UNIVERSIDAD DE ORIENTE
NÚCLEO DE ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

DESARROLLO DE UN SOFTWARE PARA EL CONTROL DE INVENTARIO DE PRODUCTOS TERMINADOS PARA LOS DEPARTAMENTOS DE ATENCIÓN AL CLIENTE, LA LÍNEA DE PRODUCCIÓN "SECTOR BETA", Y DESPACHO EN UNA EMPRESA ALIMENTOS

REALIZADO POR:

Rubén Darío García Pérez

Trabajo de grado presentado en la Universidad de Oriente como requisito parcial para optar al título de

INGENIERO EN COMPUTACIÓN

Barcelona, Abril de 2009

UNIVERSIDAD DE ORIENTE
NÚCLEO DE ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

DESARROLLO DE UN SOFTWARE PARA EL CONTROL DE INVENTARIO DE PRODUCTOS TERMINADOS PARA LOS DEPARTAMENTOS DE ATENCIÓN AL CLIENTE, LA LÍNEA DE PRODUCCIÓN "SECTOR BETA", Y DESPACHO EN UNA EMPRESA ALIMENTOS

ASESORADO POR:

_____

Ing. Mónica Saettone

Barcelona, Abril de 2009

UNIVERSIDAD DE ORIENTE
NÚCLEO DE ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

DESARROLLO DE UN SOFTWARE PARA EL CONTROL DE INVENTARIO DE PRODUCTOS TERMINADOS PARA LOS DEPARTAMENTOS DE ATENCIÓN AL CLIENTE, LA LÍNEA DE PRODUCCIÓN "SECTOR BETA", Y DESPACHO EN UNA EMPRESA ALIMENTOS

JURADO CALIFICADOR:

Ing. Víctor Mújica	**Ing. Gabriela Veracierta**
Jurado Principal	Jurado Principal

Ing. Mónica Saettone
Asesor Académico

Barcelona, Abril de 2009

RESOLUCIÓN

ARTÍCULO N° 44 Del Reglamento de Trabajo de Grado

"Los trabajos de grado son de exclusiva propiedad de la Universidad y sólo podrán ser utilizados para otros fines con el conocimiento del Consejo de Núcleo respectivo, quién lo participará al Consejo

RESUMEN

En el presente trabajo de grado, consiste en el desarrollo de un software para la automatizacion de las operaciones de, realización, aprobación y despacho de las ordenes de carga, asi como la creacion y actualizacion del inventario,en la empresa alimentos Super S planta Barcelona C.A, perteneciente al grupo la caridad, este software facilitara los procesos,llevados a cabo los operadores de los diferentes departamentos involucrados.

Este proyecto plantea optimizar y reducir los tiempos y los errores, en la realización de las operaciones llevadas a cabo en los departamentos de: Atención al Cliente, Cuentas por Cobrar, Despacho, Linea de Producción, Romana, considerando los adelantos tecnologicos en el area de software.esto con el fin de obtener reportes mas confiables y reales.

Los parámetros y requisitos seran obtenidos a traves del analisis de los reportes realizados manualmente, por los diferentes departamentos involucrados en el proceso, asi como el analisis de los requisitos y necesitades del cliente, a fin de obtener una herramienta que permita, obtener reportes y resultados confiables, y reales, acorde con la existencia tanto de ordenes como de productos.Esta herramienta se realizará mediante la utilización de algoritmos lógicos y matemáticos que integrados permitan llevar desarrollar las diferentes operaciones por los operadores en los departamentos involucrados en el proceso, de forma, confiable, rápida y eficaz, asi como la manipulación de la data almacenada por los diferentes procesos.

DEDICATORIA

A mi madre por su apoyo y convicion, a mi padre y abuela que estan observandome desde la eternidad, a mi hermano David, demostrandole que sin importar las circunstancias se pueden lograr los sueños y las metas, con dedicacion y perseverancia, a mi hermano Cruz que me dio su apoyo y consejo, en los momentos difíciles, de este largo trayecto.

AGRADECIMIENTOS

- A mi madre: Maritza, por su ayuda y permitirme, lograr esta meta.
- A la profesora Monica Saettone por aconsejarme y asesorarme.
- A mi tia Yenni por su colaboracion, apoyo, y consejos.
- A mi supervevisor industrial: Juan Carlos Ezqueda por su ayuda en mi periodo de pasantia en la empresa Super S.
- A mi asesor industrial: Adrian Rivero por su colaboracion, y confianza que depositó en mí.
- Al grupo de T.I y S.T.I, del grupo la caridad por su apoyo logistico tecnico en las operciones realizadas en las instalaciones del grupo la caridad.
- A mis amigos Ricardo Presilla, Gonzalo Borges, Jesús Malave, Carmen Paruta (Clemen), Rossana Sanchez, Jesús Otamendi, Lisandro, Julio Martinez por su: constancia, solidaria, amistad, y soportarme en estos años de carrera.
- A mi amiga Ivy Arzola, por su peculiar apoyo y amistad
- A mi amiga Daniela Amodio “Kitty, Dan”, por su amistad, comprensión, y cariño, en los primeros año de carrera que considero los mas difíciles, por recordarme y demostrarme: que si no hay nada o poco que perder se puede ganar todo.
- A mis hermanos: Victor, Cruz, Daniel, y David; por su apoyo y constancia.

Rubén Darío García Pérez

CONTENIDO

INDICE DE FIGURAS

Descripción de Figura **Pag**

INDICE DE TABLAS

Descripción de Tabla **Pag**

CAPITULO I
EL PROBLEMA

1.1 Planteamiento del problema

En al año 1892 se fundó el primer molino de harina de trigo de la Internacional Milling Company en Minneapolis, Minesota, E.E.U.U., empresa que luego cambia su nombre por Internacional Multifoods. En 1908 la Compañía se expande hacia Canadá, y para 1925, introduce en Venezuela Harina de su Producción, lo cual propicia para 1956 la constitución en el país de la Empresa Molinos Nacionales C.A. (MONACA).

Molinos Nacionales, es una de las principales filiales de la International Multifoods, empresa Norteamericana mundialmente prestigiosa en el ramo de la elaboración y distribución de productos alimenticios. Es una Empresa Multinacional de las más grandes productoras de Alimentos de Venezuela, fundamentalmente dedicada a la producción y distribución de Harina de Trigo, Harina de Maíz, Arroz, Avena, Mezclas especiales y exclusivas para panaderías y pastelería, Productos Avícolas, Alimentos Balanceados para Animales y Condimentos, con una exitosa trayectoria desde su constitución en 1956.

Ya para Junio de 1999 la empresa La Caridad, adquiere gran parte de la empresa Monaca, en cuanto al ramo de fabricación de Alimentos para Animales, de igual forma cambia de razón social de Monaca a Alimentos Super-S, C.A. Dedicándose de esta manera solo y exclusivamente a la fabricación de alimentos para animales.

La empresa Alimentos Super-S, tiene como misión mejorar continuamente la posición como líder suplidor en el mercado Venezolano, satisfaciendo siempre las necesidades de los clientes, mediante la prestación de un servicio de calidad; a través de la identificación y análisis de las exigencias de nuestros clientes, alcanzando recíprocamente los objetivos financieros de la Corporación.

En esta empresa se tiene como visión; mantener una organización capaz de enfrentar situaciones presentes y/o inmediatas difíciles, nuestros esfuerzos tendrán que verse colmados con previsiones para un futuro retador y lleno de imprevistos. Que nos mantenga el orgullo por lo que somos y lo que hacemos.

Alimentos Súper S es una empresa manufacturera de productos alimenticios balanceados para animales, que se caracteriza por tener un proceso productivo completo. Esta empresa prepara su materia prima, la procesa, fabrica el producto final, y lo empaca para finalmente ser distribuido. Un 70% de la producción es utilizado por sus propias granjas y el 30% restante se distribuye a los clientes a nivel nacional.

Esta empresa cuenta con los siguientes objetivos:

- Satisfacer las necesidades de nuestros clientes.
- Mantener un sistema que nos permita asegurar la calidad de nuestros productos y servicios.
- Preparar a nuestro personal en la ejecución correcta de sus actividades para el desarrollo continuo de los procesos de fabricación, a fin de que sean ellos los protagonistas del proceso de calidad.
- Mantener nuestra tecnología al día, incorporando los adelantos ofrecidos por nuestra sociedad cambiante.

La empresa Alimentos Súper – S, C.A. se encuentra ubicada en la zona industrial sur II, Av. Domingo Olavarria, vía aeropuerto, entre Alimentos Polar y la Kraft, Valencia, Edo. Carabobo.

En las ultimas cuatro décadas la empresa Alimentos Súper – S, C.A ha estado integrando a sus operaciones tanto de producción como administrativas, a las nuevas tendencias tecnológicas en el campo de la ingeniería de software, y de esta forma efectuarlas de manera más eficiente, y tener una mejor competitividad en el mercado en el cual se desenvuelvan sus actividades.

El uso de la tecnología presente en la empresa se orienta a la ingeniería de software mediante el desarrollo de programas o sistemas de información, los cuales se han desarrollado según las necesidades de la empresa, y estos le permitirá a la misma, responder a las necesidades de sus clientes de forma más rápida y efectiva que llevar sus operaciones de manera manual.

La Empresa Alimentos Súper S, Planta de Barcelona es una compañía dedicada a la producción y venta de alimentos balanceados para animales. Esta empresa cuenta con siete departamentos, los cuales son: Venta y Cobranza, Administración, Recursos Humanos, Operaciones, Atención Al cliente, Recepción Y Gerencia.

En los departamentos de Atención al Cliente y la Línea de Producción "Sector βeta", las operaciones de venta y despacho de materia terminada, la elaboración de reportes u ordenes de despacho son realizadas de manera manual y con un amplio margen de errores humanos, lo que produce un atraso en el desarrollo normal de sus operaciones.

Esta empresa en la actualidad no cuenta con un software de administración de base de datos que permita: Almacenar, Modificar y/o Consultar la información

correspondiente a las transacciones tanto ínter departamentales (Atención al cliente y la línea de producción o sector βeta), como el reporte u orden de despacho de un cliente determinado. Debido a esto, las órdenes de despacho son verificadas y comparadas de manera manual con otros reportes realizados en el proceso de venta del producto "Orden de Pre Pedido y Hoja Mecanizada" y en ocasiones la información de estas órdenes no coincide y es errónea, lo que atrasa aun más las transacciones de venta del producto. Todo esto influye en que los reportes; que se manejan para realizar sus despachos, son tardíos o llega con horas de retraso, lo que produce una pérdida de tiempo considerable en el despacho del producto terminado y desperdicio en horas hombre. Los reportes son almacenados en archivos móviles lo que dificulta la búsqueda de una orden previa a la fecha presente o actual, y en algunos casos se realizan entregas a transportes equivocados y en el peor de los casos pérdida de la información.

Ante la problemática que presenta la empresa se plantea lo siguiente: El desarrollo de un software con una interfaz amigable al usuario final y una base de datos, que permitirá realizar de forma correcta la manipulación de la información, y emisión de reportes de las operaciones en los departamentos de atención al cliente y al línea de producción "Sector βeta".

Esta aplicación se desarrollará con el fin de automatizar los procesos de tramitación de los pedidos del cliente y su posterior despacho mejorando el desarrollo de dos actividades importantes en la mencionada empresa como lo son: La manipulación de la información interdepartamental y de las respectivas ordenes de despacho de productos terminados por parte de la línea de producción o sector βeta, disminuyendo así el tiempo de espera en las órdenes de despacho, y reduciendo la pérdida de horas hombre, lo que producirá un aumento en la productividad y evitar errores a futuro en la emisión de ordenes, en estos dos sectores de la empresa, y brindando de esta forma una mejor atención al cliente.

Este proyecto tendrá las siguientes limitaciones para ser llevado a buen término:

- **Físicas:** La aplicación se desarrollará y se aplicará en los departamentos de Atención al Cliente y la línea de producción o Sector Beta y en la oficina de pesada o Romana, contando con los equipos computacionales e instalaciones y equipos de red necesarios para la puesta en marcha de la aplicación.

- **Lógicas:** a solicitud de la empresa la aplicación se codificará con el Software de desarrollo de aplicaciones java 2.0 y el manejador de Base de Datos My Sql 4.0.

- **Temporales**: Para desarrollar la aplicación se cuenta con un periodo de 6 meses.

1.2 Objetivos

1.2.1 Objetivo General

Desarrollar un software para el control de inventario de productos terminados en los departamentos de Atención al Cliente, la Línea de Producción "Sector Beta", y Despacho en una empresa Alimentos.

1.2.2 Objetivos Específicos

1. Analizar la información recopilada y las operaciones de la empresa.

2. Definir los actores del sistema.

3. Establecer los requerimientos del sistema que satisfagan las necesidades de los usuarios para el desarrollo correcto del sistema.

4. Diseñar los módulos, interfaz y base de datos del sistema.

5. Codificar de los procesos e interfaz del sistema.

6. Realizar las pruebas, la integración y la documentación del software.

1.3 Marco metodológico

Para el desarrollo de este proyecto se utilizara la : Metodología Orientada a Objetos mediante Proceso Unificado de Desarrollo de Software a través de el Modelo en Espiral utilizando Lenguaje Unificado de Modelado (UML) mediante el desarrollo de casos de Uso, que permite representar el comportamiento general del sistema desde una perspectiva externa, donde se especifica que debería hacer el sistema sin definir su implementación, es decir, especificar que debería hacer y no cómo lo debería hacer. Este modelo también sirve para definir cuales son los límites del sistema, quiénes van a ser los usuarios del sistema, cuales son sus funciones o roles y cuáles son los casos de uso que interactuarán con los actores identificados para el desarrollo de software

Fase de Inicio: En esta fase se plantean, se refinan, y se concretan las ideas para llevara a cabo concretamente la siguiente fase. En esta fase se desarrollan las siguientes etapas:

Requisitos: En esta etapa se describe los requisitos del sistema, los cuales se obtienen después de reuniones y entrevistas con el cliente (incluyendo los usuarios) y los desarrolladores del sistema, acerca de lo que el sistema debe hacer y lo que no.

Análisis: En esta etapa se analizan los requisitos descritos en la captura de requisitos, mediante su refinamiento y estructuración., en esta fase se pretenden lograr dos objetivos :(1) lograr una comprensión mas precisa de los requisitos, y (2)

obtener una descripción de los requisitos que sea fácil de mantener y que nos ayude a dar estructura al sistema en su conjunto incluyendo su arquitectura.

Fase de Elaboración: En esta fase se define la arquitectura. Del software.

Esta fase comprende la siguiente etapa:

Diseño: Esta etapa tiene como propósito: formular modelos que se centran en los requisitos no funcionales y el dominio de la solución, que prepara la implementación y pruebas del sistema.

Fase de Construcción: fase en la que el software es desarrollado a partir de una línea base de la arquitectura ejecutable, hasta el punto en el que se esta listo para ser implementado y utilizado por las comunidades de usuarios.

Esta fase comprende las siguientes etapas:

Pruebas: En esta etapa se implementan los resultados de las transacciones erróneas de sistema, para lograr la validación e integridad de los datos de entrada del sistema

Implementación: Esta etapa tiene como propósito implementar el sistema en términos de componentes, es decir código fuente guiones, ficheros binarios, ejecutables, etc.

Fase transición: En esta fase del ciclo de vida del software es puesto en manos de la comunidad de usuarios, con la documentación, del sistema.

CAPÍTULO II
MARCO TEÓRICO

2.1 Antecedentes de la investigación

En la actualidad se han desarrollado programas que permiten la automatización de las operaciones de las empresas y así facilitar sus operaciones, así como en la Universidad de Oriente en cuanto al Proceso Unificado de Desarrollo de Software, proceso utilizado para el diseño y construcción del Software Integral de Gestión de Necesidades y Oportunidades, ya se han realizado trabajos de investigación donde se utiliza esta metodología. Entre los trabajos de investigación realizados, que se encuentran a la disposición para ser consultados están:

INFACO-COMERCIAL Versión 9.3 (PARA EMPRESAS COMERCIALES) Sistema o programa informático desarrollado por la empresa Magnomercado, utilizado para facturación comercial, este es una aplicación completa que le permite tener un control total sobre los inventarios, compras, facturas, notas de crédito, notas de venta; reportes de retenciones en compras y ventas, libros de compras y ventas, kardex, inventarios valorizados, contabilidad, SRI, comprobantes contables, libro diario, mayor general, balance general, de comprobación, estado de resultados, estados de cuenta, esto bajo el concepto de multi empresa y multi usuario, controla de acceso al sistema mediante clave por tipo de usuario y maneja restricciones a las opciones del sistema, presenta historiales de tareas de usuarios, incluye todo tipo de reportes, configuración por usuario de impresión de formato de facturas, notas de venta y notas de crédito, asignación a impresoras por usuario, generación de código de barras e impresión de estas, además contabilización automática de transacciones en facturación [1]

El Sistema de Información Geográfica (SIG) El SIG, desarrollado por los técnicos del consorcio de la Ciudad Monumental de Mérida, el cual permite cruzar en un instante innumerables datos hasta ahora dispersos, abre nuevas vías aún no exploradas de investigaciones científicas y también facilita a los vecinos de Mérida el acceso a la información arqueológica de todos los solares y así planificar su gestión. [2]

La Corporación Financiera Internacional (CFI), filial del Banco Internacional para la Reconstrucción y el Desarrollo, cuyo sistema de transacciones del mismo nombre, desarrollado por el equipo de la empresa Magnomercado, funciona de la siguiente manera: El CFI busca inversores interesados en los países más desarrollados y el capital proveído por éstos, es transferido a empresas privadas de países subdesarrollados cuyo capital privado no basta. [3]

Otro ejemplo de este sistema se utiliza en el de la industria naviera, el cual, por medio de su sistema de transacciones internacionales transportan diferentes tipos de carga de acuerdo a pedidos en diferentes países, siendo uno de los más transportados el petróleo, cuyos pedidos pueden ser ya sea privado o por contrato.

Electronic Data Interchange (EDI), elaborado por la empresa webMethods, es un servicio entregado por ciertas organizaciones, las cuales cobran tarifas por dar enlaces de comunicación entre empresas, además, pueden dar ciertos niveles de seguridad. EDI puede ser utilizado como un medio de transmisión de transacciones. [4]

“Desarrollo de un Software que permita la automatización de las actividades asociadas al Departamento de Admisión y Control de Estudios de la extensión región centro/sur del Núcleo de Anzoátegui de la Universidad de Oriente” realizado por Luís

Eduardo Millán González y Luís Carlos Garelli Boada, (Junio de 2007). Trabajo de Grado presentado como requisito parcial para obtener el título de Ingeniero en Computación en la Universidad de Oriente – Núcleo de Anzoátegui. Fue desarrollado usando la metodología del proceso unificado de desarrollo de software y programación orientada a objetos. [5]

"Desarrollo de una Aplicación para el Control Administrativo de una Organización Farmacéutica" realizado por Juan Carlos Otero Groba, (Febrero de 2004). Trabajo presentado como requisito parcial para obtener el título de Ingeniero en Computación en la Universidad de Oriente – Núcleo de Anzoátegui. [6]

"Desarrollo de un Software que permita el Monitoreo de la Información de Estado de un Portal Web Alojado en la Intranet Corporativa de PDVSA, utilizando la Nueva Plataforma de Microsoft.NET" realizado por Mauro London, (Mayo 2004). Trabajo de grado presentado en la Universidad de Oriente, Núcleo Anzoátegui para optar al título de Ingeniero en Computación. Este trabajo surgió debido a un requerimiento de la Gerencia de Automatización, Informática y Telecomunicaciones (AIT), en Puerto la Cruz, Estado Anzoátegui, teniendo como fin el desarrollo de un Sistema que permita el monitoreo de la información de estado de un portal web alojado en la intranet corporativa de PDVSA, S.A. (SIMPOWEB). Para el desarrollo del proyecto se empleó el Proceso Unificado de Desarrollo de Software tomando como base el Lenguaje Unificado de Modelado (UML). [7]

"Diseño De Un Sistema De Información De Inventario En El Departamento IT (Información Tecnología) de una Empresa Petrolera. Puerto La Cruz, Sector Venecia", realizado por: Hiramer Caguaripano, trabajo de grado presentado en la Universidad de Oriente Núcleo de Anzoátegui para optar al titulo de Ingeniero de Sistemas, en el cual se rediseñó un sistema de inventario por razones logísticas y de estructura interna en la empresa. Gracias a la aplicación de UML se diseñó Sisvent,

un Sistema de Inventario mejorado que toma en cuenta factores como la asignación de equipos a personal foráneo de la empresa, control de licencias para el software, generación de reportes automáticos por pantalla o impresora, también permite que un usuario pueda tener dos o más modelos iguales de un mismo equipo, en fin se diseñó de tal forma que la migración a la intranet queda servida. [8]

2.2 La empresa

La Empresa Alimentos Súper S, nace para Junio de 1999, cuando la empresa La Caridad, adquiere gran parte de la empresa Monaca, en cuanto al ramo de fabricación de Alimentos para Animales, de igual forma cambia de razón social de Monaca a Alimentos Super-S, C.A. Dedicándose de esta manera solo y exclusivamente a la fabricación de alimentos para animales.

2.2.1 Procesos que cubre

Alimentos Súper S es una empresa manufacturera de productos alimenticios balanceados para animales, que se caracteriza por tener un proceso productivo completo. Esta empresa prepara su materia prima, la procesa, fabrica el producto final, y lo empaca para finalmente ser distribuido. Un 70% de la producción es utilizado por sus propias granjas y el 30% restante se distribuye a los clientes a nivel nacional.

2.2.2 Objetivos de la Empresa:

- Satisfacer las necesidades de nuestros clientes.

- Mantener un sistema que nos permita asegurar la calidad de nuestros productos y servicios.

- Preparar a nuestro personal en la ejecución correcta de sus actividades para el desarrollo continuo de los procesos de fabricación, a fin de que sean ellos los protagonistas del proceso de calidad.

- Mantener nuestra tecnología al día, incorporando los adelantos ofrecidos por nuestra sociedad cambiante.

2.3 Marco teorico

2.3.1. Sistema de Información

Es un conjunto de funciones o componentes interrelacionados que forman un todo, es decir, obtiene, procesa, almacena y distribuye información para apoyar la toma de decisiones y el control en una organización. Igualmente apoya la coordinación, análisis de problemas, visualización de aspectos complejos entre otros. **[3]**

2.3.2. Componentes de un Sistema de Información

- **El equipo computacional:** el hardware necesario para que el sistema de información pueda operar.

- El recurso humano que interactúa con el Sistema de Información, el cual está formado por las personas que utilizan el sistema.

- El sistema de información en sí (generalmente una aplicación software).

Un sistema de información realiza cuatro actividades básicas: entrada, almacenamiento, procesamiento y salida de información. **[3]**

Entrada de Información

Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son aquellas que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos. Esto último se denomina interfases automáticas. **[3]**

Ejemplos

- Datos generales del cliente: nombre, dirección, tipo de cliente, etc.
- Políticas de créditos: límite de crédito, plazo de pago, etc.
- Facturas (interfase automático).
- Pagos, depuraciones, etc.

Almacenamiento de información:

El almacenamiento es una de las actividades o capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos. **[3]**

Ejemplos:

- Movimientos del mes (pagos, depuraciones).
- Catálogo de clientes.
- Facturas.

Procesamiento de Información

Es la capacidad del Sistema de Información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecida. Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que están almacenados. Esta característica de los sistemas permite la transformación de datos fuente en información que puede ser utilizada para la toma de decisiones lo que hace posible, entre otras cosas, que un tomador de decisiones genere una proyección financiera a partir de los datos que contiene un estado de resultados o un balance general de un año base. **[3]**

Ejemplos:

- Cálculo de antigüedad de saldos.
- Cálculo de intereses moratorios.
- Cálculo del saldo de un cliente.

2.3.3. Salida de Información:

La salida es la capacidad de un Sistema de Información para sacar la información procesada o bien datos de entrada al exterior. Es importante aclarar que la salida de un Sistema de Información puede constituir la entrada a otro Sistema de Información o módulo. En este caso, también existe una interfaz automática de salida. Por ejemplo, el Sistema de Control de Clientes tiene una interfaz automática de salida con el Sistema de Contabilidad, ya que genera las pólizas contables de los movimientos procesales de los clientes. **[3]**

Las diferentes actividades que realiza un Sistema de Información se pueden observar en el diseño conceptual ilustrado en la en la figura 2.1

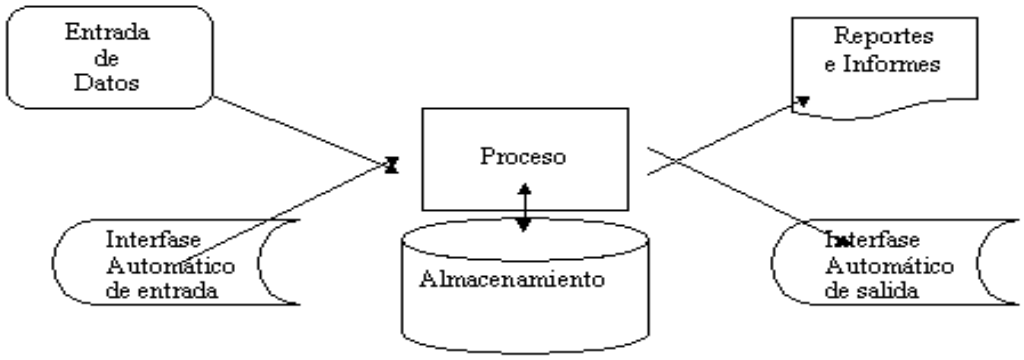


Figura 2.1 Diagrama de actividades realizadas por un Sistema de Información (Tecnomaestros, febrero 2007)

2.3.4. Tipos y Usos de los Sistemas de Información

Sistemas Transaccionales.

Los Sistemas de Información que logran la automatización de procesos operativos dentro de una organización, son llamados frecuentemente Sistemas Transaccionales, ya que su función primordial consiste en procesar transacciones tales como pagos, cobros, pólizas, entradas, salidas, etc. Sus principales características son:

- A través de éstos suelen lograrse ahorros significativos de mano de obra, debido a que automatizan tareas operativas de la organización.

- Con frecuencia son el primer tipo de Sistemas de Información que se implanta en las organizaciones. Se empieza apoyando las tareas a nivel operativo de la organización.

- Son intensivos en entrada y salida de información; sus cálculos y procesos suelen ser simples y poco sofisticados.

- Tienen la propiedad de ser recolectores de información, es decir, a través de estos sistemas se cargan las grandes bases de información para su explotación posterior.

- Son fáciles de justificar ante la dirección general, ya que sus beneficios son visibles y palpables. **[4]**

Sistemas de Apoyo de las Decisiones.

Por otra parte, los Sistemas de Información que apoyan el proceso de toma de decisiones son los Sistemas de Soporte a la Toma de Decisiones, Sistemas para la

Toma de Decisión de Grupo, Sistemas Expertos de Soporte a la Toma de Decisiones y Sistema de Información para Ejecutivos. Las principales características de estos son:

- Suelen introducirse después de haber implantado los Sistemas Transaccionales más relevantes de la empresa, ya que estos últimos constituyen su plataforma de información.

- La información que generan sirve de apoyo a los mandos intermedios y a la alta administración en el proceso de toma de decisiones.

- Suelen ser intensivos en cálculos y escasos en entradas y salidas de información. Así, por ejemplo, un modelo de planeación financiera requiere poca información de entrada, genera poca información como resultado, pero puede realizar muchos cálculos durante su proceso.

- No suelen ahorrar mano de obra. Debido a ello, la justificación económica para el desarrollo de estos sistemas es difícil, ya que no se conocen los ingresos del proyecto de inversión.

- Suelen ser Sistemas de Información interactivos y amigables, con altos estándares de diseño gráfico y visual, ya que están dirigidos al usuario final.

- Apoyan la toma de decisiones que, por su misma naturaleza son repetitivos y de decisiones no estructuradas que no suelen repetirse. Por ejemplo, un Sistema de Compra de Materiales que indique cuándo debe hacerse un pedido al proveedor o un Sistema de Simulación de Negocios que apoye la decisión de introducir un nuevo producto al mercado.

- Estos sistemas pueden ser desarrollados directamente por el usuario final sin la participación operativa de los analistas y programadores del área de informática.

- Este tipo de sistemas puede incluir la programación de la producción, compra de materiales, flujo de fondos, proyecciones financieras, modelos de simulación de negocios, modelos de inventarios, etc. **[4]**

Sistemas Estratégicos

El tercer tipo de sistema, de acuerdo con su uso u objetivos que cumplen, es el de los Sistemas Estratégicos, los cuales se desarrollan en las organizaciones con el fin de lograr ventajas competitivas, a través del uso de la tecnología de información. Sus principales características son:

- Su función primordial no es apoyar la automatización de procesos operativos ni proporcionar información para apoyar la toma de decisiones.

- Suelen desarrollarse in house, es decir, dentro de la organización, por lo tanto no pueden adaptarse fácilmente a paquetes disponibles en el mercado.

- Típicamente su forma de desarrollo es a base de incrementos y a través de su evolución dentro de la organización. Se inicia con un proceso o función en particular y a partir de ahí se van agregando nuevas funciones o procesos.

Su función es lograr ventajas que los competidores no posean, tales como ventajas en costos y servicios diferenciados con clientes y proveedores. En este contexto, los Sistema Estratégicos son creadores de barreras de entrada al negocio. Por ejemplo, el uso de cajeros automáticos en los bancos en un Sistema Estratégico, ya que brinda ventaja sobre un banco que no posee tal servicio. Si un banco nuevo decide abrir sus puertas al público, tendrá que dar este servicio para tener un nivel similar al de sus competidores.

Apoyan el proceso de innovación de productos y proceso dentro de la empresa debido a que buscan ventajas respecto a los competidores y una forma de hacerlo es innovando o creando productos y procesos. **[4]**

Los tipos y usos de los Sistemas de Información se muestran en la figura 2.2

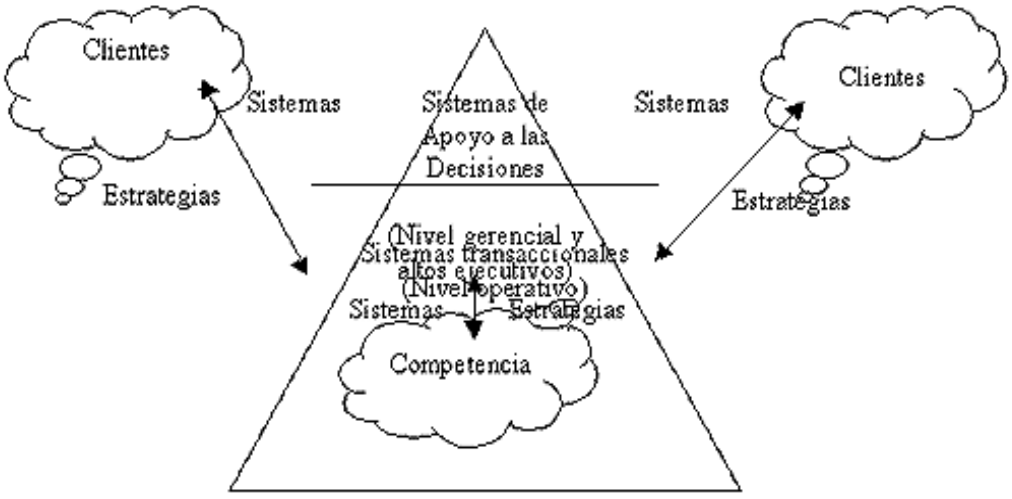


Figura 2.2 Tipos y usos de los Sistemas de Información
(Manuel Peralta, febrero 2000)

2.3.5. Software:

Es la suma total de los programas de computadora, procedimientos, reglas, la documentación asociada y los datos que pertenecen a un sistema de cómputo. Cuyo resultado es un producto diseñado para un usuario. **[7]**

2.3.6. Ingeniería de Software

Es un enfoque sistemático del desarrollo, operación, mantenimiento y retiro del software", que en palabras más llanas, se considera que la Ingeniería de Software; es la rama de la ingeniería que aplica los principios de la ciencia de la computación y las matemáticas para lograr soluciones costo-efectivas (eficaces en costo o económicas) a los problemas de desarrollo de software", es decir, "permite elaborar consistentemente productos correctos, utilizables y costo-efectivos" **[7]**

2.3.7. Objetivos de la Ingeniería de Software

En la construcción y desarrollo de proyectos se aplican métodos y técnicas para resolver los problemas; la informática aporta herramientas y procedimientos sobre los que se apoya la Ingeniería de Software.

- Poseer la capacidad para procesar transacciones con rapidez y eficiencia.

- Mejorar la calidad de los productos de software.
- Llevar a cabo el seguimiento de los costos de mano de obra, bienes y gastos generales.

- Aumentar la productividad y trabajo de los ingenieros del software.

- Facilitar el control del proceso de desarrollo de software.

- Ampliar la comunicación y facilitar la integración de funciones individuales.

- Suministrar a los desarrolladores las bases para construir software de alta calidad en una forma eficiente.

- Definir una disciplina que garantice la producción y el mantenimiento de los productos software desarrollados en el plazo fijado y dentro del costo estimado.

2.3.8. Objetivos de la Ingeniería de Software en los Proyectos de Sistemas

Para que los objetivos se cumplan las empresas emprenden proyectos por las siguientes razones:

- **Capacidad**: Las actividades de la organización están influenciadas por la capacidad de ésta para procesar transacciones con rapidez y eficiencia. Los sistemas de información mejoran esta capacidad en tres formas:

  - Aumentan la velocidad de procesamiento: Los sistemas basados en computadoras pueden ser de ayuda para eliminar la necesidad de cálculos tediosos y comparaciones repetitivas. Un sistema automatizado puede ser de gran utilidad si lo que se necesita es un procesamiento acelerado.

- Aumento en el volumen: La incapacidad para mantener el ritmo de procesamiento no significa el abandono de los procedimientos existentes. Quizá éstos resulten inadecuados para satisfacer las demandas actuales. En estas situaciones el analista de sistemas considera el impacto que tiene la introducción de procesamiento computarizado, si el sistema existente es manual. Es poco probable que únicamente el aumento de la velocidad sea la respuesta. El tiempo de procesamiento por transacción aumenta si se considera la cantidad de actividades comerciales de la empresa junto con su patrón de crecimiento.

- Recuperación más rápida de la información: Las organizaciones almacenan grandes cantidades de datos, por eso, debe tenerse en cuenta donde almacenarlos y como recuperarlos cuando se los necesita. Cuando un sistema se desarrolla en forma apropiada, se puede recuperar en forma rápida la información.

➢ **Costo**: Esta capacidad es mejora de la siguiente forma:

- Vigilancia de los costos: Para determinar si la compañía evoluciona en la forma esperada, de acuerdo con lo presupuestado, se debe llevar a cabo el seguimiento de los costos de mano de obra, bienes y gastos generales. La creciente competitividad del mercado crea la necesidad de mejores métodos para seguir los costos y relacionarlos con la productividad individual y organizacional.

- Reducción de costos: Los diseños de sistemas ayudan a disminuir los costos, ya que toman ventaja de las capacidades de cálculo automático y de recuperación de datos que están incluidos en procedimientos de programas en computadora. Muchas tareas

son realizadas por programas de cómputo, lo cual deja un número muy reducido de éstas para su ejecución manual, disminuyendo al personal.

- **Control**: esta capacidad posee las siguientes funciones para mejorar el sistema:

  - Mayor seguridad de información: Algunas veces el hecho de que los datos puedan ser guardados en una forma adecuada para su lectura por medio de una máquina, es una seguridad difícil de alcanzar en un medio ambiente donde no existen computadoras. Para aumentar la seguridad, generalmente se desarrollan sistemas de información automatizados. El acceso a la información puede estar controlado por un complejo sistema de contraseñas, limitado a ciertas áreas o personal, si está bien protegido, es difícil de acceder.

  - Menor margen de error (mejora de la exactitud y la consistencia): Esto se puede lograr por medio del uso de procedimientos de control por lotes, tratando de que siempre se siga el mismo procedimiento. Cada paso se lleva a cabo de la misma manera, consistencia y exactitud: por otra parte se efectúan todos los pasos para cada lote de transacciones. A diferencia del ser humano, el sistema no se distrae con llamadas telefónicas, ni olvidos e interrupciones que sufre el ser humano. Si no se omiten etapas, es probable que no se produzcan errores.

- **Comunicación**: La falta de comunicación es una fuente común de dificultades que afectan tanto a cliente como a empleados. Sin embargo, los sistemas de información bien desarrollados amplían la comunicación y facilitan la integración de funciones individuales.

  - Interconexión (aumento en la comunicación): Muchas empresas aumentan sus vías de comunicación por medio del desarrollo de redes para este

fin, dichas vías abarcan todo el país y les permiten acelerar el flujo de información dentro de sus oficinas y otras instalaciones que no se encuentran en la misma localidad. Una de las características más importantes de los sistemas de información para oficinas es la transmisión electrónica de información, como por ejemplo, los mensajes y los documentos.

- Integración de áreas en las empresas: Con frecuencia las actividades de las empresas abarcan varias áreas de la organización, la información que surge en un área se necesita en otra área, por ejemplo. Los sistemas de información ayudan a comunicar los detalles del diseño a los diferentes grupos, mantienen las especificaciones esenciales en un sitio de fácil acceso y calculan factores tales como el estrés y el nivel de costos a partir de detalles proporcionados por otros grupos.

2.3.9. Proceso de Ingeniería de Software

El proceso de Ingeniería del Software son etapas en las que las necesidades del usuario son traducidas en requerimientos del software, estos requerimientos transformados en diseño y el diseño implementado en código, el código es probado, documentado y certificado para uso operativo. Concretamente define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo. A este proceso también se le llama el ciclo de vida del software que comprende cuatro grandes fases:

- Concepción*:* Define el alcance del proyecto y desarrolla un caso de negocio.

- Elaboración*:* Define un plan del proyecto, especifica las características y fundamenta la arquitectura.

- Construcción: Crea el producto.

- Transición: Transfiere el producto a los usuarios.

2.3.10. Proceso de Desarrollo de Software:

Conjunto total de actividades necesarias para transformar los requisitos de un cliente en un conjunto consistente de artefactos que representan un producto software y en punto posterior en el tiempo para transformar el cambio en dichos requisitos en nuevas versiones del producto. **[8]**

2.3.11. Lenguaje Unificado de Modelado (UML):

Lenguaje estándar para el modelado de software lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software. Lenguaje usado por el Proceso Unificado. Lenguaje que permite a los desarrolladores visualizar el producto de su trabajo (Artefactos) en esquemas o diagramas estandarizados. **[8]**

2.3.12. Proceso Unificado:

Proceso de desarrollo de software basado en el Lenguaje Unificado de Modelado y que es iterativo, centrado en la arquitectura y dirigido por los casos de uso y los riesgos. Proceso que se organiza en cuatro fases: inicio, elaboración, construcción y transición, y que se estructura en torno a cinco flujos de trabajo fundamentales: recopilación de requisitos, análisis, diseño, implementación y pruebas. Proceso que se describe en términos de un modelo de negocio, el cual esta a

su vez estructurado en función de tres bloques de construcción primordiales trabajadores, actividades y artefactos.

La estructuración del proceso unificado es una contraposición de las etapas del proceso contra las fases del mismo lo cual determina el tiempo de vida de un software la cuales puede apreciar en la figura 2.3 **[8]**

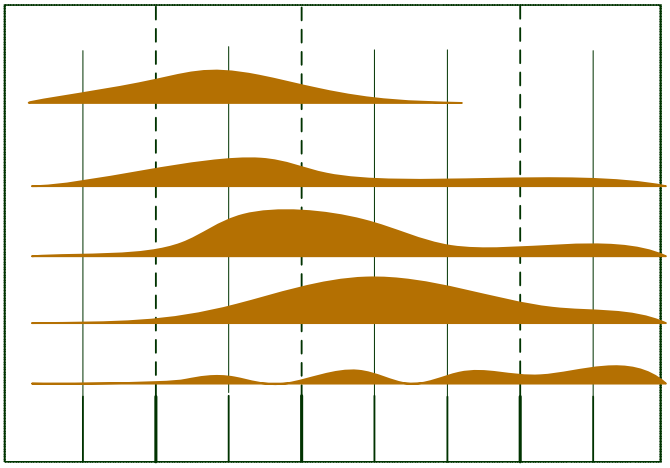


Figura 2.3. Tiempo de vida de un software
(Rumbaugh, J., 2000)

Cada ciclo produce una nueva versión del sistema, y cada versión es un producto preparado para su entrega; consta de un cuerpo de código fuente incluido en componentes que puede compilarse y ejecutarse, además de manuales y otros productos asociados. Sin embargo, el producto terminado no sólo debe ajustarse a las necesidades de los usuarios, sino también a las de todos los interesados, es decir, toda la gente que trabajará con el producto. El producto terminado incluye los requisitos, casos de uso, especificaciones no funcionales y casos de prueba. Incluye el modelo de la arquitectura y el modelo visual (artefactos modelados con el UML).

Flujo de trabajos Fundamentales

Inicio

E

Requisitos

Análisis

Diseño

Implementación

La Figura 2.3 muestra en la columna izquierda los flujos de trabajo (requisitos, análisis, diseño, implementación y prueba). Las curvas son una aproximación de hasta donde se llevan a cabo los flujos de trabajo en cada fase. Una iteración típica pasa por los cinco flujos de trabajo.

Durante la fase de inicio, se desarrolla una descripción del producto final a partir de una buena idea y se presenta el análisis de negocio para el producto. Esencialmente esta fase responde a las siguientes preguntas:

- ¿Cuáles son las principales funciones del sistema para sus usuarios más importantes?
- ¿Cómo podría ser la arquitectura del sistema?
- ¿Cuál es el plan de proyecto y cuánto costará desarrollar el producto?

La respuesta a la primera pregunta se encuentra en un modelo de casos de uso simplificado que contenga los casos de uso más críticos. Una vez obtenidos, la arquitectura es provisional, y consiste típicamente en un simple esbozo que muestra los subsistemas más importantes. En esta fase, se identifican y priorizan los riesgos más importantes, se planifica en detalle la fase de elaboración, y se estima el proyecto de manera aproximada.

Durante la fase de elaboración, se especifican en detalle la mayoría de los casos de uso del producto y se diseña la arquitectura del sistema. La relación entre la arquitectura del sistema y el propio sistema es primordial. Por tanto la arquitectura se expresa en forma de vistas de todos los modelos del sistema, los cuales juntos representan al sistema entero. Esto implica que hay vistas arquitectónicas del modelo

de casos de uso, del modelo de análisis, del modelo de diseño, del modelo de implementación y modelo de despliegue. La vista del modelo de implementación incluye componentes para probar que la arquitectura es ejecutable. Durante esta fase del desarrollo, se realizan los casos de uso más críticos que se identificaron en la fase de inicio. El resultado de esta base es la línea base de la arquitectura.

Durante la fase de construcción, se crea el producto, la línea base de la arquitectura crece hasta convertirse en el sistema completo. La descripción evoluciona hasta convertirse en un producto preparado para ser entregado a la comunidad de usuarios. El grueso de los recursos requeridos se emplea durante esta fase del desarrollo. Sin embargo la arquitectura del sistema es estable, aunque se pueden describir formas mejores de estructurar el sistema. Al final de esta fase, el producto contiene todos los casos de uso que la dirección y el cliente han acordado para el desarrollo de esta versión. Sin embargo, puede que no este completamente libre de defectos. Muchos de estos defectos se descubrirán y solucionarán durante la fase de transición.

Durante la fase de transición, se cubre el período durante el cual el producto se convierte en versión beta. En la versión beta un número reducido de usuarios con experiencia prueba el producto e informa de defectos y deficiencias. Esta fase conlleva actividades como la fabricación, formación del cliente, el proporcionar una línea de ayuda y asistencia, y la corrección de los defectos que se encuentren tras la entrega.

2.3.13. Iteración Genérica del Proceso Unificado de Desarrollo de Software

Los flujos fundamentales: Requisitos, Análisis, Diseño, Implementación y Pruebas, se distinguen entre los flujos de trabajos fundamentales e iterativos. Dichos flujos no ocurren una sola vez, en el Proceso Unificado de Modelado, como sucede

teóricamente en el modelo en cascada; sino que se repiten más bien en cada iteración, una y otra vez, como flujos de trabajos iterativos. Cada repetición, sin embargo, se diferencia en los detalles que se enfrentan o asuntos centrales de cada iteración.**[16]**

2.3.14. Flujos de Trabajos para una Iteración

Cada iteración está formada por cinco grupos de flujos de trabajos fundamentales, de entre los cuales estos flujos se adaptan según sea la fase para la cual se está desarrollando. En la siguiente figura, se observa como, para una iteración de cualquier fase, coexisten los cinco flujos de trabajos.

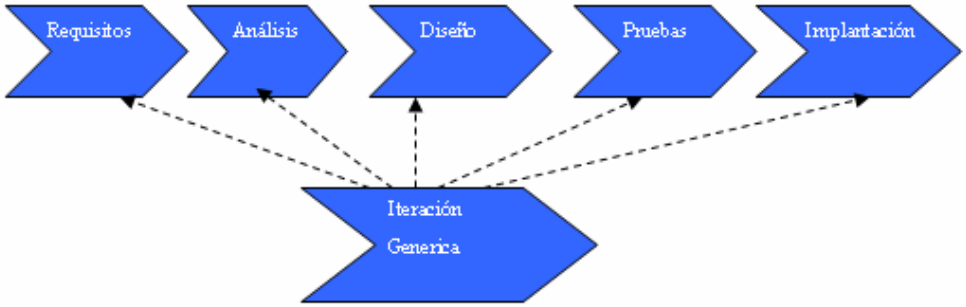


Figura 2.4. Los cinco flujos de trabajo Requisitos, Análisis, Diseño, Implementación y Prueba de una iteración genérica.

2.3.15. Ventajas del Proceso Unificado de Desarrollo de Software

- La Iteración controlada acelera el ritmo de esfuerzo de desarrollo en su totalidad debido a que los desarrolladores trabajan de manera más eficiente para obtener resultados claros a corto plazo.

- La iteración controlada reconoce una realidad que a menudo ignora que las necesidades del usuario y sus correspondientes requisitos no pueden definirse

completamente al principio. Mediante proceso unificado racional los requisitos se adquieren y refinan en sucesivas iteraciones.

- En cada fase del proceso unificado se genera un producto; así mismo el desarrollo de cada fase puede contener algunos de los siguientes modelos: requisitos, análisis, diseño, implementación, despliegue y pruebas.

- Los modelos son ideales porque mediante éstos se representa el trabajo evolutivo que desarrollan los ingenieros de software; y no incluye un producto final, como lo es la versión Beta del sistema; es decir no se requiere esperar la culminación del trabajo para observar un producto.

- Los modelos juntos representan al sistema como un todo.

- El proceso unificado está basado en componentes; y utiliza el estándar del lenguaje unificado de modelado para llevar a cabo sus modelos.

2.3.16. Desventaja del Proceso Unificado de Desarrollo de Software

A pesar de sus características de adaptabilidad y flexibilidad, el Proceso Unificado no permite representar en su totalidad los elementos representativos de los agentes, los cuales son: habilidad social, autonomía, reactividad, pro-actividad, nociones mentales, adaptabilidad o aprendizaje, veracidad, racionalidad**.**

2.3.17 Lenguaje unificado de modelado

Es un lenguaje de modelado para la especificación, visualización, construcción y documentación de los artefactos de un proceso de sistema intensivo.

1 Dentro de un *proceso de sistema intensivo*, un método es aplicado para llegar o evolucionar un sistema.

2 Como un **l*enguaje***, es usado para la comunicación. Es decir, un medio para capturar el conocimiento (semánticas) respecto a un tema y expresar el conocimiento (sintaxis) resguardando el tema propósito de la comunicación. El tema es el sistema en estudio.

3 Como un lenguaje para ***modelamiento***, se enfoca en la comprensión de un tema a través de la formulación de un modelo del tema (y su contexto respectivo). El modelo abarca el conocimiento cuidando del tema, y la apropiada aplicación de este conocimiento constituye inteligencia.

4 Cuidando la ***unificación***, integra las mejores prácticas de la ingeniería de la industria tecnológica y sistemas de información pasando por todos los tipos de sistemas (software y no - software), dominios (negocios versus software) y los procesos de ciclo de vida.

5 En *cuanto* a cómo se aplica para *especificar* sistemas, puede ser usado para comunicar "qué" se requiere de un sistema y "cómo" un sistema puede ser realizado.
6 En cuanto a cómo se aplica para *visualizar* sistemas, puede ser usado para describir visualmente un sistema antes de ser realizado.

7 En cuanto a cómo se aplica para *construir* sistemas, puede ser usado para guiar la realización de un sistema similar a los "planos".

8 En cuanto a cómo se aplica para *documentar* sistemas, puede ser usado para capturar conocimiento respecto a un sistema a lo largo de todo el proceso de su ciclo de vida.

UML no es:

9 Un lenguaje de programación visual, sino un lenguaje de modelamiento visual.

10 Una herramienta o depósito de especificación, sino un lenguaje para modelamiento de especificación.

11 Un proceso, sino que habilita procesos.

Fundamentalmente, UML está relacionado con la captura, comunicación y nivelación (disgregación en niveles) de conocimientos.

2.3.18. Vista General de UML

La explicación se basará en los diagramas, en lugar de vistas o notaciones, ya que son éstos la esencia de UML. Cada diagrama usa la notación pertinente y la suma de estos diagramas crean las diferentes vistas. Las vistas existentes en UML son:

1 Vista de casos de uso: Se forma con los diagramas de casos de uso, colaboración, estados y actividades.

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

3 Vista de procesos: Se forma con los diagramas de la vista de diseño. Recalcando las clases y objetos referentes a procesos.

4 Vista de implementación: Se forma con los diagramas de componentes, colaboración, estados y actividades.

5 Vista de despliegue: Se forma con los diagramas de despliegue, interacción, estados y actividades.

2.3.19. Utilidad del uso de UML

UML es un lenguaje para modelamiento de propósito general evolutivo, ampliamente aplicable, debe ser soportado por herramientas e industrialmente estandarizado. Se aplica a una multitud de diferentes tipos de sistemas, dominios, y métodos o procesos.

1 Como lenguaje de *propósito general*, se enfoca en el corazón de un conjunto de conceptos para la adquisición, compartición y utilización de conocimientos emparejados con mecanismos de extensión.

2 Como un lenguaje para modelamiento *ampliamente aplicable*, puede ser aplicado a diferentes tipos de sistemas (software y no - software), dominios (negocios versus software) y métodos o procesos.

3 Como un lenguaje para modelamiento *soportable por herramientas*, las herramientas ya están disponibles para soportar la aplicación del lenguaje para especificar, visualizar, construir y documentar sistemas.

4 Como un lenguaje para modelamiento *industrialmente estandarizado*, no es un lenguaje cerrado, propiedad de alguien, sino más bien, un lenguaje abierto y totalmente extensible reconocido por la industria.

UML posibilita la captura, comunicación y nivelación de conocimiento estratégico, táctico y operacional para facilitar el incremento de valor, aumentando la calidad, reduciendo costos y reduciendo el tiempo de presentación al mercado; manejando riesgos y siendo proactivo para el posible aumento de complejidad o cambio.

2.3.20. Modelo Conceptual de UML

Los tres elementos que forman el modelo conceptual de UML son: los bloques básicos de construcción del lenguaje, las reglas que se aplican sobre esos bloques y los mecanismos comunes de UML.

Existen tres tipos de bloques de construcción:

1 Elementos: Son los modelos UML (clases, casos de uso, estados, anotaciones...)

2 Relaciones: Ligan elementos entre sí, establecen la forma en que interactúan.

3 Diagramas: Representación gráfica de un grupo de elementos y sus relaciones.

2.3.21. Elementos de UML

- **Casos de Uso:** Un caso de uso es una descripción de un conjunto de acciones ejecutadas por el sistema tras la orden de un agente (llamado actor) que puede ser el usuario de la aplicación, la propia aplicación, otro caso de uso o un elemento externo (hardware). Los casos de uso suelen representar funcionalidades del sistema; se representan como una elipse en cuyo interior figura el nombre (lo más descriptivo posible) del caso de uso.

- **Clases**: En una clase se agrupan todos los objetos que comparten los mismos atributos, métodos y relaciones. Los atributos son características y propiedades comunes en todos los objetos de la clase. Los métodos son operaciones que deben cumplir las instancias de la clase. Las clases se representan como un rectángulo donde figuran el nombre de la clase, sus atributos y sus métodos.

2.3.22. Relaciones de UML

- **Dependencia**: Una dependencia es una relación de uso entre dos elementos (un elemento utiliza a otro). Una relación de dependencia entre dos elementos implica que los cambios que se produzcan en un elemento pueden afectar al otro pero no necesariamente a la inversa. Las dependencias se representan con una línea dirigida discontinua.

- **Asociación**: Una asociación es una relación estructural entre varios elementos. Una relación de asociación implica que los objetos de los distintos elementos de la relación están conectados entre sí y se pueden comunicar. Una relación de asociación se representa gráficamente con una línea continua entre los elementos relacionados.

- **Generalización**: Una generalización es una relación de especialización. Los elementos especializados (hijos) son elementos que derivan de un elemento general (padre). Los elementos hijos mantienen la estructura y el funcionamiento del elemento padre pero de una forma más especializada. Su representación gráfica es la de una línea dirigida con punta triangular.**[14]**

2.4 Diagramas de UML

2.4.1. Diagrama de Casos de Uso

Un Diagrama de Casos de Uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa. En el diagrama de casos de uso se representa también el sistema como una caja rectangular con el nombre en su interior. Los casos de uso están en el interior de la caja del sistema, y los actores fuera, y cada actor está unido a los casos de uso en los que participa mediante una línea. En la Figura 2.5 se muestra un ejemplo de Diagrama de Casos de Uso para un cajero automático. **[9]**

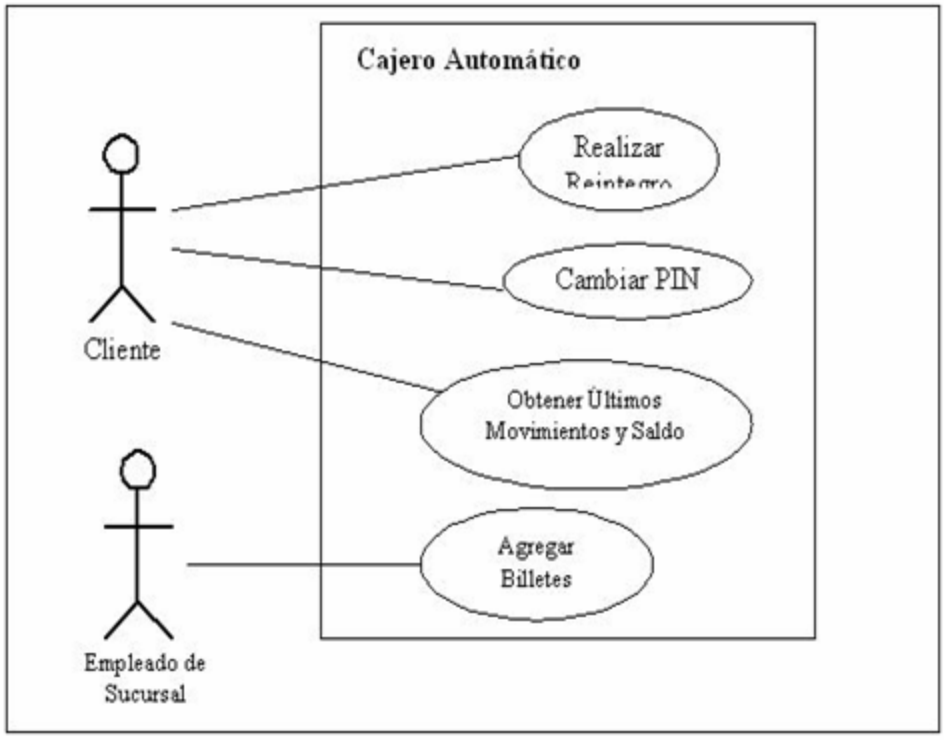


Figura 2.5: Diagrama de Casos de Uso (Ferré Grau 2004)

2.4.2 Elementos de lo Diagramas de Caso de Uso

Los elementos que pueden aparecer en un Diagrama de Casos de Uso son: Actores, Casos de Uso y Relaciones entre Casos de Uso. **[9]**

2.4.3 Actores

Un actor es algo con comportamiento, como una persona (identificada por un rol), un sistema informatizado u organización, y que realiza algún tipo de interacción con el sistema. Se representa mediante una figura humana dibujada con palotes. Esta representación sirve tanto para actores que son personas como para otro tipo de actores. **[9]**

2.4.4 Casos de Uso

Un caso de uso es una descripción de la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para llevar a cabo una tarea específica. Expresa una unidad coherente de funcionalidad, y se representa en el Diagrama de Casos de Uso mediante una elipse con el nombre del caso de uso en su interior. El nombre del caso de uso debe reflejar la tarea específica que el actor desea llevar a cabo usando el sistema. **[9]**

2.4.5 Relaciones entre Casos de Uso

Un caso de uso, en principio, debería describir una tarea que tiene un sentido completo para el usuario. Sin embargo, hay ocasiones en las que es útil describir una interacción con un alcance menor como caso de uso. La razón para utilizar estos casos de uso no completos en algunos casos, es mejorar la comunicación en el equipo de desarrollo, el manejo de la documentación de casos de uso. Para el caso de que queramos utilizar estos casos de uso más pequeños, las relaciones entre estos y los casos de uso ordinarios pueden ser de los siguientes tres tipos: • Incluye (<>): Un

caso de uso base incorpora explícitamente a otro caso de uso en un lugar especificado en dicho caso base. Se suele utilizar para encapsular un comportamiento parcial común a varios casos de uso. En la Figura 2.6 se muestra cómo el caso de uso Realizar Reintegro puede incluir el comportamiento del caso de uso Autorización. **[9]**

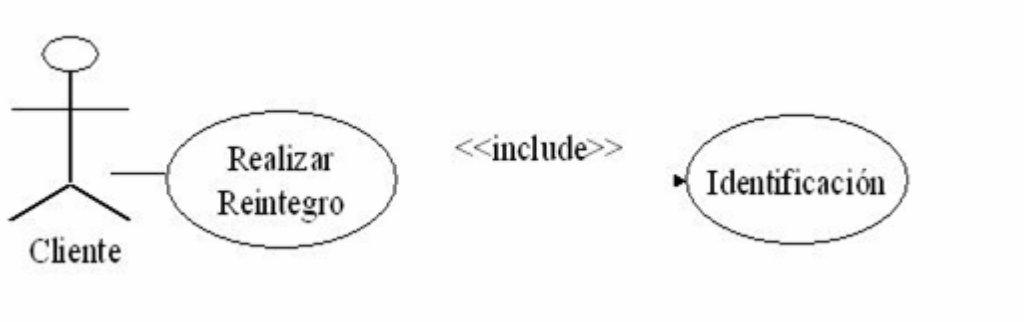


Figura 2.6: Ejemplo de caso de uso con incluye
(Ferré Grau 2004)

En la figura 2.7, se muestra un ejemplo de Relación <> • Extiende (<>): Cuando un caso de uso base tiene ciertos puntos (puntos de extensión) en los cuales, dependiendo de ciertos criterios, se va a realizar una interacción adicional. El caso de uso que extiende describe un comportamiento opcional del sistema (a diferencia de la relación incluye que se da siempre que se realiza la interacción descrita) En la Figura 2.7 se muestra como el caso de uso Comprar Producto permite explícitamente extensiones en el siguiente punto de extensión: info regalo. La interacción correspondiente a establecer los detalles sobre un producto que se envía como regalo están descritos en el caso de uso Detalles Regalo.

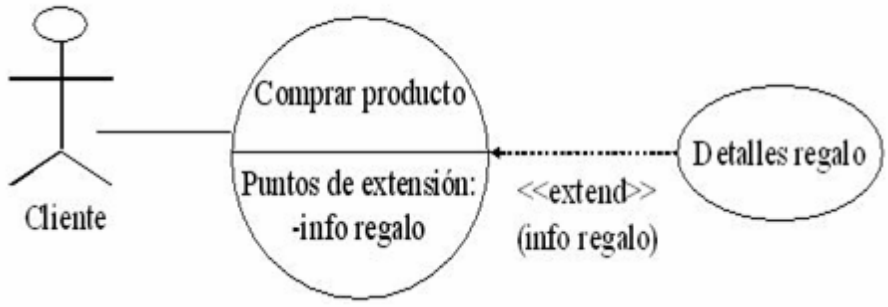


Figura 2.7: Ejemplo de caso de uso con extend

(Ferré Grau 2004)

2.4.6 Diagramas de Secuencia

Un diagrama de secuencia es un diagrama de interacción UML. Estos diagramas muestran la secuencia de mensajes que se van lanzando los objetos implicados en una determinada operación del programa. Dentro del diagrama los objetos se alinean en el eje X respetando su orden de aparición. En el eje Y se van mostrando los mensajes que se envían, también respetando su orden temporal. Cada objeto tiene una línea de vida donde se sitúa su foco de control. El foco de control es un rectángulo que representa el tiempo durante el que un objeto está activo ejecutando una acción. Con este sencillo esquema podemos visualizar la comunicación y sincronización bajo un estricto orden temporal de los objetos implicados en las distintas funcionalidades de un sistema.

2.4.7 Diagrama de Clases de Análisis

Es utilizado por los desarrolladores de software para determinar los requerimientos funcionales, considerando una o varias clases, o sub-sistemas del sistema a desarrollar. Los casos de uso se describen mediante clases de análisis y sus objetos. El diagrama de clases de análisis se construye examinando los casos de usuarios, cerrando sus reacciones e identificando los roles de los clasificadores.

2.4.8 Diagrama de Colaboración

Este diagrama es utilizado para modelar interacciones entre objetos. En el análisis de este diagrama es más importante el paso de información de un objeto a otro, constituido por los mensajes, que en su diseño se detallan. Cada caso de uso se desarrolla como una realización de casos de uso, donde cada uno tiene un conjunto de clasificadores participantes que desempeñan diferentes roles. Cada clase de análisis representa un objeto o instancia de una clase en el diagrama de colaboración donde se establece la cooperación existente entre ellas. La secuencia en que estos objetos aparecen en un caso de uso se muestra usando números y comienza con un evento que llega a una interfaz, se sigue con un análisis de los eventos siguientes y la posible respuesta del sistema.

2.4.9 Diagrama de Clases de Diseño

Se emplean para modelar la estructura estática de las clases en el sistema, sus tipos, sus contenidos y las relaciones que se establecen entre ellos. A través de este diagrama se definen las características de cada una de las clases, interfaces, colaboraciones y relaciones de dependencia y generalización**.**

2.5 Bases de datos

2.5.1 Dato:

Conjunto de caracteres con algún significado, pueden ser numéricos, alfabéticos, o alfanuméricos. **[10]**

2.5.2 Información:

Es un conjunto ordenado de datos los cuales son manejados según la necesidad del usuario, para que un conjunto de datos pueda ser procesado eficientemente y pueda dar lugar a información, primero se debe guardar lógicamente en archivos. **[10]**

2.6 Conceptos básicos de Bases de Datos.

Campo:

Es la unidad más pequeña a la cual uno puede referirse en un programa. Desde el punto de vista del programador representa una característica de un individuo u objeto. **[10]**

Registro:

Colección de campos de iguales o de diferentes tipos. **[10]**

Archivo:

Colección de registros almacenados siguiendo una estructura homogénea. **[10]**

Base de Datos:

Es una colección de archivos interrelacionados, son creados con un DBMS. El contenido de una base de datos engloba a la información concerniente (almacenadas en archivos) de una organización, de tal manera que los datos estén disponibles para los usuarios, una finalidad de la base de datos es almacenar y eliminar, los datos. **[10]**

Manejador de Bases de Datos

El sistema manejador de bases de datos es la porción más importante del software de un sistema de base de datos. Un DBMS es una colección de numerosas rutinas de software interrelacionadas, cada una de las cuales es responsable de alguna tarea específica. **[10]**

Las funciones principales de un DBMS son:

- Crear y organizar la Base de datos.
- Establecer y mantener las trayectorias de acceso a la base de datos de tal forma que los datos puedan ser accesados rápidamente.
- Manejar los datos de acuerdo a las peticiones de los usuarios.
- Registrar el uso de las bases de datos.
- Interacción con el manejador de archivos

Esto a través de las sentencias en DML al comando del sistema de archivos. Así el Manejador de base de datos es el responsable del verdadero almacenamiento de los datos. **[10]**

Respaldo y Recuperación

Consiste en contar con mecanismos implantados que permitan la recuperación fácilmente de los datos en caso de ocurrir fallas en el sistema de base de datos. **[10]**

Control de concurrencia.

Consiste en controlar la interacción entre los usuarios concurrentes para no afectar la inconsistencia de los datos. **[10]**

Seguridad e integridad.

Consiste en contar con mecanismos que permitan el control de la consistencia de los datos evitando que estos se vean perjudicados por cambios no autorizados o previstos. **[10]**

Esquema de Base de Datos:

Es la estructura por la que esta formada la base de datos, se especifica por medio de un conjunto de definiciones que se expresa mediante un lenguaje especial llamado lenguaje de definición de datos. (DDL) **[10]**

Administrador de base de datos (DBA):

Es la persona o equipo de personas profesionales responsables del control y manejo del sistema de base de datos, generalmente tiene(n) experiencia en DBMS, diseño de bases de datos, Sistemas operativos, comunicación de datos, hardware y programación. **[10]**

2.7 Tipos de relaciones

Relación uno a uno.

Se presenta cuando existe una relación como su nombre lo indica uno a uno, denominado también relación de matrimonio. Una entidad del tipo A solo se puede relacionar con una entidad del tipo B, y viceversa. **[10]**

Relación uno a muchos.

Significa que una entidad del tipo A puede relacionarse con cualquier cantidad de entidades del tipo B, y una entidad del tipo B solo puede estar relacionada con una entidad del tipo A. **[10]**

Muchos a uno

Indica que una entidad del tipo B puede relacionarse con cualquier cantidad de entidades del tipo A, mientras que cada entidad del tipo A solo puede relacionarse con solo una entidad del tipo B. **[10]**

Muchos a muchos

Establece que cualquier cantidad de entidades del tipo A pueden estar relacionadas con cualquier cantidad de entidades del tipo B. **[10]**

CAPITULO III
FASE DE INICIO

3.1 INTRODUCCION

En este capitulo le desarrollara la primera fase del proceso unificado de desarrollo de software, conocida como la fase de inicio, en donde se define el sistema y se realiza un diseño general, y de esta forma comprender el alcance, a través de sus funciones, requisitos y actores que intervienen en el proceso, todo esto con la finalidad para desarrollar modelos preliminares que representen el comportamiento del sistema. En esta fase se definen los flujos de trabajo requisitos y funciones del sistema.

Esta fase se elabora con la finalidad de obtener mediante de la recolección de información; los pasos fundamentales con los cuales se desarrolla el proceso, así como los requerimientos de los usuarios para el sistema, y de esta forma poder modelar y estructurar un software, efectivo, eficiente, flexible, amigable, y dinámico al usuario final.

La elaboración del proceso que cumplirá el sistema, se logra después de analizar la información recopilada, y de esta forma general un bosquejo del sistema muy general, ya que la función de la fase de inicio no es modelar el sistema en si, es bosquejar o tener una opinión racional del potencial sistema decidir si merece la pena invertir en un estudio más profundo. Por lo tanto, la fase de inicio deberá ser relativamente corta en la mayoría de los proyectos, una duración de unas pocas semanas

3.2 Iteraciones Previas en el Modelo en Espiral

A traves de la metodologia de Proceso Unificado de Desarrollo de Software a través del Modelo en Espiral, se desarrollaron tres iteraciones, de donde las dos primeras se obtuvieron los siguientes resultados:

Primera Iteracion de Desarrollo.

En esta seccion del desarrollo del software, posteriromente de haber realizdo una recopilacion, analisis de informacion, y de los requisitos funcionales del cliente, se obtuvo una version prototipo; el cual abarco las funciones del departamaneto de atencion al cliente,y fue presentada al cliente y los usuarios finales.

Segunda Iteracion de Desarrollo

En esta seccion del desarrollo del software, luego de realizar la fase de analisis, de los requisitos funcionales, obtenidos en la iteracion anterior, se logro codificar una version, mas completa que abarca los cuatro departamentos involucrados, en el desarrollo del proceso de aprobación y despacho de una orden, obteniendose una versión del software con caracteristicas multiusuario, y con capacidad de actualizacion y modificacion del inventerio existente, y fue presentada al cliente y los usuarios finales.

De estas dos secciones del desarrollo se obtuvieron los requisitos, y limitaciones funcionales del sotfware final; esta es la iteracion final y mas completa del desarrollo del software, y esta es la que sera explicada al detalle en los capitulos subsiguientes.

En la figura 3.1 se muestra el desarrollo iterativo de la aplicación S.D.T, asi como el número de iteraciones y diferentes fases desarrolladas en cada una de las iteraciones.

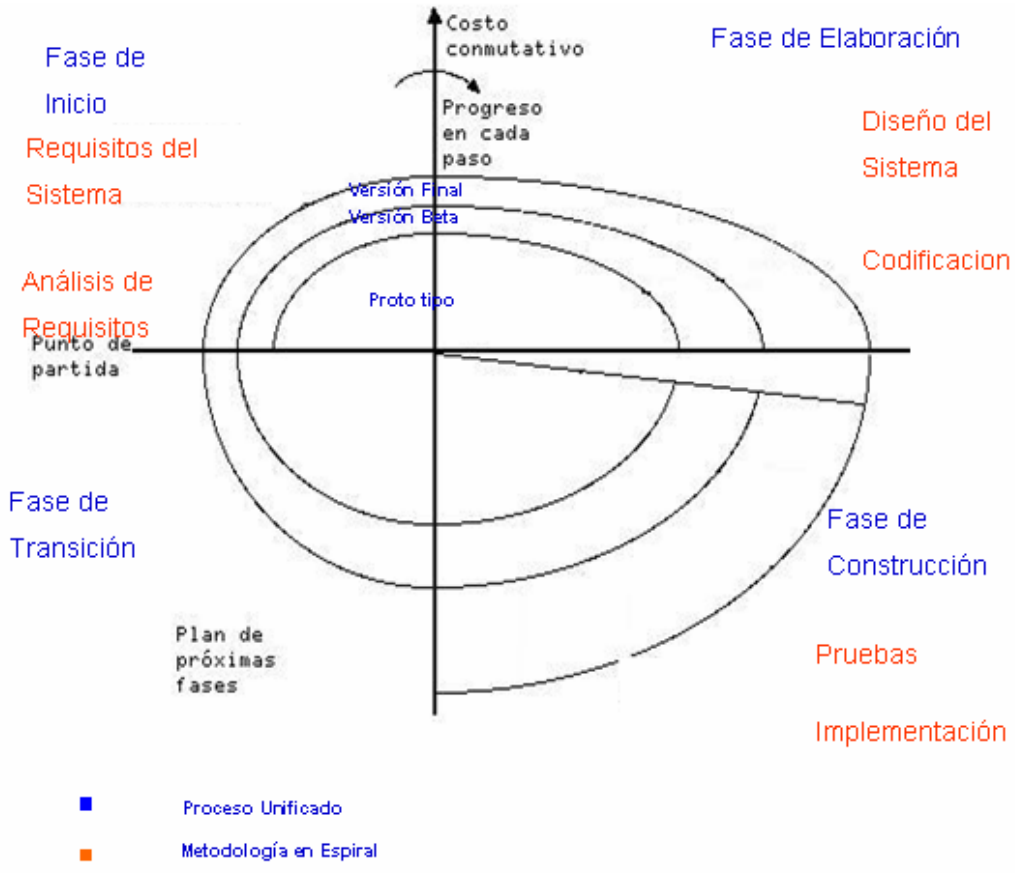


Figura 3.1: Diagrama del Proceso Unificado a traves de la Metodología en Espiral Utlizada en el Desarrollo de la aplicacion .Fuente Propia

3.2 Evaluación de la fase de inicio

Para llevar a cabo esta fase se realizará un análisis del sistema tomando en cuenta varios aspectos esenciales; en primer lugar, se estudiará el contexto en el cual se desarrollan todas las actividades humanas relacionadas con el tema de interés,

analizando todos los procesos y procedimientos involucrados; que serán representados a través del modelo de dominio. En segundo lugar, se identificarán los riesgos que serán mitigados con la realización de las fases en el desarrollo del proyecto. En tercer lugar, se capturarán los requisitos y se describirá la interacción de los actores implicados en el sistema a través de los diagramas de casos de uso. Una vez elaborados estos diagramas, se procederá a modelarlos a través de los diagramas de clases de análisis, para comprender cómo se deberán realizar dichos casos de uso. Posteriormente, se realizarán los respectivos diagramas de colaboración, para representar las interacciones entre los objetos. Y por último, se construirá el diagrama de paquetes de análisis para encapsular los casos de uso que fueron definidos al realizar el análisis del sistema.

El propósito del desarrollo de esta fase es establecer una visión común inicial de los objetivos del proyecto, determinar y decidir cuales elementos e investigaciones valen la pena llevar a cabo en las próximas etapas.

3.3 Estudio del contexto del sistema

En todo proyecto de desarrollo de software se debe tener una perspectiva del ambiente donde se piensa implementar el sistema. El presente proyecto se realizo un estudio de las actividades de despacho de productos terminados de la planta de alimentos súper S en La cuidad de Barcelona, con el propósito de comprender las actividades que se que se llevan a cabo para cumplir dicho proceso, y poder esquematizar y diseñar el sistema. Para logra esto fue necesario lo siguiente:

- Realizar entrevistas y visitas a los funcionarios que participan en el procesamiento de requisiciones.

- Revisar y analizar, aplicando ingeniería inversa, los diferentes reportes emitidos por los distintos departamentos para determinar los procesos realizados por cada uno de ellos.

Todo esto con el fin de conocer el proceso que se lleva a cabo e identificar las debilidades que existen, en los diferentes departamentos que intervienen en el proceso, de manera que éstas puedan ser corregidas a futuro con la automatización del sistema de control de despacho de productos terminados. Una vez estructuradas la información recopilada se consiguió una aproximación del contexto del sistema representado mediante del modelo de dominio, describiendo los conceptos importantes del contexto como objeto del dominio y enlazando estos departamentos y actores entre sí.

3.3.1 Roles y Responsabilidades

Departamento de Atención al cliente

Encargado de elaborar las y modificar las órdenes de pedido,, tendiendo la posibilidad de consultar las mismas y el inventario, así como generar los reportes de transporte, y ordenes de despacho, este departamento es uno de los encargados de poner en funcionamiento el proceso de despacho de productos terminados

Departamento de Despacho

Encargado de dar la aprobación y la orden de carga a los transportes para luego ser pesado por el departamento de romana, también cumple las función de actualizar el inventario los productos terminados, y también tiene la posibilidad de consultar las ordenes, este es el otro departamento encargado de de poner en funcionamiento el proceso de despacho de productos terminados.

Departamento de créditos y cobranzas

Su principal función es la de aprobar la orden de pedido, para su posterior despacho.

Departamento de Romana

Encargado de dar salida a los vehículos mediante la toma del peso del mismo así como de darle entrada a las instalaciones de la planta.

El siguiente grafico, figura 3.2, explica el proceso realizado a elaborar y aprobar la orden en la planta Súper S Barcelona

Este diagrama muestra las actividades, entradas de información y salidas de los diferentes departamentos en el proceso de aprobación de una orden de Despacho así como los reportes que se generan en este proceso.

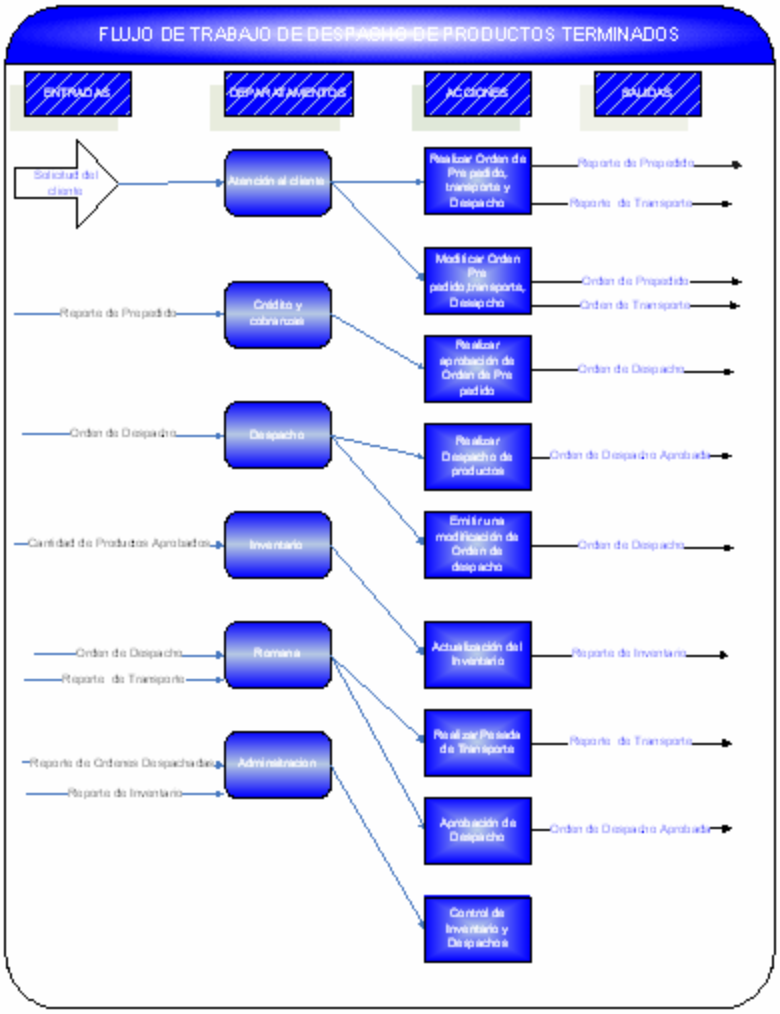


Figura 3.2. Flujo de trabajo para la aprobación de las Órdenes de Despacho. Fuente propia

ELEMENTO	DEFINICION
Solicitud del cliente	Es la petición que realiza el cliente y en donde se obtiene los datos del cliente así como también quien lo transportara, y que tipo de producto asi como la cantidad del mismo
Reporte de Pre pedido	Este es la orden que se emite con los datos que son obtenidos de la solicitud del cliente, la cual luego será enviada al departamento de créditos y cobranza para su aprobación.
Orden de Despacho	Esta orden es el reporte que se emite desde el departamento de créditos para ser enviada al departamento de despacho, y los productos puedan ser cargados en el transporte.
Reporte de Transporte	Reporte que contiene los datos de quien realizara el traslado del cargamento de productos, y este reporte permite determinar que vehiculo estará en las inmediaciones de la planta para posteriormente ser cargado con los productos.
Reporte de inventario	Reporte que contiene las cantidades de los productos, en existencia, a producir, dañados y en espera .Este reporte es realizado por el personal de inventario conjuntamente con el personal de despacho
Romana	Departamento encargado de tomar el peso del transporte luego de haber sido cargado con los productos.,así como permitirle la entrada de los vehículos a la planta
Despacho	Departamento encargado de realizar la carga del vehiculo así como realizar los reportes de inventario y la aprobación de la orden de despacho
Crédito y cobranza	Departamento encargado de realizar la aprobación de las órdenes de Pre pedido emitidas pr el departamento de atención al cliente.

Figura 3.3: Elementos del flujo de trabajo en la aprobación de la orden de despacho.

3.3.2 Modelo de Dominio

Un modelo de dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. También se les domina modelos conceptuales, modelo de objetos del dominio y modelos de objetos de análisis.

Los objetos del dominio se relacionan a través de asociaciones, agregaciones y composiciones para modelar el desenvolvimiento de las actividades; ayudando a definir "qué" deberá hacer el sistema para resolver el problema y no "cómo" lo hará.

Con el modelo del dominio se capturan los objetos más importantes en el contexto del Software STD, Los objetos del dominio representan las "cosas" que existen o los eventos que suceden en el entorno del sistema. El objetivo de este modelado es comprender el contexto y los requerimientos del sistema propuesto; en otras palabras el modelado del dominio contribuye a entender el problema, que se supone que el sistema resuelve en relación a su contexto.

Utilizando la notación UML, un modelo del dominio se representa con un conjunto de diagramas de clases en los que no se define ninguna operación. Pueden mostrar objetos del dominio o clases conceptuales, asociaciones entre las clases conceptúales y atributos de las clases conceptuales.

Las clases de dominio más relevantes en el modelo de dominio del sistema SDT, mostrado en la figura 3.4, son: Dpto. Despacho, Dpto. atención Al cliente, Dpto. Romana, Dpto. CyC, Dpto, el personal del Dpto. atención Al cliente interactúa con los otras clases del proceso realizando unas ordenes, modificandolas y elmimandolas, asi como enviando las ordenes o reportes (prepedidos) al la clase Dpto. CyC,y el envio de ordenes aprobadas al Dpto de despacho, presentado por la clase Dpto. Despacho, el personal del Dpto. Credito y cobranzas,representado por la

clase Dpto. CyC, realiza la aprobación de las ordenes, asi como la emision de nuevas ordenes, su modificacion y eliminación, de igual forma el Dpto. Despacho, realiza las funciones del Dpto de atención al cliente, y el envio de la orden despachada al departamaneto de romana presentado por la clase Dpto. Romana, dey el Dpto de romana, realiza la pesada de los transportes, y la aprobación final d ela orden y la elaboración de reportes, su modificacion y eliminación , todas estas operaciones se realizan de forma manual.

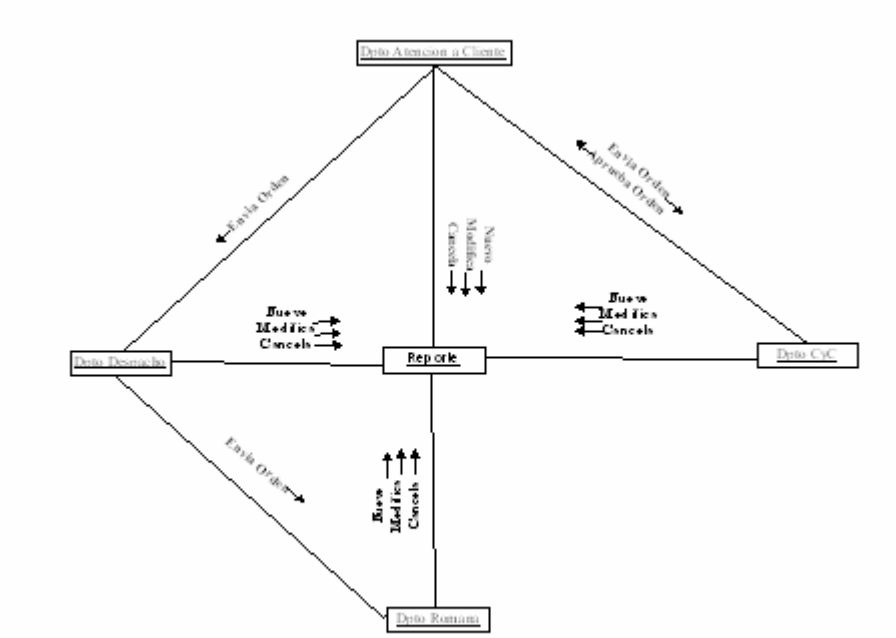


Figura 3.4 Modelo de Dominio del Sistema SDT.Fuente propia

3.3.3 Glosario de términos del modelo de dominio

Tabla 3.1 Glosario de términos del Modelo de dominio (1/2).Fuente propia

ELEMENTO	DEFINICION
Administrador:	Personal encargado de realizar la configuración del sistemas así, como funciones específicas como respaldar la base de datos y restaurarla, así como las eliminaciones, y la conexión entre los diferentes usuarios,

	este usuario, puede realizar todas las funciones de los operadores mas las propias de administración del sistemas, es el súper usuario del sistema
BD SDT	Base de datos global del sistema, mantenida y administrada por el administrador del sistema
Operaciones Sistema	Son todos aquellas clases y métodos que modifican las cantidades de productos almacenados en la base de datos, ya sea un aprobación de una orden, o una modificación, o una actualización del inventario
Reporte	Son las salidas del sistema, ya sea impreso o pro display del sistema, estos son realizados como soporte físico de las transacciones realizadas
Opciones Administrativas	Son las operaciones de configuración del sistemas como lo son , la conexión de los usuarios con la base de datos así como su desconexión, estas operaciones solo pueden ser llevadas a cabo por el administrador del sistema
Departamento de Despacho	Encargado de dar la aprobación y la orden de carga a los transportes para luego ser pesado por el departamento de romana, también cumple la función de actualizar el inventario los productos terminados, y también tiene la posibilidad de consultar las órdenes así como de realizar modificaciones a las mismas.

Tabla 3.1 Glosario de términos del Modelo de dominio (2/2). Fuente propia

ELEMENTO	DEFINICION
Departamento de créditos y cobranzas	Su principal función es la de aprobar la orden de pedido, para su posterior despacho, también opción de consultar la orden de pedido.
Departamento de Romana	Encargado de dar salida a los vehículos mediante la toma del peso del mismo así como de darle entrada a las instalaciones de la planta, también tiene la función de

	generar los reportes de transporte con los cuales se determina que vehiculo y conductor puede estar dentro las instalaciones, también tiene la, posibilidad de consultar las ordenes asignadas a los clientes

3.4 Riesgos del sistema

En todo desarrollo de sistema es importante considerar los riesgos; entiéndase por riesgos la variable del proyecto que pone en peligro o impide el éxito del mismo. Los riegos constituyen la probabilidad de que un proyecto sufra sucesos no deseables.

Es importante detectar los riesgos que podría experimentar el sistema, durante la fase de inicio, y tratar aquellos que podrían en un futuro afectar a la aplicación sobre todo en las últimas fases del proceso unificado, este es el motivo por el cual se debe llevar a cabo iteraciones que examinen los riesgos.

Una vez que se han identificado los riesgos, pueden ser tratados de diferentes maneras. Algunos pueden ser evitados llevando a cabo acciones como replanear el proyecto, pueden ser también evitados haciendo cambios en los requisitos, otros podrían ser detectados y aislados de otras partes del proyecto, a manera de afectar solo una parte.

3.4.1 Riegos Críticos del Sistema

- **Desconocimiento del ámbito del sistema**

Descripción: El no conocer el ámbito del sistema en base al cual se está trabajando, significa un riesgo crítico que debe ser mitigado en la fase de inicio. El

diseñador debe tener un conocimiento del contexto y entender sus aspectos fundamentales.

Responsable: Analistas y Diseñadores del Sistema.

- **Requisitos mal definidos**

Descripción: No disponer de unos requisitos claros y que abarquen todas las necesidades de los usuarios, pueden significar un riesgo para el sistema. Se deben recolectar todas las necesidades de los usuarios.

Responsable: Analistas y Diseñadores de Sistemas

- **Falta de robustez de la arquitectura**

Descripción: Un riesgo crítico que debe ser mitigado en la fase de inicio es el riesgo de falta de robustez de la arquitectura. El sistema debe poseer una arquitectura robusta que le permita adaptarse a los cambios y al mantenimiento de manera elegante y poco traumática.

Responsable: Analistas y Diseñadores del Sistema.

3.5 Requisitos del sistema

3.5.1 Requisitos Funcionales

1. El sistema debe contar con una base de datos unica centralizada y depurada donde esté depositada toda la información necesaria para el funcionamiento del sistema.

2 El sistema debe permitir a los usuarios la realización de consultas la base de datos centralizada.

2. El sistema debe permitir realizar reportes de las actividades realizadas dentro de un periodo de tiempo determinado.

3. El sistema debe permitir la carga de las órdenes, su consulta y su modificación desde diferentes estaciones de trabajo y departamentos.

3.5.2 Requisitos no funcionales

1. El sistema debe estar realizado con herramientas de Software Libre cumpliendo con el decreto 3.390, a manera de reducir los costos de desarrollo y su código pueda ser modificado sin ninguna restricción.

2. EL sistema debe poseer una interfaz gráfica, amigable e intuitiva que permita la fácil interacción con el usuario.

3. El sistema debe estar diseñado con una arquitectura que pueda adaptarse fácilmente a cualquier cambio y mejora estructural.

4. El sistema debe ser intuitivo, de fácil manejo y aprendizaje.

3.5.3 Requisitos de Software

1. Sistema operativo del servidor: Windows Server o xp serve pack 2.

2. Sistema operativo del cliente: Windows 9x/2000/XP.

3. Manejador de Base de Datos MySQL.4.5

3.5.4 Requisitos de la Plataforma Hardware

- Servidor:
    - 1 procesador x86 o equivalente a 1GHz o más.
    - 512 MB de memoria de acceso aleatorio (RAM).
    - Disco Duro de 40 Gb.
    - Monitor a color con una resolución máxima de 1280 x 1024.
    - Tarjeta de red Ethernet.

- Clientes:
    - Procesador x86 o equivalente a 750 MHz o más.
    - 256 MB de memoria de acceso aleatorio (RAM).
    - Monitor a color con resolución de 800x600 píxeles como mínimo.
    - Tarjeta de red Ethernet.

3.6 Modelado de casos de uso

El modelado de casos de uso es una forma complementaria de crear y documentar requisitos, permite que los desarrolladores de software y los clientes lleguen a un acuerdo sobre los mismos.

El caso de uso es el artefacto fundamental utilizado en la captura de requisitos. Cada forma en que los actores usan el sistema se representa con un caso de uso. Los casos de uso son fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores.

3.6.1. Identificación de actores del Sistema

Un actor especifica un rol que cierta entidad externa adopta cuando interactúa con su sistema directamente. Puede representar un rol de usuario, o un rol desempañado por otro sistema o hardware que toca el límite del sistema. Para identificar los actores, se necesita considerar quién y que utiliza el sistema y qué roles desempeñan en su interacción con el sistema.

- ***Administrador:*** Este actor es el encargado de mantener el sistema, gestionar la base de datos (respaldos, restauraciones, sincronizaciones y conexión), configura el sistema, también posee el privilegio de borrar la información de la base de datos en caso de ser necesario, y además generar los reportes dentro de un rango de tiempo determinado (Históricos) además este actor tiene todos los privilegios del actor Operador.

- ***Operador:*** Este actor posee las funciones de: crear nuevas órdenes, consultarlas aprobarlas, y realizar su despacho, actualizar el inventario, así como modificar tanto las órdenes como el inventario, y generar reportes, e ingresar nueva data al sistema.

- ***Romana:*** Este actor posee las funciones de: pesar los pedidos, consultar los pedidos despachados, imprimir la orden de despacho.

- ***CyC:*** Este actor posee la funciónes de: aprobar las órdenes de prepedido, y consultar los pedidos en espera y los pedidos aprobados.

- ***Despachador:*** Este actor tiene las funciones de: realizar el despacho de las órdenes, modificar las órdenes aprobadas, realizar el inventario, consultarlo e imprimir el reporte de despacho.

- **Base de Datos SDT:** Este es considerado un actor abstracto dentro del sistema que se encarga de ejecutar las transacciones relacionadas con el manejo de los datos, solicitados por los usuarios.

3.7 Casos de usos del sistema

3.7.1 Identificación de caso de uso General

Este caso de uso permite una visión general del sistema, que se predente desarrollar, permitiendo ver las funciones principales del sistema en este caso son: Procesar orden administrador y Procesar orden Operador, y los actores involucrados en el proceso, como lo son el actor Administrador y el actor Operdor, asi como la base de datos SDT. Como puede observar en la figura 3.5, el actor Administrador hereda las propiedades del actor Operador, permitiéndole realizar las operaciones de los Operadores al usuario Administrador, y como se puede observar se define la base de datos como un actor abstracto, y ambas funciones afectan tanto a la base de datos como generan salida hacia el usuario.

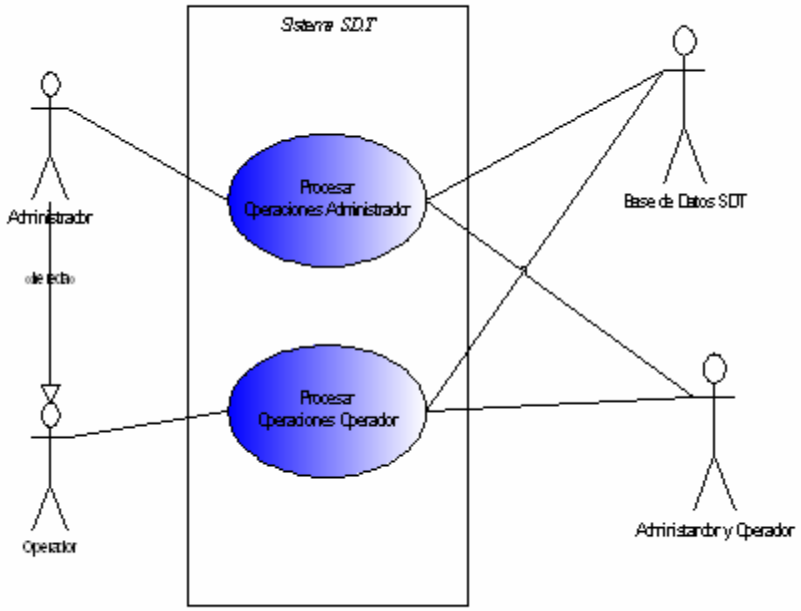


Figura 3.5 caso de uso general del sistema SDT.Fuente propia

3.7.2 Caso de Uso Procesar Operaciones Administrador

El caso de uso Procesar Operaciones Administrador están contempladas todas las operaciones que realiza el administrador del sistema, siendo estas las siguientes: Realizar backups, Restaurar BD, Cambiar Claves, Conectar, Desconectar Generar Históricos Ordenes, Generar Históricos Inventario, Consultar usuarios, estas operaciones son operaciones administrativas que solo son realizadas por el administrador del sistema, y el mismo por estar definido como actor administrativo del sistema no es necesario la confirmación de passsword, estos casos de usos, pueden consultar o modificar la base de datos, la cual es representada por un actor abstracto, los casos de uso aquí presentado se describirán mas adelante en esta misma fase , estas funciones se pueden observar en la figura 3.6.

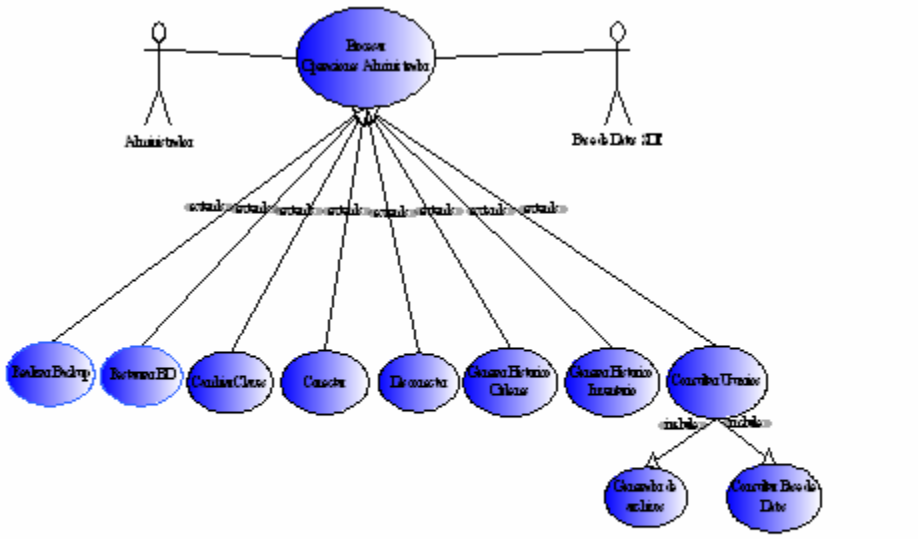


Figura 3.6 Caso de Uso Procesar Operacioes Administrador.Fuente propia

3.7.3 Caso de Uso Porcesar Operaciones Operador

El caso de uso Procesar operaciones Operador, que se pueden observar en la figura 3.7, están contemplados todos los usuarios de tipo operador del sistema, siendo estos presentados por un caso de uso:es decir el usuario promotor esta presentado por el caso de uso Realizar transacciones Promotor,en donde cada una de sus activades des interectauran con el actor abstracto Base de Datos SDT, para acceder a un usuario especifico se realiza prviamente una confirmacion de contraseña, y posteriormente se ingresa al caso de uso extendido.

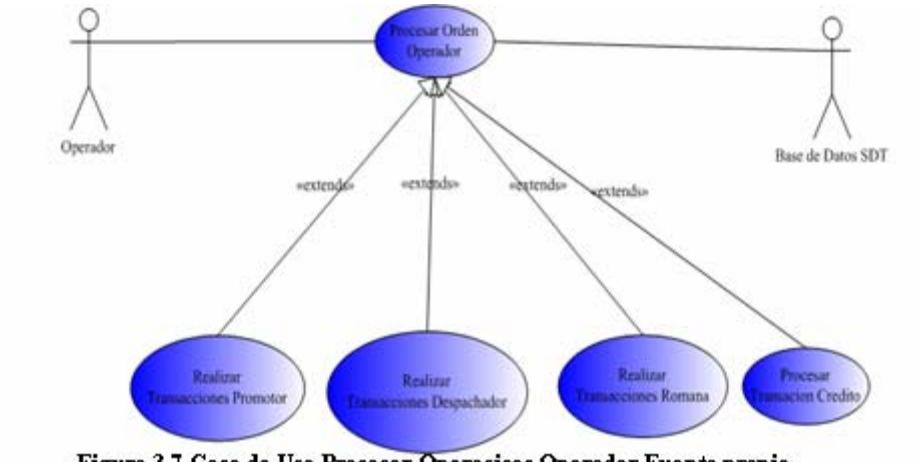


Figura 3.7 Caso de Uso Procesar Operacioes Operador.Fuente propia

3.7.4 Caso de Uso Realizar Transacciones Promotor

En este caso de uso determina las funciones que lleva acabo el operador del sistema, a través de las funciones definidas en el caso de usorealzar transacciones Promotor, en donde para poder realizar cualquier función se verifica primero el password del usuario en este caso puede ser el operador o el Administrador, las operaciones que se pueden llevar a cabo son las siguientes: Procesar Orden, modificar, modificar orden, eliminar orden, consultar el inventario, consultar orden, emitir orden. Este caso de uso se muestra en la figura 3.8

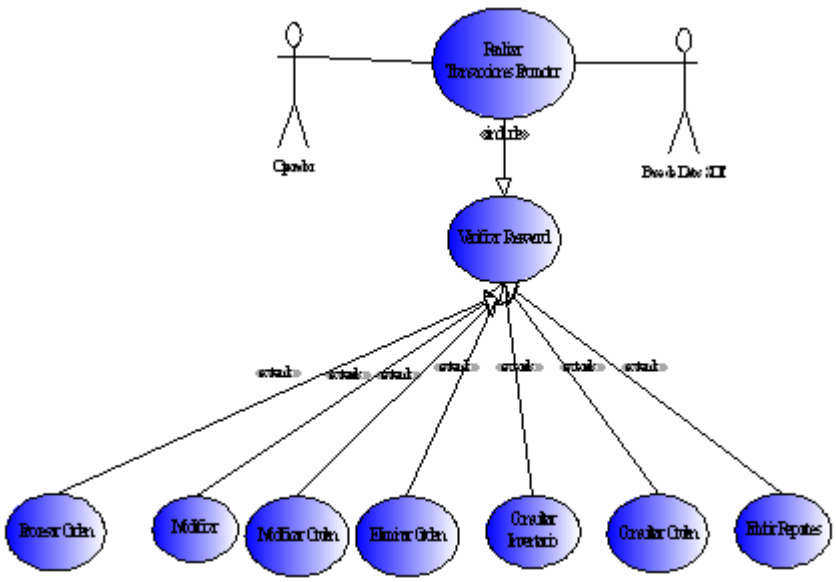


Figura 3.8 Caso de uso Realizar Transacciones Promotor. Fuente propia

3.7.5 Caso de Uso Realizar Transacciones Despachador

Caso de uso donde el operador lleva acabo las operaciones de actualizar el inventario, función que permite llevar el control de la existencia de los productos en el almacén, también permite modificarlo consultarlo, consultar la orden, procesar el despacho de las orden, y emitir los reportes.Este caso de uso se muestra en la figura 3.9.

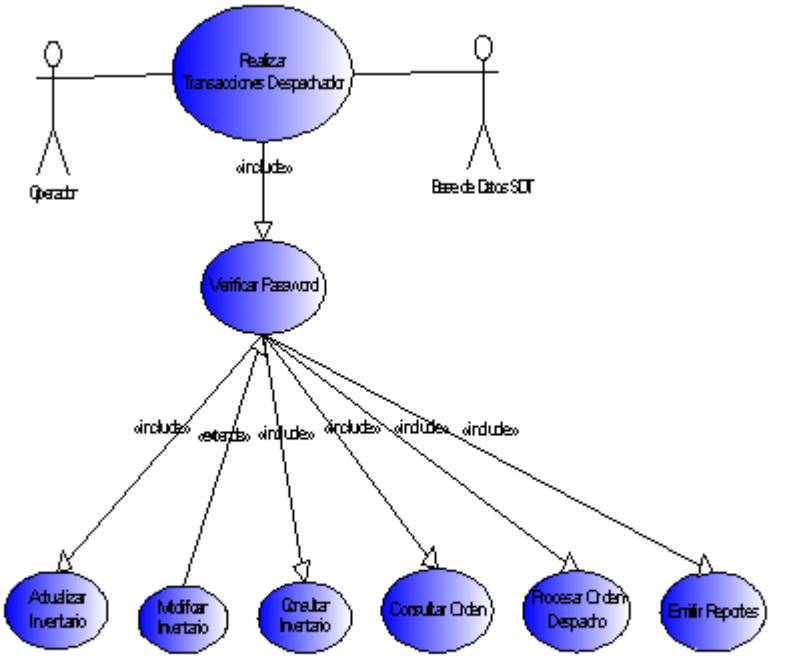


Figura 3.9 caso de uso Realizar Transacciones Despachador.Fuente propia

3.7.6 Caso de uso Realizar Transacciones Romana

Caso de uso mediante el cual el operador puede llevara a cabo las funciones de consultar despachos, es decir la ordenes que fueron pesadas y están listas para ser pesadas, emitir reportes, y Realizar pesada, función mediante la cual la orden es totalizada y descontada del inventario. Este caso de uso se muestra en la figura 3.10.

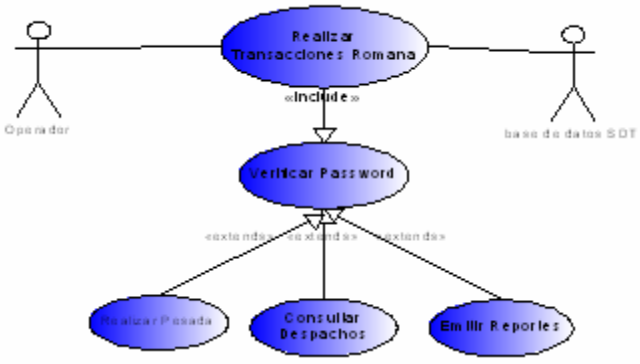


Figura 3.10 caso de uso Realizar Transacciones Romana.Fuente propia

3.7.7 Caso de Uso Procesar Transacciones Créditos

Caso de uso mediante el cual el operador lleva acabo las funciones de consultar despachos, operación mediante la cual el operador puede visualizar las ordenes en que han sido aprobadas así como las que están en espera para su aprobación, y realizar aprobación, función mediante la cual el operador aprueba la orden y automáticamente se genera la orden de despacho .Este caso de uso se muestra en la figura 3.11.

3.7.8 Caso de uso Eliminar

Este caso de uso permite a el operador eliminar la información almacenada en la base de datos, tanto de los clientes como de los productos Este caso de uso se muestra en la figura 3.12.

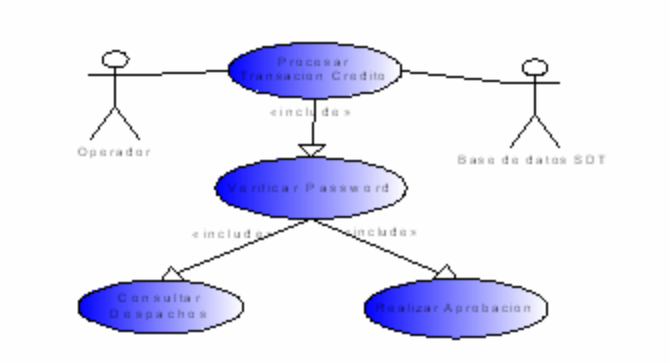


Figura 3.11 Caso de uso Realizar Transacciones Créditos.Fuente propia

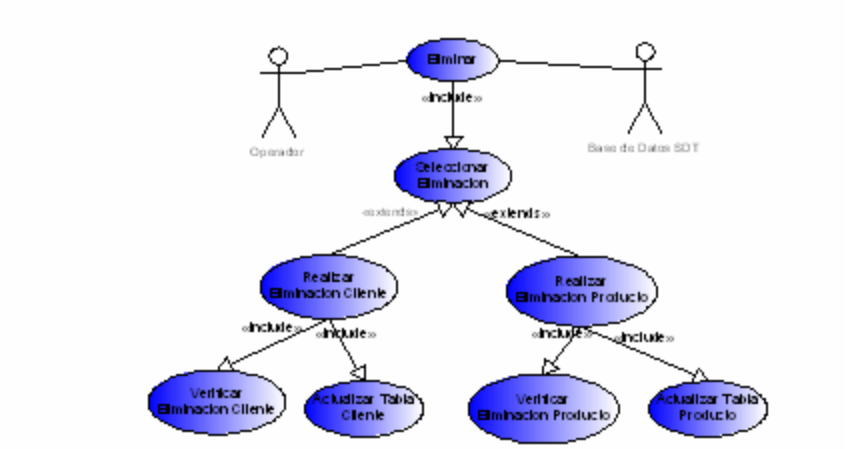


Figura 3.12 caso de uso Eliminar. Fuente propia

3.7.9 Diagrama de Caso de uso eliminar Orden

Este Caso de uso permite eliminar una orden cargada, primeramente seleccionando una orden específica y posteriormente actualizando la base de datos. Este caso de uso se muestra en la figura 3.13.

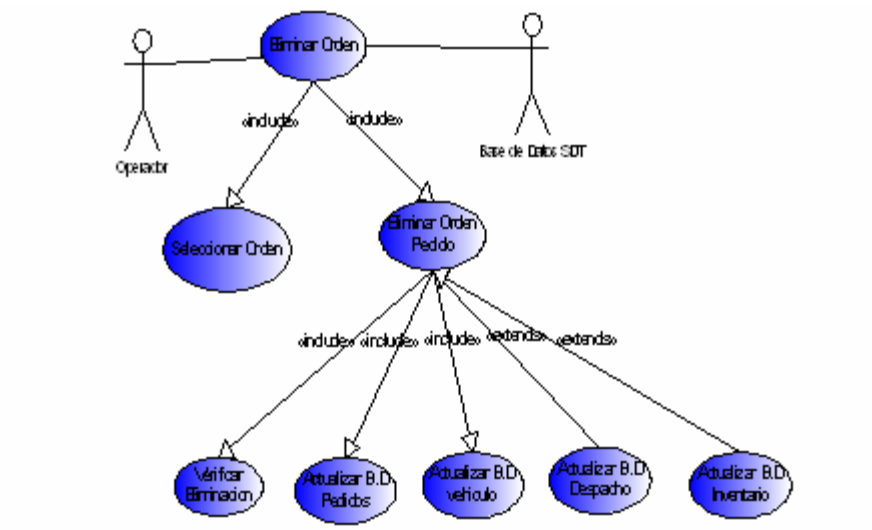


Figura 3.13 Caso de uso Eliminar Orden. Fuente propia

3.7.10 Diagrama de Caso de Uso: Procesar Nueva Orden

Mediante este Caso de uso el operador puede ingresar una nueva orden de pedido al sistema , para llevar a cabo el proceso se realiza una verificación del usuario del sistema, luego se procede a ir hacia los subcasos de uso y seleccionar que operación se quiere realizar e insertar una nueva orden o realizar la aprobación de una orden determinada. La verificación del usuario se realiza para darle acceso a un usuario especifico a ciertas operaciones del programa, por ejemplo: el promotor tiene acceso a la elaboración de la orden , pero no tiene acceso a la aprobación de la misma Este caso de uso se puede observar en la figura 3.14.

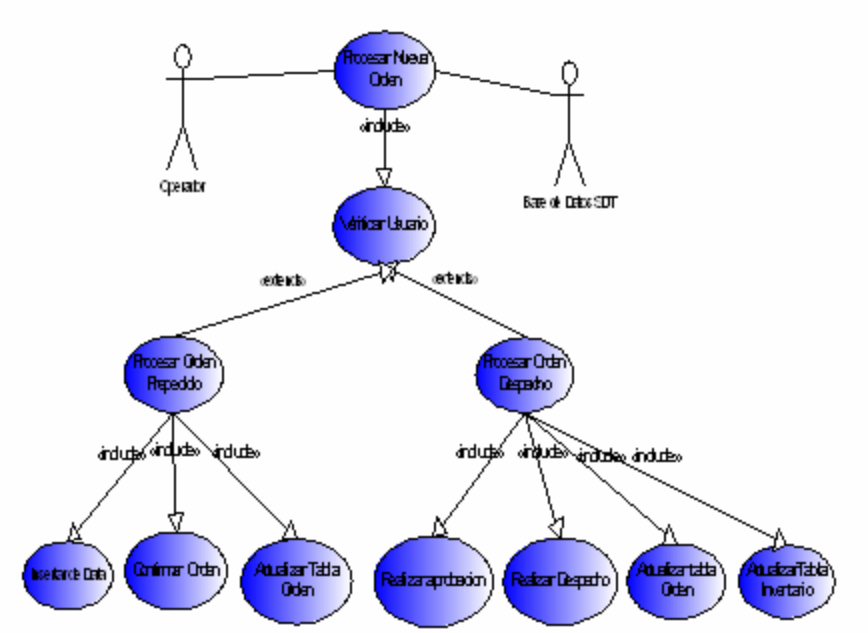


Figura 3.14 Caso de uso: Procesar Orden. Fuente propia

3.7.11 Diagrama de Caso de uso: Modificar Orden

A través de este caso de uso el operador del sistemas puede llevar a cabo la modificación de una orden de pedido especifica, realizando primero una verificación de nuevo ingreso del operador, luego se realiza una verificación del código de la orden, y posteriormente se realiza una actualización de las tablas especificas de la base de datos. Este caso de uso se puede observar en la figura 3.15.

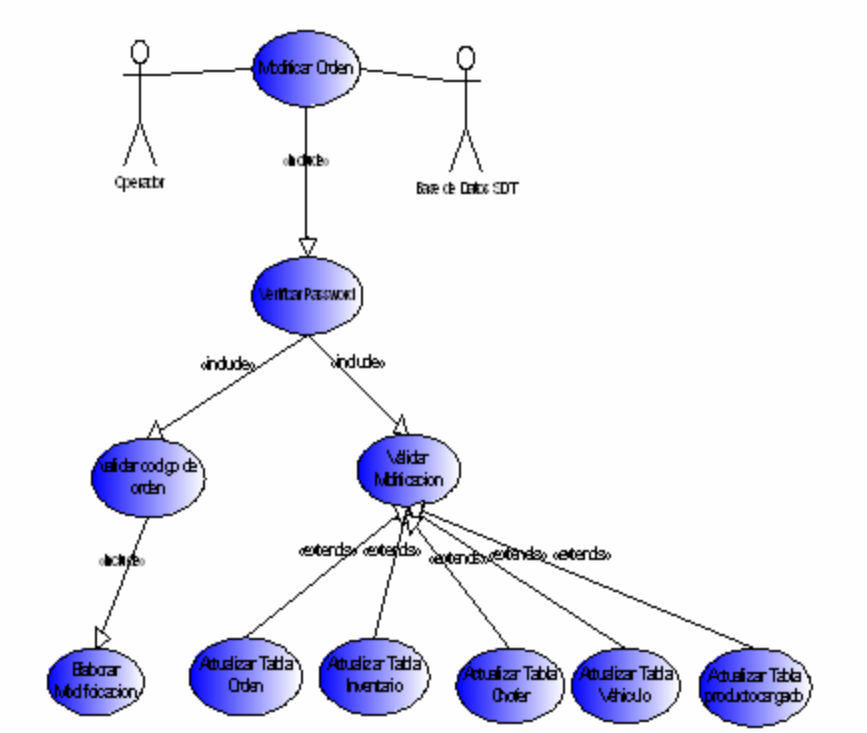


Figura 3.15 Diagrama de Caso de uso: Modificar Orden.Fuente propia

3.7.12 Diagrama de caso de uso Ingresar nuevo

A través de este caso de uso el operador puede ingresar nueva información a la base de datos, para realizar este proceso se lleva a cabo una verificación de password del operador luego se puede ingresar la nueva data tanto de los productos como de los clientes. Este caso de uso se puede observar en la figura 3.16.

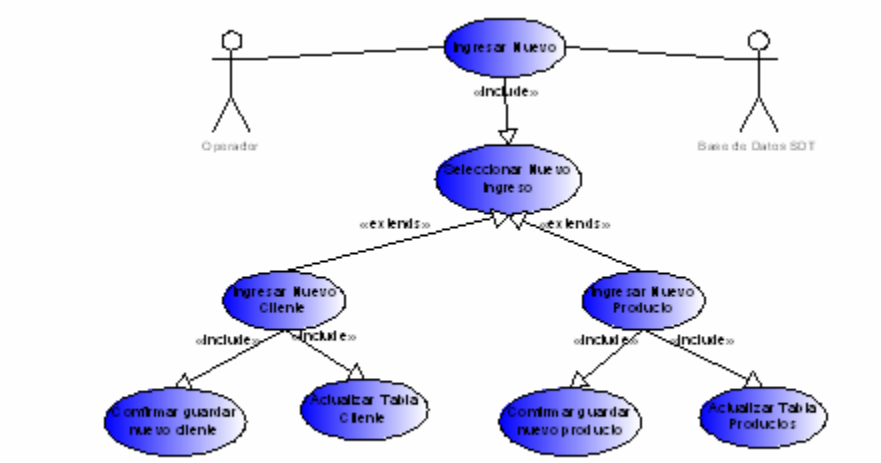


Figura 3.16 caso de uso ingresar Nuevo.Fuente propia

3.7.13 Diagrama de caso de uso Modificar

A través de este caso de uso el operador puede realizar las operaciones de modificar tanto los datos de los clientes y de los productos, accensando el caso de uso correspondiente y posteriormente actualizando la base de datos. Este caso de uso se puede observar en la figura 3.17.

3.7.14 Diagrama de Caso de uso Actualizar Inventario

Mediante este caso de uso el operador de puede actualizar el inventario y realizar el reporte del inventario, la modificación se realiza o ingresando nuevas cantidades posteriormente se verifica y por ultimo se actualiza la base de datos. Este caso de uso se puede observar en la figura 3.18.

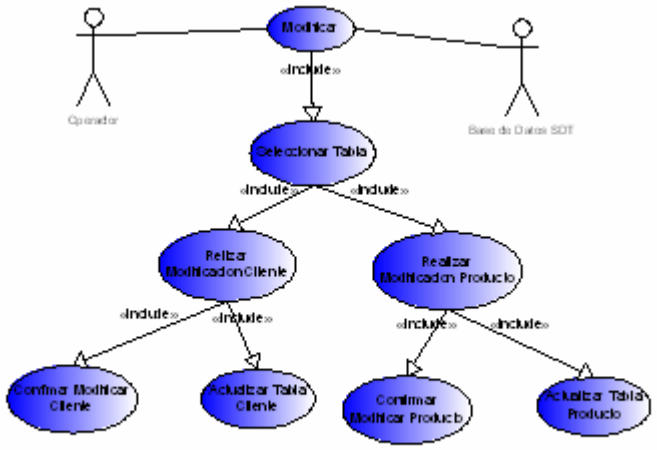


Figura 3.17 Caso de uso Modificar.Fuente propia

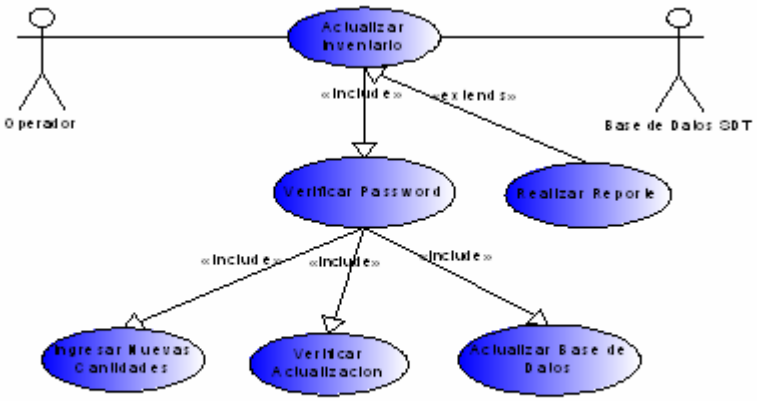


Figura 3.18 Caso de uso Actualizar Inventario.Fuente propia

3.7.15 Diagrama de caso de Uso Consultar Inventario

Mediante este caso de uso el operador podrá realizar la consulta del inventario ya realizado, ingresando primeramente el código del inventario, luego se realiza la consulta por parte de sistema a la base de datos y luego se emitirá un reporte. Este caso de uso se puede observar en la figura 3.19.

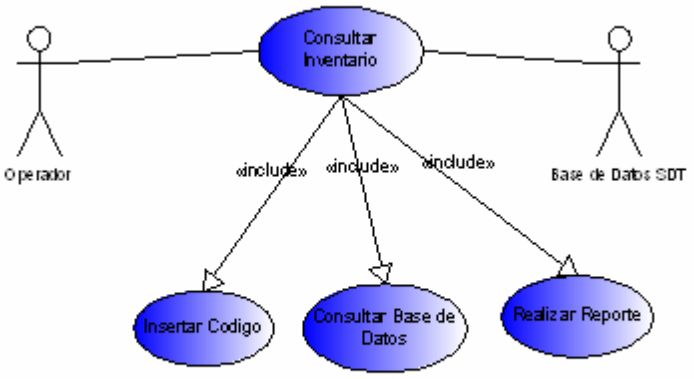


Figura 3.19 Caso de Uso Consultar Inventario.Fuente propia

3.7.16 Diagrama de Caso de Uso Modificar Inventario

Este Caso de uso permite la modificación de un inventario previamente creado, logrando modificar las cantidades de los productos en el inventario así como los carriles en los cuales colocaran la mercancía, para luego actualizar la base de datos posteriormente se envía un reporte. Este caso de uso se puede observar en la figura 3.20.

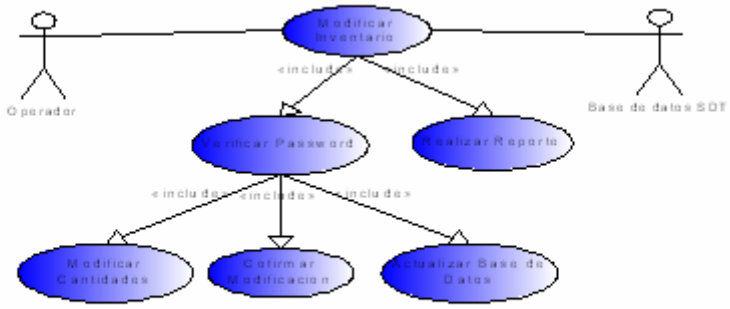


Figura 3.20 Caso de uso Modificar inventario.Fuente propia

3.7.17 Diagrama de casos de Uso Emitir Reporte

Mediante este caso de uso el operador puede emitir los diferentes reportes del sistema que estos vendrían a representar gran parte de la salida del sistema. Este caso de uso se muestra en la figura 3.20.

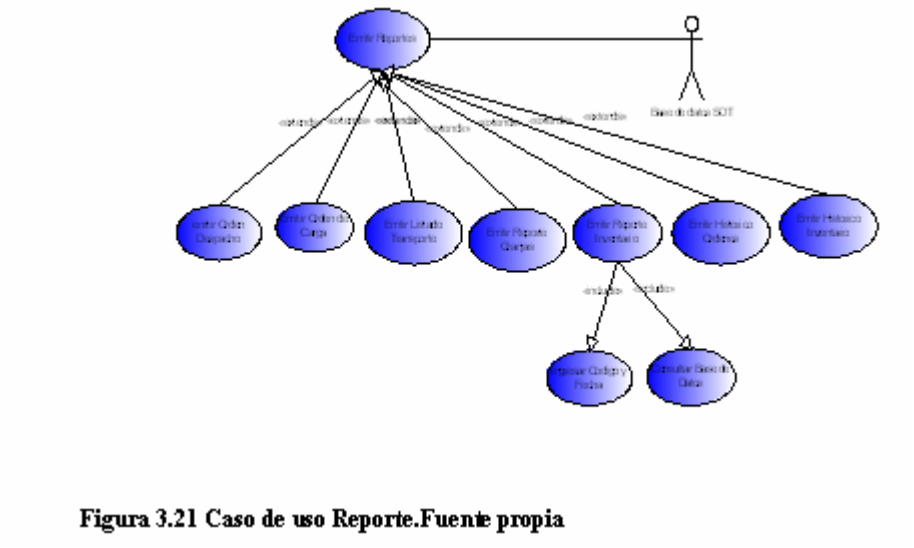


Figura 3.21 Caso de uso Reporte.Fuente propia

3.8 Diagrama de Clase de Análisis

Los diagramas de clases de análisis plasman las posibles clases del diseño, encajándolas en los tres estereotipos básicos: de interfaz, de control y de entidad. Cada uno de ellos implica una semántica específica, lo cual constituye un método consistente de identificar y describir cada clase, contribuyendo a la creación de un modelo de objetos y una arquitectura robusta.

3.8.1 Diagrama de Clases de Análisis Sesiones

En el diagrama de clases de análisis Sesiones el usuario realiza la escogencia de cual sesión va a iniciar el sistema, se identifica una clase interfaz que modela la

interfaz principal con la que el usuario interactúa, una clase de control que se encargue de procesar la acción de inicio de sesión y las entidades Opciones_Admistrativas y Opciones_Operador, son las sesiones de los usuarios principales del sistema. Este diagrama se muestra en la figura 3.22.

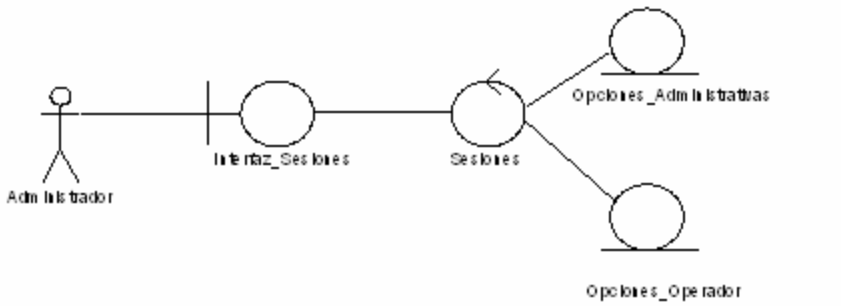


Figura 3.22 Diagrama de Claes de análisis Sesiones.Fuente propia

3.8.2 Diagrama de Clases de Análisis Nueva Orden

El diagrama clases de análisis Nueva orden, mostrado en la figura 3.23, permite realizar la elaboración de una nueva orden y cargarla a la base de datos, en este diagrama se identifica una clase interfaz que representa la interfaz para elaborar la nueva orden y un clase de control a traves e la cual se verifican y anexan todos los valores en la base y una clase entidad que representa la base de datos que va a hacer donde se guardaran los nuevos datos.

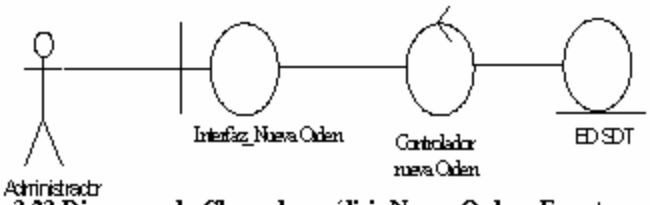


Figura 3.23 Diagrama de Clases de análisis Nueva Orden. Fuente propia

3.8.3 Diagrama de Clases de Análisis Ingresar Nueva Data

El diagrama clase de Análisis Nueva Data, mostrado en la figura 3.24, permite realizar una inserción de nuevos datos, y según la selección del usuario se

puede tener dos flujos de datos, uno para ingresar unos datos de productos, cuyas operaciones de verificación e inserción son gestionadas por el controlador Nuevo Producto, y otro para ingresarlos datos de los clientes, cuyas operaciones son gestionadas por el controlador Nuevo Cliente, y estas consultas van hasta la entidad BD SDT

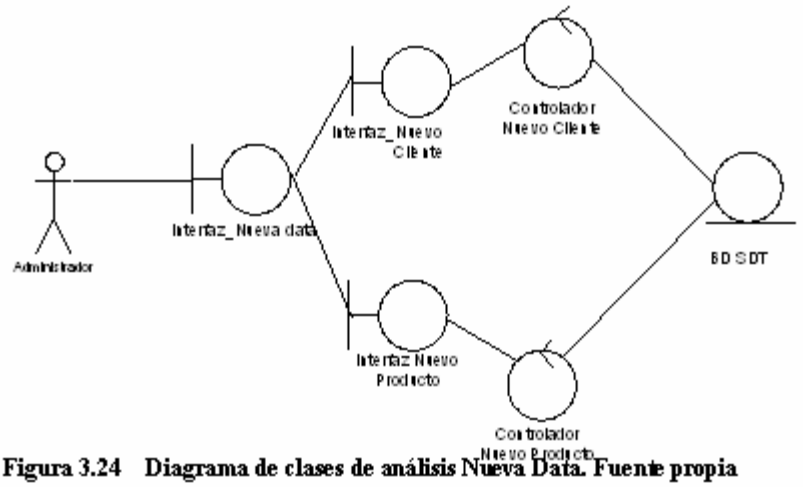


Figura 3.24 Diagrama de clases de análisis Nueva Data. Fuente propia

3.8.4 Diagrama de Clase de Análisis Modificar

El diagrama de clases de análisis Modificar, mostrado en la figura 3.25, le permite al usuario modificar tanto las ordenes los datos del cliente, así como los datos del producto, este flujo es generado activando la interfaz modificar la cual le da acceso al usuario seleccionar uno de estos tres procedimientos, presentados por su interfaz especifica, es decir; la modificar cliente posee una interfaz donde se podrán modificar los datos al usuario, la cual es seguida de un control que realiza todas las operaciones y verificaciones y posteriormente estos datos van a la entidad BD STD.

El flujo de datos modificar producto le da acceso a través de una clase interfaz al el usuario de modificar los daos de un producto especifico, y las operaciones de verificación y carga son realizadas por la clase de control modificar producto y estos datos son almacenados en la entidad BD STD.

El tercer flujo de datos le concede la posibilidad de modificar los datos de entrada a una orden determinada, y cuyas operaciones de verificación y carga de los datos son llevadas a cabo por el controlador Modificar Orden y estos datos son depositados en la entidad BD STD.

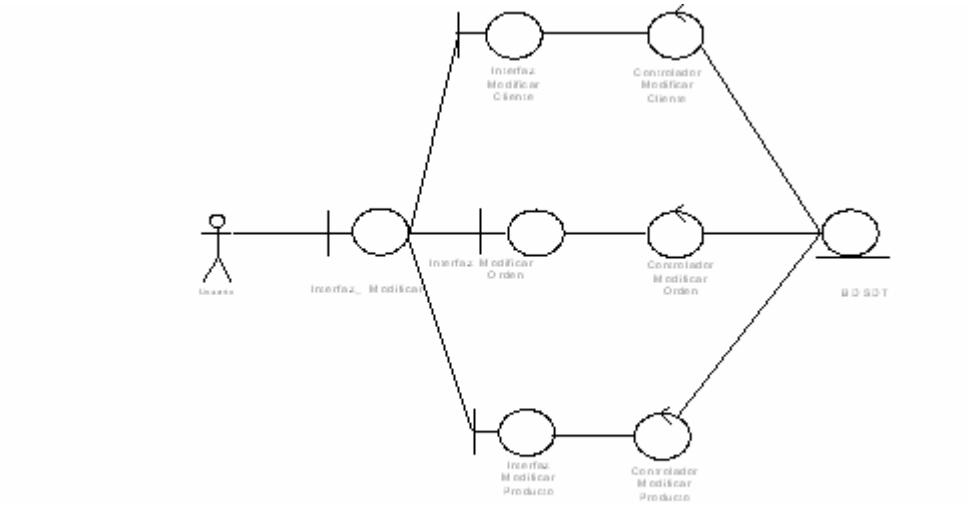


Figura 3.25 Diagrama de clases de clases Modificar.Fuente propia

3.8.5 Diagrama de Clase de Análisis Consultar

Este Diagrama permite llevar a cabo las consultas del sistema, el cual posee cuatro flujos de informacion los cuales son:

El primer flujo de información es invocado desde la interfaz consultar la cual realiza una llamada a la clase interfaz consultar pedidos aprobados el cual muestra todos aquellos pedidos que han sido aprobados y sus operaciones de obtener los datos de la base de datos son llevadas a cabo por el controlador pedidos aprobados y estos van al controlador BD STD.

Para el segundo flujo de información es invocado la interfaz consultar la cual realiza una llamada a la clase interfaz consultar pedidos en espera, el cual muestra

todos los pedidos en espera y sus operaciones son llevadas acabo por el controlador pedidos aprobados y estos son extraidos a través de la clase entidad BD STD.

Para el tercer flujo de información es invocado la interfaz consultar la cual realiza una llamada a la clase interfaz consultar ordenes, el cual muestra los valores de las ordenes y sus operaciones de consulta de los datos son llevadas acabo por el controlador consultar orden y estos son extraídos a través de la clase entidad BD STD.

Para el cuarto flujo de información es invocado la interfaz consultar la cual realiza una llamada a la clase interfaz consultar inventario, el cual muestra todos los valores de los productos existentes, y sus operaciones son llevadas acabo por el controlador pedidos consultar inventario y estos son extraidos a través de la clase entidad BD STD.Estos flujos son diagramados en la figura 3.26.

3.8.6 Diagrama de Clases de Análisis Eliminar

El diagrama de clases eliminar permite la eliminación de los clientes, de los productos así como de los pedidos, teniendo acceso a esta función solo el operador promotor y el administrador del sistema siendo estos discriminados por las sesiones respectivas, este modelo de análisis posee tres flujos de datos en donde para llevar acabo la eliminación de un cliente se realiza una invocación de la interfaz eliminar que realiza una llamada a la interfaz eliminar cliente y esta permite ingresar los datos necesarios para llevara a cabo la eliminación la cual es gestionada por el controlador eliminar cliente realizando un llamado a la entidad BD STD la cual actualiza los nuevos valores de la base de datos. Este diagrama es mostrado en la figura 3.27.

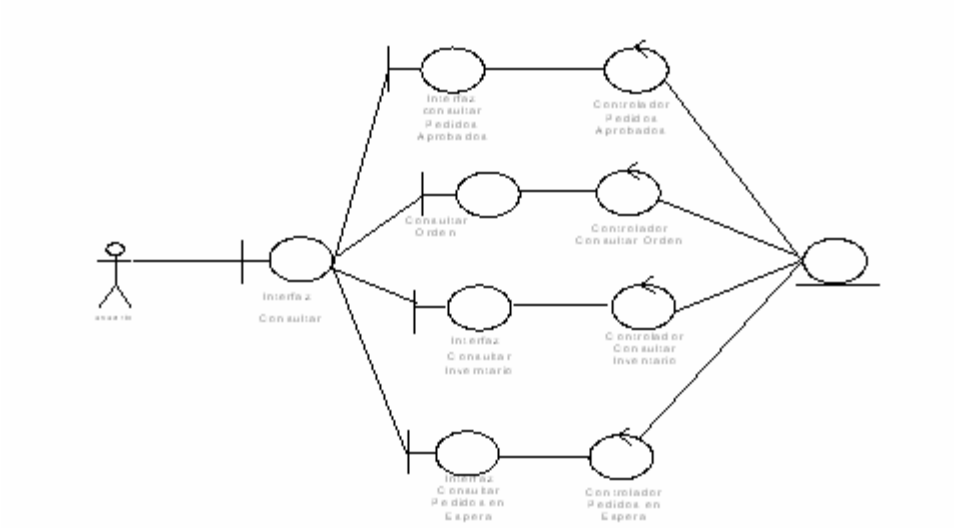


Figura 3.26 diagrama de Clases Consultar.Fuente propia.

En un segundo flujo de datos para llevar acabo la eliminación de un producto se realiza una invocación de la interfaz eliminar que realiza una llamada a la interfaz eliminar producto y esta permite ingresar los datos necesarios para llevara a cabo la eliminación la cual es gestionada por el controlador eliminar producto realizando un llamado a la entidad BD STD la cual actualiza os nuevo valores de la base de datos.

Para el tercer flujo de información para llevar acabo la eliminación de un pedido se realiza una invocación de la interfaz eliminara que realiza una llamada a la interfaz eliminar pedido y esta permite ingresar los datos necesarios para llevara a cabo la eliminación la cual es gestionada por el controlador eliminar pedido realizando un llamado ala entidad BD STD la cual actualiza os nuevo valores de la base de datos.

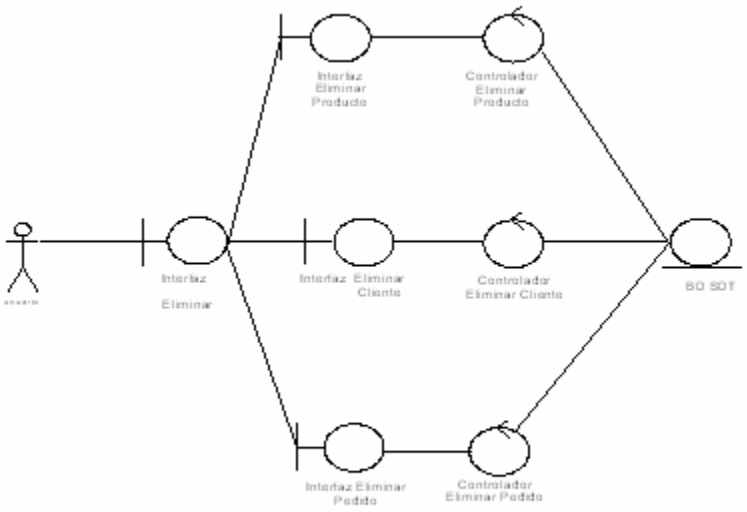


Figura 3.27 Diagrama de clases de Análisis Eliminar.Fuente propia

3.8.7 Diagrama de Clases de análisis Funciones Administrativas

Este diagrama muestra las funciones administrativas del sistemas, estas funciones son solo llevadas a cabo por el administrador del sistema, y le permiten hacer cambios en la base de datos y en el sistema de suma importancia, estas funciones incluyen tanto el mantenimiento como la puesta en marcha del sistema, estas funciones administrativas son : Respaldar la Base de datos , restaurar la base de datos , realizar la conexión con el servidor desde los clientes así como terminar la conexión con el servidor y realizar el cambio de claves de los operadores del sistema, todas estas funciones poseen un controlador que lleva a cabo las función es de consulta o ingreso a la base de datos que es representada por la entidad BD STD. Este diagrama es mostrado en la figura 3.28.

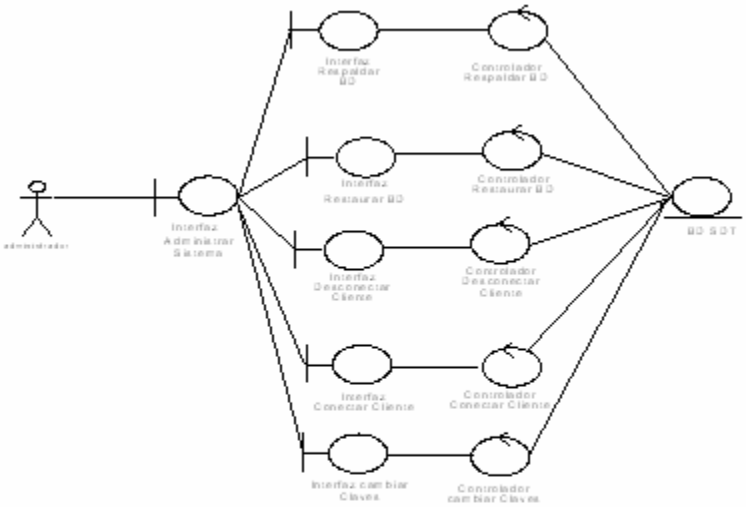


Figura 3.28 Diagrama de clase de análisis Funciones Administrativas.Fuente propia

3.8.8 Diagrama de Clases de Análisis Nuevo Inventario

El diagrama clases de análisis Nuevo Inventario permite realizar la elaboración de un nuevo inventario de los productos en existencia con relacion a la existencia previa de los msimos, y cargarlo a la base de datos, en este diagrama se identifica una clase interfaz que representa la interfaz para elaborar el nuevo inventario y un clase de control a traves e la cual se verifican y anexan todos los valores en la base y una clase entidad que representa la base de datos que va a hacer donde se guardaran los nuevos datos.El diagrama de clases de analisis, nuevo inventario es mostrado en la figura 3.29.

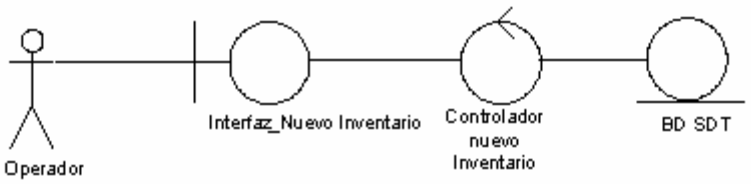


Figura 3.29 Diagrama de clase de análisis Nuevo Inventario.Fuente propia

CAPITULO IV
FASE DE ELABORACION

El objetivo principal de esta fase es alcanzar la línea base de la arquitectura, recopilando la mayoría de los requisitos que aún quedan pendientes. En esta fase se van a transformar y refinar los modelos de la fase de inicio en otra serie de modelos que vayan perfilando una solución más cercana al mundo real.

Al final se tiene la recopilación de todos los requisitos funcionales y la elaboración de un modelo de análisis y de diseño, lo suficientemente sólidos como para construir la arquitectura base del sistema.

Cabe resaltar que SDT es una aplicación de escritorio, desarrollada tomando como guía la metodología orientada a objetos de proceso unificado en espiral a través de UML.

4.1 Diagrama de Clases Detallado

El diagrama de clases permite determinar todas las clases que intervienen en el desarrollo de un flujo de trabajo determinado, y como se encuentran relacionadas entre si.

4.1.1 Diagrama de Clases Detallado Elaboración de una Nueva Orden

A continuación se mostrara en la figura 4.1 el diagrama de clases de análisis para la elaboración de una nueva orden

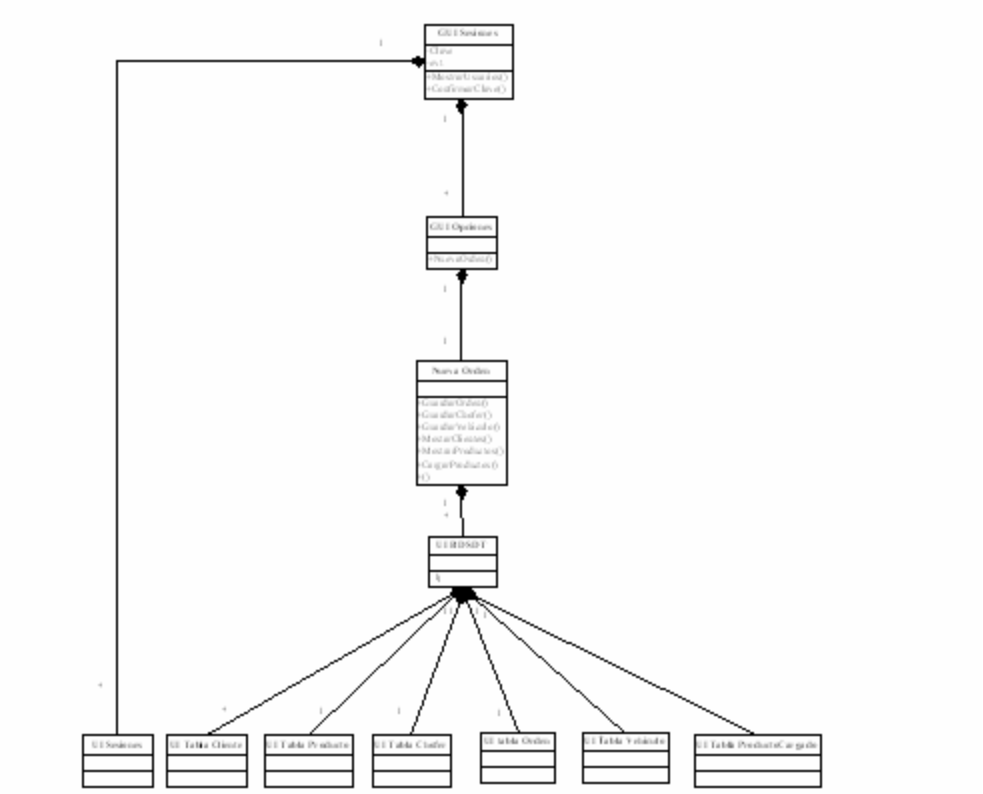


Figura 4.1: Diagrama de Clases de Análisis Detallado para la elaboración de una nueva orden. Fuente propia.

Clase Sesiones

Esta clase permite seleccionar la sesión en la cual va a ser realizada la nueva orden, esta relacionada directamente con la tabla sesiones la cual contiene todos los usuarios autorizados para usar el software.

Clase Opciones

Esta clase le permite al usuario del sistema escoger una opción de un numero de acciones que puede ser llevadas a cabo por el sistema, en este caso se selecciona la opción: nuevo/orden

Clase Nueva Orden

En esta clase en donde se realiza el proceso de selección e inserción de los datos correspondientes a la orden, y los métodos necesarios para ejecutar esta acción.

Clase UI BD SDT

Esta clase representa la base de datos y el enlace entre esta y las tablas que la componen, es decir que la inserción o modificación de los datos de alguna de las tablas existiría una relación entre esta clase y la clase que envía los datos.

Clase UI Tabla Sesiones

Esta clase representa la tabla sesiones que conforma la base de datos, la cual contiene los usuarios y las claves acceso al sistema.

Clase UI tabla Cliente

Clase que representa la tabla cliente que conforma la base de datos del sistema y la cual contiene todos los clientes que realizan operaciones de compra de los productos.

Clase UI Tabla Producto

Clase que representa la tabla producto que constituye parte de la base de datos del sistema, y en donde se almacenan los todos los productos que pueden ser despachados al cliente.

Clase UI Tabla Chofer

Clase que representa la tabla chofer la cual conforma la base de datos el sistema, y en donde se almacenan los chóferes de los transportes a los cuales se les serán despachados los productos terminados.

Clase UI Tabla vehiculo

Clase que representa la tabla vehiculo, la cual conforma la base de datos el sistema, y en donde se almacenan los transportes a los cuales se les serán despachados los productos terminados.

Clase UI Tabla Orden

Clase que representa la tabla orden, la cual conforma la base de datos el sistema, y en donde se almacenan los datos de las órdenes que fueron realizadas y serán despachadas en los transportes.

Clase UI ProductosCargados

Clase que representa la tabla productoscargados, la cual conforma la base de datos el sistema, y en donde se almacenan los productos que fueron cargados en una orden determinada.

4.1.2 Diagrama de Clases Detallado Elaboración de un Nuevo Inventario

A continuación se mostrara en la figura 4.2 el diagrama de clases de análisis para la elaboración de una nuevo Inventario.

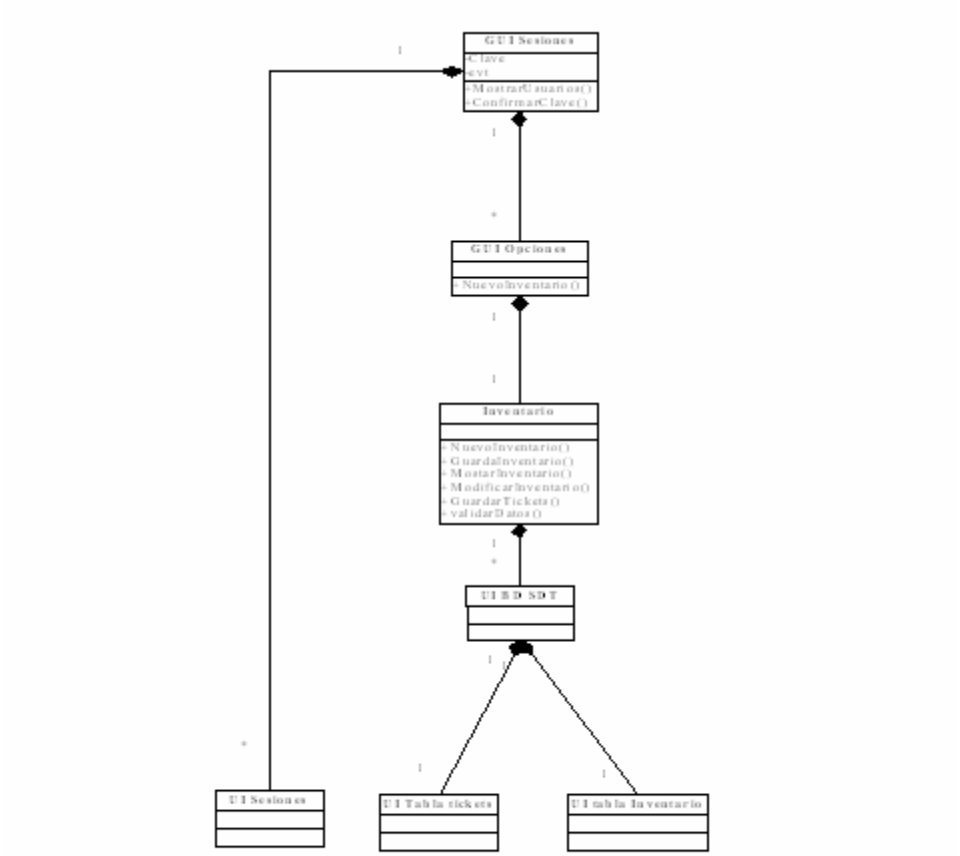


Figura 4.2: Diagrama de Clases de Análisis Detallado para la elaboración de un nuevo Invetario.fuente propia.

Clase Sesiones

Esta clase permite seleccionar la sesión en la cual va a ser realizada la nueva orden, esta relacionada directamente con la tabla sesiones la cual contiene todos los usuarios autorizados para usar el software.

Clase Opciones

Esta clase le permite al usuario del sistema escoger una opción de un numero de acciones que puede ser llevadas a cabo por el sistema, en este caso se selecciona la opción: nuevo/inventario.

Clase Inventario

En esta clase en donde se realiza el proceso de selección e inserción de los datos correspondientes a el nuevo invenatrio, y los métodos necesarios para llevar a cabo esta acción.

Clase UI BD SDT

Esta clase representa la base de datos y el enlace entre esta y las tablas que la componen, es decir que la inserción o modificación de los datos de alguna de las tablas existiría una relación entre esta clase y la clase que envía los datos.

Clase UI Tabla Sesiones

Esta clase representa la tabla sesiones que conforma la base de datos, la cual contiene los usuarios y las claves acceso al sistema.

Clase UI tabla Inventario

Clase que representa la tabla inventario que conforma la base de datos del sistema y la cual contiene los datos correspondientes a un determinado inventario, permitiendo asi llevar a acabo las operaciones en un derterminado inventario.

Clase UI Tabla tickets

Clase que representa la tabla tickets que constituye parte de la base de datos del sistema, y en donde se almacenan las cantidades de un terminado producto, según su número de tickets, y asi llevar un mejor control de los productos, en existencia que seran despachados al cliente.

4.1.3 Diagrama de Secuencia para realizar una Nueva Orden

Diagrama mediante el cual se muestra el intercambio de mensajes (es decir la forma en que se invocan) entre los diferentes objetos que se involucran en el proceso de elaboración de una nueva orden de despacho, es decir Objeto: NUEVA_ORDEN, Transporte, BD_STD, BD_ORDEN, BD_Cliente, entre otros; en un momento dado, asi como el tiempo de vida de cada uno de los procesos que se involucran en el desarrollo de este proceso. La figura 4.3 muestra el diagrama de secuencia para realizar una nueva orden.En este diagrama se puede observar que para realiza una nueva orden el usuario envia un mensaje de selección de nuevo, a traves del objeto nueva orden,la cual reliza una envio deun mensaje de nueva orden a la base de datos del sistema, la cual realiza una consulta a las tablas clientes y productos, y envia los clientes y los productos existentes en la base de datos a la pantalla reprsentada por UI NUEVA ORDEN, posteriormente se realiza un ingresoa traves de la UI NUEVA ORDEN, activando el objeto orden, el cual realiza la validación del codigo de la orden y de los productos y validación del cliente, asi como los datos del chofer y del vehiculo donde sera transportados los productos.posteriormente envia un mensaje con

los dtso anteriormente mencionados al objeto BD_SDT, la cusl realiza un envio de mensaje a el objeto BD_ORDEN, y se realiza un envio de mensaje de confirmacion de guardado a la UI NUEVA ORDEN.

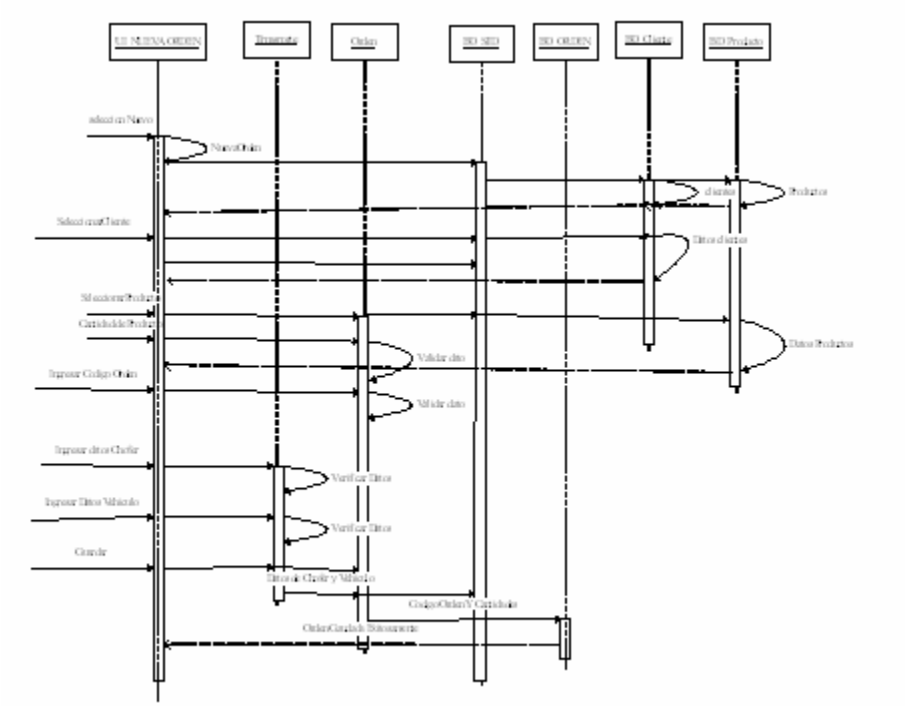


Figura 43 diagrama de secuencia denueva Orden. Fuente propia.

4.1.4 Diagrama de Secuencia para Realizar un Nuevo Inventario

Diagrama mediante el cual se muestra el intercambio de mensajes (es decir la forma en que se invocan) entre los diferentes objetos que se involucran en el proceso de elaboración de un nuevo inventario, es decir Objeto: Inventario, Tickets, BD_STD, BD_INVENTARIO, BD_TICKETS; entre otros, asi como el tiempo de vida de cada uno de los procesos que se involucran en el desarrollo de este proceso. La figura 4.4 muestra el diagrama de secuencia para realizar un nuevo iventario En este diagrama se puede observar que para ealiza un nuevo inventario, se realiza una envio de mensaje a la UI_INVENTARIO, y realiza una confirmacion de insertar inventario, a la UI_INVENTARIO, posteriormente se envian mensajes seleccionanado producto y lsus cantidades atraves del objeto UI_INVENTARIO, esta obejto envia un mensaje al objeto inventario, en la cual se realizan una validación de integridad de datos y envia un mensaje al objeto BD_SDT, que realiza un envio de datos a traves de un mensaje al objeto BD_INVENTARIO,y BD_TIKETS, posteriormente este reenvia un mensaje a la UI_Inventario y consecutivamente un mensaje de confirmacion de guardado.

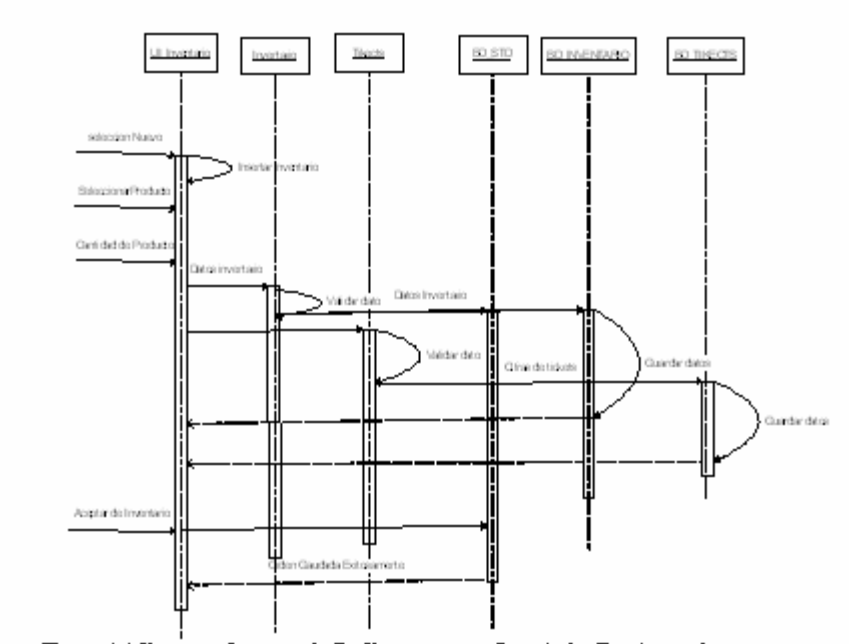


Figura 4.4 diagrama de secuencia Realizar un nuevo Inventario. Fuente propia

4.2 Base de datos del Sistema

Para realizar el diseño de la base de datos, se determinaron las posibles estructuras de las tablas que requiere el sistema, según los requerimientos previamente estudiados. A las tablas identificadas durante este proceso se les aplicará el modelo relacional, el cual mejorará eficientemente la implementación de la base de datos. La figura 4.3 muestra el diagrama relacional (modelo de datos); en este diagrama se observa que consta de nueve tablas en donde, ocho de las cuales guardan relacion entre si, y una tabla aislada, que corresponde a las claves del sistema, en donde se almacenan los usuarios del sistema asi como sus claves,y tambien se puede apreciar la cardinalidad entre las tablas, por ejemplo: la tabla cliente posee una realcion de uno a muchos con la tabla orden, ya que un cliente puede tener varias ordenes almacenadas, asi como la relacion entre la tabla orden y la tabla

productoscaragados, en donde una orden posee varios productos cargados, otra relacion que se puede observar es la recion que existe entre la tabal orden y la tabal chofer , que es de uno a uno, ya que una orden posee un solo chofer, y asu ves un chofer conduce un vehiculo por orden. La tabla claves permite que los usuarios entren al sistema, la primera vez que se inicia el mismo es llenada por un manejador de archivos que realiza la lectura de los usuarios del sistema asi como sus claves desde un archivo de datos y de esta forma permite el acceso de los ususarios al sistema. Para el diseño de esta base de se implemento como manejador de base de datos mysql 4.5.

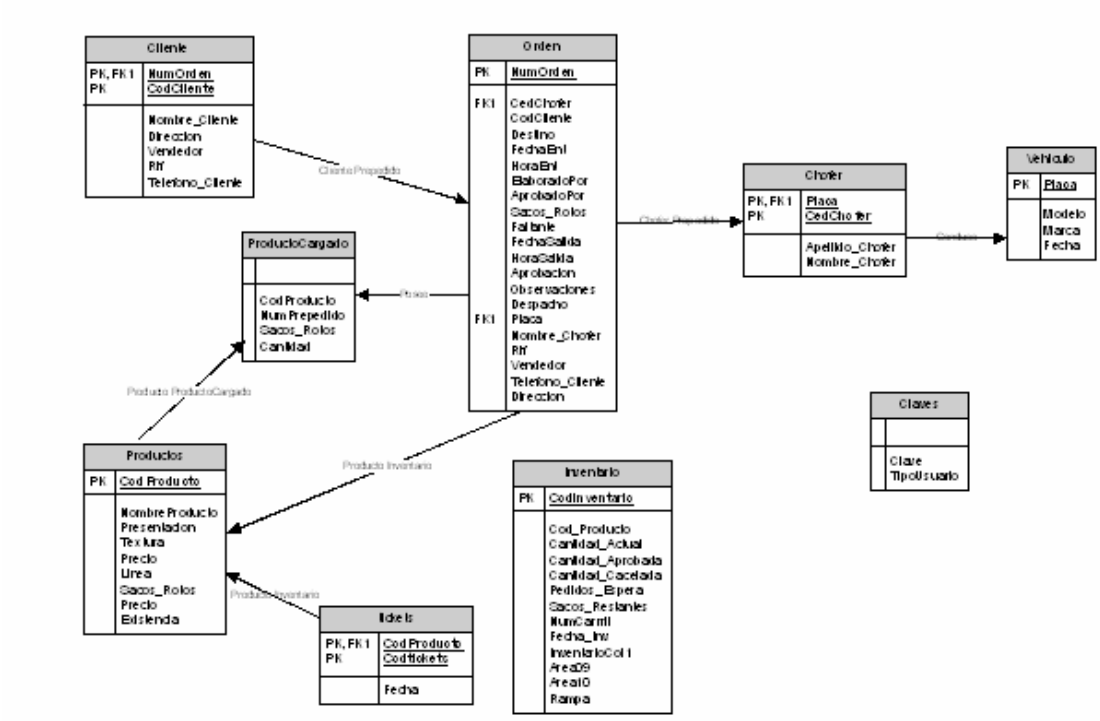


Figura 4.5 Modelo relacional de la base de datos STD .Fuente Propia

4.3 Identificación de Tablas

Durante el proceso de definición de la estructura de las tablas, se encontraron 10 tablas que conformarán el sistema de datos del software, 5 al almacenamiento información correspondiente a la inserción de las ordenes,2 tablas destinadas a la inserción de los datos correspondiente a los clientes y los productos, los cuales son consultados al momento de realizar la nueva orden,1 tabla destinada al almacenamiento de datos relacionados con usuarios2 tabal relacionada con el proceso de actualización del inventario, luego de un análisis profundo se pudo reducir el modelo a 9 tablas En la notación del modelo relacional, los campos clave de las tablas o relaciones son señalados mediante un subrayado.

Tabla Orden

Tabla orden contiene la información correspondiente a la Orden de aprobación de un pedido realizado por un cliente , y el registro correspondiente a ese pedido es alterado a lo largo del proceso de aprobación y despacho del producto.Los campos, tipos de datos, asi como su tamaño pueden ser observados en las tablas 4.1 y 4.2.

Tabla 4.1: Tabla Orden (1/2). Fuente propia

Campo	Tipo	Tamaño
NumOrden	Interger	10
CedChofer	integer	8
CedChofer	integer	8
Destino	varchar	30
FechaEnt :	varchar	10
HoraEnt	varchar	10

Tabla 4.2: Tabla Orden (2/2). Fuente propia

Campo	Tipo	Tamaño
ElaboradoPor	varchar	10
AprobadoPor	varchar	10
Sacos_Rotos	integer	8
Faltante	integer	8
FechaSalida	varchar	10
HoraSalida	varchar	10
Aprobación	varchar	5
Observaciones	Text	30
Despacho	varchar	5
Placa	varcha	10
Rif	Varchar	10
Vendedor	Varchar	15
Telefono_Cliente	Int	10

Tabla Cliente

La tabla cliente contiene los datos correspondientes a los clientes a los cuales se les realiza las órdenes de pedido, esta tabla es consultada al momento de realizar una nueva orden. .Los campos y tipo de datos asi como su tamaño pueden ser observados en la tabla 4.3

Tabla 43: tabla Cliente. Fuente propia

Campo	Tipo	Tamaño
CodCliente	integer	10
NumOrden	integer	10
Nombre_Cliente	varchar	10
Direccion	varchar	20
Vendedor	varchar	10
Rif	varchar	10
Telefono_Cliente	varchar	10

Tabla Productos

La tabla producto contiene los datos correspondientes a los productos que son manufacturados por la planta y estos son los cargados en las órdenes de pedido, esta tabla es consultada al momento de realizar una nueva orden. Los campos y tipo de datos asi como su tamaño pueden ser observados en la tabla 4.4.

Tabla 4.4: Tabla Producto. Fuente propia

Campo	Tipo	Tamaño
CodProducto	integer	10
NombreProducto	varchar	25
Presentacion	varchar	10
Textura	varchar	10
Precio	real	10
Linea	varchar	10
Sacos_Rotos	integer	10
Existencia	integer	10

Tabla ProductoCargado

La tabla ProductoCargado es en donde se almacenan los valores de los productos que son seleccionados en una orden específica por un cliente determinado. Los campos y tipo de datos asi como su tamaño pueden ser observados en la tabla 4.5

Tabla 4.5: Tabla productocargado. Fuente propia		
Campo	**Tipo**	**Tamaño**
CodProducto	integer	10
NumPrepedido	integer	10
Sacos_Rotos	integer	10
Cantidad	integer	10

Tabla Inventario

La tabla inventario contiene la información necesaria para llevar a cabo el control de los productos terminados de la planta, los cuales son depositados en el almacén, esta tabla sirve para llevar un control de la existencia de los mismos. Los campos y tipo de datos asi como su tamaño pueden ser observados en las tablas 4.6 y 4.7.

Tabla 4.6: Tabla inventario (1/2). Fuente propia		
Campo	**Tipo**	**Tamaño**
CodInventario	integer	10
Cod_Producto	integer	10
Cantidad_Actual	integer	10
Cantidad_Aprobada	integer	10

Tabla 4.7: Tabla inventario (2/2). Fuente propia

Campo	Tipo	Tamaño
Cantidad_Cancelada	integer	10
Pedidos_Espera	integer	10
Sacos_Restantes	integer	10
NumCarrril	integer	10
Fecha_Inv	varchar	10
Area09	Integer	10
Area10	Integer	10
Rampa	Integer	10

Tabla Tickets

La tabla ticket contiene la información relacionada al producto elaborado por la planta ya que un mismo producto puede poseer varios códigos de producción y formulas y estos datos son relacionados por un número de ticket. Los campos y tipo de datos asi como su tamaño pueden ser observados en la tabla 4.8

Tabla 4.8: Tabla tickets. Fuente propia

Campo	Tipo	Tamaño
Codtickets	integer	10
CodProducto	integer	10
fecha	varchar	10

Tabla Chofer

La tabla chofer contiene los datos correspondiente a la persona encargada de transportar y conducir el vehiculo que transporta los productos despachados por la planta, estos datos son cargados al momento de elaborar la orden de pedido. Los campos y tipo de datos asi como su tamaño pueden ser observados en la tabla 4.9

.

Tabla 4.9: Tabla chofer.Fuente propia

Campo	Tipo	Tamaño
Placa	varchar	10
CedChofer	integer	10
Apellido_Chofer	varchar	10
Nombre_Chofer	varchar	10

Tabla Vehiculo

La tabla vehiculo contiene la información correspondiente a el vehiculo que transportara los productos que fueron autorizados en la orden de despacho, desde la planta hasta su destino. Los campos y tipo de datos asi como su tamaño pueden ser observados en la tabla 4.10

Tabla 4.10: Tabla vehiculo.Fuente propia

Campo	Tipo	Tamaño
Placa	varchar	10
Modelo	varchar	10
Marca	varchar	10
Fecha	varchar	10

Tabla Claves

La tabla claves contiene la información de los usuarios que pueden realizar operaciones y transacciones en el sistema, sta es una tabla aislada debido a que no influye directamente en el proceso de elaboración y aprobación de una orden de pedido. Los campos y tipo de datos asi como su tamaño pueden ser observados en la tabla 4.11.

Tabla 4.11: Tabla claves.Fuente propia

Campo	Tipo	Tamaño
Clave	varchar	10
TipoUsuario	varchar	10

4.4. Diseño de las interfaces del Sistema

En esta sección de la fase de elaboración se codifican los casos de usos diseñados en la fase de diseño. Las ventanas diseñadas fueron elaboradas utilizando el lenguaje de programación java 2.0 a través de la metodología orientada a objetos, utilizando diagramas uml, mediante la IDE netbeans y como manejador de base de datos MySql 4.5.

La figura 4.6 corresponde a la ventana de elaboración de una nueva orden correspondiente al caso de uso elaborar nueva orden.

En el diseño de las pantallas se desarrollaron combobox para listar los clientes y los productos, para mostrar los productos y sus características se implemento un tabla y cuadros de texto así como etiquetas para Mostrar la información de las pantallas y

para moverse o desplazarse a través del sistema se hizo uso de menús y sub. menús verticales integrándoles controles de acceso rápido.

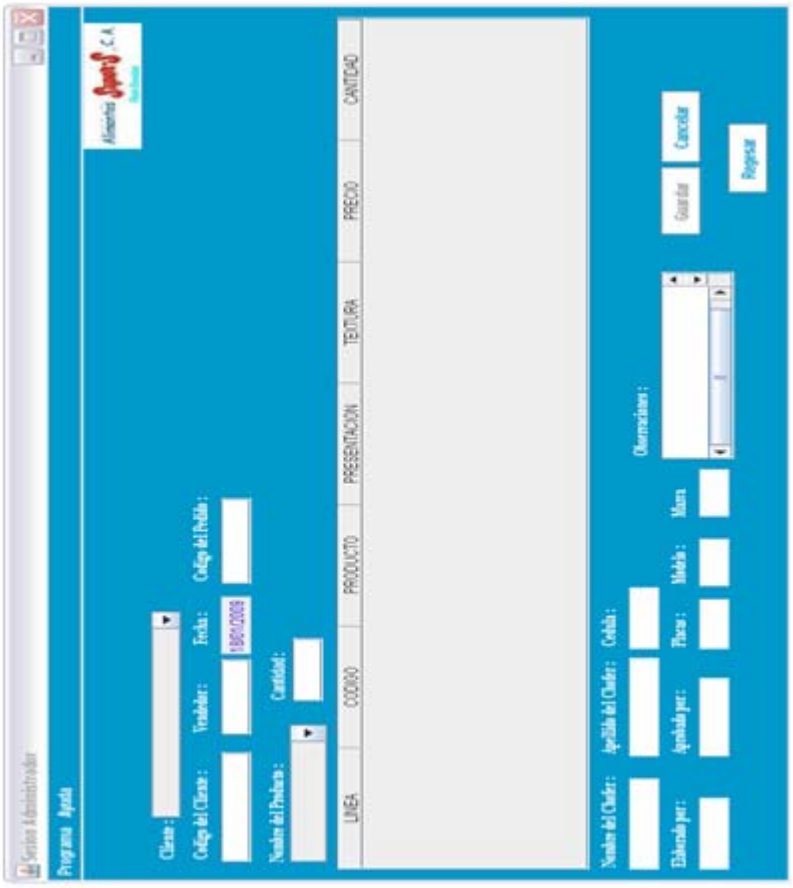


Figura 4.6: Ventana de elaboración de una nueva orden. Fuente propia

CAPITULO V

FASE DE CONSTRUCCION Y TRANSICION

5 1 CONSTRUCCION

En esta fase se implementara la codificación, la cual fue realizada con el lenguaje de programación java 2.0 mediante de la IDE netbeans 5.5 así como las pruebas para lograr los reportes y los posibles errores del sistema; de los caso de uso: Sesiones, Opciones, Procesar orden pre pedido, con el propósito de obtener la versión final del sistema SDT. Para efectos de la explicación sólo será mostrado el código fuente de la interfaces, sesiones, Opciones y nueva orden.

5.1.1 Implementación del caso de uso Sesiones

Figura 5.1: Pantalla sesiones del sistema SDT.Fuente propia

5 1 2. Nombre del fichero: Sesiones.java

Código fuente

```
/*
 * Sesiones.java
 *
 * Created on 18 de mayo de 2008, 05:01 AM
 */

package sapt;

import java.io.*;
import java.sql.*;
import java.io.File;
import java.lang.String;
import java.util.ArrayList;
import javax.swing.JOptionPane;
import javax.swing.JFileChooser;
import javax.swing.DefaultListModel;

/**
 *
 * @author  Ing Ruben Garcia
 */
public class Sesiones extends javax.swing.JFrame {
    private  TemporalUsuarios TempUsuario;
    private conexiones enlace;
    private ResultSet contenedor;
```

```
    private ArrayList<String[]> listas;
    public  Menu ventMenu;
    private Opciones opciones;
    private OpcionesPro OpcPro;
    private OpcionesCredito OpcCre;
    private OpcionesDespacho OpcDesp;
    private OpcionesRomana OpcRom;
    private int codigo;
    private int User;
    /** Creates new form Sesiones */
    public Sesiones(Menu M)
    {
       initComponents();
       this.setLocation(250,150);
       ventMenu =M;
       //Verifica si exite el archivo de configuración
       File Archivo = new File("config.txt");
       if(!Archivo.exists())
       {
          JOptionPane.showMessageDialog(null,"Error: El archivo de
configuración no exite.", "Error",JOptionPane.ERROR_MESSAGE);
          //PConexion configuracion = new PConexion(this);
          this.setVisible(false);
          //configuracion.setVisible(true);
       }
       try{
          enlace=new conexiones();
       }catch(Exception e){}
       TempUsuario= new TemporalUsuarios();
```

```
        try{
            contenedor= enlace.MostrarUsuarios();
            CBSesiones.setModel(new
ResultSetComboBoxModel(contenedor,"Clave","TipoUsuario"));

        // El contenedor es un resultset, "claves" y "TipoUsuarios" son los
nombres de los campos en la base de datos.
        }catch (Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Error: Al mostrar los
Usuarios.\nLa operación fue cancelada.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
        }



        }
    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        ButtonRegresar = new javax.swing.JButton();
```

```
        CBSesiones = new javax.swing.JComboBox();
        jLabel2 = new javax.swing.JLabel();
        TClave = new javax.swing.JPasswordField();
        jButton1 = new javax.swing.JButton();
        MenuSesiones = new javax.swing.JMenuBar();
        Menu = new javax.swing.JMenu();
        Cancelar = new javax.swing.JMenuItem();
        Salir = new javax.swing.JMenuItem();
        Ayuda = new javax.swing.JMenu();
        Pantalla = new javax.swing.JMenuItem();
        Sistema = new javax.swing.JMenuItem();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jPanel1.setBackground(new java.awt.Color(0, 153, 204));

        ButtonRegresar.setBackground(new java.awt.Color(255, 255, 255));
        ButtonRegresar.setFont(new java.awt.Font("Tahoma", 1, 11));
        ButtonRegresar.setForeground(new java.awt.Color(0, 153, 204));
        ButtonRegresar.setText("Cancelar");
        ButtonRegresar.setName("SRegresar"); // NOI18N
        ButtonRegresar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                ButtonRegresarActionPerformed(evt);
            }
        });

        CBSesiones.setForeground(new java.awt.Color(51, 0, 255));
```

```
        CBSesiones.setName("SSesiones"); // NOI18N
        CBSesiones.addItemListener(new java.awt.event.ItemListener() {
            public void itemStateChanged(java.awt.event.ItemEvent evt) {
                CBSesionesItemStateChanged(evt);
            }
        });
        CBSesiones.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                CBSesionesActionPerformed(evt);
            }
        });

        jLabel2.setBackground(new java.awt.Color(255, 255, 255));
        jLabel2.setIcon(new javax.swing.ImageIcon("C:\\Documents and
Settings\\Ing\\Escritorio\\sapt_1\\imagenes\\logo_s.JPG")); // NOI18N
        jLabel2.setText("logo");

        TClave.setFont(new java.awt.Font("Times New Roman", 1, 18));
        TClave.setForeground(new java.awt.Color(51, 0, 255));
        TClave.setCaretColor(new java.awt.Color(51, 0, 255));
        TClave.setEnabled(false);
        TClave.setName("Clave"); // NOI18N
        TClave.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                TClaveActionPerformed(evt);
            }
        });
        TClave.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
```

```
            TClaveKeyTyped(evt);
        }
    });

    jButton1.setBackground(new java.awt.Color(255, 255, 255));
    jButton1.setFont(new java.awt.Font("Tahoma", 1, 11));
    jButton1.setForeground(new java.awt.Color(0, 153, 204));
    jButton1.setText("Aceptar");
    jButton1.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton1ActionPerformed(evt);
        }
    });

    javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
            .addContainerGap(253, Short.MAX_VALUE)
            .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 146,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap())
        .addGroup(jPanel1Layout.createSequentialGroup()
```

```
.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
TRAILING)
                    .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
jPanel1Layout.createSequentialGroup()
                        .addGap(43, 43, 43)
                        .addComponent(ButtonRegresar,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    91,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,       43,
Short.MAX_VALUE)
                        .addComponent(jButton1,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    83,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED))
                    .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
jPanel1Layout.createSequentialGroup()
                        .addGap(28, 28, 28)
                        .addComponent(CBSesiones,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   117,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(15, 15, 15)
                        .addComponent(TClave,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    90,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap(149,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addComponent(jLabel2)
                .addGap(31, 31, 31)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
BASELINE)
                    .addComponent(TClave,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(CBSesiones,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    20,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(66, 66, 66)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
BASELINE)
                    .addComponent(ButtonRegresar)
                    .addComponent(jButton1))
                .addContainerGap(80, Short.MAX_VALUE))
        );

        TClave.getAccessibleContext().setAccessibleName("Clave");

        MenuSesiones.setBackground(new java.awt.Color(0, 153, 204));
```

```
        Menu.setBackground(new java.awt.Color(0, 153, 204));
        Menu.setForeground(new java.awt.Color(255, 255, 255));
        Menu.setText("Menu");


Cancelar.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEv
ent.VK_C, java.awt.event.InputEvent.CTRL_MASK));
        Cancelar.setForeground(new java.awt.Color(0, 153, 204));
        Cancelar.setText("Cancelar");
        Cancelar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                CancelarActionPerformed(evt);
            }
        });
        Menu.add(Cancelar);


Salir.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.
VK_S, java.awt.event.InputEvent.CTRL_MASK));
        Salir.setForeground(new java.awt.Color(0, 153, 204));
        Salir.setText("Salir");
        Salir.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                SalirActionPerformed(evt);
            }
        });
        Menu.add(Salir);

        MenuSesiones.add(Menu);
```

```
        Ayuda.setBackground(new java.awt.Color(0, 153, 204));
        Ayuda.setForeground(new java.awt.Color(255, 255, 255));
        Ayuda.setText("Ayuda");

        Pantalla.setBackground(new java.awt.Color(0, 153, 204));
        Pantalla.setForeground(new java.awt.Color(255, 255, 255));
        Pantalla.setText("Guia de Usuario");
        Ayuda.add(Pantalla);

        Sistema.setBackground(new java.awt.Color(0, 153, 204));
        Sistema.setForeground(new java.awt.Color(255, 255, 255));
        Sistema.setText("Acerca De");
        Ayuda.add(Sistema);

        MenuSesiones.add(Ayuda);

        setJMenuBar(MenuSesiones);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
            );
            layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jPanel1,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            );

            pack();
        }// </editor-fold>//GEN-END:initComponents

        private void TClaveKeyTyped(java.awt.event.KeyEvent evt) {//GEN-
FIRST:event_TClaveKeyTyped



        }//GEN-LAST:event_TClaveKeyTyped

        private void CBSesionesActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_CBSesionesActionPerformed
       // TODO: Agrege su codigo aqui:

codigo=((ResultSetComboBoxModelObject)CBSesiones.getSelectedItem()).getCodig
o();
            //
codigo=((ResultSetComboBoxModelObject)CBSesiones.getSelectedItem()).getCodig
o();
```

```
            //          JOptionPane.showMessageDialog(null,"          "+codigo+"          ",
"Atención",JOptionPane.ERROR_MESSAGE);
            try
            {
               TempUsuario=enlace.MostrarUsuarios(codigo);
            }
            catch (Exception ex) {}
             //**********************Determinacion del Usuario del
sistema*********************************
               if(TempUsuario.Usuario.equals("Administrador"))
               {
                  User=1;
               }
               if(TempUsuario.Usuario.equals("Promotor"))
               {
                  User=2;
               }
               if(TempUsuario.Usuario.equals("Despachador"))
               {
                  User=3;
               }
               if(TempUsuario.Usuario.equals("CyC"))
               {
                  User=4;
               }
               if(TempUsuario.Usuario.equals("Romana"))
               {
                  User=5;
               }
```

```
        }//GEN-LAST:event_CBSesionesActionPerformed

        private void TClaveActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_TClaveActionPerformed
        // TODO: Agrege su codigo aqui:
            String Cl= TClave.getText();

             //*********Administrador************
               switch(User)
               {
                  case 1:
                  {
                   if (codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.valueOf(codigo))*/)
                   {

                       opciones = new Opciones(null);
                       opciones.setVisible(true);
                       this.setVisible(false);

                       try{
                          enlace.Cerrar_Conexion();
                       }catch (Exception ex) {}
                   }
              else
              {
                JOptionPane.showMessageDialog(null,"Error: Clave incorrecta.\n", "Atención",JOptionPane.ERROR_MESSAGE);
```

```
                TClave.setText("");
              }
                  break;
             }
                case 2:
              //*********Promotor************
                {
              if
(codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.value
Of(codigo))*/)
                 {

                    OpcPro = new OpcionesPro(null);
                    OpcPro.setVisible(true);
                    this.setVisible(false);

                    try{
                       enlace.Cerrar_Conexion();
                    }catch (Exception ex) {}
                }
             else
             {
                JOptionPane.showMessageDialog(null,"Error:    Clave    incorrecta.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
                TClave.setText("");
              }
               break;
             }
           case 3:
```

```
            {
                //*********Despachador************

                if
(codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.value
Of(codigo))*/)
                  {

                      OpcDesp = new OpcionesDespacho(null);
                      OpcDesp.setVisible(true);
                      this.setVisible(false);

                      try{
                         enlace.Cerrar_Conexion();
                      }catch (Exception ex) {}
                  }
                else
                {
                  JOptionPane.showMessageDialog(null,"Error:    Clave    incorrecta.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
                  TClave.setText("");
                }
                break;
            }
            case 4:
            {
                //*********Creditos************
```

```
            if
(codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.value
Of(codigo))*/)
              {

                  OpcCre = new OpcionesCredito(null);
                  OpcCre.setVisible(true);
                  this.setVisible(false);

                  try{
                     enlace.Cerrar_Conexion();
                  }catch (Exception ex) {}
              }
            else
            {
              JOptionPane.showMessageDialog(null,"Error:  Clave  incorrecta.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
              TClave.setText("");
            }
             break;
          }
          case 5:
          {
             //*********Romana***********

             if
(codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.value
Of(codigo))*/)
              {
```

```
                    OpcRom = new OpcionesRomana(null);
                    OpcRom.setVisible(true);
                    this.setVisible(false);

                    try{
                       enlace.Cerrar_Conexion();
                    }catch (Exception ex) {}
                  }
                else
                {
                  JOptionPane.showMessageDialog(null,"Error: Clave incorrecta.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
                  TClave.setText("");
                }
                 break;
             }
           }//GEN-LAST:event_TClaveActionPerformed
         }
          private void SalirActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_SalirActionPerformed
        // TODO: Agrege su codigo aqui:
             //ventMenu.dispose();
             //this.dispose();
             System.exit(0);
          }//GEN-LAST:event_SalirActionPerformed

          private void ButtonRegresarActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_ButtonRegresarActionPerformed
```

```
        // TODO: Agrege su codigo aqui:

codigo=((ResultSetComboBoxModelObject)CBSesiones.getSelectedItem()).getCodig
o();
            TClave.setText("");
        }//GEN-LAST:event_ButtonRegresarActionPerformed

        private void CBSesionesItemStateChanged(java.awt.event.ItemEvent evt)
{//GEN-FIRST:event_CBSesionesItemStateChanged
        // TODO: Agrege su codigo aqui:
            TClave.setEnabled(true);

        }//GEN-LAST:event_CBSesionesItemStateChanged

        private void CancelarActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_CancelarActionPerformed
        // TODO: Agrege su codigo aqui:

codigo=((ResultSetComboBoxModelObject)CBSesiones.getSelectedItem()).getCodig
o();
            TClave.setText("");
        }//GEN-LAST:event_CancelarActionPerformed

        private void jButton1ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jButton1ActionPerformed
            // TODO add your handling code here:
            String Cl= TClave.getText();

            //*********Administrador************
```

```
                switch(User)
                {
                   case 1:
                   {
                    if
(codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.value
Of(codigo))*/)
                    {

                        opciones = new Opciones(null);
                        opciones.setVisible(true);
                        this.setVisible(false);

                        try{
                           enlace.Cerrar_Conexion();
                        }catch (Exception ex) {}
                    }
               else
               {
                  JOptionPane.showMessageDialog(null,"Error:    Clave    incorrecta.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
                  TClave.setText("");
               }
                        break;
              }
                  case 2:
                 //*********Promotor***********
                  {
```

```
            if
(codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.value
Of(codigo))*/)
                {

                    OpcPro = new OpcionesPro(null);
                    OpcPro.setVisible(true);
                    this.setVisible(false);

                    try{
                       enlace.Cerrar_Conexion();
                    }catch (Exception ex) {}
                }
              else
              {
                JOptionPane.showMessageDialog(null,"Error:    Clave    incorrecta.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
                TClave.setText("");
              }
               break;
             }
           case 3:
           {
              //*********Despachador***********

              if
(codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.value
Of(codigo))*/)
                {
```

```
                    OpcDesp = new OpcionesDespacho(null);
                    OpcDesp.setVisible(true);
                    this.setVisible(false);

                    try{
                       enlace.Cerrar_Conexion();
                    }catch (Exception ex) {}
                  }
                else
                {
                  JOptionPane.showMessageDialog(null,"Error:  Clave  incorrecta.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
                  TClave.setText("");
                }
                 break;
              }
              case 4:
              {
                 //*********Creditos***********

                 if
(codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.value
Of(codigo))*/)
                   {
                    OpcCre = new OpcionesCredito(null);
                    OpcCre.setVisible(true);
                    this.setVisible(false);
                    try{
```

```
                    enlace.Cerrar_Conexion();
                }catch (Exception ex) {}
              }
            else
            {
              JOptionPane.showMessageDialog(null,"Error:  Clave  incorrecta.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
              TClave.setText("");
            }
             break;
          }
          case 5:
          {
            //*********Romana***********

            if
(codigo==Integer.parseInt(TClave.getText())/*TClave.getText().equals(String.value
Of(codigo))*/)
              {

                OpcRom = new OpcionesRomana(null);
                OpcRom.setVisible(true);
                this.setVisible(false);

                try{
                  enlace.Cerrar_Conexion();
                }catch (Exception ex) {}
              }
            else
```

```
            {
              JOptionPane.showMessageDialog(null,"Error:  Clave  incorrecta.\n",
"Atención",JOptionPane.ERROR_MESSAGE);
              TClave.setText("");
            }
             break;
          }
         }
         }//GEN-LAST:event_jButton1ActionPerformed
         /**
          * @param args the command line arguments
          */
         public static void main(String args[]) {
            java.awt.EventQueue.invokeLater(new Runnable() {
                public void run() {
                   new Sesiones(null).setVisible(true);
                }
            });
         }
         private int equal(String Cl, String string) {
            return 0;
         }
         public void Cerrar()
         {
            try {
                enlace.Cerrar_Conexion();
            } catch (SQLException ex) {}
            this.dispose();
         }
```

```
    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JMenu Ayuda;
    private javax.swing.JButton ButtonRegresar;
    private javax.swing.JComboBox CBSesiones;
    private javax.swing.JMenuItem Cancelar;
    private javax.swing.JMenu Menu;
    private javax.swing.JMenuBar MenuSesiones;
    private javax.swing.JMenuItem Pantalla;
    private javax.swing.JMenuItem Salir;
    private javax.swing.JMenuItem Sistema;
    private javax.swing.JPasswordField TClave;
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JPanel jPanel1;
    // End of variables declaration//GEN-END:variables
    private Object var1;
    private String Cl;
    private String Cl1;
}
```

5.1.3 Implementación del caso de uso: Opciones

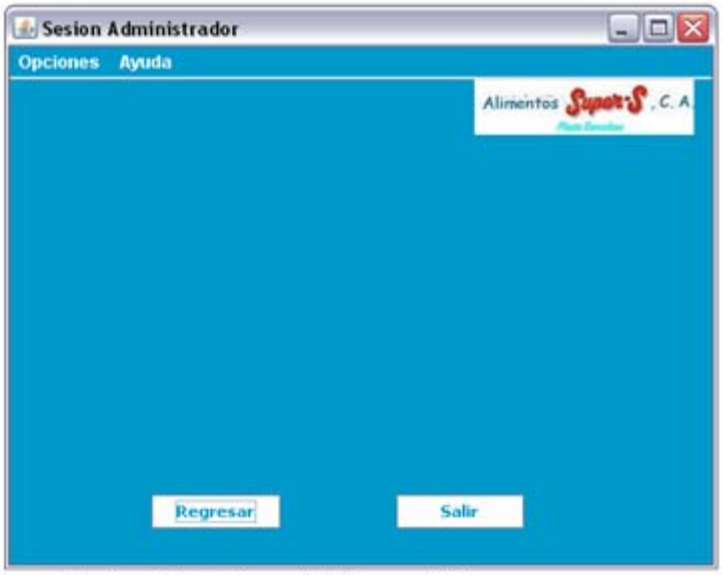


Figura 5.1: Pantalla Opciones del Sistema SDT

5.1.3 Nombre del Fichero: Opciones.java

Código fuente

```
/*
 * Opciones.java
 *
 * Created on 17 de mayo de 2008, 06:43 AM
 */

package sapt;

import java.awt.print.Book;
import java.awt.print.PageFormat;
import java.awt.print.PrinterJob;
```

```
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import javax.swing.JOptionPane;

/**
 *
 * @author  Ing Ruben Garcia
 */
public class Opciones extends javax.swing.JFrame {
    private Creditos Cre;
    private Pedidos pedido;
    private Eliminar elimi;
    private eliminarCliente elC;
    private ConsultaCliente consultaC;
    private ModificarPr ModifPr;
    private Existencia exist;
    private eliminarProducto elP;
    private nuevaOrden NOrden;
    private NuevoCliente NCli;
    private nuevoProducto NPro;
    private Sesiones Sesion;
    private PInventario inv;
    private BackupsBD Back;
    private PConexion conex;
    private DespachoOrd ConsR;
    private CambiarClaves CambC;
    private ModificarCl MCl;
    private MostarOrden ConsC;
    private ConfEnlace CEn;
```

```
    private Impresion Trans;
    private OrdenDesp OrdDesp;
    private conexiones enlace;
    private Historico Hist;
    public  int aux=0; // para determinar la sesion en la cual es abierta la aplicacion
    public int Band=0;
    private String Fech;
    private int Hora;
    private int Min;
    private int Seg;
    private String HoraSist=null;
    /** Creates new form Opciones */
    public Opciones(Sesiones S) {
       initComponents();
       this.setLocation(250,150);
       Sesion = S;
       try{
          enlace=new conexiones();
       }catch (Exception ex) {}

    }
     public void Cerrar()
       {
        try{
          enlace.Cerrar_Conexion();
                // Cerrar_Coneccion();
       }catch (Exception ex) {}
```

```
            try
                {
                   java.util.Date actual=new java.util.Date();

                    Hora=actual.getHours();
                    Min=actual.getMinutes();
                    Seg=actual.getSeconds();

HoraSist=String.valueOf(Hora)+":"+String.valueOf(Min)+":"+String.valueOf(Seg);

                }catch(Exception ex) {
                    JOptionPane.showMessageDialog(null,"Error con la
hora.\n"+ex.getMessage(), "Atención",JOptionPane.ERROR_MESSAGE);
                }
            }

        /** This method is called from within the constructor to
         * initialize the form.
         * WARNING: Do NOT modify this code. The content of this method is
         * always regenerated by the Form Editor.
         */
        // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents
        private void initComponents() {

            jPanel1 = new javax.swing.JPanel();
            logo = new javax.swing.JLabel();
            BRegresar = new javax.swing.JButton();
            BSalir = new javax.swing.JButton();
```

```
jMenuBar1 = new javax.swing.JMenuBar();
Opciones = new javax.swing.JMenu();
CambiarClave = new javax.swing.JMenuItem();
Consultar = new javax.swing.JMenu();
Despacho = new javax.swing.JMenuItem();
Romana = new javax.swing.JMenuItem();
CCodigo = new javax.swing.JMenuItem();
Existencia = new javax.swing.JMenuItem();
En_Espera = new javax.swing.JMenuItem();
Aprobados = new javax.swing.JMenuItem();
Cretidos = new javax.swing.JMenuItem();
Eliminar = new javax.swing.JMenu();
EProducto = new javax.swing.JMenuItem();
ECliente = new javax.swing.JMenuItem();
EOrden = new javax.swing.JMenuItem();
Enlace = new javax.swing.JMenuItem();
inventario = new javax.swing.JMenuItem();
Imprimir = new javax.swing.JMenu();
ITransporte = new javax.swing.JMenuItem();
historico1 = new javax.swing.JMenuItem();
ReporteG = new javax.swing.JMenuItem();
Modificar = new javax.swing.JMenu();
MProducto = new javax.swing.JMenuItem();
CCliente = new javax.swing.JMenuItem();
MOrden = new javax.swing.JMenuItem();
Nuevo = new javax.swing.JMenu();
Producto = new javax.swing.JMenuItem();
Cliente = new javax.swing.JMenuItem();
Orden = new javax.swing.JMenuItem();
```

```
        IRegresar = new javax.swing.JMenuItem();
        Backups = new javax.swing.JMenuItem();
        Salir = new javax.swing.JMenuItem();
        Ayuda = new javax.swing.JMenu();
        Pantalla = new javax.swing.JMenuItem();
        Sistema = new javax.swing.JMenuItem();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Sesion Administrador");
        setBackground(new java.awt.Color(204, 204, 204));

        jPanel1.setBackground(new java.awt.Color(0, 153, 204));
        jPanel1.setForeground(new java.awt.Color(51, 0, 255));

        logo.setBackground(new java.awt.Color(255, 255, 255));
        logo.setIcon(new javax.swing.ImageIcon("C:\\Documents and
Settings\\Ing\\Escritorio\\sapt_1\\imagenes\\logo_s.JPG")); // NOI18N
        logo.setText("logo");

        BRegresar.setBackground(new java.awt.Color(255, 255, 255));
        BRegresar.setFont(new java.awt.Font("Tahoma", 1, 11));
        BRegresar.setForeground(new java.awt.Color(0, 153, 204));
        BRegresar.setText("Regresar");
        BRegresar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                BRegresarActionPerformed(evt);
            }
        });
```

```
        BSalir.setBackground(new java.awt.Color(255, 255, 255));
        BSalir.setFont(new java.awt.Font("Tahoma", 1, 11));
        BSalir.setForeground(new java.awt.Color(0, 153, 204));
        BSalir.setText("Salir");
        BSalir.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                BSalirActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                .addContainerGap(311, Short.MAX_VALUE)
                .addComponent(logo,
javax.swing.GroupLayout.PREFERRED_SIZE,                    146,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(94, 94, 94)
                .addComponent(BRegresar)
                .addGap(77, 77, 77)
```

```
                    .addComponent(BSalir,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap(124, Short.MAX_VALUE))
            );
            jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addComponent(logo)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,    264,
Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
BASELINE)
                        .addComponent(BSalir)
                        .addComponent(BRegresar))
                    .addGap(25, 25, 25))
            );

            jMenuBar1.setBackground(new java.awt.Color(0, 153, 204));
            jMenuBar1.setForeground(new java.awt.Color(51, 0, 255));

            Opciones.setBackground(new java.awt.Color(0, 153, 204));
            Opciones.setForeground(new java.awt.Color(255, 255, 255));
            Opciones.setText("Opciones");
```

```
CambiarClave.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.K
eyEvent.VK_C, java.awt.event.InputEvent.CTRL_MASK));
        CambiarClave.setForeground(new java.awt.Color(0, 153, 204));
        CambiarClave.setText("Cambiar Clave");
        CambiarClave.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                CambiarClaveActionPerformed(evt);
            }
        });
        Opciones.add(CambiarClave);

        Consultar.setForeground(new java.awt.Color(0, 153, 204));
        Consultar.setLabel("Consultar");

        Despacho.setForeground(new java.awt.Color(0, 153, 204));
        Despacho.setText("Despacho");
        Despacho.setAutoscrolls(true);
        Despacho.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                DespachoActionPerformed(evt);
            }
        });
        Consultar.add(Despacho);

        Romana.setForeground(new java.awt.Color(0, 153, 204));
        Romana.setText("Romana");
        Romana.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
        RomanaActionPerformed(evt);
    }
});
Consultar.add(Romana);

CCodigo.setForeground(new java.awt.Color(0, 153, 204));
CCodigo.setText("Codigo de Orden");
CCodigo.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        CCodigoActionPerformed(evt);
    }
});
Consultar.add(CCodigo);

Existencia.setForeground(new java.awt.Color(0, 153, 204));
Existencia.setText("Existencia");
Existencia.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        ExistenciaActionPerformed(evt);
    }
});
Consultar.add(Existencia);

En_Espera.setForeground(new java.awt.Color(0, 153, 204));
En_Espera.setText("Pedidos en Espera");
En_Espera.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        En_EsperaActionPerformed(evt);
    }
```

```
});
Consultar.add(En_Espera);

Aprobados.setForeground(new java.awt.Color(0, 153, 204));
Aprobados.setText("Pedidos Aprobados");
Aprobados.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      AprobadosActionPerformed(evt);
   }
});
Consultar.add(Aprobados);

Opciones.add(Consultar);

Cretidos.setForeground(new java.awt.Color(0, 153, 204));
Cretidos.setText("Creditos");
Cretidos.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      CretidosActionPerformed(evt);
   }
});
Opciones.add(Cretidos);

Eliminar.setForeground(new java.awt.Color(0, 153, 204));
Eliminar.setLabel("Eliminar");

EProducto.setForeground(new java.awt.Color(0, 153, 204));
EProducto.setText("Producto");
EProducto.addActionListener(new java.awt.event.ActionListener() {
```

```
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        EProductoActionPerformed(evt);
    }
});
Eliminar.add(EProducto);

ECliente.setForeground(new java.awt.Color(0, 153, 204));
ECliente.setText("Cliente");
ECliente.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        EClienteActionPerformed(evt);
    }
});
Eliminar.add(ECliente);

EOrden.setForeground(new java.awt.Color(0, 153, 204));
EOrden.setText("Orden Prepedido");
EOrden.setAutoscrolls(true);
EOrden.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        EOrdenActionPerformed(evt);
    }
});
Eliminar.add(EOrden);

Opciones.add(Eliminar);
```

```
Enlace.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK_E, java.awt.event.InputEvent.CTRL_MASK));
        Enlace.setForeground(new java.awt.Color(0, 153, 204));
        Enlace.setText("Enlace");
        Enlace.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                EnlaceActionPerformed(evt);
            }
        });
        Opciones.add(Enlace);

        inventario.setForeground(new java.awt.Color(0, 153, 204));
        inventario.setText("Inventario");
        inventario.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                inventarioActionPerformed(evt);
            }
        });
        Opciones.add(inventario);

        Imprimir.setForeground(new java.awt.Color(0, 153, 204));
        Imprimir.setText("Imprimir");

        ITransporte.setForeground(new java.awt.Color(0, 153, 204));
        ITransporte.setText("Listado Transporte");
        ITransporte.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                ITransporteActionPerformed(evt);
```

```
    }
});
Imprimir.add(ITransporte);

historico1.setForeground(new java.awt.Color(0, 153, 204));
historico1.setText("Historico de Ordenes");
historico1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        historico1ActionPerformed(evt);
    }
});
Imprimir.add(historico1);

ReporteG.setForeground(new java.awt.Color(0, 153, 204));
ReporteG.setText("Reporte de Granjas");
ReporteG.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        ReporteGActionPerformed(evt);
    }
});
Imprimir.add(ReporteG);

Opciones.add(Imprimir);

Modificar.setForeground(new java.awt.Color(0, 153, 204));
Modificar.setLabel("Modificar");

MProducto.setForeground(new java.awt.Color(0, 153, 204));
MProducto.setText("Producto");
```

```
MProducto.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        MProductoActionPerformed(evt);
    }
});
Modificar.add(MProducto);

CCliente.setForeground(new java.awt.Color(0, 153, 204));
CCliente.setText("Cliente");
CCliente.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        CClienteActionPerformed(evt);
    }
});
Modificar.add(CCliente);

MOrden.setForeground(new java.awt.Color(0, 153, 204));
MOrden.setText("Orden Prepedido");
MOrden.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        MOrdenActionPerformed(evt);
    }
});
Modificar.add(MOrden);

Opciones.add(Modificar);

Nuevo.setForeground(new java.awt.Color(0, 153, 204));
Nuevo.setLabel("Nuevo");
```

```
Producto.setForeground(new java.awt.Color(0, 153, 204));
Producto.setText("Producto");
Producto.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        ProductoActionPerformed(evt);
    }
});
Nuevo.add(Producto);

Cliente.setForeground(new java.awt.Color(0, 153, 204));
Cliente.setText("Cliente");
Cliente.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        ClienteActionPerformed(evt);
    }
});
Nuevo.add(Cliente);

Orden.setForeground(new java.awt.Color(0, 153, 204));
Orden.setText("Orden Prepedido");
Orden.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        OrdenActionPerformed(evt);
    }
});
Nuevo.add(Orden);

Opciones.add(Nuevo);
```

```
IRegresar.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEv
ent.VK_R, java.awt.event.InputEvent.CTRL_MASK));
        IRegresar.setForeground(new java.awt.Color(0, 153, 204));
        IRegresar.setText("Regresar");
        IRegresar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                IRegresarActionPerformed(evt);
            }
        });
        Opciones.add(IRegresar);

        Backups.setForeground(new java.awt.Color(0, 153, 204));
        Backups.setText("Respaldos");
        Backups.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                BackupsActionPerformed(evt);
            }
        });
        Opciones.add(Backups);


Salir.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.
VK_S, java.awt.event.InputEvent.CTRL_MASK));
        Salir.setForeground(new java.awt.Color(0, 153, 204));
        Salir.setLabel("Salir");
        Salir.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
        SalirActionPerformed(evt);
    }
});
Opciones.add(Salir);

jMenuBar1.add(Opciones);

Ayuda.setBackground(new java.awt.Color(0, 153, 204));
Ayuda.setForeground(new java.awt.Color(255, 255, 255));
Ayuda.setText("Ayuda");
Ayuda.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        AyudaActionPerformed(evt);
    }
});

Pantalla.setForeground(new java.awt.Color(0, 153, 204));
Pantalla.setText("Guia de Usuario");
Ayuda.add(Pantalla);

Sistema.setForeground(new java.awt.Color(0, 153, 204));
Sistema.setText("Acerca De");
Ayuda.add(Sistema);

jMenuBar1.add(Ayuda);

setJMenuBar(jMenuBar1);
```

```
            javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
            getContentPane().setLayout(layout);
            layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            );
            layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            );

            java.awt.Dimension screenSize = java.awt.Toolkit.getDefaultToolkit().getScreenSize();
            setBounds((screenSize.width-475)/2, (screenSize.height-399)/2, 475, 399);
        }// </editor-fold>//GEN-END:initComponents

        private void historico1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_historico1ActionPerformed
        // TODO: Agrege su codigo aqui:
            this.setVisible(false);
            Hist= new Historico(this);
            Hist.setVisible(true);
        }//GEN-LAST:event_historico1ActionPerformed
```

```
        private void CambiarClaveActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_CambiarClaveActionPerformed
    // TODO: Agrege su codigo aqui:
        this.setVisible(false);
        CambC=new CambiarClaves(this);
        CambC.setVisible(true);
        }//GEN-LAST:event_CambiarClaveActionPerformed

        private void CClienteActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_CClienteActionPerformed
    // TODO: Agrege su codigo aqui:
        this.setVisible(false);
        MCl=new ModificarCl(this);
        MCl.setVisible(true);
        aux=2;
        }//GEN-LAST:event_CClienteActionPerformed

        private void MProductoActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_MProductoActionPerformed
    // TODO: Agrege su codigo aqui:
        ModifPr= new ModificarPr(this);
        ModifPr.setVisible(true);
        aux=1;
    }//GEN-LAST:event_MProductoActionPerformed

        private void inventarioActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_inventarioActionPerformed
    // TODO: Agrege su codigo aqui:
        this.setVisible(false);
```

```
            inv=new PInventario(this);
            inv.setVisible(true);

        }//GEN-LAST:event_inventarioActionPerformed

        private void BackupsActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_BackupsActionPerformed
        // TODO: Agrege su codigo aqui:
            this.setVisible(false);
            Back= new BackupsBD(this);
            Back.setVisible(true);

        }//GEN-LAST:event_BackupsActionPerformed

        private void IRegresarActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_IRegresarActionPerformed
        // TODO: Agrege su codigo aqui:
            this.setVisible(false);
            Sesion.setVisible(true);
             this.dispose();
        }//GEN-LAST:event_IRegresarActionPerformed

        private void SalirActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_SalirActionPerformed
            //this.Sesion.dispose();
            //this.dispose();
             System.exit(0);
             this.dispose();
            //Sesion.ventMenu.dispose();*/
```

```
            //System.exit(0);
        }//GEN-LAST:event_SalirActionPerformed

        private void CretidosActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_CretidosActionPerformed
    // TODO: Agrege su codigo aqui:
            this.setVisible(false); //*****nota: Realizar esto para todo los enlaces*****
             Cre= new Creditos(this);
             Cre.setVisible(true);
        }//GEN-LAST:event_CretidosActionPerformed

        private void AprobadosActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_AprobadosActionPerformed
    // TODO: Agrege su codigo aqui:
            this.setVisible(false);
            pedido= new Pedidos(this);
            pedido.setVisible(true);
        }//GEN-LAST:event_AprobadosActionPerformed

        private void AyudaActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_AyudaActionPerformed
    // TODO: Agrege su codigo aqui:

        }//GEN-LAST:event_AyudaActionPerformed

        private void En_EsperaActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_En_EsperaActionPerformed
    // TODO: Agrege su codigo aqui:
```

```
            this.setVisible(false);
            pedido=new Pedidos(this);
            pedido.setVisible(true);
        }//GEN-LAST:event_En_EsperaActionPerformed

        private void EOrdenActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_EOrdenActionPerformed
    // TODO: Agrege su codigo aqui:
            this.setVisible(false);
            elimi= new Eliminar(this);
            elimi.setVisible(true);

        }//GEN-LAST:event_EOrdenActionPerformed

        private void EClienteActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_EClienteActionPerformed
    // TODO: Agrege su codigo aqui:
             this.setVisible(false);
             elC=new eliminarCliente(this);
             elC.setVisible(true);
        }//GEN-LAST:event_EClienteActionPerformed

        private void ExistenciaActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_ExistenciaActionPerformed
    // TODO: Agrege su codigo aqui:
             this.setVisible(false);
             exist=new Existencia(this);
             exist.setVisible(true);
```

```
    }//GEN-LAST:event_ExistenciaActionPerformed

        private void CCodigoActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_CCodigoActionPerformed
    // TODO: Agrege su codigo aqui:
            this.setVisible(false);
            ConsC= new MostarOrden(this);
            ConsC.setVisible(true);
        }//GEN-LAST:event_CCodigoActionPerformed

        private void EProductoActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_EProductoActionPerformed
    // TODO: Agrege su codigo aqui:
            this.setVisible(false);
            elP=new eliminarProducto(this);
            elP.setVisible(true);
        }//GEN-LAST:event_EProductoActionPerformed

        private void RomanaActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_RomanaActionPerformed
    // TODO: Agrege su codigo aqui:
             this.setVisible(false);
             ConsR=new DespachoOrd(this);
             ConsR.setVisible(true);
        }//GEN-LAST:event_RomanaActionPerformed

        private void OrdenActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_OrdenActionPerformed
    // TODO: Agrege su codigo aqui:
```

```
            //new nuevaOrden(null).setVisible(true);
             this.setVisible(false);
            NOrden= new nuevaOrden(this);
             NOrden.setVisible(true);
        }//GEN-LAST:event_OrdenActionPerformed

        private void ClienteActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_ClienteActionPerformed
    // TODO: Agrege su codigo aqui:
             this.setVisible(false);
             NCli=new NuevoCliente(this);
             NCli.setVisible(true);
        }//GEN-LAST:event_ClienteActionPerformed

        private void ProductoActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_ProductoActionPerformed
    // TODO: Agrege su codigo aqui:
             this.setVisible(false);
             NPro=new nuevoProducto(this);
             NPro.setVisible(true);
        }//GEN-LAST:event_ProductoActionPerformed

        private void DespachoActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_DespachoActionPerformed
            // TODO: Agrege su codigo aqui:
            this.setVisible(false);
            OrdDesp=new OrdenDesp(this);
            OrdDesp.setVisible(true);
        }//GEN-LAST:event_DespachoActionPerformed
```

```
        private void EnlaceActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_EnlaceActionPerformed
            // TODO add your handling code here:
            this.setVisible(false);
            CEn=new ConfEnlace(this);
            CEn.setVisible(true);
        }//GEN-LAST:event_EnlaceActionPerformed

        private void ITransporteActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_ITransporteActionPerformed
            // TODO add your handling code here:

         PrinterJob pj = PrinterJob.getPrinterJob();
            Book libro = new Book();
            PageFormat formatoPagina = new PageFormat();
            PageFormat formatoPagina2 = new PageFormat();
            formatoPagina.setOrientation(PageFormat.LANDSCAPE);
            Impresion pag=new Impresion("PEDIDO SECUENCIA CLIENTES CHOFER DESTINO OBSERVACIONES",enlace.MostrarTransporte()," "," ");
            libro.append(pag,formatoPagina2);
            pj.setPageable(libro);
            if (pj.printDialog())
            {
               try {
                  pj.print();
               } catch (Exception PrinterException) {
                  PrinterException.printStackTrace();
```

```
            }
        }

    }//GEN-LAST:event_ITransporteActionPerformed

    private void ReporteGActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_ReporteGActionPerformed
        // TODO add your handling code here:
          //**************************************Imprimir Orden de Carga*******************************************
          try
             {
                java.util.Date dia=new java.util.Date();
                SimpleDateFormat formato = new SimpleDateFormat("dd/MM/yyyy");
                Fech=formato.format(dia);

             }catch(Exception ex) {}

           PrinterJob pj = PrinterJob.getPrinterJob();
        Book libro = new Book();
         PageFormat formatoPagina = new PageFormat();
        PageFormat formatoPagina2 = new PageFormat();
        formatoPagina.setOrientation(PageFormat.LANDSCAPE);
        ArrayList <String[]> Granjas;
        Granjas=new ArrayList <String[]>();
        Granjas=enlace.BuscarDatosGranjas();
```

```
            ImpresionGranjas pag=new ImpresionGranjas("FECHA Nro DE
TICKETS CHOFER CLIENTES DESTINO Nro DE PEDIDO
OBSERVACION",Granjas,Fech,HoraSist);
            libro.append(pag,formatoPagina);
            pj.setPageable(libro);
            if (pj.printDialog())
            {
               try {
                  pj.print();
               } catch (Exception PrinterException) {
                  PrinterException.printStackTrace();
               }
            }

        }//GEN-LAST:event_ReporteGActionPerformed

        private void MOrdenActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_MOrdenActionPerformed
            // TODO add your handling code here:
      }//GEN-LAST:event_MOrdenActionPerformed

        private void BRegresarActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_BRegresarActionPerformed
            // TODO add your handling code here:
            /*Opcion.setVisible(true);
            this.dispose();*/
            Sesion.setVisible(true);
             this.dispose();
        }//GEN-LAST:event_BRegresarActionPerformed
```

```
    private void BSalirActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_BSalirActionPerformed
        // TODO add your handling code here:
        System.exit(0);
         this.dispose();
    }//GEN-LAST:event_BSalirActionPerformed

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Opciones(null).setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JMenuItem Aprobados;
    private javax.swing.JMenu Ayuda;
    private javax.swing.JButton BRegresar;
    private javax.swing.JButton BSalir;
    private javax.swing.JMenuItem Backups;
    private javax.swing.JMenuItem CCliente;
    private javax.swing.JMenuItem CCodigo;
    private javax.swing.JMenuItem CambiarClave;
    private javax.swing.JMenuItem Cliente;
```

```
private javax.swing.JMenu Consultar;
private javax.swing.JMenuItem Cretidos;
private javax.swing.JMenuItem Despacho;
private javax.swing.JMenuItem ECliente;
private javax.swing.JMenuItem EOrden;
private javax.swing.JMenuItem EProducto;
private javax.swing.JMenu Eliminar;
private javax.swing.JMenuItem En_Espera;
private javax.swing.JMenuItem Enlace;
private javax.swing.JMenuItem Existencia;
private javax.swing.JMenuItem IRegresar;
private javax.swing.JMenuItem ITransporte;
private javax.swing.JMenu Imprimir;
private javax.swing.JMenuItem MOrden;
private javax.swing.JMenuItem MProducto;
private javax.swing.JMenu Modificar;
private javax.swing.JMenu Nuevo;
private javax.swing.JMenu Opciones;
private javax.swing.JMenuItem Orden;
private javax.swing.JMenuItem Pantalla;
private javax.swing.JMenuItem Producto;
private javax.swing.JMenuItem ReporteG;
private javax.swing.JMenuItem Romana;
private javax.swing.JMenuItem Salir;
private javax.swing.JMenuItem Sistema;
private javax.swing.JMenuItem historico1;
private javax.swing.JMenuItem inventario;
private javax.swing.JMenuBar jMenuBar1;
private javax.swing.JPanel jPanel1;
```

```
    private javax.swing.JLabel logo;
    // End of variables declaration//GEN-END:variables

}
```

5.1.4 Implementación del caso de uso Procesar Orden Pre Pedido

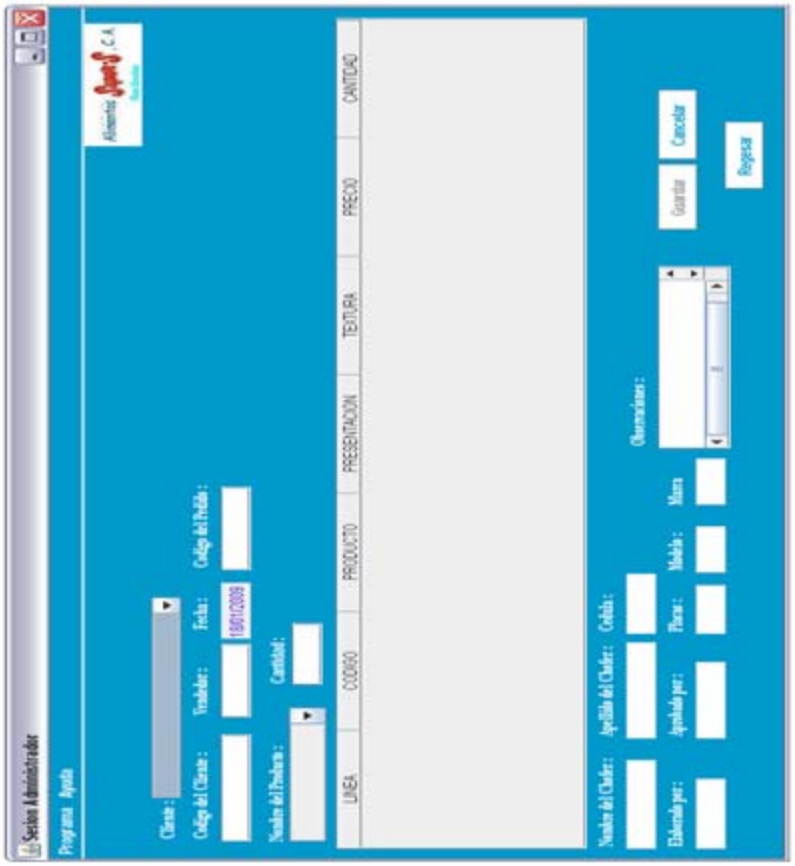


Figura 5.3: Pantalla Nueva Orden del Sistema SDT

5 1 5. Nombre del fichero: Orden.java

Código fuente

```
/*
 * Nuevaorden2.java
```

```
 *
 * Created on 20 de septiembre de 2008, 09:12 AM
 */

package sapt;
import java.sql.*;
import java.lang.String;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import javax.swing.JOptionPane;
/**
/**
 *
 * @author  Ruben García
 */
public class nuevaOrden extends javax.swing.JFrame {
    private DBGrid ModeloOrden;
   //private DBGrid Inventario;
   //private BDTableModel TablaProductos;
    private static conexiones enlace;
    private ResultSet contenedor;
    private ResultSet contenedor2;
    private  ArrayList<String[]> listas;
    private ArrayList <String[]> contenido=null;
    public TemporalCliente TempCliente;
    private TemporalProducto TempProducto;
    private TemporalOrden TempOrden;
    private TemporalTransporte TempTrans;
    private nuevaOrden NOrden;
    //private Opciones Opcion;
    private  String NombrePro;
    private int Cant;
    private int n;
    private int Band=0;
    private String Destino;
```

```
        private int Codigo;
        private int Codigo1;
        private int Codigo2;
        private String NombChofer;
        private String Plac;
        private int Hora;
        private int Min;
        private int Seg;
        private String HoraSist=null;
         private Opciones Opcion;
         private validar validacion;
         private String fech=null;
        /** Creates new form Nuevaorden2 */
        public nuevaOrden(Opciones op) {
            initComponents();
            TSalida.setVisible(false);
            TSalida1.setVisible(false);
            validacion=new validar();
            try{
               enlace=new conexiones();
            }catch (Exception ex) {}
            Opcion =op;
            //*****
            String[]
Titulos={"LINEA","CODIGO","PRODUCTO","PRESENTACION","TEXTURA","PRECI
O","CANTIDAD"};
            ModeloOrden= new DBGrid();
            ModeloOrden.Columnas(7);
            ModeloOrden.AgregarTitulos_Columnas(Titulos);
            TablaProductos.setModel(ModeloOrden);
            /*TablaProductos.setEnabled(true);
            TablaProductos.setVisible(true);*/
            TempProducto=new TemporalProducto();
            TempOrden=new TemporalOrden();
            TempCliente=new TemporalCliente();
```

```
            TempTrans=new TemporalTransporte();

            //Pedidos.setModel(ModeloPedido);
            //********
            try{
                contenedor= enlace.MostrarClientes();
                CBCliente.setModel(new
ResultSetComboBoxModel(contenedor,"codCliente","nombreCliente"));
                contenedor=enlace.MostrarProductos();
                CBProducto.setModel(new
ResultSetComboBoxModel(contenedor, "CodProducto","NombreProducto"));
            //  El  contenedor  es  un  resultset,  "CodProducto"  y
"NombreProducto" son los nombres de los campos en la base de datos.
            }catch (Exception ex) {}

            //Opcion =op;
            try{
           //********************** Se toma la fecha del Sistema
                        java.util.Date dia=new java.util.Date();
                        SimpleDateFormat     formato     =     new
SimpleDateFormat("dd/MM/yyyy");
                        TFecha.setText(formato.format(dia) );
                        TFecha.setEditable(false);
                        fech=TFecha.getText();
                    }catch(Exception ex) {
                        JOptionPane.showMessageDialog(null,"Error
con                   la                   fecha.\n"+ex.getMessage(),
"Atención",JOptionPane.ERROR_MESSAGE);
                    }

             //*******************************Hora del Sistema
             try
                   {
                       java.util.Date actual=new java.util.Date();
```

```
                        Hora=actual.getHours();
                        Min=actual.getMinutes();
                        Seg=actual.getSeconds();

HoraSist=String.valueOf(Hora)+":"+String.valueOf(Min)+":"+String.val
ueOf(Seg);

                    }catch(Exception ex) {
                        JOptionPane.showMessageDialog(null,"Error
con                    la                    hora.\n"+ex.getMessage(),
"Atención",JOptionPane.ERROR_MESSAGE);
                    }

        }
        public void Cerrar()
            {
              try{
                enlace.Cerrar_Conexion();
                       // Cerrar_Coneccion();
            }catch (Exception ex) {}

        }
        public void nuevo()
        {
           TCantidad.setText("0");
            TCodigoClient.setText(" ");
            TVendedor.setText(" ");
            CodPedido.setText(" ");
            try {

                ModeloOrden.BorrarContenido();
                TablaProductos.setModel(ModeloOrden);
                TablaProductos.repaint();
                TablaProductos.setEnabled(false);
                TablaProductos.setEnabled(false);
```

```
                CBProducto.removeAllItems();
                CBCliente.setEnabled(true);
                CBCliente.setEditable(true);
                CBCliente.removeAllItems();
            } catch (Exception ex) {
                JOptionPane.showMessageDialog(null,"Error con la Carga de Los Productos\n"+ex.getMessage(), "Alerta",JOptionPane.ERROR_MESSAGE);
            }

        }
        /** This method is called from within the constructor to
         * initialize the form.
         * WARNING: Do NOT modify this code. The content of this method is
         * always regenerated by the Form Editor.
         */
        //  <editor-fold  defaultstate="collapsed"  desc="Generated Code">//GEN-BEGIN:initComponents
        private void initComponents() {

            PanelOrden = new javax.swing.JPanel();
            Cliente = new javax.swing.JLabel();
            CBCliente = new javax.swing.JComboBox();
            Vendedor = new javax.swing.JLabel();
            TVendedor = new javax.swing.JTextField();
            codigoClient = new javax.swing.JLabel();
            Fecha = new javax.swing.JLabel();
            jLabel2 = new javax.swing.JLabel();
            TFecha = new javax.swing.JTextField();
            CodPedido = new javax.swing.JTextField();
            jLabel1 = new javax.swing.JLabel();
            CBProducto = new javax.swing.JComboBox();
            jLabel6 = new javax.swing.JLabel();
            NombreChofer = new javax.swing.JTextField();
```

```
jLabel7 = new javax.swing.JLabel();
ApChofer = new javax.swing.JTextField();
jLabel8 = new javax.swing.JLabel();
Cedula = new javax.swing.JTextField();
jLabel9 = new javax.swing.JLabel();
Placa = new javax.swing.JTextField();
jLabel10 = new javax.swing.JLabel();
Modelo = new javax.swing.JTextField();
jLabel11 = new javax.swing.JLabel();
jScrollPane1 = new javax.swing.JScrollPane();
Observaciones = new javax.swing.JTextArea();
jLabel12 = new javax.swing.JLabel();
Elaborado = new javax.swing.JTextField();
jLabel13 = new javax.swing.JLabel();
Aprobado = new javax.swing.JTextField();
Guardar = new javax.swing.JButton();
jLabel14 = new javax.swing.JLabel();
Cantidad = new java.awt.Label();
TCantidad = new javax.swing.JTextField();
TCodigoClient = new javax.swing.JTextField();
jScrollPane2 = new javax.swing.JScrollPane();
TablaProductos = new javax.swing.JTable();
TSalida = new javax.swing.JTextField();
TSalida1 = new javax.swing.JTextField();
jLabel15 = new javax.swing.JLabel();
Marca = new javax.swing.JTextField();
BCancelar = new javax.swing.JButton();
BRegresar = new javax.swing.JButton();
jMenuBar1 = new javax.swing.JMenuBar();
Poyecto = new javax.swing.JMenu();
CerrarSesion = new javax.swing.JMenuItem();
MCancelar = new javax.swing.JMenuItem();
Salir = new javax.swing.JMenuItem();
Regresar = new javax.swing.JMenuItem();
Ayuda = new javax.swing.JMenu();
```

```
            Pantalla = new javax.swing.JMenuItem();
            Sistema = new javax.swing.JMenuItem();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
            setTitle("Sesion Administrador ");

            PanelOrden.setBackground(new   java.awt.Color(0,   153,
204));

            Cliente.setBackground(new    java.awt.Color(255,    255,
255));
            Cliente.setFont(new java.awt.Font("Times New Roman", 1,
12));
            Cliente.setForeground(new    java.awt.Color(255,    255,
255));
            Cliente.setText(" Cliente :");

            CBCliente.setForeground(new java.awt.Color(51, 0, 255));
            CBCliente.setToolTipText("");
            CBCliente.addItemListener(new
java.awt.event.ItemListener() {
                public                                          void
itemStateChanged(java.awt.event.ItemEvent evt) {
                    CBClienteItemStateChanged(evt);
                }
            });

            Vendedor.setBackground(new   java.awt.Color(255,   255,
255));
            Vendedor.setFont(new java.awt.Font("Times New Roman", 1,
12));
            Vendedor.setForeground(new   java.awt.Color(255,   255,
255));
            Vendedor.setText(" Vendedor :");
```

```
            TVendedor.setForeground(new java.awt.Color(51, 0, 255));

            codigoClient.setBackground(new  java.awt.Color(255,  255,
255));
            codigoClient.setFont(new    java.awt.Font("Times    New
Roman", 1, 12));
            codigoClient.setForeground(new  java.awt.Color(255,  255,
255));
            codigoClient.setText(" Codigo del Cliente :");

            Fecha.setBackground(new java.awt.Color(204, 204, 204));
            Fecha.setFont(new  java.awt.Font("Times  New  Roman",  1,
12));
            Fecha.setForeground(new java.awt.Color(255, 255, 255));
            Fecha.setText(" Fecha : ");

            jLabel2.setBackground(new    java.awt.Color(204,    204,
204));
            jLabel2.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel2.setForeground(new    java.awt.Color(255,    255,
255));
            jLabel2.setText(" Codigo del Pedido :");

            TFecha.setForeground(new java.awt.Color(51, 0, 204));
            TFecha.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    TFechaActionPerformed(evt);
                }
            });

            CodPedido.setForeground(new java.awt.Color(51, 0, 204));
```

```
            CodPedido.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    CodPedidoActionPerformed(evt);
                }
            });

            jLabel1.setBackground(new    java.awt.Color(204,    204,
204));
            jLabel1.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel1.setForeground(new    java.awt.Color(255,    255,
255));
            jLabel1.setText("Nombre del Producto :");

            CBProducto.setForeground(new    java.awt.Color(51,    0,
255));
            CBProducto.addItemListener(new
java.awt.event.ItemListener() {
                public                                          void
itemStateChanged(java.awt.event.ItemEvent evt) {
                    CBProductoItemStateChanged(evt);
                }
            });
            CBProducto.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    CBProductoActionPerformed(evt);
                }
            });

            jLabel6.setBackground(new    java.awt.Color(204,    204,
204));
```

```
            jLabel6.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel6.setForeground(new    java.awt.Color(255,    255,
255));
            jLabel6.setText(" Nombre del Chofer :");

            NombreChofer.setForeground(new   java.awt.Color(51,   0,
204));
            NombreChofer.setEnabled(false);
            NombreChofer.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    NombreChoferActionPerformed(evt);
                }
            });

            jLabel7.setBackground(new    java.awt.Color(204,    204,
204));
            jLabel7.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel7.setForeground(new    java.awt.Color(255,    255,
255));
            jLabel7.setText(" Apellido del Chofer :");
            jLabel7.setAutoscrolls(true);

            ApChofer.setForeground(new java.awt.Color(51, 0, 204));
            ApChofer.setEnabled(false);
            ApChofer.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    ApChoferActionPerformed(evt);
                }
            });
```

```
            jLabel8.setBackground(new    java.awt.Color(204,    204,
204));
            jLabel8.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel8.setForeground(new    java.awt.Color(255,    255,
255));
            jLabel8.setText(" Cedula :");

            Cedula.setForeground(new java.awt.Color(51, 0, 204));
            Cedula.setEnabled(false);
            Cedula.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    CedulaActionPerformed(evt);
                }
            });

            jLabel9.setBackground(new    java.awt.Color(204,    204,
204));
            jLabel9.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel9.setForeground(new    java.awt.Color(255,    255,
255));
            jLabel9.setText(" Placas :");

            Placa.setForeground(new java.awt.Color(51, 0, 204));
            Placa.setDisabledTextColor(new  java.awt.Color(180,  168,
153));
            Placa.setEnabled(false);
            Placa.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                    PlacaActionPerformed(evt);
                }
            });

            jLabel10.setBackground(new    java.awt.Color(204,    204,
204));
            jLabel10.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel10.setForeground(new    java.awt.Color(255,    255,
255));
            jLabel10.setText(" Modelo :");

            Modelo.setForeground(new java.awt.Color(51, 0, 204));
            Modelo.setEnabled(false);
            Modelo.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    ModeloActionPerformed(evt);
                }
            });

            jLabel11.setBackground(new    java.awt.Color(204,    204,
204));
            jLabel11.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel11.setForeground(new    java.awt.Color(255,    255,
255));
            jLabel11.setText(" Observaciones :");

            Observaciones.setColumns(20);
            Observaciones.setForeground(new   java.awt.Color(51,   0,
255));
            Observaciones.setRows(5);
            Observaciones.setEnabled(false);
```

```
            jScrollPane1.setViewportView(Observaciones);

            jLabel12.setBackground(new   java.awt.Color(204,   204,
204));
            jLabel12.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel12.setForeground(new   java.awt.Color(255,   255,
255));
            jLabel12.setText(" Elaborado por :");

            Elaborado.setForeground(new java.awt.Color(51, 0, 204));
            Elaborado.setEnabled(false);
            Elaborado.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    ElaboradoActionPerformed(evt);
                }
            });

            jLabel13.setBackground(new   java.awt.Color(204,   204,
204));
            jLabel13.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel13.setForeground(new   java.awt.Color(255,   255,
255));
            jLabel13.setText(" Aprobado por :");

            Aprobado.setForeground(new java.awt.Color(51, 0, 204));
            Aprobado.setEnabled(false);
            Aprobado.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    AprobadoActionPerformed(evt);
```

```
                }
            });

            Guardar.setBackground(new    java.awt.Color(255,    255,
255));
            Guardar.setForeground(new java.awt.Color(0, 153, 204));
            Guardar.setText("Guardar");
            Guardar.setEnabled(false);
            Guardar.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    GuardarActionPerformed(evt);
                }
            });

            jLabel14.setBackground(new    java.awt.Color(255,    255,
255));
            jLabel14.setFont(new java.awt.Font("Times New Roman", 1,
11));
            jLabel14.setForeground(new java.awt.Color(51, 0, 204));
            jLabel14.setIcon(new
javax.swing.ImageIcon("C:\\Documents                              and
Settings\\Ing\\Escritorio\\sapt_1\\imagenes\\logo_s.JPG"));        //
NOI18N
            jLabel14.setText("Logo");

            Cantidad.setBackground(new java.awt.Color(0, 153, 204));
            Cantidad.setFont(new java.awt.Font("Times New Roman", 1,
12));
            Cantidad.setForeground(new    java.awt.Color(255,    255,
255));
            Cantidad.setText("Cantidad :");

            TCantidad.setForeground(new java.awt.Color(51, 0, 255));
```

```
            TCantidad.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    TCantidadActionPerformed(evt);
                }
            });

            TCodigoClient.setForeground(new  java.awt.Color(51,  0,
255));

            TablaProductos.setModel(new
javax.swing.table.DefaultTableModel(
                new Object [][] {
                    {null, null, null, null, null, null, null},
                    {null, null, null, null, null, null, null},
                    {null, null, null, null, null, null, null},
                    {null, null, null, null, null, null, null}
                },
                new String [] {
                    "Title 1", "Title 2", "Title 3", "Title 4",
"Título 5", "Título 6", "Título 7"
                }
            ));
            TablaProductos.setEnabled(false);
            jScrollPane2.setViewportView(TablaProductos);

            jLabel15.setBackground(new   java.awt.Color(204,   204,
204));
            jLabel15.setFont(new java.awt.Font("Times New Roman", 1,
12));
            jLabel15.setForeground(new   java.awt.Color(255,   255,
255));
            jLabel15.setText("Marca");
```

```
            Marca.setForeground(new java.awt.Color(51, 0, 204));
            Marca.setEnabled(false);
            Marca.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    MarcaActionPerformed(evt);
                }
            });

            BCancelar.setBackground(new   java.awt.Color(255,   255,
255));
            BCancelar.setForeground(new    java.awt.Color(0,    153,
204));
            BCancelar.setText("Cancelar");
            BCancelar.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    BCancelarActionPerformed(evt);
                }
            });

            BRegresar.setBackground(new   java.awt.Color(255,   255,
255));
            BRegresar.setForeground(new    java.awt.Color(0,    153,
204));
            BRegresar.setText("Regesar");
            BRegresar.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    BRegresarActionPerformed(evt);
                }
            });
```

```
            javax.swing.GroupLayout     PanelOrdenLayout     =     new
javax.swing.GroupLayout(PanelOrden);
            PanelOrden.setLayout(PanelOrdenLayout);
            PanelOrdenLayout.setHorizontalGroup(

PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
                .addGroup(PanelOrdenLayout.createSequentialGroup()
                    .addContainerGap()

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(PanelOrdenLayout.createSequentialGroup()

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                .addComponent(NombreChofer,
javax.swing.GroupLayout.PREFERRED_SIZE,                          109,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(jLabel12)
                                .addComponent(Elaborado,
javax.swing.GroupLayout.PREFERRED_SIZE,                           86,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGap(26, 26, 26)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                .addComponent(jLabel7)
                                .addComponent(ApChofer,
javax.swing.GroupLayout.PREFERRED_SIZE,                          118,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(jLabel13)
```

```
                                .addComponent(Aprobado,
javax.swing.GroupLayout.PREFERRED_SIZE,                              94,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGap(10, 10, 10)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                .addComponent(jLabel8)
                                .addComponent(Cedula,
javax.swing.GroupLayout.PREFERRED_SIZE,                              74,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(PanelOrdenLayout.createSequentialGroup()

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                        .addComponent(jLabel9)
                                        .addComponent(Placa,
javax.swing.GroupLayout.PREFERRED_SIZE,                              60,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                    .addGap(15, 15, 15)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                        .addComponent(Modelo,
javax.swing.GroupLayout.PREFERRED_SIZE,                              57,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                        .addComponent(jLabel10))))
                            .addGap(26, 26, 26)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                .addComponent(Marca,
javax.swing.GroupLayout.PREFERRED_SIZE,                              57,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                                .addComponent(jLabel15))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(PanelOrdenLayout.createSequentialGroup()
                                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
125, Short.MAX_VALUE)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
PanelOrdenLayout.createSequentialGroup()
                                        .addComponent(Guardar)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(BCancelar)
                                        .addGap(100, 100, 100))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
PanelOrdenLayout.createSequentialGroup()
                                        .addComponent(BRegresar)
                                        .addGap(139,        139,
139))))

.addGroup(PanelOrdenLayout.createSequentialGroup()
                                    .addComponent(jLabel11)
```

```
                                        .addContainerGap(461,
Short.MAX_VALUE))))

.addGroup(PanelOrdenLayout.createSequentialGroup()
                            .addComponent(jLabel6,
javax.swing.GroupLayout.PREFERRED_SIZE,                             120,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addContainerGap(911,
Short.MAX_VALUE))))
                .addGroup(PanelOrdenLayout.createSequentialGroup()
                    .addGap(21, 21, 21)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(PanelOrdenLayout.createSequentialGroup()
                            .addComponent(Cliente)
                            .addGap(3, 3, 3)
                            .addComponent(CBCliente,
javax.swing.GroupLayout.PREFERRED_SIZE,                             246,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(PanelOrdenLayout.createSequentialGroup()

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                .addComponent(codigoClient,
javax.swing.GroupLayout.PREFERRED_SIZE,                             120,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(TCodigoClient,
javax.swing.GroupLayout.PREFERRED_SIZE,                             131,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGap(21, 21, 21)
```

```
.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                .addComponent(TVendedor,
javax.swing.GroupLayout.PREFERRED_SIZE,                              90,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(Vendedor,
javax.swing.GroupLayout.PREFERRED_SIZE,                              71,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.TRAILING)

.addGroup(PanelOrdenLayout.createSequentialGroup()
                                    .addComponent(Fecha)
                                    .addGap(31, 31, 31))

.addGroup(PanelOrdenLayout.createSequentialGroup()
                                    .addComponent(TFecha,
javax.swing.GroupLayout.PREFERRED_SIZE,                              69,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addGap(15, 15, 15)))

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                .addComponent(jLabel2)

.addGroup(PanelOrdenLayout.createSequentialGroup()
                                    .addComponent(CodPedido,
javax.swing.GroupLayout.PREFERRED_SIZE,                             102,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addGap(49, 49, 49)
```

```
                                    .addComponent(TSalida,
javax.swing.GroupLayout.PREFERRED_SIZE,                              99,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addGap(31, 31, 31)
                                    .addComponent(TSalida1,
javax.swing.GroupLayout.PREFERRED_SIZE,                             135,
javax.swing.GroupLayout.PREFERRED_SIZE))))

.addGroup(PanelOrdenLayout.createSequentialGroup()

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                .addComponent(jLabel1)
                                .addComponent(CBProducto,
javax.swing.GroupLayout.PREFERRED_SIZE,                             164,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGap(28, 28, 28)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                .addComponent(Cantidad,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(TCantidad,
javax.swing.GroupLayout.PREFERRED_SIZE,                              75,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                    .addContainerGap(272, Short.MAX_VALUE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
PanelOrdenLayout.createSequentialGroup()
                    .addContainerGap(872, Short.MAX_VALUE)
                    .addComponent(jLabel14,
javax.swing.GroupLayout.PREFERRED_SIZE,                             150,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                    .addGap(19, 19, 19))
                .addGroup(PanelOrdenLayout.createSequentialGroup()
                    .addContainerGap()
                    .addComponent(jScrollPane2,
javax.swing.GroupLayout.DEFAULT_SIZE, 1021, Short.MAX_VALUE)
                    .addContainerGap())
            );
            PanelOrdenLayout.setVerticalGroup(

PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
                .addGroup(PanelOrdenLayout.createSequentialGroup()
                    .addComponent(jLabel14,
javax.swing.GroupLayout.PREFERRED_SIZE,                          38,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                        .addComponent(CBCliente,
javax.swing.GroupLayout.PREFERRED_SIZE,                          20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(Cliente))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                        .addComponent(Fecha)
                        .addComponent(jLabel2)
                        .addComponent(codigoClient)
                        .addComponent(Vendedor,
javax.swing.GroupLayout.PREFERRED_SIZE,                          14,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                        .addComponent(TVendedor,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(CodPedido,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(TCodigoClient,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(TSalida,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(TSalida1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(TFecha,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                        .addComponent(jLabel1)
```

```
                        .addComponent(Cantidad,
javax.swing.GroupLayout.PREFERRED_SIZE,                              15,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGap(0, 0, 0)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                        .addComponent(TCantidad)
                        .addComponent(CBProducto))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE,                             166,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(PanelOrdenLayout.createSequentialGroup()
                            .addGap(6, 6, 6)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(PanelOrdenLayout.createSequentialGroup()
                                    .addComponent(jLabel8)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(Cedula,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                                        .addComponent(jLabel13)
                                        .addComponent(jLabel9)
                                        .addComponent(jLabel10)
                                        .addComponent(jLabel15))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                                        .addComponent(Placa,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                        .addComponent(Modelo,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                        .addComponent(Marca,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addGroup(PanelOrdenLayout.createSequentialGroup()

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                                        .addComponent(jLabel7)
                                        .addComponent(jLabel6))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING, false)
                                          .addComponent(NombreChofer)
                                          .addComponent(ApChofer))

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(PanelOrdenLayout.createSequentialGroup()
                                              .addGap(6, 6, 6)
                                              .addComponent(jLabel12)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                              .addComponent(Elaborado,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(PanelOrdenLayout.createSequentialGroup()
                                              .addGap(26, 26, 26)
                                              .addComponent(Aprobado,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))))

.addGroup(PanelOrdenLayout.createSequentialGroup()
                              .addGap(27, 27, 27)
                              .addComponent(jLabel11)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
```

```
                                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,                          48,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(PanelOrdenLayout.createSequentialGroup()

.addGroup(PanelOrdenLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                                        .addComponent(Guardar)
                                        .addComponent(BCancelar))
                                    .addGap(18, 18, 18)
                                    .addComponent(BRegresar)))))
                        .addGap(31, 31, 31))
            );

            jMenuBar1.setBackground(new    java.awt.Color(0,    153,
204));

            Poyecto.setBackground(new java.awt.Color(0, 153, 204));
            Poyecto.setForeground(new    java.awt.Color(255,    255,
255));
            Poyecto.setText("Programa");


CerrarSesion.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.
awt.event.KeyEvent.VK_S, java.awt.event.InputEvent.ALT_MASK));
            CerrarSesion.setForeground(new   java.awt.Color(0,   153,
204));
            CerrarSesion.setText("Cerra Sesion");
            CerrarSesion.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    CerrarSesionActionPerformed(evt);
                }
```

```
            });
            Poyecto.add(CerrarSesion);


MCancelar.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt
.event.KeyEvent.VK_C, java.awt.event.InputEvent.CTRL_MASK));
            MCancelar.setForeground(new    java.awt.Color(0,    153,
204));
            MCancelar.setText("Cancelar");
            MCancelar.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    MCancelarActionPerformed(evt);
                }
            });
            Poyecto.add(MCancelar);


Salir.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.eve
nt.KeyEvent.VK_S, java.awt.event.InputEvent.CTRL_MASK));
            Salir.setForeground(new java.awt.Color(0, 153, 204));
            Salir.setText("Salir");
            Salir.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    SalirActionPerformed(evt);
                }
            });
            Poyecto.add(Salir);


Regresar.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.
event.KeyEvent.VK_R, java.awt.event.InputEvent.CTRL_MASK));
```

```
            Regresar.setForeground(new java.awt.Color(0, 153, 204));
            Regresar.setText("Regresar");
            Regresar.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    RegresarActionPerformed(evt);
                }
            });
            Poyecto.add(Regresar);

            jMenuBar1.add(Poyecto);

            Ayuda.setBackground(new java.awt.Color(0, 153, 204));
            Ayuda.setForeground(new java.awt.Color(255, 255, 255));
            Ayuda.setText("Ayuda");
            Ayuda.addActionListener(new
java.awt.event.ActionListener() {
                public                                          void
actionPerformed(java.awt.event.ActionEvent evt) {
                    AyudaActionPerformed(evt);
                }
            });

            Pantalla.setForeground(new java.awt.Color(0, 153, 204));
            Pantalla.setText("Guia de Usuario");
            Ayuda.add(Pantalla);

            Sistema.setForeground(new java.awt.Color(0, 153, 204));
            Sistema.setText("Acerca De");
            Ayuda.add(Sistema);

            jMenuBar1.add(Ayuda);

            setJMenuBar(jMenuBar1);
```

```
            javax.swing.GroupLayout        layout        =        new
javax.swing.GroupLayout(getContentPane());
            getContentPane().setLayout(layout);
            layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addComponent(PanelOrden,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            );
            layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addComponent(PanelOrden,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            );

            java.awt.Dimension              screenSize              =
java.awt.Toolkit.getDefaultToolkit().getScreenSize();
            setBounds((screenSize.width-1049)/2, (screenSize.height-
519)/2, 1049, 519);
        }// </editor-fold>//GEN-END:initComponents

        private                                                 void
CBClienteItemStateChanged(java.awt.event.ItemEvent    evt)    {//GEN-
FIRST:event_CBClienteItemStateChanged
            // TODO: Agrege su codigo aqui:
```

```
            int
codigo=((ResultSetComboBoxModelObject)CBCliente.getSelectedItem()).g
etCodigo();
            Codigo=codigo;
            try{

                TempCliente=enlace.MostrarCliente(codigo);

TCodigoClient.setText(String.valueOf(TempCliente.Codigo));
                TVendedor.setText(TempCliente.Vendedor);
                Destino=enlace.BuscarDestino(codigo);
            }catch (Exception ex) {}
        }//GEN-LAST:event_CBClienteItemStateChanged

        private                                                 void
TFechaActionPerformed(java.awt.event.ActionEvent     evt)     {//GEN-
FIRST:event_TFechaActionPerformed
            // TODO: Agrege su codigo aqui:
        }//GEN-LAST:event_TFechaActionPerformed

        private                                                 void
CodPedidoActionPerformed(java.awt.event.ActionEvent    evt)    {//GEN-
FIRST:event_CodPedidoActionPerformed
            // TODO add your handling code here:
            boolean valor =CodPedido.getText().isEmpty();
            if(CodPedido.getText()!=null && valor==false)
            {
            try
            {
             TCantidad.setEnabled(true);

TempOrden.NPedido=Integer.parseInt(CodPedido.getText());
            }
            catch(Exception ex)
            {
```

```
                JOptionPane.showMessageDialog(null, "Error ingrese
solo       numeros\n"       +       ex.getMessage(),       "Alerta",
JOptionPane.ERROR_MESSAGE);
                CodPedido.setText("");
             }
             }

             else
             {
              JOptionPane.showMessageDialog(null, "Error ingrese un
valor numerico \n", "Alerta", JOptionPane.ERROR_MESSAGE);

             }
         }//GEN-LAST:event_CodPedidoActionPerformed

         private                                                void
CBProductoItemStateChanged(java.awt.event.ItemEvent   evt)   {//GEN-
FIRST:event_CBProductoItemStateChanged
             // TODO: Agrege su codigo aqui:
              String[] LineaCont=null;



             int
codigo=((ResultSetComboBoxModelObject)CBProducto.getSelectedItem()).
getCodigo();
             Codigo1=codigo;
             contenedor=null;
             contenido  =  enlace.BuscarProducto(Codigo1);//  busqueda
de los datos en la base de datos
             for(int i=0;i<contenido.size();i++)
                 {
                   LineaCont=contenido.get(i);
               }
             // enviamos el codigo del cliente por pantalla
```

```
            //int
codigo1=((ResultSetComboBoxModelObject)CBCliente.getSelectedItem()).
getCodigo();
            //CodPedido.setText(String.valueOf(codigo1));
            TSalida1.setText(String.valueOf(codigo));

            try{
                TablaProductos.removeAll();//debe de removel todo el
contenido de la tabla
                TablaProductos.repaint();
               String[] linea1={" "," "," "," "," "," "," "};

                int k=TablaProductos.getRowCount();

                System.out.println("k en combo pro:"+k);

                if(k==0)
                {
                    ModeloOrden.AgregarLinea(linea1);
                   TablaProductos.setModel(ModeloOrden);
                   //Agergando los elementos en el jtable
                    TablaProductos.setValueAt(LineaCont[0], k, 0);

TablaProductos.setValueAt(String.valueOf(codigo),k , 1);
                    TablaProductos.setValueAt(LineaCont[2], k, 2);
                    TablaProductos.setValueAt(LineaCont[3], k, 3);
                    TablaProductos.setValueAt(LineaCont[4], k, 4);
                    TablaProductos.setValueAt(LineaCont[5], k, 5);
                }

Codigo2=Integer.parseInt(String.valueOf(ModeloOrden.getValueAt(k-1,
1)));
                if(k>=1 && codigo!=Codigo2)
                {
                    ModeloOrden.AgregarLinea(linea1);
```

```
                    TablaProductos.setModel(ModeloOrden);
                    TablaProductos.setValueAt(LineaCont[0], k, 0);

TablaProductos.setValueAt(String.valueOf(codigo),k , 1);
                    TablaProductos.setValueAt(LineaCont[2], k, 2);
                    TablaProductos.setValueAt(LineaCont[3], k, 3);
                    TablaProductos.setValueAt(LineaCont[4], k, 4);
                    TablaProductos.setValueAt(LineaCont[5], k, 5);
                    TablaProductos.removeAll();
                }


                TSalida.setText(String.valueOf(Codigo1));
                TablaProductos.removeAll();//debe de remover todo el
contenido de la tabla
                TablaProductos.remove(k);
             }catch (Exception ex) {}

         }//GEN-LAST:event_CBProductoItemStateChanged

         private                                                  void
CedulaActionPerformed(java.awt.event.ActionEvent     evt)     {//GEN-
FIRST:event_CedulaActionPerformed
             // TODO: Agrege su codigo aqui:

TempTrans.cedula=validacion.validarNumero(Cedula.getText());
            if(TempTrans.cedula==0)
            {
              Cedula.setText("");
            }
            else
            {
             Elaborado.setEnabled(true);
             Aprobado.setEnabled(true);
```

```
            Observaciones.setEnabled(true);
            TempOrden.CedChofer=Integer.parseInt(Cedula.getText());
            TempTrans.cedula=TempOrden.CedChofer;
            TempTrans.Apellido=ApChofer.getText();
            TempTrans.Nombre=NombreChofer.getText();
            TempTrans.placa=Placa.getText();
            TempTrans.modelo=Modelo.getText();
            TempTrans.FechEntr=TempOrden.fech;
        }//GEN-LAST:event_CedulaActionPerformed
        }
        private                                                 void
ModeloActionPerformed(java.awt.event.ActionEvent     evt)     {//GEN-
FIRST:event_ModeloActionPerformed
            // TODO: Agrege su codigo aqui:
           TempTrans.modelo=validacion.Validar(Modelo.getText());
                if(TempTrans.modelo.equals("null"))
                {
                  Modelo.setText("");
                }



        }//GEN-LAST:event_ModeloActionPerformed

        private                                                 void
GuardarActionPerformed(java.awt.event.ActionEvent    evt)    {//GEN-
FIRST:event_GuardarActionPerformed
            // TODO: Agrege su codigo aqui:
            n=TablaProductos.getRowCount();
            String auxF=TFecha.getText();
            String [] Fecha=auxF.split("/");
            TempOrden.NPedido=Integer.parseInt(CodPedido.getText());


            TempOrden.Aprobado=Aprobado.getText();
```

```
            TempOrden.Elaborado=Elaborado.getText();
            TempOrden.des=Destino;
            TempOrden.CodCliente=Codigo;
            // Se guardan los datos del chofer y el vehiculo

            //Se toman los valores de los Productos del DBGrid y se
Guardan en la Tabla productocargado
                TempTrans.Nombre=NombreChofer.getText();
                TempTrans.Apellido=ApChofer.getText();
                TempTrans.Nombre=NombreChofer.getText();
                TempTrans.cedula=Integer.parseInt(Cedula.getText());
                TempTrans.placa=Placa.getText();
                TempTrans.marca=Marca.getText();
                TempTrans.modelo=Modelo.getText();

//TempTrans.FechEntr=Integer.parseInt(Fecha.getText());

            for(int i=0; i<n;i++)
            {
                //
"LINEA","CODIGO","PRODUCTO","PRESENTACION","TEXTURA","PRECIO","CANTI
DAD"


TempProducto.CodProducto=Integer.parseInt((String)(ModeloOrden.getVa
lueAt(i,1)));

TempProducto.Cantidad=Integer.parseInt((String)(ModeloOrden.getValue
At(i,6)));
                TempProducto.NombProducto=(String)
ModeloOrden.getValueAt(i,2);
                TempProducto.Presentacion=(String)
ModeloOrden.getValueAt(i,3);
                TempProducto.Textura=(String)
ModeloOrden.getValueAt(i,4);;
```

```
TempProducto.Precio=Float.parseFloat((String)(ModeloOrden.getValueAt
(i,5)));
                TempProducto.Sacos_Rotos=0;
                //guarda los datos de los productos
               //Cantidad,sacosRotos,NumPedido,Codigo_Producto

enlace.CargarProducto(TempProducto.Cantidad,TempProducto.Sacos_Rotos
,TempOrden.NPedido,TempProducto.CodProducto);
                Band=1;
            }
                //guarda los datos de la orden
                //int NumPedido,String  fecha, String hora, String
Destino, String ElaboradoPor, String AprobadoPor
               if(Band==1)
               {
                Destino=enlace.BuscarDestino(Codigo);

                enlace.AgregarOrden(TempOrden.NPedido,fech,HoraSist,
Destino,TempOrden.Elaborado,
TempOrden.Aprobado,TempTrans.cedula,TempOrden.CodCliente);
                Band=2;
               }
            if(Band==2)
            {
                    String NombChofer=null;

NombChofer=enlace.BuscarChofer(TempTrans.cedula);
                    if(NombChofer==null)// si el chofer no existe lo
guarda en la base de datos
                    {

enlace.AgregarChofer(TempTrans.cedula,TempTrans.Nombre,TempTrans.Ape
llido,TempOrden.NPedido);
                    }
```

```
enlace.AgregarVehiculo(TempTrans.placa,TempTrans.modelo,TempTrans.ma
rca,fech,TempTrans.cedula);
                JOptionPane.showMessageDialog(null,"La   orden   se
guardo exitosamente", "ATENCION",JOptionPane.INFORMATION_MESSAGE);
                // Para generar la panatalla y realizar una nueva
orden
                new nuevaOrden(null).setVisible(true);
                //NOrden= new nuevaOrden(null);
             //NOrden.setVisible(true);
                this.dispose();
            }
            //guarda los datos del chofer y el vehiculo


        }//GEN-LAST:event_GuardarActionPerformed

        private                                                 void
TCantidadActionPerformed(java.awt.event.ActionEvent   evt)   {//GEN-
FIRST:event_TCantidadActionPerformed

             // TODO: Agrege su codigo aqui:
            try {
                TablaProductos.repaint();

                Cant = Integer.parseInt(TCantidad.getText());
                if(TCantidad.getText()==null)
                {
                try
                {
                  TempProducto.AgregarCantidad(Cant);
                }
                catch(Exception ex)
                {
```

```
                    JOptionPane.showMessageDialog(null, "Error ingrese
un    valor    numerico    \n"    +    ex.getMessage(),    "Alerta",
JOptionPane.ERROR_MESSAGE);
                    TCantidad.setText("");
                }
                }
                //contenedor=null;
                //contenedor = enlace.BuscarProducto(Codigo1);

                //Activacion de las cajas de texto de los datos del
Conductor
                NombreChofer.setEnabled(true);
                ApChofer.setEnabled(true);
                Cedula.setEnabled(true);


//"LINEA","CODIGO","PRODUCTO","PRESENTACION","TEXTURA","PRECIO","CAN
TIDAD"
                if(contenido.size()!=0) {
                    try {

                        int k=TablaProductos.getRowCount();
                        int antk=0;
                        if(k>1) {

                            antk=k-1;
                            System.out.println("k en cant:"+k);

                        }
                        System.out.println("antk en cant:"+antk);
                        TablaProductos.removeAll();
                        //TablaProductos.repaint();
                        //String[]              linea             =
{contenedor.getString("Linea"),
String.valueOf(contenedor.getInt("CodProducto")),
```

```
contenedor.getString("NombreProducto"),
contenedor.getString("Presentacion"),
contenedor.getString("Textura"),
String.valueOf(contenedor.getDouble("Precio")),
String.valueOf(Cant)};

                        // ModeloOrden.AgregarLinea(linea);
                        // al DBtable de producto
                        TablaProductos.setDoubleBuffered(true);
                        //TablaProductos.setModel(ModeloOrden);
                        TablaProductos.repaint();


TablaProductos.setValueAt(String.valueOf(Cant),antk,6);

                        //TSalida1.setText(linea[2]);

                        for(int i=0;i<7;i++) {
                       /*    System.out.println(linea[i]);
                           String Salida=linea[i];
                           TSalida.setText(Salida);*/
                        }



                        TablaProductos.setEnabled(true);
                        TablaProductos.setDoubleBuffered(true);
                        TablaProductos.repaint();
                        CBCliente.setEditable(false);
                        CBCliente.setEnabled(false);
                        TablaProductos.repaint();
                        int indice=CBProducto.getSelectedIndex();
                        TSalida.setText( String.valueOf(indice));
                        TablaProductos.repaint();

                        CodPedido.setEditable(false);
```

```
                        TCodigoClient.setEditable(false);
                        TVendedor.setEditable(false);
                        TCantidad.setText("");

                    } catch (Exception ex) {
                        JOptionPane.showMessageDialog(null,   "Error
con la Carga de la tabla Productos\n" + ex.getMessage(), "Alerta",
JOptionPane.ERROR_MESSAGE);

                    }


                } else {
                    JOptionPane.showMessageDialog(null, "Error con
la Carga de Inventario. contenedor no actualiza\n", "Alerta",
JOptionPane.ERROR_MESSAGE);
                }
            } catch (Exception ex) {
                JOptionPane.showMessageDialog(null, "Error ingrese
solo     numeros     \n"     +     ex.getMessage(),     "Alerta",
JOptionPane.ERROR_MESSAGE);
                TCantidad.setText("");
            }


            CBProducto.setSelectedItem(null);
            TablaProductos.setModel(ModeloOrden);
            TablaProductos.setEnabled(true);
            TablaProductos.repaint();
            TablaProductos.repaint();
            TablaProductos.removeAll();
        }//GEN-LAST:event_TCantidadActionPerformed

        private void MarcaActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_MarcaActionPerformed
```

```
            // TODO: Agrege su codigo aqui:
             TempTrans.marca=validacion.Validar(Marca.getText());
                 if(TempTrans.marca.equals("null"))
                 {
                   Marca.setText("");
                 }
                 else
                 {
                  Observaciones.setEnabled(true);
                  Guardar.setEnabled(true);
                 }
    }//GEN-LAST:event_MarcaActionPerformed

        private                                                 void
CerrarSesionActionPerformed(java.awt.event.ActionEvent  evt)  {//GEN-
FIRST:event_CerrarSesionActionPerformed
            // TODO: Agrege su codigo aqui:
        }//GEN-LAST:event_CerrarSesionActionPerformed

        private                                                 void
MCancelarActionPerformed(java.awt.event.ActionEvent   evt)   {//GEN-
FIRST:event_MCancelarActionPerformed
            // TODO: Agrege su codigo aqui
            new nuevaOrden(null).setVisible(true);
            //NOrden= new nuevaOrden(null);
            // NOrden.setVisible(true);
            this.dispose();
           /* TCantidad.setText("0");
            TCodigoClient.setText(" ");
            TVendedor.setText(" ");
            CodPedido.setText(" ");
            TCodigoClient.setEditable(true);
            TVendedor.setEditable(true);
            CodPedido.setEditable(true);
            CBCliente.setEnabled(true);
```

```
            /* try {
                 //CBCliente.setEditable(true);
                 CBCliente.setEnabled(true);
                 CBCliente.removeAll();
                 //CBCliente.setEditable(false);
                  } catch (Exception ex) {
                 JOptionPane.showMessageDialog(null,"Error    en    la
inicalizacion      del      Combobox      Clientes\n"+ex.getMessage(),
"Alerta",JOptionPane.ERROR_MESSAGE);
             }
             try
             {
                 TablaProductos.setEnabled(true);
                 CBProducto.removeAllItems();
             }catch(Exception ex) {
                 JOptionPane.showMessageDialog(null,"Error    en    la
inicalizacion      del      Combobox      Productos\n"+ex.getMessage(),
"Alerta",JOptionPane.ERROR_MESSAGE);
             }*/
             /*try {

                 ModeloOrden.BorrarContenido();
                 TablaProductos.setModel(ModeloOrden);
                 TablaProductos.repaint();
                 TablaProductos.setEnabled(false);



             } catch (Exception ex) {
                 JOptionPane.showMessageDialog(null,"Error    en    el
borrado           de           la           Orden\n"+ex.getMessage(),
"Alerta",JOptionPane.ERROR_MESSAGE);
             }*/
     }//GEN-LAST:event_MCancelarActionPerformed
```

```
        private void SalirActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_SalirActionPerformed
            // TODO: Agrege su codigo aqui
            System.exit(0);
        }//GEN-LAST:event_SalirActionPerformed

        private                                                 void
RegresarActionPerformed(java.awt.event.ActionEvent    evt)    {//GEN-
FIRST:event_RegresarActionPerformed
            // TODO: Agrege su codigo aqui:
            new Opciones(null).setVisible(true);
            this.Cerrar();
            //Opcion.setVisible(true);
            this.dispose();
        }//GEN-LAST:event_RegresarActionPerformed

        private void AyudaActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_AyudaActionPerformed
            // TODO: Agrege su codigo aqui:
        }//GEN-LAST:event_AyudaActionPerformed

        private                                                 void
CBProductoActionPerformed(java.awt.event.ActionEvent   evt)   {//GEN-
FIRST:event_CBProductoActionPerformed
            // TODO add your handling code here:
        }//GEN-LAST:event_CBProductoActionPerformed

        private                                                 void
BRegresarActionPerformed(java.awt.event.ActionEvent    evt)    {//GEN-
FIRST:event_BRegresarActionPerformed
            // TODO add your handling code here:
            new Opciones(null).setVisible(true);
            this.Cerrar();
            this.dispose();
        }//GEN-LAST:event_BRegresarActionPerformed
```

```
        private                                                 void
BCancelarActionPerformed(java.awt.event.ActionEvent    evt)    {//GEN-
FIRST:event_BCancelarActionPerformed
            // TODO add your handling code here:
            new nuevaOrden(null).setVisible(true);
            this.dispose();
        }//GEN-LAST:event_BCancelarActionPerformed

        private                                                 void
NombreChoferActionPerformed(java.awt.event.ActionEvent  evt)  {//GEN-
FIRST:event_NombreChoferActionPerformed
            // TODO add your handling code here:


TempTrans.Nombre=validacion.Validar(NombreChofer.getText());
               if(TempTrans.Nombre.equals("null"))
               {
                 NombreChofer.setText("");
               }
               else
                  ApChofer.setEnabled(true);


        }//GEN-LAST:event_NombreChoferActionPerformed

        private                                                 void
ApChoferActionPerformed(java.awt.event.ActionEvent    evt)    {//GEN-
FIRST:event_ApChoferActionPerformed
            // TODO add your handling code here:

TempTrans.Apellido=validacion.Validar(ApChofer.getText());
               if(TempTrans.Apellido.equals("null"))
               {
                 ApChofer.setText("");
               }
```

```
                else
                   Cedula.setEnabled(true);


        }//GEN-LAST:event_ApChoferActionPerformed

        private                                                 void
ElaboradoActionPerformed(java.awt.event.ActionEvent    evt)    {//GEN-
FIRST:event_ElaboradoActionPerformed
            // TODO add your handling code here:

TempOrden.Elaborado=validacion.Validar(Elaborado.getText());
                if(TempOrden.Elaborado.equals("null"))
                {
                  Elaborado.setText("");
                }



        }//GEN-LAST:event_ElaboradoActionPerformed

        private                                                 void
AprobadoActionPerformed(java.awt.event.ActionEvent    evt)    {//GEN-
FIRST:event_AprobadoActionPerformed
            // TODO add your handling code here:

TempOrden.Aprobado=validacion.Validar(Aprobado.getText());
                if(TempOrden.Aprobado.equals("null"))
                {
                  Aprobado.setText("");
                }

        }//GEN-LAST:event_AprobadoActionPerformed
        private void PlacaActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_PlacaActionPerformed
```

```
            // TODO add your handling code here:

        }//GEN-LAST:event_PlacaActionPerformed

        /**
         * @param args the command line arguments
         */
        public static void main(String args[]) {
            java.awt.EventQueue.invokeLater(new Runnable() {
                public void run() {
                    new nuevaOrden(null).setVisible(true);
                }
            });
        }
        //   Variables   declaration   -   do   not   modify//GEN-
BEGIN:variables
        private javax.swing.JTextField ApChofer;
        private javax.swing.JTextField Aprobado;
        private javax.swing.JMenu Ayuda;
        private javax.swing.JButton BCancelar;
        private javax.swing.JButton BRegresar;
        private javax.swing.JComboBox CBCliente;
        private javax.swing.JComboBox CBProducto;
        private java.awt.Label Cantidad;
        private javax.swing.JTextField Cedula;
        private javax.swing.JMenuItem CerrarSesion;
        private javax.swing.JLabel Cliente;
        private javax.swing.JTextField CodPedido;
        private javax.swing.JTextField Elaborado;
        private javax.swing.JLabel Fecha;
        private javax.swing.JButton Guardar;
        private javax.swing.JMenuItem MCancelar;
        private javax.swing.JTextField Marca;
        private javax.swing.JTextField Modelo;
        private javax.swing.JTextField NombreChofer;
```

```
    private javax.swing.JTextArea Observaciones;
    private javax.swing.JPanel PanelOrden;
    private javax.swing.JMenuItem Pantalla;
    private javax.swing.JTextField Placa;
    private javax.swing.JMenu Poyecto;
    private javax.swing.JMenuItem Regresar;
    private javax.swing.JMenuItem Salir;
    private javax.swing.JMenuItem Sistema;
    private javax.swing.JTextField TCantidad;
    private javax.swing.JTextField TCodigoClient;
    private javax.swing.JTextField TFecha;
    private javax.swing.JTextField TSalida;
    private javax.swing.JTextField TSalida1;
    private javax.swing.JTextField TVendedor;
    private javax.swing.JTable TablaProductos;
    private javax.swing.JLabel Vendedor;
    private javax.swing.JLabel codigoClient;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    // End of variables declaration//GEN-END:variables
}
```

5.2 PRUEBAS

En esta etapa se implementan la prueba de identidad que se reflejan en la tabla siguiente, los resultados de las transacciones erróneas de sistema, para lograr la validación e integridad de los datos de entrada del sistema

Tabla 5.1 manejo de mensajes de error del sistema SDT (1/4)

Tipo de variable	Variable de entrada	Error Arrojado por el Sistema	Explicación
Alfanumérica	Alfanumérica	Error: Clave incorrecta	El usuario del sistema introdujo una clave incorrecta para su sesión.
Entero	Caracteres	Error: ingrese solo números	El operador del sistema introdujo letras en un cuadro de texto donde solo pueden colocar números
Entero		Error: ingrese un valor numérico	El usuario del sistema no introdujo ningún valor en el cuadro y procedió a aceptar la entrada del dato
Carácter	Entero	Error: ingrese solo letras	El operador del sistema introdujo letras en un cuadro de texto donde solo pueden colocar números
Carácter		Error: ingrese un valor alfabético	El usuario del sistema no introdujo ningún valor en el cuadro y procedió a aceptar la entrada del dato
		Error :en el controlador de la operación	La conexión con el MySQL no puede realizarse; el controlador de la conexión se encuentra dañado. Se cancela la operación actual del sistema.

Tabla 5.1 manejo de mensajes de error del sistema SDT (2/4)

Tipo de variable	Variable de entrada	Error Arrojado por el Sistema	Explicación
		Error :Base de datos no encontrada	La base de datos no existe en la estación servidor, debido a que no ha sido creada o se encuentra dañada.. Se cancela la operación actual del sistema
		Error al crear la base de datos	La creación de la base de datos no fue terminada con éxito, se cancela la operación actual del sistema.
		Error al guardar el nuevo producto	Error en la inserción del nuevo producto a la base de datos debido a que el código del producto se encuentra repetido, o los datos son incorrectos
		Error al guardar el nuevo cliente	Error en la inserción del nuevo cliente a la base de datos debido a que el código del cliente se encuentra repetido, o los datos son incorrectos
		Error al guardar la nueva orden	Error en la inserción del nueva orden a la base de datos debido a que el código de la orden se encuentra repetido, o los datos son incorrectos
		Error al modificar la cantidad de los productos	no se puede modificar el producto debido a que no se pueden remplazar los datos actuales
		Error al consultar la tabla clientes	No se logra extraer los datos del código de la orden

			o debido a que es incorrecto o no existe

Tabla 5.1 manejo de mensajes de error del sistema SDT (3/4)

Tipo de variable	Variable de entrada	Error Arrojado por el Sistema	Explicación
		Error al consultar la tabla productos	No se logra extraer los datos del código de la orden o debido a que es incorrecto o no existe
		Error al modificar el inventario	No se puede modificar el inventario debido a que no se pueden remplazar los datos actuales
		Error al modificar la cantidad del producto	No se puede modificar el producto debido a que no se pueden remplazar los datos actuales
		Error al restaurar la base de datos la operación fue cancelada	El archivo no puede ser cargado debido a que su estructura no concuerda con la de la base de datos actual
		Error realizar la impresión la operación fue cancelada	Problema con el dispositivo de impresión
Alfanumérica	Alfanumérica	Error: Clave incorrecta	El usuario del sistema introdujo una clave incorrecta para su sesión.
Entero	Caracteres	Error: ingrese solo números	El operador del sistema introdujo letras en un cuadro de texto donde solo pueden colocar números
Entero		Error: ingrese un valor numérico	El usuario del sistema no introdujo ningún valor en el cuadro y procedió a aceptar la entrada del dato
Carácter	Entero	Error: ingrese solo letras	El operador del sistema introdujo letras en un cuadro de texto donde solo pueden colocar números

Tabla 5.1 manejo de mensajes de error del sistema SDT (3/4)

Tipo de variable	Variable de entrada	Error Arrojado por el Sistema	Explicación
Entero		Error: ingrese un valor numérico	El usuario del sistema no introdujo ningún valor en el cuadro y procedió a aceptar la entrada del dato
Carácter	Entero	Error: ingrese solo letras	El operador del sistema introdujo letras en un cuadro de texto donde solo pueden colocar números
Carácter		Error: ingrese un valor alfabético	El usuario del sistema no introdujo ningún valor en el cuadro y procedió a aceptar la entrada del dato
		Error :en el controlador de la operación	La conexión con el MySQL no puede realizarse; el controlador de la conexión se encuentra dañado. Se cancela la operación actual del sistema.
		Error :Base de datos no encontrada	La base de datos no existe en la estación servidor, debido a que no ha sido creada o se encuentra dañada.. Se cancela la operación actual del sistema
		Error al crear la base de datos	La creación de la base de datos no fue terminada con éxito, se cancela la operación actual del sistema.
		Error al guardar el nuevo producto	Error en la inserción del nuevo producto a la base de datos debido a que el código del producto se encuentra

			repetido, o los datos son incorrectos

Tabla 5.1 manejo de mensajes de error del sistema SDT (4/4)

Tipo de variable	Variable de entrada	Error Arrojado por el Sistema	Explicación
		Error al guardar el nuevo cliente	Error en la inserción del nuevo cliente a la base de datos debido a que el código del cliente se encuentra repetido, o los datos son incorrectos
		Error al guardar la nueva orden	Error en la inserción del nueva orden a la base de datos debido a que el código de la orden se encuentra repetido, o los datos son incorrectos
		Error al modificar la cantidad de los productos	no se puede modificar el producto debido a que no se pueden remplazar los datos actuales
		Error al consultar la tabla clientes	No se logra extraer los datos del código de la orden o debido a que es incorrecto o no existe
		Error al consultar la tabla productos	No se logra extraer los datos del código de la orden o debido a que es incorrecto o no existe
		Error al modificar el inventario	No se puede modificar el inventario debido a que no se pueden remplazar los datos actuales
		Error al modificar la cantidad del producto	No se puede modificar el producto debido a que no se pueden remplazar los datos actuales
		Error al restaurar la base de datos la operación fue cancelada	El archivo no puede ser cargado debido a que su estructura no concuerda con la de la base de datos actual
		Error realizar la	Problema con el dispositivo de

		impresión la operación fue cancelada	impresión

5.2 FASE TRANSICIÓN

En esta fase del ciclo de vida del software es puesto en manos de la comunidad de usuarios, a través de los manuales de usuarios y mantenimientos descritos en los apéndices A y B

CONCLUSIONES

1. Se realizó un análisis detallado de las necesidades del sistema propuesto donde se especificaron todos los procesos requeridos para la elaboración de un reporte final "Orden de Despacho" y así lograr el objetivo inicialmente planteado.

2. Mediante la metodología de proceso unificado a través del modelado de casos de uso, se diseñaron clara y eficientemente los módulos que conforman el sistema, lo que permitió desarrollar una arquitectura sólida del mismo.

3. Se logró definir un modelo eficiente de la base de datos del sistema por medio del modelo entidad relación, la cual permite recopilar adecuadamente toda la información pertinente a los reportes generados por el mismo.

4. Se logro diseñar y desarrollar a través de los diferentes diagramas de UML un software, robusto, amigable y de fácil manipulación por el usuario final; y de esta forma se reduce el margen de error por parte de los usuarios del sistema.

5. La función multiusuario del sistema SDT permite realizar la apropiada identificación del autor de cada reporte del sistema, con lo cual se facilita la auditoria de la data almacenada en el servidor.

6. El manual de usuario del sistema, proporciona el adecuado soporte y una mejor comprensión de su funcionamiento del sistema por parte de los usuarios.

7. La herramienta diseñada es capaz de ofrecer respuestas e información detallada con altísima certidumbre y mayor rapidez que el estudio tradicional realizado manualmente.

8. La certeza de los resultados que arroja la herramienta depende directamente del proceso de depuración de los datos suministrados y de la veracidad de los mismos.

9. Mediante la aplicación de esta herramienta, se logran disminuir los tiempos de respuesta; en cuanto a elaboración de las ordenes de pedido y despacho, en función al control de despacho y de inventario

10. La herramienta permite realizar cómputos de muy aceptable certidumbre cuando es alimentada con datos pre-cargados tales como, los productos y los clientes de la empresa, por lo que este paso es previo a la aplicación de la misma y debe ser realizado por personal con la experticia adecuada.

11. Se definió satisfactoriamente el alcance del sistema logrando con ello la adecuada implementación del mismo.

RECOMENDACIONES

1. Se recomienda realizar un respaldo de la base de datos trimestralmente.

2. Se aconseja un mantenimiento periódico a los equipos usuarios del sistema.

3. Instalar el sistema siguiendo los pasos en el manual de instalación del apéndice A para lograr la mayor eficiencia y efectividad del sistema SDT.

4. Desarrollar un módulo que permita, a partir de la entrada de materia prima de determinar la cantidad de consumo realizado en la producción por producto de entrada y de esta forma integrar el departamento de materia prima al sistema, y de esta forma lograr una mayor automatización de las plantas.

5. Desarrollar un modulo que permita la integración del sistema SDT al sistemas de control de procesos CKF.

BIBLIOGRAFIA

[1] MAGNOMERCADO (2006), Sistemas de Gestión Empresarial, Consultado en febrero 2008, tomado de http://www.magnomercado.com

[2] HERRERA Celia (2004), Consultado en febrero 2008, tomado de http://www.saber.ula.ve.com información

[3] TECNOMAESTROS, Tipos de Sistemas de Información, Consultado en febrero 2008 tomado de http://www.TECNOMAESTROS

[4] PERALTA Manuel, Sistemas transaccionales tecnologías de información sistemas de información sia EDI, Consultada en febrero 2008, tomado de http://www.Sistemas transaccionales

[5] MILLAN L. GARELLI L. "Desarrollo de un Software que permita la automatización de las actividades asociadas al Departamento de Admisión y Control de Estudios de la extensión región centro/sur del Núcleo de Anzoátegui de la Universidad de Oriente". Tesis. (2007).

[6] OTERO J. "Desarrollo de una Aplicación para el Control Administrativo de una Organización Farmacéutica". Tesis. (2004).

[7] ZAVALA R. (2000), Ingeniería de Software. Tesis de Maestría en Ciencias de la Computación. Universidad Autónoma Metropolitana Consultado en febrero, 2008, tomado de www.uam.es.com/Ingeneria de Software

[8] RUMBAUGH James (2000) Proceso Unificado de Desarrollo de Software Mc Graghill Segunda edición

[9] FERRE Xavier (2004), Manual de UML (Univ. Politécnica de Madrid España) Consultado en febrero, 2008, tomado de www.upm.es.com/Manual de Uml

[10] KORTH, Henry (2003), Fundamentos de Bases de Datos, Prentice Hall 3ª edición, Cáp. 1, 2, 3,13, AP A,

[11] KENDALL K. & KENDALL, J. (1997) Análisis y Diseño de Sistemas. de Información Prentice Hall 3ª Edición.

[12] PRESSMAN Roger, (2005) Ingeniería de Software un Enfoque Practico Mc Graghill Tercera edición

[13] SCHMULLER Joseph (2004), Aprendiendo UML en 24 Horas Manual Prentice Hall

ANEXOS

A.1.1 COMO INSTALAR EL PROGRAMA

Para instalar el programa se debe:

- Colocar el CD de instalación en la unidad de CD-ROM y ejecutar el paquete "**SDT**".

- Se debe extraer el programa en el directorio C, para qu el mismo pueda extraer y configurarse correctamente.

- Una vez finalizada la ejecución del instalador se instala primero el programa mysql essential 4.1, con el usuario ***root*** y clave ***12345***. Este programa se encuentran dentro del directorio creado por el instalador.

- Luego se instala el JDK 6, este programa se encuentran dentro del directorio creado por el instalador. La clave por defecto del administrador es **12345**.

- Se ejecuta el programa, y se selecciona el usuario administrador, se coloca la clave de acceso, la cual inicialmente es: ***12345,*** para que el programa cree y configure la Base de Datos del sistema, previamente en vacía.

- Por ultimo, se recomienda cambiar la clave del administrador para mayor seguridad. Para mayor referencia diríjase a el apartado .

Para ejecutar el programa se debe hacer doble click al icono con el nombre **SDT** ubicado en el escritorio del PC ó en el directorio **C:\SDT\sapt.exe**, en caso de

ejecutarlo en el sistema operativo Windows. Para otros sistemas operativos consulte con soporte técnico, o el administrador del sistema.

A.12 INICIANDO EL SISTEMA SDT

Al ejecutar el programa **SDT,** desde el acceso directo del escritorio, o desde el **C:\SDT\sapt.exe,** la primera ventana que se mostrara será la ,presentación del programa la cual posee el siguiente aspecto:

En esta pantalla se realiza click en la fecha y se desplegara una lista con los usuarios del sistema, como lo muestra la siguiente figura

En esta ventana, se puede seleccionar un usuario. Presionando "**enter"**, a haciendo click en el botón **Acepta**r,, y proceder a entrar a su sesión con sus funciones, previamente definidas, o cancelar la selección en el botón **Cancelar**, o por el menú/cancelar

A.1.3 INICIANDO LA SECCION ADMINISTRADOR

Para entrar en la sesión administrador se procede a seleccionar el usuario Administrador de la lista de usuarios, realizando click en la fecha posteriormente ingresar su clave en la casilla adyacente, y luego Presionando "**enter"**, a haciendo click en el botón **Acepta**r, y proceder a entrar a su sesión con sus funciones, previamente definidas, o cancelar la selección en el botón **Cancelar**, o por el menú/cancelar

Opciones y funciones del Usuario Administrador

El usuario administrador tiene el control sobre todo el sistemas, esta sesión posee una amplia gama de opciones las cuales se muestran en la pantalla siguiente, que luego de se despliega luego de entrar en la sesión Administrador

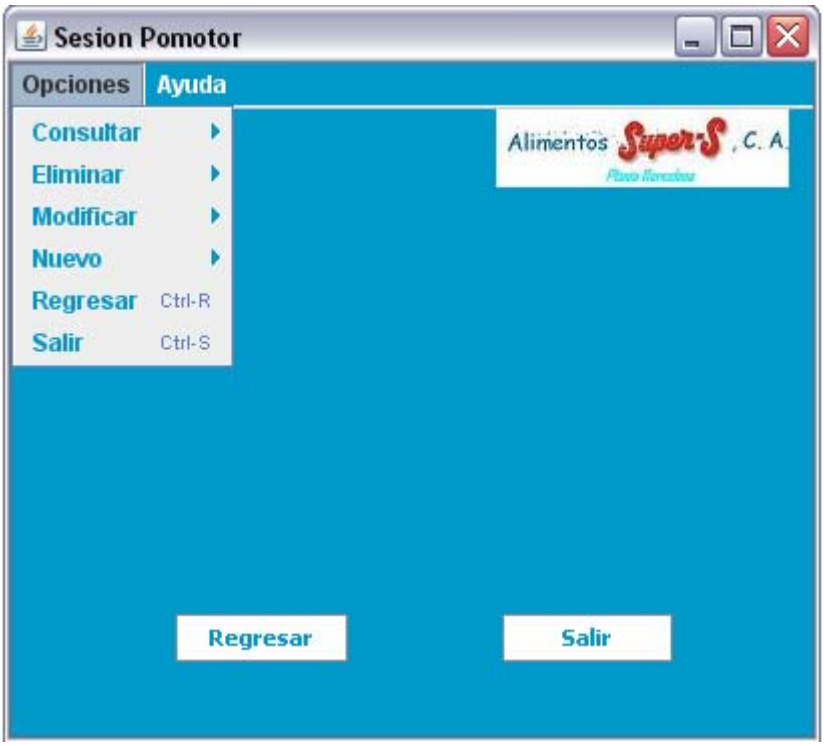
En esta pantalla se puede consultar orden aprobada, en espera, el inventario, despacho, romana, y orden por su código, también se puede ingresar un nuevo producto, cliente y orden de precedido, realizar un nuevo inventario, modificar las ordenes de despacho, los productos y los clientes, imprimir el listado de transporte que serán cargados según la aprobación de la orden, un historial de las ordenes aprobadas dentro de un rango de fechas determinado, y el reporte de las granjas según el producto despachado, respaldar y restaurar la base de datos, así como cambiar las claves de los usuarios, y realizar el enlace entre las estaciones de trabajo y el servidor, así como salir del sistema **SDT**. Todas estas opciones son seleccionadles haciendo click sobre ellas.

A.1.4 INICIANDO LA SECCION PROMOTOR

Para entrar en la sesión Promotor se procede a seleccionar el usuario promotor de la lista de usuarios, realizando click en la fecha, seleccionar el usuario Administrador, posteriormente ingresar su clave en la casilla adyacente, y luego Presionando "**enter"**, a haciendo click en el botón **Aceptar**, y proceder a entrar a su sesión con sus funciones, previamente definidas, o cancelar la selección en el botón **Cancelar**, o por el menú/cancelar

Opciones y funciones del Usuario promotor

El usuario promotor, posee las siguientes funciones dentro del sistema SDT, las cuales se muestran en la siguiente ventana.



En esta ventana el usuario promotor puede consultar, una orden aprobada, o en espera, el inventario actual, una orden a través de su código, ingresar productos , clientes y ordenes de prepedido, modificar productos , clientes y ordenes de precedido, eliminar, productos , clientes y ordenes de precedido, del sistema SDT, regresar a las sesiones a través de el botón Regresar o en el menú/regresar, y salir del sistema mediante del botón salir o a del menú/salir.

A.1.5 INICIANDO LA SECCION DESPACHADOR

Para entrar en la sesión Despachador se procede a seleccionar el usuario despachador de la lista de usuarios, realizando click en la fecha, seleccionar el usuario Despachador, posteriormente ingresar su clave en la casilla adyacente, y luego Presionando "**enter"**, a haciendo click en el botón **Aceptar**, y proceder a entrar a su sesión con sus funciones, previamente definidas, o cancelar la selección en el botón **Cancelar**, o por el menú/cancelar

Opciones y funciones del Usuario Despachador

El usuario despachador, posee las siguientes funciones dentro del sistema SDT, las cuales se muestran en la siguiente ventana.

En esta ventana el usuario despachador puede consultar, el despacho de las ordenes, una orden a través de su código, puede regresar a las sesiones a través de el botón Regresar o en el menú/regresar, y salir del sistema mediante del botón salir o a del menú/salir.

A.1.6 INICIANDO LA SECCION CYC

Para entrar en la sesión CYC (Créditos y Cobranzas) se procede a seleccionar el usuario CyC de la lista de usuarios, realizando click en la fecha, y seleccionar el usuario CyC, posteriormente ingresar su clave en la casilla adyacente, y luego Presionando "**enter"**, a haciendo click en el botón **Aceptar**, y proceder a entrar a su sesión con sus funciones, previamente definidas, o cancelar la selección en el botón **Cancelar**, o por el menú/cancelar

Opciones y funciones del Usuario CYC

El usuario despachador, posee las siguientes funciones dentro del sistema SDT, las cuales se muestran en la siguiente ventana.

En esta ventana el usuario CyC puede consultar, los pedidos en espera así como los pedidos aprobados, realizar la aprobación de las ordenes a través del menú créditos, puede regresar a las sesiones a través de el botón Regresar o en el menú/regresar, y salir del sistema mediante del botón salir o a del menú/salir.

A.1.7 INICIANDO LA SECCION ROMANA

Para entrar en la sesión Romanero se procede a seleccionar el usuario Romanero de la lista de usuarios, realizando click en la fecha, y seleccionar el usuario Romanero, posteriormente ingresar su clave en la casilla adyacente, y luego Presionando "**enter"**, a haciendo click en el botón **Aceptar**, y proceder a entrar a su sesión con sus funciones, previamente definidas, o cancelar la selección en el botón **Cancelar**, o por el menú/cancelar

Opciones y funciones del Usuario CYC

El usuario despachador, posee las siguientes funciones dentro del sistema SDT, las cuales se muestran en la siguiente ventana.

En esta ventana el usuario Romanero puede consultar, los pedidos que serán despachados en el menú Romana, y una orden determinada a través de su código de orden, puede regresar a las sesiones a través de el botón Regresar o en el menú/regresar, y salir del sistema mediante del botón salir o a del menú/salir.

Ingresando un nuevo cliente Al Sistema

Para ingresar un nuevo cliente al sistema se desliza hacia el menú opciones y deslizando el cursor hacia el menú nuevo, realiza un click en el mismo, y se desplaza hasta el submenú cliente y realiza un click.o a través de la siguiente **Ruta : opciones/nuevo/cliente**, lo cual permitirá desplegar la siguiente ventana:

Se procede a escribir en los cuadros de texto los valores adecuados, y correctamente, ya que si se coloca un numero donde debería ir una cadena de letras arrojara un error de inserción de datos, y borrara la casilla, al igual sucede con los

cuadros de texto donde deba de ir un numero y se intente ingresar un cadena de letras, se sugiere si los nombres de los clientes son muy largos abreviarlos, por ejemplo: *agropecuaria* ***por*** *agro*, luego de haber llenado todos los cuadros de texto se activa el botón Aceptar, si presionar botón Regresar, regresara a las opciones del usuario, al presionar el botón Aceptar y se procede a desplegar la siguiente pantalla .

En esta pantalla se confirma la inserción de los nuevos datos, si esta deacuerdo con ellos, se procede a presionar el botón "si" y si no esta deacuerdo el botón no y regresara a la pantalla anterior, permitiéndole corregir los datos, al presionar el botón "si" se le desplegara la siguiente ventana:

Esta pantalla le ofrece la opción de almacenar otro cliente, si presiona el botón "si" le emergerá la ventana de nuevo cliente, y si presiona el botón "no" se desplegara la ventana de opciones del usuario.

Ingresando un nuevo producto Al Sistema

Para ingresar un nuevo producto al sistema se desliza hacia el menú opciones y deslizando el cursor hacia el menú nuevo, realiza un click en el mismo, y se desplaza hasta el submenú producto y realiza un click.o a través de la siguiente **Ruta : opciones/nuevo/producto**, lo cual permitirá desplegar la siguiente ventana:

Se procede a escribir en los cuadros de texto los valores adecuados, y correctamente, ya que si se coloca un numero donde debería ir una cadena de letras arrojara un error de inserción de datos, y borrara la casilla, al igual sucede con los cuadros de texto donde deba de ir un numero y se intente ingresar un cadena de letras, se sugiere si los nombres de los productos son muy largos abreviarlos, por ejemplo: *pollo crecimiento enerpro* ***por*** *pollo cre enerpro*, luego de haber llenado todos los cuadros de texto se activa el botón Aceptar, si presionar botón Regresar, regresara a

las opciones del usuario, al presionar el botón Aceptar y se procede a desplegar la siguiente pantalla .

En esta pantalla se confirma la inserción de los nuevos datos, si esta deacuerdo con ellos, se procede a presionar el botón “si” y si no esta deacuerdo el botón no y regresara a la pantalla anterior, permitiéndole corregir los datos, al presionar el botón “si” se le desplegara la siguiente ventana:

Esta pantalla le ofrece la opción de almacenar producto, si presiona el botón “si” le emergerá la ventana de nuevo producto, y si presiona el botón “no” se desplegara la ventana de opciones del usuario.

Ingresando un nueva orden Al Sistema

Para ingresar una nueva orden al sistema, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el menú nuevo, realiza un click en el mismo, y se desplaza hasta el submenú orden y realiza un click.o a través de la siguiente **Ruta : opciones/nuevo/orden prepedido**, lo cual permitirá desplegar la siguiente ventana:

En esta ventana se realiza la inserción de una nueva orden de la siguiente manera:

1.- Seleccionar un determinado cliente, el cual fue previamente cargado a través de la opción: nuevo/cliente, y para seleccionarlo dirige el cursor hacia la pestaña ubicada en el cuadro de texto adyacente a el nombre cliente, realiza un click y se desplegara la lista de los clientes es el sistema SDT, luego se selecciona un determinado cliente, haciendo click sobre el nombre, esto permitirá visualizar los datos asociados al cliente.

2- Se procede a ingresar el código de la orden, el cual debe ir en el cuadro de texto que se encuentra en la parte inferior al nombre código pedido, y luego presione “enter”,

3- Luego se procede a insertar los productos, esto se hace dirigiendo el cursor hacia la pestaña adyacente del nombre: productos, y realizando click sobre la pestaña, lo cual desplegara una lista de los productos cargados en el sistema SDT, se selecciona un determinado producto haciendo click sobre el nombre, esto le permitirá escribirlo en la tabla inferior.

4- Se procede a ingresar las cantidades de los productos en el cuadro de texto que se encuentra debajo de nombre cantidad, se sugiere ingresar solo números debido a que si ingresa letras se arrojara un error de inserción de datos y borrar la casilla, luego de ingresar el numero presione “enter”, y la cantidad aparecerá en la tabla inferior al lado del producto seleccionado

5- Se procede a ingresar los datos del chofer, del vehiculo que transportara la carga y del personal encargado de elaborar la orden ya probarla, después de llenar los cuadros de texto correctamente, se procede a presionar el botón Guardar,

6- Se presiona el botón Cancelar de cancela la orden, o a través del menú/cancelar. Si presiona el botón Regresar regresara a las opciones del Usuario o a través del menú/regresar, se puede salir del sistema SDT mediante el menú/salir.

Aprobando una orden en el Sistema

Para realizar la aprobación de una orden en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el menú creditos, realiza un click en el mismo,o a través de la siguiente **Ruta : opciones/cretidos**, lo cual permitirá desplegar la siguiente ventana:

En esta ventana se realiza la aprobación de la siguiente manera:

1. Se desliza el Mouse hasta el menú programa se realiza un click en el submenú pedidos en espera, hace clic en el mismo, si se procederá a llenar la tabla inferior con los datos de las ordenes por aprobar existentes en el sistema.

2.-se procede a hacer click, sobre una orden determinada dentro de la tabla, y posteriormente en la parte inferior aparecerá, el nombre del cliente, y el precio total de la orden, y se activara la opción "Aprobación".

3.-Luego se selecciona en el botón de opción al lado de la palabra opción y la orden procede a ser aprobada.

Esta ventana también permite, visualizar las ordenes que han sido aprobadas, a través de la ruta menú/ordenes aprobadas

Despachando una orden en el Sistema

Para realizar el despacho de una orden en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el menú consulta realiza un click en el mismo, luego de dirige al submenú despacho, o a través de la siguiente **Ruta: opciones/consulta/despacho**, lo cual permitirá desplegar la siguiente ventana:

1.- Se procede a dirigir el cursor al cuadro de texto al lado de la oración cedula chofer, y se presiona “enter”.

2.-luego se procede a ingresar la placa del vehiculo en el cuadro de texto al lado de la palabra placa y se presiona “enter”.

3.-luego se selecciona de la tabla una orden determinada, y se presiona el botón despacho., ya la orden a sido despachada.

En esta pantalla se puede regresar a la pantalla opciones del usuario presionando el botón regresar o a través de la ruta: menú programa/regresar o a traves de crtl + R, también se puede salir a través de la ruta: menú programa/salir o a traves del crtl + S.

También se puede limpiar los cuadros de texto y la tabla a través de la ruta: menú programa/limpiar

Pesando una orden en el Sistema

Para realizar el pesada de una orden en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el menú consulta realiza un click en el mismo, luego de dirige al submenú Romana, o a través de la siguiente **Ruta: opciones/consulta/Romana**, lo cual permitirá desplegar la siguiente ventana :

1.- Se procede a dirigir el cursor al cuadro de texto al lado de la oración cedula chofer, y se presiona “enter”.

2.-luego se procede a ingresar la placa del vehiculo en el cuadro de texto al lado de la palabra placa y se presiona “enter”.

3.-luego se selecciona de la tabla una orden determinada, y se presiona el botón Pesado., ya la orden a sido despachada.

4.-luego se procede a presionar el botón imprimir y saldrá la orden aprobada, despachada y pesada.

En esta pantalla se puede regresar a la pantalla opciones del usuario presionando el botón regresar o a través de la ruta: menú programa/regresar o a través de crtl + R, también se puede salir a través de la ruta: menú programa/salir o a través del crtl + S.

También se puede limpiar los cuadros de texto y la tabla a través de la ruta: menú programa/limpiar

Realizando un inventario en el Sistema

Para realizar un inventario en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el menú inventario realiza un click en el mismo, o a través de la siguiente **Ruta: opciones/inventario**, lo cual permitirá desplegar la siguiente ventana:

1.- Se dirige el cursor hacia e menú programa, se realiza click, sobre el mismo, dirigirse al submenu, nuevo, y hacer click en el mismo, esto permitirá llenar la tabla en la parte inferior.

2.-Se escoge un producto determinado, presionando una fila determinada en la tabla, esto permitirá Mostar el producto en la parte superior.

3.-se procede a insertar las cantidades del producto, en sus respectivos cuadros de texto, siento el área 09 , el área de productos buenos y a probados para la venta, el área 10, los productos para reproceso, y la rampa , los productos en los vehículos que no han sido, despachados de la planta.

4.- se procede a insertar el número de ticket para ese producto, y el carril en donde se encuentra el mismo en el almacén.

5.- Luego se dirige hacia el botón actualizar, y se actualizaran los datos del producto.

6.- al terminar de actualizar los productos se procede a presionar el botón imprimir, el cual arrojara un reporte con la existencia de los productos en el almacén.

Esta pantalla también permite mostrar un inventario determinado, a través de su código de inventario, a través de la ruta programa/ mostrar.

También se puede regresar a la pantalla opciones del usuario presionando el botón regresar o a través de la ruta: menú programa/regresar o a través de crtl + R, también se puede salir a través de la ruta: menú programa/salir o a través del crtl + S.

Realizando un respaldo del Sistema

Para realizar un respaldo en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el menú **Base de Datos** realiza un click en el mismo, o a través de la siguiente **Ruta: opciones/Base de Datos**, lo cual permitirá desplegar la siguiente ventana:

1.- Se dirige el cursor hacia e menú ejecutar, se realiza click, sobre el mismo, dirigirse al submenu, Respaldos, y hacer click en el mismo, esto permitirá desplegar la siguiente ventana:

2.- se procede a respaldar una tabla específica realizando click, en cuadro de selección, o se puede respaldar toda la base de datos realizando un click en el cuadro de selección Respaldo Completo.

3.-Se procede a presionar el botón Respaldar.

También se puede regresar a la pantalla anterior (Base de Datos) del usuario presionando el botón regresar o a través de la ruta: menú programa/regresar o a través de crtl + R, también se puede salir a través de la ruta: menú programa/salir o a través del crtl + S.

Realizando un respaldo del Sistema

Para realizar una restauración de la base de datos, en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el menú **Base de Datos** realiza un click en el mismo, o a través de la siguiente **Ruta: opciones/Base de Datos**, lo cual permitirá desplegar la siguiente ventana:

Se sugiere que antes de respaldar el o los archivos a restaurar estén en un directorio de primer nivel es decir C:/ , F: / , etc.

1.- Se dirige el cursor hacia e menú ejecutar, se realiza click, sobre el mismo, dirigirse al submenu, Respaldos, y hacer click en el mismo, esto permitirá desplegar la siguiente ventana:

2.- se procede a restaurar una tabla específica realizando click, en cuadro de selección, esto desplegara un pantalla de búsqueda, como la siguiente:

4.- en esta pantalla se busca un archivo con extensión txt, el cual debe de poseer el nombre de la tabla seguido de la palabra Backup, por ejemplo: backups_cliente, se selecciona, el archivo.

3.-Se procede a presionar el botón Restaurar.

También se puede regresar a la pantalla anterior (Base de Datos) del usuario presionando el botón regresar o a través de la ruta: menú programa/regresar o a través de crtl + R, también se puede salir a través de la ruta: menú programa/salir o a través del crtl + S.

Enlace con el servidor del Sistema

Para realizar un enlace con la base de datos, en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el item **Enlace** realiza un click en el mismo, o a través de la siguiente **Ruta: opciones/Enlace**, lo cual permitirá desplegar la siguiente ventana de confirmación de usuario:

1.-se coloca el usuario Administrador, y se procede a colocar la clave en el cuadro de texto inferior, y luego se presiona “enter” o el botón aceptar, esto desplegara la siguiente ventana:

2.- en esta pantalla se coloca la dirección ip del equipo que será el servidor del sistema, y el puerto por el cual se realizara el envió de paquetes del sistema, y se procede a presionar el botón conectar. Se sugiere darle permiso a la aplicación a través del muro de fuego o desactivarlo.

3.- si se desea cancelar se presiona el botón Regresar, el cual regresara a la pantalla opciones del usuario.

Cambiando claves a los usuarios del Sistema

Para realizar un cambio de claves e un usuario del sistema SDT , se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el item realiza un click en el mismo, o a través de la siguiente **Ruta: opciones/Cambiar Claves**, lo cual permitirá desplegar la siguiente ventana :

1.- se coloca el usuario a cambiar la clave.

2.- se procede a ingresar la clave anterior, en el cuadro de texto al lado de la oración calve anterior.

3.- Se ingresa la nueva clave en el cuadro de textual lado de la oración Nueva Clave, y se procede a confirmar la nueva clave en el cuadro de texto inferior al lado de la oración Confirmar Clave.

4.- se procede a presionar el botón Aceptar, si se desea cancelar a operación .se procede a presionar el botón Cancelar o en el menú/Cancelar

También se puede regresar a la pantalla anterior del usuario presionando el botón regresar o a través de la ruta: menú programa/regresar o a través de crtl + R, también se puede salir a través de la ruta: menú programa/salir o a través del crtl + S.

REALIZANDO LOS REPORTES DE TRANSPORTE

Para realizar un Reporte de transporte en sistema SDT , se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el submenú Imprimir un click en el mismo, y posteriormente se desliza el cursor hasta el ítem Listado de transporte a través de la siguiente **Ruta: opciones/Imprimir/Listado de Transporte**, lo cual permitirá imprimir el listado de trasporte de los vehículos con ordenes aprobadas:

RAELIZANDO LOS REPORTES DE GRANJAS

Para realizar un Reporte de Granjas en sistema SDT , se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el submenú Imprimir un click en el mismo, y posteriormente se desliza el cursor hasta el ítem Reporte de Granjas a través de la siguiente **Ruta: opciones/Imprimir/Reporte de Granjas**, lo cual permitirá imprimir el listado de las granjas a través de los productos despachados.

RAELIZANDO EL HISTORIAL

Para realizar un Reporte de transporte en sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el submenú Imprimir un click en el mismo, y posteriormente se desliza el cursor hasta el ítem Histórico de Ordenes a través de la siguiente **Ruta: opciones/Imprimir/Historico de Ordenes**, lo cual permitirá desplegar la siguiente ventana:

1.- Se procede a ingresar la fecha desde donde se desea realizar el historial, ingresando un numero correspondiente al dia , y presionando “enter”, luego se coloca el numero correspondiente a el mes y se presiona “enter”, luego se coloca el numero correspondiente a el año completo y se presiona “enter”.

2.- **.-** Se procede a ingresar la fecha hasta donde se desea realizar el historial, ingresando un numero correspondiente al dia , y presionando “enter”, luego se coloca el numero correspondiente a el mes y se presiona “enter”, luego se coloca el numero correspondiente a el año completo y se presiona “enter”.

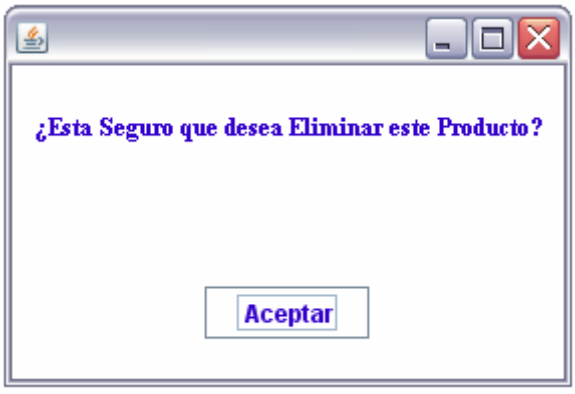
3.-Luego se procede a presionar el boton Buscar, y esto imprimirá el histórico respectivo.

En esta pantalla se puede regresar a la pantalla opciones del usuario presionando el botón regresar o a través de la ruta: menú programa/regresar o a través de crtl + R, también se puede salir a través de la ruta: menú programa/salir o a través del crtl + S.

ELIMINANDO UN PRODCUTO DEL SISTEMA

Para eliminar un producto del sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el submenú Eliminar realiza un click en el mismo, y se desliza el cursor hasta el ítem Producto o a través de la siguiente **Ruta: opciones/Eliminar/Producto**, lo cual permitirá desplegar la siguiente ventana :

1.- Se procede a ingresar el código del producto a eliminar, y se presiona el botón eliminar, lo cual desplegara la siguiente ventana. :

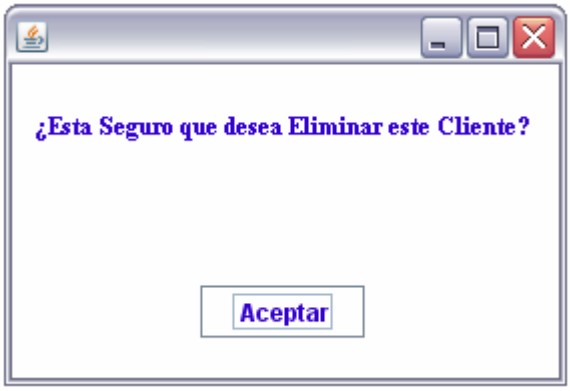


2.- Se procede a eliminar la orden presionando el botón aceptar, y se retornara a la ventana opciones el usuario.

ELIMINANDO UN PRODCUTO DEL SISTEMA

Para eliminar un cliente del sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el submenú Eliminar realiza un click en el mismo, y se desliza el cursor hasta el ítem Cliente o a través de la siguiente **Ruta: opciones/Eliminar/Cliente**, lo cual permitirá desplegar la siguiente ventana:

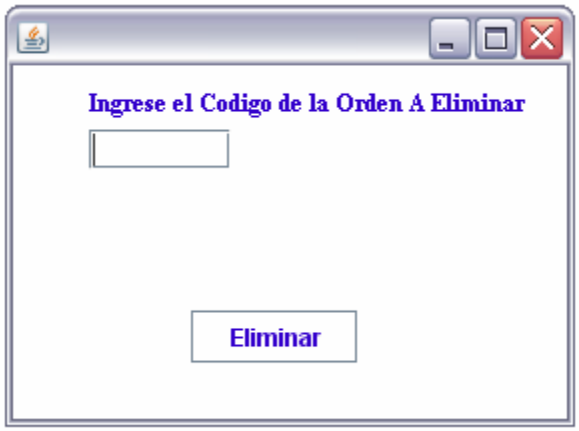
1.- Se procede a ingresar el código del producto a eliminar, y se presiona el botón eliminar, lo cual desplegara la siguiente ventana. :



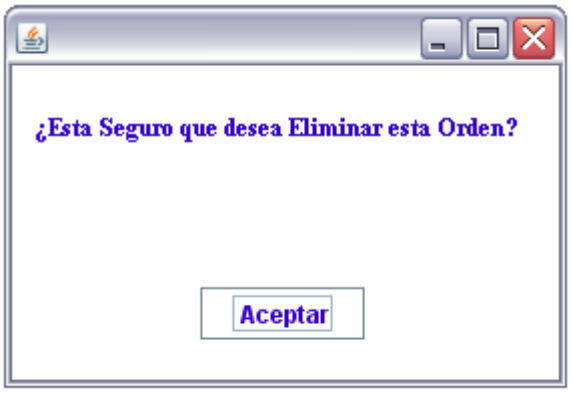
2.- Se procede a eliminar la orden presionando el botón aceptar, y se retornara a la ventana opciones el usuario.

ELIMINANDO UNA ORDEN DEL SISTEMA

Para eliminar una orden del sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el submenú Eliminar realiza un click en el mismo, y se desliza el cursor hasta el ítem Orden Prepedido o a través de la siguiente **Ruta: opciones/Eliminar/Orden Prepedido**, lo cual permitirá desplegar la siguiente ventana:



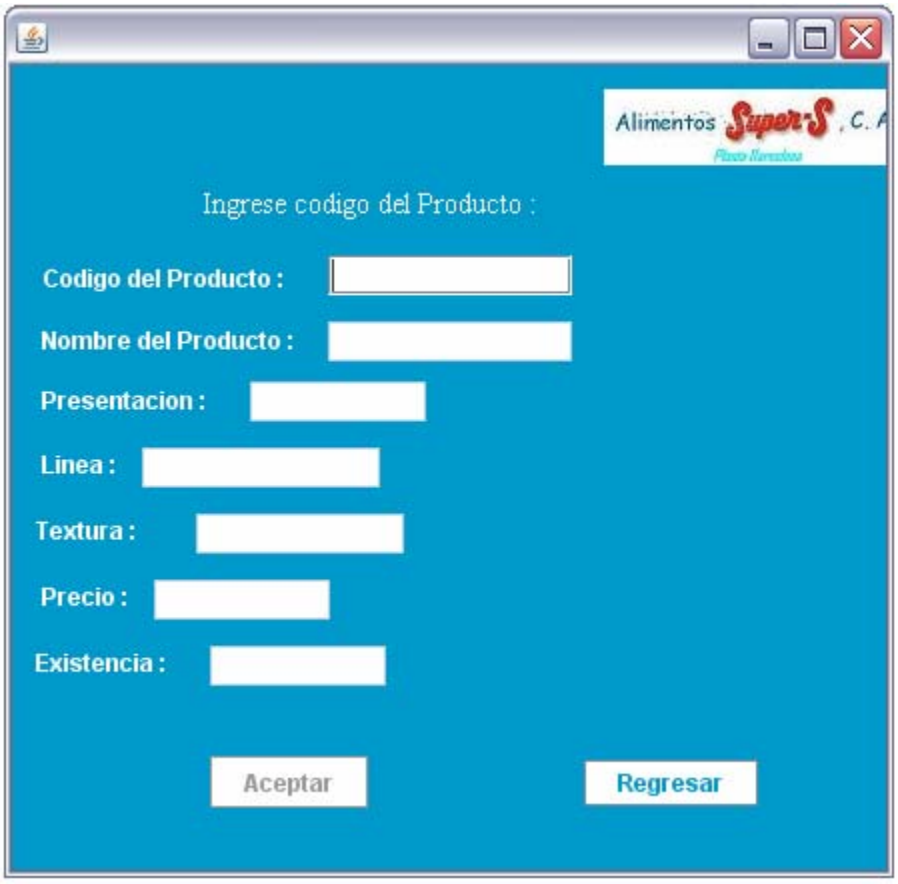
1.- Se procede a ingresar el código del producto a eliminar, y se presiona el botón eliminar, lo cual desplegara la siguiente ventana. :



2.- Se procede a eliminar la orden presionando el botón aceptar, y se retornara a la ventana opciones el usuario.

MODIFICANDO UN PRODUCTO DEL SISTEMA

Para modificar un producto en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el submenú Modifiar realiza un click en el mismo, y se desliza el cursor hasta el ítem Producto o a través de la siguiente **Ruta: opciones/Modificar/Producto**, lo cual permitirá desplegar la siguiente ventana:

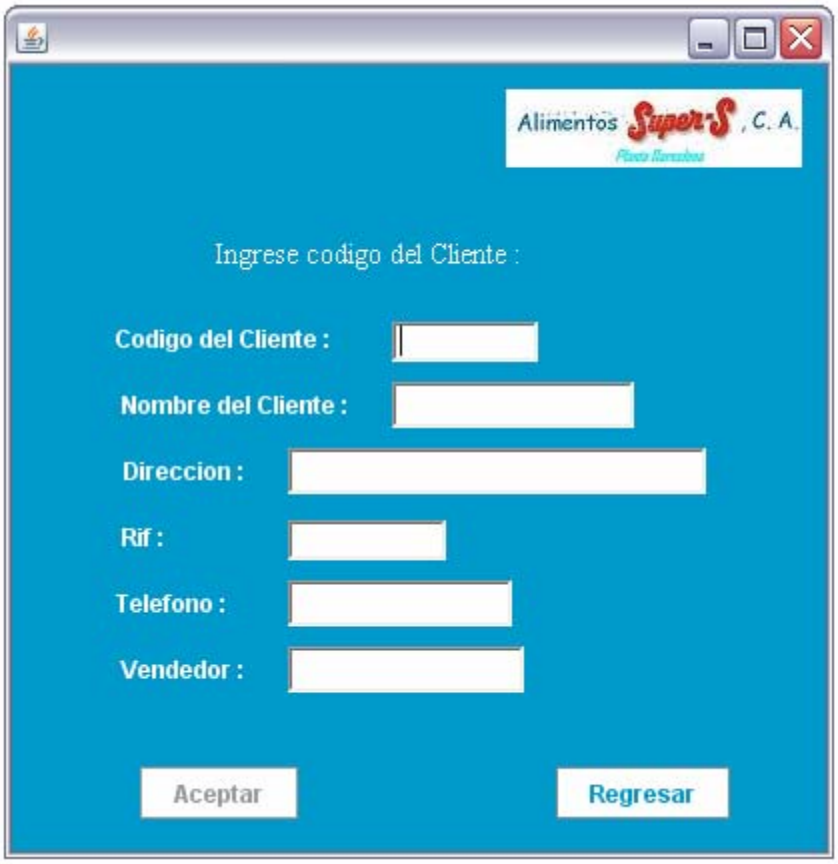


1.- Se ingresa el número del producto a modificar y presiones “enter”.

2.- después de que todos los datos del producto fueron cargados, se procede a escribir, o modificar los valores necesarios y se presiona “enter” o el botón aceptar.

MODIFICANDO UN CLIENTE DEL SISTEMA

Para modificar un cliente en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el submenú Modifiar realiza un click en el mismo, y se desliza el cursor hasta el ítem Cliente o a través de la siguiente **Ruta: opciones/Modificar/Cliente**, lo cual permitirá desplegar la siguiente ventana:



1.- Se ingresa el número del cliente a modificar y presiones “enter”.

2.- después de que todos los datos del cliente fueron cargados, se procede a escribir, o modificar los valores necesarios y se presiona “enter” o el botón aceptar.

MODIFICANDO UNA ORDEN DEL SISTEMA

Para modificar una orden en el sistema SDT, se desliza hacia el menú opciones, posteriormente deslizando el cursor hacia el submenú Modificar realiza un click en el mismo, y se desliza el cursor hasta el ítem Orden de Prepedido o a través de la siguiente **Ruta: opciones/Modificar/Orden de Prepedido**, lo cual permitirá desplegar la siguiente ventana:

1 

.- Se ingresa el número de la orden a modificar y presiones “enter”.

2.- después de que todos los datos del producto fueron cargados, se procede a escribir, o modificar los valores del chofer, los productos así, como sus cantidades, quien elaboro o aprobó la orden.

3. para eliminar un producto de la orden se selecciona la fila de la tabla de los productos, y se dirige el cursor hasta el botón de selección eliminar producto.

4.-Luego de que todos los datos están correctos se presiona el botón guardar.

En esta pantalla se puede regresar a la pantalla opciones del usuario presionando el botón regresar o a través de la ruta: menú programa/regresar o a través de crtl + R, también se puede salir a través de la ruta: menú programa/salir o a través del crtl + S.