent = UpperBound(this.Control)
this.Control[iCurrent+1] = this.dw_captura_entrega
this.Control[iCurrent+2] = this.dw_entrega_contenedores
this.Control[iCurrent+3] = this.lb_millonthis.Control[iCurrent+4]=this.st_1
this.Control[iCurrent+5] = this.gb_comandos
this.Control[iCurrent+6] = this.cb_consulta
this.Control[iCurrent+7] = this.cb_imprime
this.Control[iCurrent+8] = this.gb_opcionesend onon w_captura_entrega.destroy
call super::destroydestroy(this.dw_captura_entrega)
destroy(this.dw_entrega_contenedores)
destroy(this.lb_millon)
destroy(this.st_1)
destroy(this.gb_comandos)
destroy(this.cb_consulta)
destroy(this.cb_imprime)
destroy(this.gb_opciones)
end onevent open;call super::open;
dw_entrega_contenedores.Of_SetRowSelect(TRUE)
dw_entrega_contenedores.inv_rowselect.of_SetStyle
(dw_entrega_contenedores.inv_rowselect.EXTENDED)
cb_imprime.visible = False
end event

type cb_2 from w_sheet_captura`cb_2 within w_captura_entregainteger x = 3049
integer y = 1584integer
```

```
= 590
er height = 108
er taborder = 70
harset fontcharset = ansi!
g text = "Salir"
ype

cb_1 from w_sheet_captura`cb_1 within w_captura_entrega
eger x = 3049
eger y = 1316
eger width = 581
eger height = 108
eger taborder = 50
tcharset fontcharset = ansi!
ng text = "Aceptar Seleccion"
ype

Este botón se encarga de enviar la relación de contenedores seleccionados al
servidor y prepara una relación para ser impresa. */
t cb_1::clicked;call super::clicked;

        ll_indice, ll_num_cont, ll_indice1, id_consulta, lrgs_num_piezas[], lrgs_valor[]
        lrgs_serie[],lrgs_prefijo[],lrgs_etiqueta[],lrgs_millones[]
        ls_banco, ls_especimen, ls_envase, ls_consecutivo, ls_digito
an      lb_seleccionadon_ds ds_desgloce_etiqueta
        as_banco, as_especimen, as_cve_envase, as_consecutivo, as_digito

 lb_consultado then
 ds_desgloce_etiqueta = Create n_ds
 uo_concatena_detalle  iuo_concatena_detalle
 iuo_concatena_detalle = Create uo_concatena_detalle
 ds_desgloce_etiqueta.DataObject = "d_desgloce_etiqueta"
ndice = 1
um_cont = 0
 Si está seleccionado el contenedor lo asigna a la consulta
 while ll_indice <= il_num_regs
)_seleccionado = dw_entrega_contenedores.Isselected (ll_indice)
   if lb_seleccionado then
 ll_num_cont ++
 lrgs_serie[ll_num_cont] =  dw_entrega_contenedores.getitemstring(ll_indice,1)
 lrgs_prefijo[ll_num_cont] = dw_entrega_contenedores.getitemstring(ll_indice,2)
 lrgs_etiqueta[ll_num_cont] = dw_entrega_contenedores.getitemstring(ll_indice,3)
 lrgs_millones[ll_num_cont] = dw_entrega_contenedores.getitemstring(ll_indice,4)
 lrgs_num_piezas[ll_num_cont] = dw_entrega_contenedores getitemnumber(ll_indice,5)
 lrgs_valor[ll_num_cont] = dw_entrega_contenedores.getitemnumber(ll_indice,6)
nd if
 indice++
 num_cont > 0  then
 consultado = TRUE
  /* Muestra en la misma Data Window la Consulta resultante para poder ser impreso
  si se desea */
```

```
dw_entrega_contenedores.DataObject = "d_contenedores_seleccionados"
lb_seleccionado = FALSE
ll_indice = 1
ll_indice1 = 1
    /* Vetana que avisa a usuario que está transmitiendo
SetPointer (HourGlass!)
    OpenWithParm (w_aviso,'Transmitiendo al servidor...')
do while ll_indice <= ll_num_cont
  dw_entrega_contenedores.insertrow(0)
  dw_entrega_contenedores.Setitem(ll_indice, 1,
  dw_captura_entrega.GetItemDecimal(1,1))
  dw_entrega_contenedores.Setitem(ll_indice, 2,
  dw_captura_entrega.GetItemDatetime(1,2))
  dw_entrega_contenedores.Setitem(ll_indice, 3,
  dw_captura_entrega.GetItemString(1,3))
 dw_entrega_contenedores.Setitem(ll_indice, 4, lrgs_serie[ll_indice])
  dw_entrega_contenedores.Setitem(ll_indice, 5, lrgs_prefijo[ll_indice])
  dw_entrega_contenedores.Setitem(ll_indice, 6, lrgs_etiqueta[ll_indice])
  dw_entrega_contenedores.Setitem(ll_indice, 8, gnv_constantes.of_getfechahoy())
  dw_entrega_contenedores.Setitem(ll_indice, 9, lrgs_millones[ll_indice])
  dw_entrega_contenedores.Setitem(ll_indice, 10, lrgs_num_piezas[ll_indice])
  dw_entrega_contenedores.Setitem(ll_indice, 11, lrgs_valor[ll_indice])
  /* Desglosa los datos codificados en el código de barras
  as_banco = mid(lrgs_etiqueta[ll_indice],1,5)
  as_espécimen  =mid(lrgs_etiqueta[ll_indice],6,3)
  as_cve_envase = mid(lrgs_etiqueta[ll_indice],9,2)
  as_consecutivo = mid(lrgs_etiqueta[ll_indice],11,6)
  as_digito = mid(lrgs_etiqueta[ll_indice],16,1)
  ds_desgloce_etiqueta.insertrow(0)
  ds_desgloce_etiqueta.Setitem(ll_indice, 1, as_banco)
  ds_desgloce_etiqueta.Setitem(ll_indice, 2, as_consecutivo)
   ds_desgloce_etiqueta.Setitem(ll_indice, 3, lrgs_millones[ll_indice])
   ll_indice++
   loop
else
   MessageBox('Aviso','No ha seleccionado ningún Contenedor de la Lista')
end if
    if ll_num_cont > 0  then
  /* Concatena los contenedores seleccionados para ser enviados al servidor
  iuo_concatena_detalle.of_trans_consulta_entrega(ds_desgloce_etiqueta,
              dw_captura_entrega.GetItemDecimal(1,1),
              dw_captura_entrega.GetItemDatetime(1,2),
              dw_captura_entrega.GetItemString(1,3), is_millon, lrgs_serie[1], lrgs_prefijo[1],
              id_consulta)
  dw_entrega_contenedores.Setitem(1, 'id_consulta', id_consulta)
  close (w_aviso)
    parent.of_limpia_seleccion()

    /* Muestra al usuario el ID Consulta asignado por el Servidor
  messagebox ('id_consulta','ID_CONSULTA:    '+string(id_consulta))
  cb_imprime.visible = True
  cb_consulta.visible = False
end if
```

```
e
MessageBox('Aviso','Ya realizó la Consulta')
d if
d event


pe dw_captura_entrega from u_captura_relacion_entrega within w_captura_entrega
eger x = 389
eger y = 56
eger width = 2226
eger height = 276
eger taborder = 10
olean bringtotop = true
olean titlebar = true
ing title = "Datos para Consultar Contenedores"
ing dataobject = "dw_captura_datos_entrega"
olean vscrollbar = false
olean border = false
olean livescroll = false
rderstyle borderstyle = styleshadowbox!
ing is_coloranterior = " "
d type


/ Nombre: pfc_populateDDDW
/ Descripción: Desplegar las columnas de las DropDownDataWindow que necesitan
/              parámetros
/ Regresa: El número de renglones consultados
/ Argumentos: as_colname Nombre de la columna donde esta posicionado
/             adwc_obj   DataWindow asociada donde se hará el retrieve.
/                  Esta cambia cada vez que las PFC's recorren las DDDW.
ent pfc_populatedddw;

nteger  li_denomdatetime ld_femilong ll_returnstring ls_valor
/* Elijo la trasacción de las DDDW.this.uf_trandddws()
/* Si la columna es la del TIRO. Entonces tomo el especimen y fecha_emision
/* respectivos para pasarlo como parámetros.
f as_colname = 'fecha_emision'  then
  if this.getrow() = 0 then
    setnull(li_denom)
  else
       li_denom = this.getitemNumber(This getRow(),'denominacion')
  end if
     ll_return = adwc_obj.retrieve(li_denom)
nd if
f as_colname = 'tiro' then
    if this.getrow() = 0 then
   setnull(li_denom)
   setnull(ld_femi)
  else
       li_denom = this.getitemNumber(This getRow(),'denominacion')
       ld_femi = this.getitemdatetime(This getRow(),'fecha_emision')
  end if
```

```
    ll_return =adwc_obj.retrieve(li_denom,ld_femi)
  end if

  /* Si la columna no es el TIRO o Fecha_emision. Entonces hago un retrieve
  /* normal.
  if as_colname = 'denominacion' then
    ll_return =adwc_obj.retrieve()
  end if

  /*Regreso el resultado de la consulta.
  return ll_return
end event


// Nombre: pfc_prermbmenu
// Descripción: Inhabilitar las opciones de insertar, añadir y borrar registros
//              en la DW
event constructor;call super::constructor;this.inv_rowselect.of_setstyle(1)
  This. event pfc_insertrow()this.of_setupdateable(FALSE)
end event

event pfc_prermbmenu;call super::pfc_prermbmenu;
  am_dw.m_table.m_insert.enabled = falseam_dw.m_table.m_addrow.enabled = false
  am_dw.m_table.m_delete.enabled = falseend event
  type dw_entrega_contenedores from u_entrega_contenedores within  w_captura_entrega
  integer x = 64
    integer y = 356
    integer width = 2917
    integer height = 1372
    integer taborder = 40
    boolean bringtotop = true
    boolean titlebar = true
    string title = "Lista Resultante de Contenedores"
    string dataobject = "d_cons_envase_entrega"
    boolean hscrollbar = true
    boolean resizable = true
    borderstyle borderstyle = stylesshadowbox!
  end type

event buttonclicked;call super::buttonclicked;parent.of_limpia_seleccion()
  reset(dw_entrega_contenedores)
end event


// Nombre: pfc_prermbmenu
// Descripción: Inhabilitar las opciones de insertar, añadir y borrar registros
//              en la DW
event pfc_prermbmenu;call super::pfc_prermbmenu;
  am_dw.m_table.m_insert.enabled = falseam_dw.m_table.m_addrow.enabled = false
  am_dw.m_table.m_delete.enabled = false
end event

event constructor;call super::constructor;this.of_setupdateable(FALSE)
```

```
/ event

e lb_millon from u_lb within w_captura_entrega
nteger x = 3122
nteger y = 532
nteger width = 293
nteger height = 532
nteger taborder = 20
oolean bringtotop = true
oolean sorted = false
oolean multiselect = true
tring item[] = {"0","1","2","3","4","5","6","7","8","9"}
orderstyle borderstyle = styleshadowbox!
oolean extendedselect = true
/ type

e st_1 from statictext within w_captura_entrega
nteger x = 3118
nteger y = 412
nteger width = 288
nteger height = 80
oolean bringtotop = true
nteger textsize = -10
nteger weight = 700
ntcharset fontcharset = ansi!
ntpitch fontpitch = variable!
ntfamily fontfamily = swiss!
tring facename = "MS Sans Serif"
ng textcolor = 33554432
ng backcolor = 67108864
tring text = "Millón"
oolean border = true
orderstyle borderstyle = styleshadowbox!
oolean focusrectangle = false
/ type

' gb_comandos from groupbox within w_captura_entrega
teger x = 3013
teger y = 1108
teger width = 658
eger height = 620
eger textsize = -10
eger weight = 700
ntcharset fontcharset = ansi!
ntpitch fontpitch = variable!
ntfamily fontfamily = swiss!
ing facename = "MS Sans Serif"
ng textcolor = 33554432
g backcolor = 67108864
rderstyle borderstyle = stylelowered!
/ type

gb_consulta from u_cb within w_captura_entrega
```

```
    integer x = 3049
    integer y = 1180
    integer width = 581
    integer height = 108
    integer taborder = 30
    boolean bringtotop = true
    fontcharset fontcharset = ansi!
    string text = "Consulta Contenedores"
end type

// En este Control, se limpia la dw de_entrega_contenedores
// y manda llamar a la función of_elije_contenedores_entrega
event clicked;call super::clicked;
    reset(dw_entrega_contenedores)Parent.of_elije_contenedores_entrega()
end event

type cb_imprime from u_cb_aceptar within w_captura_entrega
    integer x = 3049
    integer y = 1452
    integer width = 585
    integer height = 108
    integer taborder = 60
    boolean bringtotop = true
    fontcharset fontcharset = ansi!
    string text = "Impresion de la Seleccion"
end type

// Este boton, se encarga de imprimir la relación de contenedores
// seleccionados
event clicked;call super::clicked;
    long retorno
    retorno = dw_entrega_contenedores.event pfc_print()
    if retorno <> 1 then
      of_messagebox('err_dberror','Print','Hay problemas para imprimir el reporte',StopSign!, Ok!, 1)
    end if
end event

type gb_opciones from groupbox within w_captura_entrega
    integer x = 3058
    integer y = 312
    integer width = 434
    integer height = 796
    integer textsize = -10
    integer weight = 700
    fontcharset fontcharset = ansi!
    fontpitch fontpitch = variable!
    fontfamily fontfamily = swiss!
    string facename = "MS Sans Serif"long textcolor = 33554432
    long backcolor = 67108864
    borderstyle borderstyle = stylelowered!
end type
```

3.- Código para la Palm

ontinuación se muestra el código para controlar los eventos de pantalla donde se pide el identificador de la relación de tenedores y la fecha en que se realizó, también se muestar la talla respectiva (Fig. 6.9.).

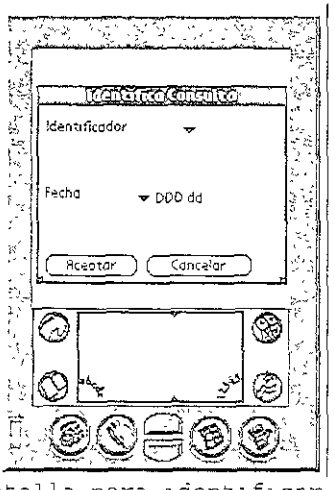


Fig. 6.9. Pantalla para identificar la consulta.

```
nclude <Pilot.h>
nclude <SysEvtMgr.h>
nclude "MainFormRsc.h"
nclude "utils.h"
nclude "sqlca.h"
nclude "entregadb.h"

********** Variables Globales **********/
teType          g_fechaCons,
temPreferencesType sysPrefs;
                g_idCons;

********** Constantes **********/
Constantes para los elementos de la lista que elige la fecha de
a consulta.
efine selectDateItem    0
efine firstItem         0

*****
FUNCION.    DeterminaFecha
DESCRIPCION. De acuerdo a la lista de opciones decido que fecha
             poner en la consulta
```

```
 * PARAMETROS:  word itemSelected- Campo seleccionado de la lista
 *                          DateType fechaConsP- fecha de la consulta.
 * REGRESA:    nada
 * REVISION:
 ***********************************************************************/
static void DeterminaFecha(Word itemSelected, DateType
        * fechaConsP)
{
Word                timeInSeconds;
SWord               year, month, day;
DateTimeType        date;
VoidHand            titleH;
CharPtr             titleP;
Boolean             lb_fecha;

    if(itemSelected  == selectDateItem)
    {
            if(*((ShortPtr) fechaConsP) == -1)
            {
                    timeInSeconds  =TimGetSeconds();
                    TimSecondsToDateTime(timeInSeconds, &date);
                    year     =date.year;
                    month   =date.month;
                    day              =date.day;
            }
            else
            {
                    year     =fechaConsP->year + firstYear;
                    month   =fechaConsP->month;
                    day              =fechaConsP->day;
            }
            titleH    =DmGetResource(strRsc, FechaTitleString);
            titleP    =(CharPtr) MemHandleLock(titleH);

            lb_fecha          =SelectDay(selectDayByDay, &month, &day, &year,
          titleP);
            if(lb_fecha)
            {
                    fechaConsP->day                  =day;
                    fechaConsP->month       =month;
                    fechaConsP->year        =year - firstYear;
            }
            MemHandleUnlock(titleH);
    }
}

/***********************************************************************
* FUNCION:    IDSetDateTrigger
* DESCRIPCION: Pongo en la etiqueta del trigger la fecha seleccionada.
* PARAMETROS:  DateType date fecha seleccionada
* REGRESA:    nada
* REVISION:
***********************************************************************/
```

```
atic void IDSetDateTrigger(DateType date);
atic void IDSetDateTrigger(DateType date)

  ListPtr         lst;
  ControlPtr      ctl;
  CharPtr         label;
  int                     dayOfWeek;

  lst     =(ListPtr) GetObjectPtr(idConsultaFechaList);
  LstSetSelection(lst, selectDateItem);

  ctl     =(ControlPtr) GetObjectPtr(idConsultaFechaPopTrigger);
  label   =CtlGetLabel(ctl);

  if(DateToInt(date) == -1)
  {
        StrCopy(label,"No fecha");
        CtlSetLabel(ctl, label);
        LstSetSelection(lst,selectDateItem);
  }
  else
  {
        PrefGetPreferences(&sysPrefs);
        dayOfWeek     =DayOfWeek(date.month, date.day, date.year +
      firstYear);
        DateToDOWDMFormat(date.month, date.day, date.year +
          firstYear, sysPrefs.dateFormat,
                                                label);
        CtlSetLabel(ctl,label);
        LstSetSelection(lst, selectDateItem);
  }

*****************************************************************
FUNCION:    IDSetConsultaTrigger
DESCRIPCION. Pongo el ID seleccionado en la etiqueta del
        trigger.
PARAMETROS:  Word itemSelected- numero del campo seleccionado.
REGRESA:    Nada
REVISION:
*****************************************************************/
tic void IDSetConsultaTrigger(Word itemSelected);
tic void IDSetConsultaTrigger(Word itemSelected)

  ListPtr         lst;
  ControlPtr      ctl;
  CharPtr         label;

  lst     =(ListPtr) GetObjectPtr(idConsultaIdList);
  label   =LstGetSelectionText(lst, itemSelected);

  ctl     =(ControlPtr) GetObjectPtr(idConsultaIdPopTrigger);
  CtlSetLabel(ctl, label);
```

```
    g_idCons        =StrAToI( label);

}

/*****************************************************************
 * FUNCION:   EntregaFormInit
 * DESCRIPCION: Inicializa la pantalla MainForm.
 * PARAMETROS:  frm – apuntador a la forma MainForm.
 * REGRESA:   nothing
 * REVISION:
 *****************************************************************/
static void EntregaFormInit(FormPtr frmP)
{
    // Inicializo con la fecha de hoy
    DateSecondsToDate(TimGetSeconds(),&g_fechaCons);
    IDSetDateTrigger(g_fechaCons);

    // Inicaøilizo el ID de consulta con el primero
    IDSetConsultaTrigger(firstItem);
}


/*****************************************************************
 * FUNCION:   EntregaFormDoCommand
 * DESCRIPCION: Lleva a cabo las acciones del comando seleccionado.
 * PARAMETROS:  command  -opción id
 * REGRESA:   nothing
 * REVISION:
 *****************************************************************/
static Boolean EntregaIDDoButton(Word buttonId)
{
    Boolean handled = false;
    switch (buttonId)
        {
            case idConsultaAceptarButton:
                UEntregaConsulta(g_idCons,
            ConvertDateToSql(g_fechaCons));
                FrmGotoForm(Code3arForm);
                handled =true;
                break;

            case idConsultaCancelarButton:
                FrmGotoForm(MainForm);
                handled =true;
                break;
        }
    return handled;
}

/*****************************************************************
 * FUNCION:   EntregaFormDoCommand
 * DESCRIPCION: Lleva a cabo las acciones del comando seleccionado.
```

```
ARAMETROS:  command  - campo id
EGRESA:    nothing
EVISION:
**********************************************************************/
ic Boolean EntregaFormDoCommand(Word command)

 Boolean handled = false;
 switch (command)
        {
        }
 return handled;


**********************************************************************
UNCION:    EntregaFormHandleEvent
ESCRIPCION: Es el manejador de la pantalla "MainForm" de la aplicación

ARAMETROS:  eventP  - apuntador a la estructura  EventType
EGRESA:    true ısı el evento es manejado.

EVISION:
**********************************************************************/
lean EntregaFormHandleEvent(EventPtr eventP);
lean EntregaFormHandleEvent(EventPtr eventP)

oolean handled = false;
ormPtr frmP;
 switch (eventP->eType)
        {
        case menuEvent:
                return EntregaFormDoCommand(eventP->data.menu.itemID);

        case ctlSelectEvent:
                handled =EntregaIDDoButton
               (eventP->data.ctlSelect.controlID);
             break;

        case frmOpenEvent:
                frmP = FrmGetActiveForm();
                EntregaFormInit( frmP);
                FrmDrawForm ( frmP),
                handled = true;
                break;

        case popSelectEvent:
                if(eventP->data.popSelect.listID ==     idConsultaFechaList)
                {
                 DeterminaFecha
             (eventP->data.popSelect.selection,&g_fechaCons);
                 IDSetDateTrigger(g_fechaCons),
                }
                else if(eventP->data.popSelect.listID ==
             idConsultaList)
```

```
                {
                        IDSetConsultaTrigger
                (eventP->data.popSelect.selection);
                }
                handled =true;
                break;
        default:
                break;
        }
    return handled;
}
```

Una vez que el usuario registra el identificador de la relació contenedores, entonces se despliega la pantalla de la figura 6. que es para registrar los códigos de barras que se van a entre La PALM valida que cada contenedor leído corresponda con los d relación de entrega. A continuación se muestra el código dond realizan las acciones antes descritas.

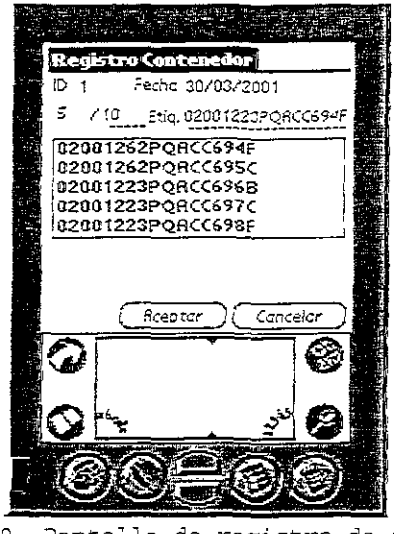


Fig. 6.10. Pantalla de registro de entrega.

```
#include <Pilot.h>
#include <SysEvtMgr.h>
#include "MainFormRsc.h"
#include "utils.h"
#include "sqlca.h"
#include "entregadb.h"

#ifdef __cpluspius //Como estoy trabajanco con C++tengo que añadir
                   //esto para el manejo del lector de codigo de
                   // barras.
extern "C"{
```

```
dif

clude "ScanMgrDef.h"        // Scan Manager constantes
clude "ScanMgrStruct.h" // Scan Manager structura
clude "ScanMgr.h"                    // Scan Manager API FUNCIONes
ef        __cplusplus

dif

****** Variables globales ********************************/
c int g_idConsulta;
c int g_totalLeidas;
c int gi_totalEtiquetas;
c DateType g_fechaConsulta;
c char g_code128[18];
c char g_etiqueta[18];
c Boolean          gb_estanTodasLeidas,

fine maxCBlen -6
**********************************************************************
UNCION:    EtiqLeidaDrawFunc
ESCRIPCION: Funcion CALLBACK para dibujar la lista
            dinamicamente
ARAMETROS:  Uint itemNum- numero de campos
                        RectanglePtr bounds- limites del rectangulo
                        CharPtr *data- No se
EGRESA:    nada
EVISION:
**********************************************************************/
f EtiqLeidaDrawFunc(UInt itemNum, RectanglePtr bounds, CharPtr *data);

f EtiqLeidaDrawFunc(UInt itemNum, RectanglePtr bounds, CharPtr *data)

 itemNum          ++;

 // hacer el llamado a la funcion de la base de datos para
 obtener la sig. etiq.
 GetNextEtiqLeida(itemNum, g_etiqueta);
 DrawCharToFitWidth(g_etiqueta, bounds);



**********************************************************************
INCION·    SetTotalLeidas
ESCRIPCION· Pongo el total de etiquetas leidas para que lo vea
            el usuario
\RAMETROS.  ninguno
EGRESA·    nada
EVISION·
**********************************************************************/
 SetTotalLeidas(void);
 SetTotalLeidas(void)
```

```
    CharPtr lc_total;
    DECL_DATETIME ldt_fecha;
    int li_id;

    ldt_fecha                =ConvertDateToSql(g_fechaConsulta);
    li_id                    =g_idConsulta;
    g_totalLeidas    =of_getEtiquetaCount(ldt_fecha, li_id, leida);

    StrIToA(lc_total,g_totalLeidas);
    SetFieldText( CodeBarTotalField, lc_total, 25, true);

}


/*****************************************************************
 * FUNCION: DecodificaDato
 * DESCRIPCION:    La llamo cuando la aplicación recibe un scanDecodeEvent,
 *                 el cual indica que una decodificación se realizó. Llamo a
 *                 "ScanGEtDecodeData" para obtener el dato leido y el tipo de CB.
 *                 Llena el control en la forma principal que despliega la
 *                 informacion. Como el codigo que leemos es grande
 *                 hay que establecer un número
 * PARAMETROS:     Ninguno.
 * RETURNED:               cero.
*****************************************************************/
Boolean DecodificaDato(void) ;
Boolean DecodificaDato(void)
{
    MESSAGE decodeDataMsg;

    int status = ScanGetDecodedData( &decodeDataMsg );
    if( status == STATUS_OK ) // Si tuve exito decodifico el dato
    {
        // Si no es leible lo trato
        if( StrNCompare( (CharPtr)decodeDataMsg.data, "NR", 2 ) == 0 )
        {
        // FrmCustomAlert(CodigoLeidoAlert,"NoLectura","","");
        }
        else
        {
         StrCopy( g_code128, (char*)&decodeDataMsg.data[maxCBlen] );
         SetFieldText( CodeBarEtiquetaField, g_code128, 25, true);
        }
    }

    return true;
}

/*****************************************************************
 * FUNCION: HandleReadCodeBar
 * DESCRIPCION:    La llamo cuando se realiza la lectura de un código
 *                 de barras. Llevo a cabo las operaciones de:
 *                              - Obtener si aún hay etiquetas sin leer
```

```
                                        - Cerrar el cursor de etiquetas leidas
                                        - Decodificar el código de barras
                                        - Validar que el código de barras pertenezca
                    a la consulta
                                        - Abrir el cursor de etiquetas leidas
                                        - Mostrar la lista nueva
ARAMETROS:      Ninguno.
EGRESA:         cero.
**********************************************************/
d HandleReadCodeBar();
d HandleReadCodeBar()

 int li_totalNoLeidas;
 DECL_DATETIME ldt_fecha;
 int       resultado;

 // Decodifico el dato
 DecodificaDato();
 ldt_fecha          =ConvertDateToSql(g_fechaConsulta);

 // Si no esta todas leidas actuo
 if( !gb_estanTodasLeidas){
        // Pongo que no estan todas leidas
        gb_estanTodasLeidas     =false;
        // Cierro el cursor de etiq. leidas para no ver datos sucios.
        CerrarCursorEtiqLeida();
        // Si la etiqueta existe  en la base de datos abro el cursor
/ de leidas
        resultado          =GetEtiqLeida(ldt_fecha, g_idConsulta, g_code128);
        if (resultado == 0){
                AbrirCursorEtiqLeida(ldt_fecha, g_idConsulta, leida);

                // Le muestro al usuario cuantas leidas lleva
                SetTotalLeidas();

                // Inicio la nueva lista de leidas
                IniciaLista(CodeBarEtiquetaList, g_totalLeidas,
       (ListDrawDataFuncPtr) EtiqLeidaDrawFunc, false);

                // Dibujo la lista
                  LstDrawList((ListPtr),
                GetObjectPtr(CodeBarEtiquetaList));
        }
        else
                if(resultado == -1001)// La etiqueta ya se leyo.
                        FrmCustomAlert(ErroretiquetaAlert,"ya ha sido
                          leida.", NULL, NULL),
                else // La etiqueta no existe.
                        FrmCustomAlert(ErroretiquetaAlert,"no existe en la
                        consulta.", NULL, NULL),
 }
 else{
        // Muestro todas las etiquetas de la consulta
```

```
            FrmCustomAlert(TotalEtiquetasAlert, "por leer", NULL, NULL);
        }
        if(gi_totalEtiquetas        == g_totalLeidas)
            gb_estanTodasLeidas =true;
}

/*****************************************************************
 * FUNCION:    CodebarFormInit
 * DESCRIPCION: Esta rutina inicializa la forma donde se leen codigos de barras.
 * PARAMETROS:  frm - Apuntador a la forma para lectura de codigo de barras.
 * REGRESA:    nada
 * REVISION:
*****************************************************************/
static void CodebarFormInit(FormPtr frmP)
{
    CharPtr lc_idConsulta;
    CharPtr lc_fechaConsulta;
    DECL_DATETIME ldt_fecha;
    int                         li_totalNoLeidas;

    SystemPreferencesType sysPrefs;

    // Inicializo el lector de   codigo de barras
    AbreLector();

    PrefGetPreferences(&sysPrefs);

    // Tomo los datos de la consulta que estan en la base de datos
    CEntregaConsulta(&g_idConsulta, &g_fechaConsulta);

    // Los pongo en forma de cadena
    StrIToA(lc_idConsulta,g_idConsulta);

    DateToDOWDMFormat(g_fechaConsulta.month, g_fechaConsulta.day,
                        g_fechaConsulta.year + firstYear, sysPrefs.longDateFormat,
                        lc_fechaConsulta);

    // Pongo los valores en los campos correspondientes
    SetFieldText( CodeBarIdField, lc_idConsulta, 18, true);
    SetFieldText( CodeBarFechaField, lc_fechaConsulta, 25, true);

    ldt_fecha       =ConvertDateToSql(g_fechaConsulta);

    // Pongo el total de etiquetas leidas para la consulta
    SetTotalLeidas();

    // Obtengo cuantas no leidas hay
    li_totalNoLeidas =of_getEtiquetaCount(ldt_fecha, g_idConsulta,
                        noLeida);
    if(li_totalNoLeidas >0)         gb_estanTodasLeidas    =false;
    else
        gb_estanTodasLeidas    =true;
```

```
// Sumo las leidas y las no leidas para poner cuantas hay
gi_totalEtiquetas= li_totalNoLeidas + g_totalLeidas;

// Las muestro al usuario
StrIToA(lc_idConsulta, gi_totalEtiquetas);
SetFieldText( CodeBarGlobalField, lc_idConsulta, 18, true);

AbrirCursorEtiqLeida(ldt_fecha, g_idConsulta, leida);
iniciaLista(CodeBarEtiquetaList, g_totalLeidas, (ListDrawDataFuncPtr) EtiqLeidaDrawFunc, true);


*************************************************************
UNCION:    TareasDeFin
ESCRIPCION: Realiza las tareas de fin de la pantalla
ARAMETROS:  ninguno
EGRESA:    nada
EVISION:
*************************************************************/
c void TareasDeFin(void)

// Cierro el lector
CierraLector();

// Cierro el cursor de etiquetas leidas
CerrarCursorEtiqLeida();

// Regreso a la forma de identificasion de la consulta
FrmGotoForm(IdConsultaForm);

*************************************************************
INCION:    CodebarDoCommand
ESCRIPCION: Ejecuta el comando especificado.
 RAMETROS:  command  - menu item id
GRESA:    nada
VISION:
*************************************************************/
 Boolean CodebarDoCommand(Word command)

Boolean handled = false;

switch (command)
        {
        break;
        }
return handled;


*************************************************************
NCION:    CodeBarDoButton
SCRIPCION: Ejecuta la acción correspondiente al boton seleccionado
RAMETROS:  buttonId  - Id del boton
CRESA:    true - si pudo procesar, en caso contrario false
```

```
 * REVISION:
 ***************************************************************************/
static Boolean CodebarDoButton(Word buttonId)
{
    DECL_DATETIME ldt_fecha;
    Boolean       lb_continuar=false;

    Boolean handled = false;
    switch (buttonId)
        {
            case CodeBarAceptarButton:
                // Realizar las tareas de fin.
                ldt_fecha    =ConvertDateToSql(g_fechaConsulta);
                if(gb_estanTodasLeidas)
        {// Si se han leido todas, las pongo como
         //aceptadas
                    CambiaEstadoConsulta(ldt_fecha, g_idConsulta,
                aceptada);
                    TareasDeFin();
                }
                else{
                    // Si no se han leido, le aviso al usuario
                    if(   FrmAlert(ParcialEtiquetasAlert) == 1)
            {// No continua. Pongo todas como no leidas.
                        CambiaEstadoConsulta(ldt_fecha,
                g_idConsulta, noLeida);
                        TareasDeFin();
                    }
                }
                handled =true;
                break;

            case CodeBarCancelarButton:
            // Realizar tarea de cancelacion y regresa al menu
     // principal
                TareasDeFin();
                handled =true;
                break;

        }
    return handled;
}

/****************************************************************
 * FUNCION:    CodebarFormHandleEvent
 * DESCRIPCION: Esta rutina es el manejador de eventos para la pantalla
 *          "CodeBarForm" de esta aplicación.
 * PARAMETROS:  eventP  - apuntador a la estrucutra EventType
 * REGRESA:    true isi el evento ha sido manejado y no debe ser pasado a otro manejador
 *          de eventos de mayor nivel.
 * REVISION:
 ****************************************************************/
Boolean CodebarFormHandleEvent(EventPtr eventP);
```

```
an CodebarFormHandleEvent(EventPtr eventP)

olean handled = false;
rmPtr frmP;


switch (eventP->eType)
        {

        case menuEvent:
                return CodebarDoCommand(eventP->data.menu.itemID);



        case ctlSelectEvent:
                handled =CodebarDoButton
                (eventP->data.ctlSelect.controlID);
        break;



        case frmOpenEvent:
                frmP = FrmGetActiveForm();
                FrmDrawForm ( frmP);
                CodebarFormInit( frmP);
                LstDrawList((ListPtr)GetObjectPtr(CodeBarEtiquetaList));
                handled = true;
                break;



        case scanDecodeEvent:
                HandleReadCodeBar();
                handled = true;
                break;

        default:
                break;
        }
eturn handled;
```

Apéndice D

INFRAESTRUCTURA DEL SISTEMA

Red de comunicaciones

istema de contenedores debe estar implementado en todas las locales de la Dirección de Emisión que se encargan del uc y el flujo de piezas, así como en las sucursales que den de ella. Las áreas locales se encuentran en un área ingida que comprende un par de edificios. Las sucursales se izan en distintas áreas geográficas del país. Ambos puntos intercambiar información entre ellos a través del sistema de nedores. Para lograr lo anterior se tiene una red de icaciones *WAN(Wide Area Network*: Red de amplia cobertura) que dentro de sus características más importantes las siguientes:

- Cubre una amplia superficie o área geográfica Abarca más allá del conjunto de edificios donde se localizan las áreas locales, para conectar equipos terminales de datos(nodos), a distancias que llegan a miles de kilómetros, como es en el caso de las sucursales.

- Utiliza medios de telecomunicación para la transmisión de datos.

- El modo de transmisión de los datos es por paquetes, o Retransmisión de tramas, como *Frame Relay*.

rotocolo de comunicación es TCP/IP (*Transmission Control co./Internet Protocol*). Este protocolo fue desarrollado por el rtamento de defensa de los Estados Unidos de América como un [illegible]

por distintos proveedores en una red de redes(Internet).

Las capas que componen el protocolo están divididas de la sigui forma:

- *IP* - Es responsable de mover los paquetes de datos d nodo hacia otro. Cada paquete es llevado a su destino base en una dirección formada de cuatro *bytes*(direc *IP*). La dirección *IP* es asignada por las autoridades regulan Internet y son distribuidas en ra determinados a las organizaciones que solic direcciones *IP*.

- *TCP* - es responsable de verificar la entrega correct los datos del cliente al servidor. Es probable qu pierdan datos durante la transmisión. *TCP* soporta detección de datos perdidos o de errores y retransmite hasta que son recibidos completamente.

- *Sockets* - es el nombre que se la da al conjunto funciones que proveen el acceso a *TCP/IP* en las may de los sistemas.

La red sobre la que trabaja el sistema de contenedores es infraestructura que el Banco Central, como organismo central que depende la Dirección de Emisión, adquirió de acuerdo a necesidades específicas de la Institución. De esta forma el sis de contenedores utiliza una red WAN y el protocolo de comunica TCP/IP. Cabe señalar que cada computadora en la que funcion sistema tiene un dirección IP única, la cual se valida cada vez se intenta entrar al sistema para verificar que realmente t derecho de hacerlo. De este modo se tiene que ninguna computa tiene puede accesar al sistema si no tiene los privilegios hacerlo.

D.2. Características del equipo de Código de Barras

El sistema de contenedores se sustenta en la aplicación de tecnología de código de barras(CB). Sin embargo, no se cuenta el equipo necesario para cumplir con los requerimientos del usu descritos en el capítulo III. Para solventar este problema reunieron un conjunto de características que el equipo de códig barras debe cumplir.

. Terminal portátil para lectura de código de barras

o a que en las bóvedas donde se almacenarán los contenedores
puede tener instalado equipo de cómputo de forma permanente,
ayude en el registro de los movimientos que ahí se llevan a
y/o en la lectura de los códigos de barras de las unidades de
ue, es necesario tener terminales portátiles para lectura de
o de barras. A continuación se describen las características
ebe cumplir la terminal portátil.

- MANEJO DE DATOS
  - Se requiere que en la terminal se puedan *almacenar* y *validar* datos de forma local, para que el usuario pueda capturar y manejar transacciones, incluso cuando no esté conectado a la base de datos institucional. De lo contrario el usuario tendría que desplazarse nuevamente a donde está el material y/o efectuar reacomodos de éste dentro de las bóvedas para volver a capturar. También se reduce el riesgo de equivocación al no haber doble captura de datos y también el uso de papel en algunas áreas.

  - Debe permitir que se instalen versiones ligeras de la base de datos *SYBASE*, ya que bajo esta plataforma es donde están las bases de datos institucionales, incluyendo la del sistema de contenedores con la que la terminal intercambiará información, tal como: catálogos institucionales, reglas de validación y operaciones. Además de que en estas versiones ligeras se pueden utilizar todas las características de *SQL* para el almacenamiento de datos, extracción de información y manipulación. También es posible incluir la información corporativa en las terminales y disponer de ella en cualquier lugar y momento en que se necesite.

- TRANSMISIÓN DE DATOS
  - Debido a las características físicas de las bóvedas, y a que no es necesario transmitir la información por radio frecuencia para tenerla al instante, se deben tener cunas de transmisión para comunicar la terminal portátil con la computadora personal, ya que el usuario conectará la terminal a la base de datos institucional para transmitir por lotes los datos capturados

- Se requiere que durante la transmisión se lle control de los datos enviados para que solamen transmita la información adecuada y no haya dupli de datos.

- DESARROLLO DE APLICACIONES
  - Las herramientas para desarrollar aplicaciones, terminal, deben ser accesibles y manejar el len C/C++ para garantizar la portabilidad de aplicaciones desarrolladas, así como para evita uso de un lenguaje de desarrollo propietario impida la creación de aplicaciones acordes a necesidades específicas del sistema de contenedore

- MOTOR DE LECTURA DE CÓDIGOS DE BARRA
  - El motor de lectura debe leer los códigos más co de una dimensión, específicamente el *código 128*, que con base en este estándar se generarán los có de barras para el proyecto.

  - La distancia mínima entre el lector y el códig barras que se esta leyendo debe ser de 50 cm, deb que los contenedores estarán apilados y se deben los códigos asociados con éstos.

- COMPATIBILIDAD CON EL AÑO DOS MIL
  - Debe ser compatible tanto en hardware como en *sof* con el año dos mil, debido a que en el registro d transacciones es muy importante saber la fecha e éstas se realizaron.

- MEMORIA
  - Debe permitir el almacenamiento de dato aplicaciones en la memoria no volátil de tal form al recargar o cambiar la pila no se borren ni datos ni las aplicaciones. En este mismo sentido, permitir que los datos se borren sólo a través aplicación correspondiente y las aplicaciones borren con la indicación específica de desarrolladores por medio de funciones exclusiva la terminal.

  - Debe tener mínimo 2 MB de RAM y ROM, ampli opcionalmente según las necesidades de la aplicaci

- DISEÑO Y ERGONOMÍA

- Facilidad para ser portadas, ya que estas lectoras serán utilizadas por el personal responsable de las bóvedas, quienes en forma continua durante la jornada de trabajo hacen movimientos de entrada/salida de material hacia los sectores de trabajo y de traslado entre las bóvedas, por eso es que la lectora debe ser ligera y cómoda para portar tal que no cause cansancio o fastidio al usuario.

- Deberán tener una cubierta resistente para uso en entornos industriales agresivos, que soporte caídas de mínimo 1m. en concreto, ya que el uso que se le dará a estos equipos será continuo y dentro de un entorno de este tipo, por lo que es necesario que la cubierta sea de un material físicamente resistente a condiciones de trabajo pesado.

- Cumplir con normas de aislamiento contra polvo y lluvia que garanticen el funcionamiento del equipo bajo esas condiciones.

- Debe ser de fácil operación, no debe ser necesario ningún conocimiento técnico. Su manejo debe ser funcional, de tal manera que no cause confusiones que puedan ocasionar errores de captura.

- Debe tener batería recargable de ion-litio, que dure cargada mínimo 10 horas.

- El cargador de baterías o soporte para el lector con cargador de baterías integrado, en ambos casos puede estar o no integrado en un solo conjunto a un soporte de para transmisión de datos. El cargador de batería debe tener un indicador que haga notar cuando se está cargando la batería y cuando termina.

- Indicador de lectura por sonido, para que el usuario sepa cuando la lectura se ha realizado y no cause confusión en si ésta se llevó o no a cabo.

- Pantalla antirreflejante y de alto contraste para poder leer los datos en lugares con mucha o poca luz, tales como bóvedas o lugares al aire libre.

Cuna de transmisión de la terminal portátil a la computadora

La transmisión de un dispositivo especial de esta lectora

remotas en la cual se coloca ésta para que se realic intercambio de información entre una computadora y la te portátil.

- RANURAS
  - El soporte debe ser de una o más ranuras sincronizar la información de una o más terminale la de la computadora.

- CABLE DE CONEXIÓN
  - El cable serial para conectar la computadora PC soporte donde se coloca la terminal portátil, ya se tendrá un puerto serial de la computadora ded exclusivamente para la terminal

  - El conector del cable serie que se enchufa computadora PC o compatible deberá ser DB9 hembra que las computadoras con que cuenta la Instit tienen este tipo de entrada para ese puerto.

  - Este cable deberá ser capaz de proveer comunic entre la computadora PC y el dispositivo transferencia de comunicación de la terminal port ya que a través de él pasarán los datos capturad un lugar a otro.

D.2.3. *Software* de desarrollo para la terminal portátil

- COMPATIBILIDAD
  - Debe correr bajo el Sistema Operativo Windows posterior porque las computadoras donde desarrollará la aplicación utilizan este Si Operativo.

  - Debe usar el lenguaje C/C++ para garantiza portabilidad de las aplicaciones desarrolladas, como para evitar el uso de un lenguaje de desar propietario que impida la creación de aplicac acordes a las necesidades específicas de institución.

  - El *software* debe incluir las librerías necesarias utilizar las funciones del lector de código de b de la terminal portátil.

- Debe tener librerías, programas o funciones para poder transmitir los programas desarrollados hacia la terminal.

- DISEÑO
  - Debe estar diseñado para desarrollar aplicaciones que funcionen en la terminal portátil antes descrita.

- DISTRIBUCIÓN
  - El proveedor debe distribuir el *software* de forma directa o indirecta.

. Terminales locales de código de barras

rminal local de código de barras es un dispositivo capaz de nficar códigos de barras y transmitirlos a la computadora natamente, cumpliendo una función similar a la del teclado. erminales locales necesitan estar conectadas a una computadora funcionar.

- LECTOR DE CÓDIGOS DE BARRA
  - Fuente de luz láser para ver el área que está leyendo, esto le da la seguridad al usuario experto e inexperto acerca de la lectura que está haciendo.

  - Decodificador integrado que lea los códigos más comunes de una dimensión, específicamente el código 128, ya que con base en este estándar se generarán los códigos de barras para el sistema de contenedores.

  - Debe permitir la configuración del decodificador del lector para insertar al menos el carácter ***CARRY RETURN*** al final del código leído y salte al siguiente renglón para de esta forma agilizar la captura de información.

  - El alcance del lector para leer un código de barras debe ser de 90 cm. mínimo, debido a que los contenedores estarán apilados y se deben leer los código asociados de los contenedores en los extremos.

DISEÑO Y ERGONOMÍA
  - Gatillo integrado al lector para que por medio de un disparo se realice la lectura, esto facilita el uso y minimiza la capacitación del usuario

- Indicador de lectura por sonido, para que el usu sepa cuando la lectura se ha realizado y no c confusión en si ésta se llevo o no a cabo.

- Cubierta resistente para uso en entornos industri agresivos, debido a la carga de trabajo y constantes movimientos que existen en las áreas en cuales operará.

- Debe contar con un soporte para manos libres agilice la lectura de productos similares.

- Cumplir con normas de aislamiento contra polv lluvia que garanticen el funcionamiento del eq bajo esas condiciones

• COMPATIBILIDAD
  - Debe ser Compatible con PC y la comunicación debe por el puerto serial, con una interface RS-232 ya el puerto estará dedicado exclusivamente para entrada de datos.

  - No debe requerir ningún *software* adicional para funcionamiento.

• INSTALACIÓN
  - La instalación, capacitación y puesta en marcha lector de código de barras deberá ser sencilla par invertir demasiado tiempo en estos aspectos.

• CABLE CONECTOR
  - El cable de conexión entre la terminal local y computadora PC o compatible debe ser tipo "Y" enchufe para teclado IBM PC/AT y compatibles, lo permite que en caso de que no se pueda hacer lectura mediante el láser entonces se capture medio del teclado sin necesidad de desconectar u otro.

  - La longitud del cable debe ser de al menos 2.5mts que el usuario lleve el lector a la separa suficiente para leer el código de barras y no li la lectura de los contenedores apilados.

• BASE PARA OPERACIÓN A MANOS LIBRES
  - Base para colocar el lector de código de barr efectuar lecturas sobre mostrador que permita t libres las manos del operador.

- La base debe tener la altura necesaria para que la terminal local capture los códigos de barras de acuerdo al alcance de la terminal.

- La base deberá ser la adecuada al mecanismo de la terminal local para proveer la lectura automática o a manos libres.

- Instalación, capacitación y puesta en marcha del mecanismo de lectura a manos libres.

Impresora de código de barras

la impresión de los códigos de barras en las etiquetas es ario tener impresoras especialmente diseñadas para este fin. A nuación se listan las características más importantes que cumplir.

TECNOLOGÍA

- La impresora debe ser de transferencia térmica, que utilice cinta entre la cabeza de la impresora y el medio en que se va a imprimir, extendiendo con ello significativamente la vida de la impresión.

- Se requiere que la tecnología de impresión utilizada, conjuntamente con los materiales sobre los que se imprimirá, aseguren que los códigos durarán legibles durante al menos 5 años.

COMPATIBILIDAD

- El equipo de cómputo utilizado para producir los Códigos de Barras será *"PC-compatible"*, por tanto las impresoras deberán poder utilizarse con este tipo de equipo.

- Interfaz serie RS-232 o sus sub-tipos seriales, para comunicarse con IBM PC/AT y compatibles.

- Debe incluir los *drivers* necesarios para su funcionamiento con el sistema operativo windows95 o posterior.

VELOCIDAD

- Por no requerirse una impresión del código en línea, una velocidad mínima de 6 pulgadas/segundo es suficiente.

- RESOLUCIÓN
  - Debe tener un resolución mínima 203 puntos por pu (8 puntos por milímetro) debido al volumen med impresiones.

  - Debe imprimir gráficos, textos y códigos de bar alta calidad.

- ALIMENTACIÓN DE ETIQUETAS
  - Se imprimirán volúmenes medianos de etiquetas p que deberá permitir la alimentación manual alimentación continua.

- TAMAÑO Y TIPO DEL PAPEL
  - Las impresoras deberán poder ser alimentadas rollos de etiquetas adheribles, y manejar tamañ etiqueta de por lo menos 5 x 10cm, por ser esta características de las etiquetas donde se impri los códigos.

  - El espesor del rollo será de 0.0635 a 0.254 mm (0 a 0.0100 pulgadas) ya que es un tamaño que la ma de las impresoras soporta.

  - El diámetro máximo de rollos de 152 mm con centr 38 mm a 76 mm (1.5 a 3 pulgadas) por ser un diá usado comúnmente.

- MEDIO AMBIENTE
  - Las impresoras se instalaran en áreas de produc donde estarán expuestas a golpes, polvo, vibraci etc. Será necesario entonces que las impresoras adaptadas para soportar un "ambiente industrial".

- MEMORIA
  - Memoria RAM mínima del modelo que asegure la impr de gráficos en las etiquetas.

- ALIMENTACIÓN DE PODER
  - La impresora debe funcionar correctamente con fu de alimentación de 127 Volts, 60 Hz, por se voltaje manejado en México.

- CÓDIGOS SOPORTADOS
  - Debe ser compatibles con al menos los códigos usados a nivel mundial tales como: EAN-8, EAN-13, A, UPC-E, UPC con 2 y 5 dígitos adicionales, c

39, entrelazado 2 de 5, código 128 (subconjuntos A, B, y C), CODABAR, código 93, simbología universal de embarques (*Universal Shipping Container Symbology*), UCC/EAN código 128, POSTNET, MSI PLESSEY, LOGMARS de tal forma que se garantice la correcta impresión de cada uno de ellos.

- CABLES DE CONEXIÓN DE LA IMPRESORA A LA COMPUTADORA
  - Cable de conexión serie (RS-232) para conectar la impresora a la computadora IBM PC/AT o compatibles.
  - El conector del cable serie que se enchufa a la computadora IBM PC/AT o compatible deberá ser DB9 hembra.
  - Si la impresora cuenta además con puerto paralelo (*centronics*) deberá proveerse del cable paralelo para la conexión con la IBM PC/AT, ya que es el estándar de todos los tipos de impresoras.

Rebobinadoras para impresoras de código de barras

gunas áreas es necesario que al imprimirse los códigos de s en las etiquetas, éstas queden enrolladas. Por esta razón es ario que las impresoras tengan rebobinadoras integradas, y por un dispositivo extra se describen sus características a nuación.

- COMPATIBILIDAD
  - Debe ser totalmente compatible con las características de la impresora antes descrita, para que una vez que se impriman las etiquetas éstas queden organizadas en un rollo.
- DISEÑO
  - Debe dejar las etiquetas enrolladas con la cara hacía afuera de tal forma que se vean los códigos de barras impresos.
  - Debe ser para uso industrial porque estarán en áreas de producción y donde hay muchos movimientos de paquetes.

FUENTE DE ALIMENTACIÓN
- Debe obtener la energía de la impresora a la que se conecte.

D.2.7. Despachador de etiquetas

El despachador de etiquetas es un dispositivo que va suministr una etiqueta a la vez. Un ejemplo se tiene en las filas de es de los bancos, donde el usuario toma una ficha de un despachado fichas y espera el turno de ser atendido. Las características despachador son las siguientes.

- *DISEÑO*
  - Debe tener avance automático para que cada vez qu usuario tome una etiqueta se tenga disponible siguiente.
  - Debe ser para uso industrial porque estarán en á de producción y donde hay muchos movimientos paquetes.
  - Debe aceptar etiquetas en rollos por que de rebobinadora se montarán en el despachador.
- DIMENSIONES
  - Debe manejar etiquetas de 5 cm de ancho x 10 c largo, porque es el tamaño de etiquetas que manejará.
  - Debe tener un ancho mínimo de 5 cm para poner cantidad suficiente de etiquetas enrolladas tal agilice las operaciones.
  - Debe tener un ancho máximo de 10 cm para que obstruya las operaciones y no ocupe mucho espacio.
  - Debe tener una longitud mínima de 2.5 cm y longitud máxima de 12.5 cm para no impedir desprendimiento de las etiquetas

D.3. Características de las etiquetas y cintas impresión

Como se mencionó anteriormente los códigos de barras deben impresos en etiquetas autoadheribles blancas.

En la tabla C.1. se muestran las dimensiones de la etiquet

Referencia	Descripción	Mínimo Milímetros	Máximo Milímetros
A	Ancho de la etiqueta	100	102
B	Ancho de sustrato o liner	104	105
C	Separación entre etiquetas	2.54	3
D	Longitud de la etiqueta	49	51
E	Espesor del sustrato o liner	0.05842	0.1270
F	Espesor de la etiqueta	0.05842	0.1270

Tabla D.1. dimensiones de la etiqueta autoadherible.

a figura D.1 se presenta el formato de las etiquetas a zar.

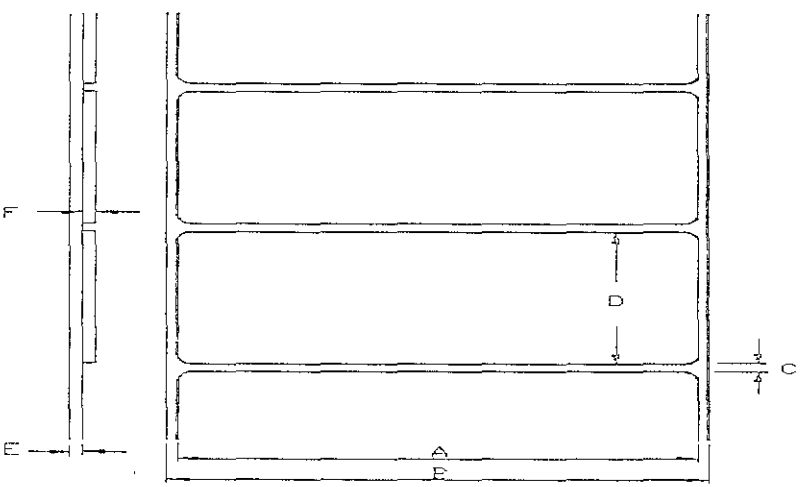


Figura D 1. Dimensiones de la etiqueta

material de las etiquetas debe ser Polipropileno ntado(BOPP, Biorientad Polipropileno), el cual es un material a la intemperie al rasgado

El adhesivo de las etiquetas debe ser agresivo, esto es presente un alto grado de dificultad al tratar de remove etiqueta de donde está adherida. El adhesivo debe tener una útil de al menos 5 años, esto significa que el adhesivo no cristalizarse y la etiqueta se despegue en menos del periodo establecido. Además, debe ser capaz de adherirse a los materi en los que están contenidas las piezas y que se lista continuación.

- Película de poliolefina multilaminada termoencogi Este es el material con el que se envuelven las pi para formar paquetes.
- Bolsa de polietileno natural. Es el material de bolsas que contienen paquetes de piezas.
- Plástico de alta y media densidad. Este es el mate del que están hechos los contenedores.

D.4. Proceso de adquisición

Con el propósito de adquirir el equipo anteriormente descrito invitó a distintos proveedores en el ramo de equipo de códig barras para que ellos elaboraran una propuesta acorde a solicitado. Después de recibir las propuestas, éstas fu analizadas y con base en este estudio se elaboró un dict técnico, que se tomó como base para adquirir el equipo.

El resultado del dictamen técnico se muestra en la tabla C.2. d cada característica es evaluada para las dos empresas, del tota 10 invitadas, que presentaron sus propuestas.

EMPRESAS →	EMPRESA 1	EMPRESA 2
PRODUCTO		

Tabla D.2. Dictamen técnico del equipo de código de barras. (Continúa)

DA A. LECTORAS TILES Y *SOFTWARE* DE ROLLO PARA LA LECTORAS TILES.		
Lectora Portátil.		
za Sybase Ultralite para nejo de información.	No especifica.	No especifica.
tible con PC	Cumple.	Cumple.
tible con el año dos mil	No especifica.	Cumple.
de RAM y 2MB de ROM.	Cumple.	Cumple.
e de luz láser.	Cumple.	Cumple.
l código 128.	Cumple.	No especifica.
ce de lectura de mínimo	Cumple.	Cumple.
uso en entornos triales y soporta a s de mínimo 1m hacía el eto.	Cumple.	Cumple.
e con la norma IP54.	Cumple.	Cumple.
orios para la ferencia de información la lectora y una PC.	Cumple.	Cumple.
ıcluye comunicación por frecuencia	Cumple.	Cumple.
ía recargable de Litio-	Cumple.	Cumple.
batería debe proveer) 6 horas de uso bajo ciones normales.	No especifica.	No especifica.

abla D.2. Dictamen técnico del equipo de código de barras.
(Continúa)

Debe soportar fuentes de alimentación de poder de 110-127V, 60HZ para recargar la batería.	No especifica.	Cumple.
Accesorios de portabilidad.	No especifica.	Cumple.
Incluye manuales.	Cumple.	Cumple.
Mantenimiento en México.	Cumple.	No especifica
Garantía del equipo por un año.	Cumple.	Cumple.
A.2. *Software* de desarrollo para lectora portátil.		
Licencia para al menos 3 PC's.	Cumple.	Cumple.
Compatible con la lectora portátil.	Cumple.	Cumple.
Debe funcionar bajo el Sistema Operativo Windows 95/98	Cumple.	Cumple.
Debe ser compatible con PC con procesador Pentium.	Cumple.	No especifica
Debe usar el lenguaje C/C++ o estar basado en este.	Cumple.	Cumple.
Compatible con el año dos mil.	Cumple.	No especifica
Librerías para utilizar el lector de código de barras.	Cumple.	No especifica
Librerías, funciones o programas para instalar y transmitir aplicaciones a la lectora.	Cumple.	No especifica

Tabla D.2. Dictamen técnico del equipo de código de barras. (Continúa)

tado al desarrollo de aciones que funcionen en ctora portátil.	Cumple.	No cumple. Lo que se pide es un *software* para que el Banco Central desarrolle aplicaciones, y lo que la empresa cotiza es el desarrollo de una aplicación.
ación de la interface en	No especifica.	No especifica.
rias para la detección errores durante el rollo de la ación(debug).	No especifica.	No especifica.
les.	No especifica.	No especifica.
DA B. LECTORAS LOCALES CÓDIGO DE BARRAS CON ORIOS.		
ectora local de código rras con accesorios.		
e de luz láser	Cumple.	Cumple.
guración del ificador	No especifica.	No especifica.
cance del lector para un código de barras ser mínimo de 20cm para os impresos de 1.00cm de X 5.00cm de longitud.	Cumple.	Cumple.
oe tener la capacidad de enar datos.	Cumple.	Cumple.
ser de uso manual.	Cumple.	Cumple.

abla D.9. Dictamen técnico del equipo de código de barras.
Cont...

Gatillo integrado al lector para que por medio de un disparo se realice la lectura.	Cumple.	Cumple.
Deberá contar con el mecanismo necesario para la lectura automática (tipo "detector de movimiento") al montar el lector sobre una base para operación a manos libres.	Cumple.	Cumple.
Indicador de lectura al menos por sonido.	Cumple.	Cumple.
Debe ser para uso en entornos industriales y soportar caídas de 1.00m hacía el concreto.	No especifica.	Cumple.
Cumplir con las normas IP54 de aislamiento contra polvo y lluvia.	No especifica.	No especifica
Debe ser compatible con PC y la comunicación debe ser al menos por el puerto serial.	Cumple.	Cumple.
No debe requerir ningún *software* adicional para su funcionamiento.	No especifica.	No especifica
B.2 Base para operación a manos libres.		
Base para colocar el lector de código de barras y efectuar lecturas automáticas que permitan tener libres las manos del operador.	Cumple.	Cumple.
Debe ser de altura ajustable.	Cumple.	No especifica

Tabla D.2. Dictamen técnico del equipo de código de barras. (Continúa)

permitir que la posición lector sea en distintos os.	Cumple.	No especifica.
ables de conexión.		
para conexión a do.	Cumple.	No cumple. Porque se requieren ambos cables y cotizan uno u otro.
para conexión a puerto l.	Cumple.	No cumple. Porque se requieren ambos cables y cotizan uno u otro.
ongitud de los cables ser de al menos 3m.	Cumple.	No cumple. Por que la longitud del cable cotizado es de 8 pies y no es la longitud mínima de 3m que se pidió.
ir manual o manuales que ifiquen claramente las terísticas de la lectora .	Cumple.	Cumple.
tener servicio de nimiento en México ocido por el fabricante.	Cumple.	No especifica.
debe garantizar el cto funcionamiento del o por lo menos un año condiciones normales de	Cumple.	Cumple.

abla D 2. Dictamen técnico del equipo de código de barras
(Continúa)

PARTIDA C. IMPRESORAS DE CODIGO DE BARRAS CON ACCESORIOS.		
C.1 Impresoras de código de barras con accesorios.		
Para conexión con PC a través del puerto paralelo.	Cumple.	Cumple.
La impresora debe ser de transferencia térmica. Debe soportar cintas(Ribbons) de cera, cera-resina, resina; con centros cilíndricos de diámetro interior de 1 pulgada(tolerancia -/- 0.1 pulgada);	Cumple.	Cumple.
longitud mínima 360m y máxima de 450m. Las cintas tendrán la cera o cera-resina o resina en la cara interior.	Cumple.	No especifica
Debe tener como mínimo una memoria total 2MB.	Cumple.	Cumple.
Debe poder imprimir gráficos en archivos: .BMP, .JPG, etc..	Cumple.	Cumple.
Es necesario que las impresoras estén adaptadas para soportar un ambiente industrial.	Cumple.	No especifica
La impresora no debe ser portátil.	Cumple.	Cumple.
Debe ser compatible con los códigos más usados a nivel mundial y al menos con el código 128.	Cumple.	No especifica

Tabla D.2. Dictamen técnico del equipo de código de barras. (Continúa)

incluir los drivers arios para su onamiento con el sistema tivo Windows 95/98 y ser tibles con el año dos	Cumple.	No especifica.
requiere una velocidad a para imprimir en una 2000 etiquetas de 10cm cho X 5cm de longitud.	No especifica.	No especifica.
tener un resolución a de 8ptos/mm(203 dpi).	Cumple.	Cumple.
imprimir gráficos, s y códigos de barras de calidad.	Cumple.	Cumple.
soportar rollos con las etas autoadheribles adas con la cara hacía a.	No especifica.	No especifica.
soportar rollos de etas autoadheribles con tro exterior de 8 das máximo	Cumple.	No especifica.
soportar rollos de etas autoadheribles iadas en centro drico de cartón con un tro interior de 3 das(tolerancia +/- 0 1 da).	Cumple.	No especifica.
manejar tamaños de eta de por lo menos 10cm cho x 5cm de longitud.	Cumple.	No especifica.
presora debe funcionar uentes de alimentacion .0-127V, 60Hz	Cumple.	No especifica.

Tabla D 2. Diseño técnico de equipo de código de barras

Debe incluir los cables de conexión de la impresora al puerto paralelo de la PC.	Cumple.	No especifica
Debe incluir los cables de conexión a la fuente de alimentación de poder.	Cumple.	No especifica
Manuales	Cumple.	No especifica
Se debe garantizar el correcto funcionamiento del equipo por lo menos un año bajo condiciones normales de uso.	Cumple.	Cumple.
Debe tener servicio de mantenimiento en México y reconocido por el fabricante.	Cumple.	No especifica
C.2 Rebobinadora para impresoras de código de barras.		
Debe estar integrada a la impresora	Cumple.	Cumple.
Debe rebobinar rollos de etiquetas autoadheribles con diámetro exterior de 8 pulgadas máximo.	Cumple.	No especifica
Debe utilizar rollos de etiquetas autoadheribles enrolladas en centro cilíndrico de cartón con un diámetro interior de 3 pulgadas(tolerancia +/- 0.1 pulgada).	No especifica.	No especifica
Debe dejar las etiquetas(de al menos 10cm de ancho X 5cm de longitud) enrolladas con	No especifica.	No especifica

Tabla D.2. Dictamen técnico del equipo de código de barr
(Continúa)

ra hacía afuera de tal que se vean los códigos rras impresos.		
incluir los manuales arios que describan las terísticas especificas quipo.	Cumple.	No especifica.
tener servicio de nimiento en México y ocido por el fabricante.	No especifica.	No especifica.
debe garantizar el cto funcionamiento del o por lo menos un año condiciones normales de	Cumple.	Cumple.
DA D. DESPACHADORES DE ETAS.		
chadores de etiquetas.		
tener avance automático que cada vez que el lo tome una etiqueta se disponible la ente.	Cumple	Cumple.
ser para uso en entornos riales.	No especifica	No especifica.
manejar etiquetas de de ancho x 5cm de ud.	No especifica.	No especifica.
soportar rollos de tas autoadheribles con ro exterior de 8 las máximo.	Cumple.	No cumple porque el rollo máximo que soporta es de 3.93 pulgadas.
soportar rollos de tas autoadheribles	Cumple.	No cumple porque el diámetro interior es de 1

Tabla D.2 Datasheet técnico del equipo de código de barras adquirido

enrolladas en centro cilíndrico de cartón con un diámetro interior de 3 pulgadas(tolerancia +/- 0.1 pulgada).	Cumple.	pulgada.
Debe tener servicio de mantenimiento en México y reconocido por el fabricante.	Cumple.	No especifica
Se debe garantizar el correcto funcionamiento del equipo por lo menos un año bajo condiciones normales de uso.	Cumple.	Cumple.
Debe incluir los manuales necesarios que describan las características especificas del equipo.	Cumple.	No especifica

Tabla D.2. Dictamen técnico del equipo de código de barras.

Después de analizar los resultados se determinó que la empres cumple con todos los requerimientos que se solicitaron, mien que la empresa 2 no cumple en algunos aspectos como es el caso *software* para el desarrollo de aplicaciones en la lectora portá ya que lo que se pide es un *software* para que el desarrollo aplicaciones, y lo que la empresa propone es el desarrollo de aplicación. De esta forma el equipo que se adquirió fue correspondiente a la empresa 1.

D.5. Características del equipo elegido

D.5.1. Lectora portátil

En el rubro de la lectora portátil se eligió el modelo SPT170 Symbol©, el cual se muestra en la figura D.2. y está basado e plataforma PALM©. Utiliza el sistema operativo palm OS 3.2© lectora SPT1700 es adecuada para el uso en ambientes industri debido a la cubierta contra golpes que posee. Es de fácil uso que el usuario introduce información y accede a las opciones

tocar la pantalla con una pluma de plástico que se incluye accesorio de la terminal. El modelo SPT1700 tiene la terística de poder almacenar los datos en una base de datos ional, cuya estructura reside en la memoria no volátil de Así, el manejo de datos se hace a través de sentencias SQL como *SELECT, INSERT, UPDATE y DELETE*.

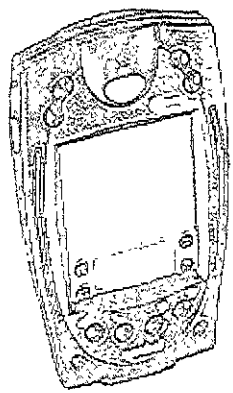


Figura D.2. Lectora portátil SPT1700.

s, durante la transferencia de datos hacia una computadora, la ra lleva el control de aquellos renglones que deben mitirse, así como de aquellos que ya fueron transmitidos.

ual forma, lleva el control de los datos que solicita a la tadora y que son transferidos a la lectora portátil, como es aso de los catálogos que ésta usa para validar que la mación que se introduce es correcta.

portante considerar que el ambiente de desarrollo es en C++, y pantallas deben estar diseñadas de tal forma que el usuario duzca la menor información posible y solo deba seleccionar la a lectora le propone. El uso de la lectora significa un avance desarrollo de aplicaciones móviles en el Banco Central.

Lectora local

ctora local seleccionada es el modelo LS2100 de Symbol©, que estra en la figura D.3. Este tipo de lectoras están diseñadas amente para funcionar en aplicaciones de inventarios, por o el uso de la misma permite una fácil de usar,

incluso durante largos periodos. El diseño ergonómico permite su uso sea intuitivo y el tiempo de capacitación con el usuari reduzca. Otra característica que distingue a la lectora es la estar cubierta por un caparazón rígido que la protege contra ca y otros factores ambientales como lluvia o polvo.

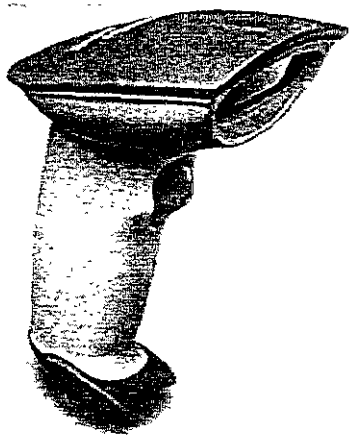


Figura D.3. Lectora local LS2100

D.5.3. Impresora de código de barras

La impresora de transferencia térmica utiliza una cabeza impresión compuesta por una fila de elementos control individualmente, que transmiten calor a una cinta de transfere térmica generalmente compuesta de resina. El calor de la impre derrite el material de la cinta (resina) y lo transfiere a etiqueta, los lubricantes de la cinta cumplen además con la fun de alargar la vida de las cabezas de impresión.

La impresora modelo DMX-I-4206 de DATAMAX©, que se muestra e figura D.4., es la impresora seleccionada. Esta impresora impri una velocidad de 15 mm por segundo, la cual se considera una velocidad.

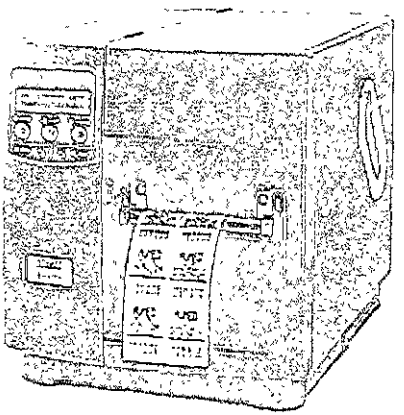


Figura D.4. Impresora DMX-I-4206.

anexo presente se describió el proceso para la adquisición y aracterísticas del equipo de código de barras utilizado para istema de contenedores. De igual forma se describió la estructura del Banco Central sobre la que se implementó el ma de contenedores.

GLOSARIO

Usuario.- Sucursal de una institución bancaria facultada para zar operaciones de efectivo con el Banco Central.

.- Unidad de empaque que consta de 5 ó 6 paquetes dependiendo es alta o baja denominación (25000 o 30000 piezas ctivamente).

a.- Lugar físico destinado al almacenamiento de billete o a.

nedor.- Máxima unidad de empaque que está formada por 10 s (250,000 o 300,000 piezas según sea el caso).

nedor Pico - Unidad de empaque que resulta al final de la da o del tiro que no llega a completar las 10 bolsas (menos de 00 o 300,000 piezas según sea el caso)

to.- Diferencia del material respecto a un patrón en el que se lecen las características, de apariencia y calidad del billete e trate.

inación alta.- Impresión solo de fondos, denominaciones altas .

ınación baja.- Impresión de fondos y reversos denominaciones (20, 50, 100).

o Físico.- Estado de desgaste que tiene cada billete o moneda.

.- Material defectuoso clasificado como correcto o material e al patrón(según criterios establecidos) que se clasifica defectuoso

.a.- Unida que consta de 100 piezas de billete terminado y que ınida por una cintilla.

: Conjunto de diez mil hojas de papel seguridad impreso (20).

Mazo.- Mínima unidad de empaque conformada por 10 fajilla 1000piezas, unida por una cintilla o fleje según sea el caso.

Maculatura.- Material mal impreso o manchado y que no se p utilizar.

Machimbre.- Cinta de plomo, para cerrar las bolsas.

Paquete Numeración Chica.- Unidad de empaque que consta de 5 ma cuyo número de folio de los billetes que la conforman, en casilla de posición, están entre 0 y 4 (5000 piezas).

Paquete Numeración Grande.- Unidad de empaque que consta de 5 m cuyo número de folio, de los billetes que la conforman, en casilla de posición, están entre 5 a 9 (5000 piezas). Conjunto 500 hojas de papel seguridad impreso.

Resma.- Conjunto de 500 hojas de papel seguridad impreso.

Ventanilla.- Lugar físico donde se realizan distintos procesos los billetes o monedas.

BIBLIOGRAFIA

r, Martín, *"UML gota a gota"*, Addison Wesley Longman de o, S.A. de C.V., México, 1999

acobson, G. Booch, J. Rumbaugh, *"El Proceso Unificado de rrollo de Software"*, Pearson Edicación S.A., Madrid, 2000

n, Craig, *"UML y Patrones, Introducción al Análisis y Diseño tado a Objetos"*, Prentice Hall, México, 1999

acobson, M. Christerson, P. Jonsson, G. Övergaard, *"Object ted Software Engineering"*, Addison Wesley, E.U.A., 1995

n James, *"Análisis y Diseño Orientado a Objetos"*, Prentice México, 1994

E. Ruble, *"Análisis y Diseño Práctico para Sistemas te/Servidor con GUI"*, Prentice Hall, México, 1998

, Rumbaugh, Jacobson, *"The Unified Modeling Language User "*, Addison Wesley, E.U.A., México, 1999

Gamma, Richard Helm, Ralph Johnson, John Vlissides, *"Design rns"*, Addison Wesley, E U.A., 1995

Giguére, *"Palm Database Programming: The Complete Developer´s "*, Wiley Computer Publishing E.U.A., 1999

Rhodes and Julie Mckeehan *"The Developer´s Guide"*, O'Reilly, ., 1999

alite Developer's Guide", SYBASE

//www.banxico.org.mx
//www.inegi.org
//www.[illegible]