UNIVERSIDAD AUTÓNOMA DE ICA

FACULTAD DE INGENIERÍA, CIENCIAS Y ADMINISTRACIÓN

TESIS:

"SISTEMA COMPUTARIZADO PARA LA ADMINISTRACION DEL DEPARTAMENTO COMERCIAL DE LA LIBRERÍA Y DISTRIBUIDORA SANCHEZ SRL CHINCHA – 2015"

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO DE SISTEMAS

PRESENTADA POR:

BACHILLER. ROBERTO CARLOS SALAS NAPA

BACHILLER. DAVIS BRYAM MONSERRATE MEDINA

ASESORA:

DRA. MAGDALENA TALLA LINDERMAN

CHINCHA - ICA – PERÚ

SISTEMA COMPUTARIZADO PARA LA ADMINISTRACIÓN DEL DEPARTAMENTO COMERCIAL DE LA LIBRERÍA Y DISTRIBUIDORA SANCHEZ SRL CHINCHA – 2015

Roberto Carlos Salas Napa

AUTOR

Davis Bryam Monserrate Medina

AUTOR

Dra. Magdalena Talla Linderman

ASESORA

Presentada a la Facultad de Ciencias, Ingeniería y administración de la Universidad Privada Autonoma de Ica. Para optar el Titulo de Ingeniería de Sistemas

APROBADO POR:

PRESIDENTE DEL JURADO

SECRETARIO DEL JURADO

VOCAL DEL JURADO

DICIEMBRE - 2015

DEDICATORIA

A Dios por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos, además de su infinita bondad y amor

A mis padres por haberme apoyado en cada momento, por sus consejos, sus valores, por la motivación constante que me ha permitido ser una persona de bien, pero más que nada, por su amor.

A mi novia karem por su amor, sus consejos, apoyo y comprensión en todo momento.

EPIGRAFE

“Nunca debe el hombre lamentarse de los tiempos en que vive, pues esto no le servirá de nada. En cambio, en su poder está siempre mejorarlos.”

“De nada sirve al hombre lamentarse de los tiempos en que vive. Lo único bueno que puede hacer es intentar mejorarlos.”

“Un cuerpo sano es cosa buena; pero un alma sana vale más que todo lo que el hombre pueda desear; un alma sana es lo más hermoso que el cielo pueda concedernos para hacer feliz esta pobre tierra nuestra.”

Thomas Carlyle

“El sabio no se sienta para lamentarse, sino que se pone alegremente a su tarea de reparar el daño hecho.”

“Nuestras dudas son traidores que muchas veces nos hacen perder el bien que podríamos ganar si no temiéramos buscarlo.”

“El hombre cauto jamás deplora el mal presente; emplea el presente en prevenir las aflicciones futuras.”

William Shakespeare

AGRADECIMIENTOS

Para dar Inicio expresamos nuestros más sincero y profundo agradecimiento a Dios nuestro Señor, por habernos colmado de bendiciones y guiado en el camino para lograr nuestros objetivos a lo largo de nuestra formación profesional.

También agradezcemos profundamente a todos los docentes, que durante los cinco años de estudio nos brindaron y orientaron sus sabias enseñanzas relacionadas con los avances Tecnológicos de cada día; damos las gracias por que aprendimos a analizar, desarrollar y brindar soluciones empresariales la cual nos ayudó mucho a desarrollar este proyecto.

A la empresa Multipapel Sanchez S.R.L. por brindarnos sus instalaciones para el desarrollo del sistema, ya que sin su colaboración no hubiera sido posible la realización de la presente Tesis titulada "**SISTEMA COMPUTARIZADO PARA EL ADMINISTRACIÓN DEL DEPARTAMENTO COMERCIAL DE LA LIBRERÍA – DISTRIBUIDORA SANCHEZ S.R.L. CHINCHA - 2015**".

Por último, agradezcemos a nuestros padres y a todas aquellas personas, que de manera directa o indirecta, ayudaron a que sea posible la culminación de este proyecto.

RESUMEN

Hoy en día la eficiente gestión comercial representa un reto en las empresas o instituciones en donde es gestionado un gran volumen de información, y que en la mayoría de los casos el registro se hace de manera manual y una vez archivada la información representa cierto grado de dificultad extraer información en torno a un documento en específico en relación a la gestión que se le ha dado.

El desarrollo de esta tesis titulada **"SISTEMA COMPUTARIZADO PARA LA ADMINISTRACIÓN DEL DEPARTAMENTO COMERCIAL DE LA LIBRERIA DISTRIBUIDORA SANCHEZ S.R.L.CHINCHA - 2015",** Tiene como objetivo principal establecer una propuesta para darle una mejor eficiencia y rapidez a la gestión de los productos que se ingresan y salen, a su vez controlar el stock de forma automatizada, de manera que se Acelere el proceso de atención hacia los ciudadanos con la ayuda del sistema, Facilitando Reportes de todos los ingresos y salidas de la empresa **LIBRERIA DISTRIBUIDORA SANCHEZ S.R.L**.

En la actualidad nos encontramos inmersos al avance tecnológico, es por ello que esta tesis comprende en desarrollar un sistema de comunicación utilizando herramientas que nos brinda hoy en día dichos avances, para así mejorar la eficiencia de la empresa que hoy en día se encuentra involucrado en esta problemática.

Con la elaboración de este software el encargado de la empresa seria beneficiado porque tendría toda la información más personalizada y al instante, con este proyecto se ahorraría tiempo a la vez para los ciudadanos brindándole un servicio de calidad a un tiempo estimado.

ABSTRACT

Today the efficient business management is a challenge in enterprises or institutions where it is handled a large volume of information, which in most cases registration is done manually and once archived information represents some degree of difficulty extracting information about a specific document relating to the management which has given him.

The development of this thesis entitled "**COMPUTER SYSTEM FOR BUSINESS ADMINISTRATION DEPARTMENT OF LIBRERIA DISTRIBUIDORA SANCHEZ S.R.L.CHINCHA - 2015"** Its main objective is to establish a proposal to give greater efficiency and speed management products are input and exit in turn control the stock automatically, so that the care process is accelerated towards the citizens with the help of the system, facilitating reports all income and company **LIBRERIA DISTRIBUIDORA SANCHEZ S.R.L.**

Today we find ourselves to technological progress, which is why this thesis consists in developing a communication system using tools provided to us today such advances, in order to improve the efficiency of the company today is involved in this problem.

With the development of this software in charge of the company it would have benefited because all the more personalized and instant information, this project will save time both for citizens by providing a quality service to an estimated time.

INDICE

INTRODUCCION

El rol que desempeña la tecnología en el mundo de hoy es una gran contribución para la sociedad y el hombre. Dentro del desarrollo de la tecnología los sistemas gestores de base de datos son una herramienta útil para el trabajo desempeñado lo que a permitirá fortalecer la seguridad de datos de la organización y permitir que la información sea transparente y de mejor calidad.

El ser humano ha aprendido a utilizar y aprovechar la tecnología en su beneficio en las diferentes actividades cotidianas tanto industriales, comercio.

Uno de los usos más vanguardista se le ha asignado a los avances tecnológicos el cual sirve de apoyo para la toma de decisiones y la gestión de información de la empresa

Dado la importancia que está teniendo la tecnología en el desarrollo como en la evolución de los negocios.

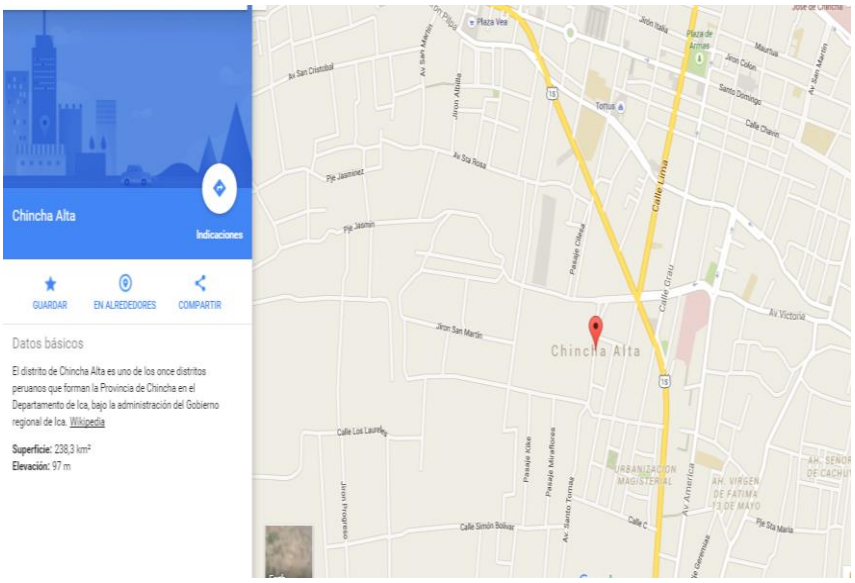
El presente proyecto plantea “implementación de un sistema Computarizado en el departamento comercial para la **LIBRERIA DISTRIBUIDORA SANCHEZ S.R.L.CHINCHA -2015”**

ASPECTO INFORMATIVO

➢ UBICACIÓN

Geográfica

El distrito de Chincha Alta es uno de los once distritos peruanos que forman la Provincia de Chincha en el Departamento de Ica, bajo la administración del Gobierno regional de Ica.



Limites

El Distrito de Chincha Alta limita:

Por el Norte:

Con el distrito de Sunampe y Grocio Prado.

Por el Este:

Con el distrito de Chincha Baja y El Carmen.

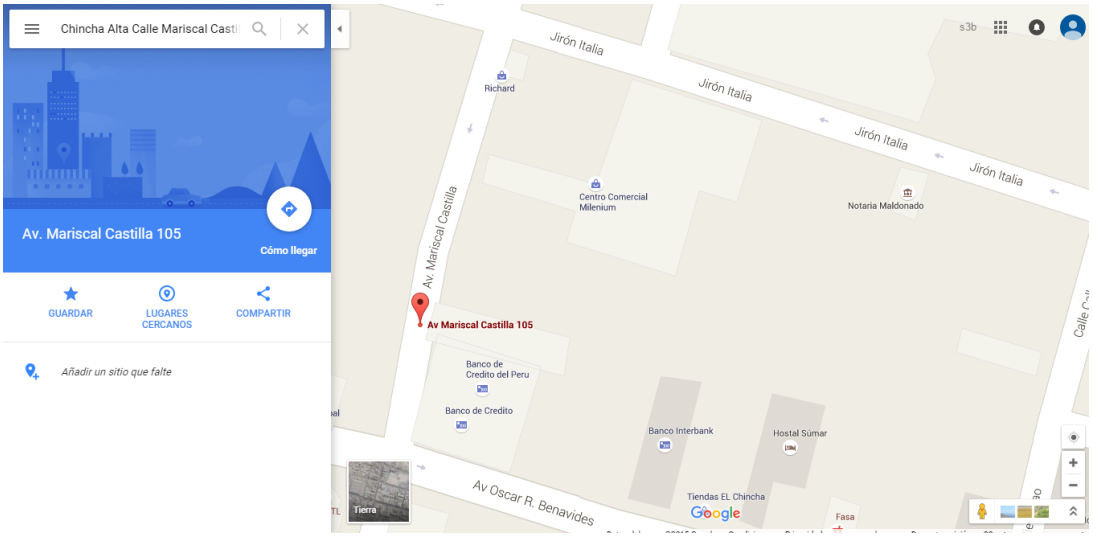
Por el Oeste:

Con el Océano Pacífico.

Extensión

Tiene una superficie aproximada de 209,45 km².

Plano de Ubicación



➢ DATOS GENERALES DEL DISTRITO

Referencias históricas

Chincha Alta es la capital de la provincia de Chincha. Uno de los principales atrativos de este distrito es la iglesia de Santo Domingo, iglesia mayor.

Dentro del perímetro de la ciudad, se encuentran los más importantes centros comerciales, los centros de abastos más grandes e importantes de la ciudad, las sedes de las instituciones públicas y privadas más represetantivas del gobierno central, regional y local, como la Prefectura, el Gobierno Regional Ica y la Municipalidad Provincial de Chincha, así como la Corte Superior y otras. Los terminales de transportes con buses hacia todas las ciudades del país.

➢ DATOS GENERALES

- **Nombre de Empresa** **:** LIBRERIA DISTRIBUIDORA SANCHEZ S.R.L.
- **RUC** **:** 20410333806
- **Fecha de Fundación** **:** 29/02/2000
- **Tipo de Sociedad** **:** Sociedad Comercial de Responsabilidad Limitada
- **Estado de la Empresa** **:** Activo
- **Actividad Económica** **:** Prestacion de Servicios a la Comunidad
- **Dirección Principal** **:** Calle Mariscal castilla 105
- **Población** **:** Ica / Chincha Alta
- **Teléfono** **:** 267957
- **Nro. Trabajadores** **:** 20

VISIÓN DE LA EMPRESA

Exceder las expectativas de calidad y servicio del consumidor de bienes escolares y de oficina.

MISIÓN DE LA EMPRESA

Ser la empresa más exitosa en la venta de artículos escolares y de oficina y que nuestras marcas sean las más reconocidas a nivel nacional.

ACTIVIDAD PRINCIPAL DE LA EMPRESA

La LIBRERIA DISTRIBUIDORA SANCHEZ S.R.L. brinda servicio de comercialización de productos escolares a la comunidad de la provincia de Chincha.

Actividades:

- Venta al por mayor.
- Venta al por menor de juegos y juguetes en comercios especializados.
- Venta a por mayor de otros enseres domesticos.

DEFINICION DE TERMINOS Y CONCEPTOS

VENTA

El volumen de facturación dependerá del tipo de librería especializada o general y, sobre todo, de su ubicación. Los negocios situados en enclaves turísticos pueden tener una mayor actividad durante el verano y la Semana Santa, facturando en estos momentos entre el 60 y el 70% de su facturación anual. En cuanto a los ciclos generales con mayores ventas, según Michèle Chevallier, “existen varias épocas importantes en el año para las librerías. La primera coincide con el inicio del curso escolar que, tradicionalmente, era la época fuerte de venta de útiles escolares, libros, etc. Otro periodo importante corresponde a las fechas de Navidades y Reyes y a la feria del libro de cada ciudad, si la hay”

COMPRAS

Al iniciar la actividad, lo más habitual es que los distribuidores exijan un pedido en firme a abonar a 30, 60 o 90 días. Además y dependiendo del poder de negociación, de la experiencia, de los contactos, la ubicación... se puede conseguir un depósito de libros o útiles escolares. Es decir, los proveedores prestan los ejemplares y el librero los paga según se van vendiendo. Normalmente, esta forma de trabajar es habitual en las librerías más consolidadas, pero en nuestro supuesto asumimos que el emprendedor hace un pedido en firme de 3000 a 8000 nuevos soles en depósito.

EXISTENCIAS

La inversión para constituir el stock inicial de una librería depende de la superficie del local, la ubicación, el tipo de productos que se van a comercializar, etc. Además hay que tener en cuenta que el fondo de una librería especializada es mucho más caro que el de una general. Otro aspecto a analizar es la frecuencia de la rotación del stock, ya que ésta condicionará el volumen de ventas.

Un buen control de existencias es imprescindible para el adecuado funcionamiento del negocio. En una empresa pequeña el inventario se puede realizar una vez al año, pero es conveniente realizarlo con mayor frecuencia, ya que permite contrastar los datos obtenidos por el programa informático y conocer la necesidad de incrementar las compras en determinados ejemplares o de devolver otros si hay un exceso de stock.

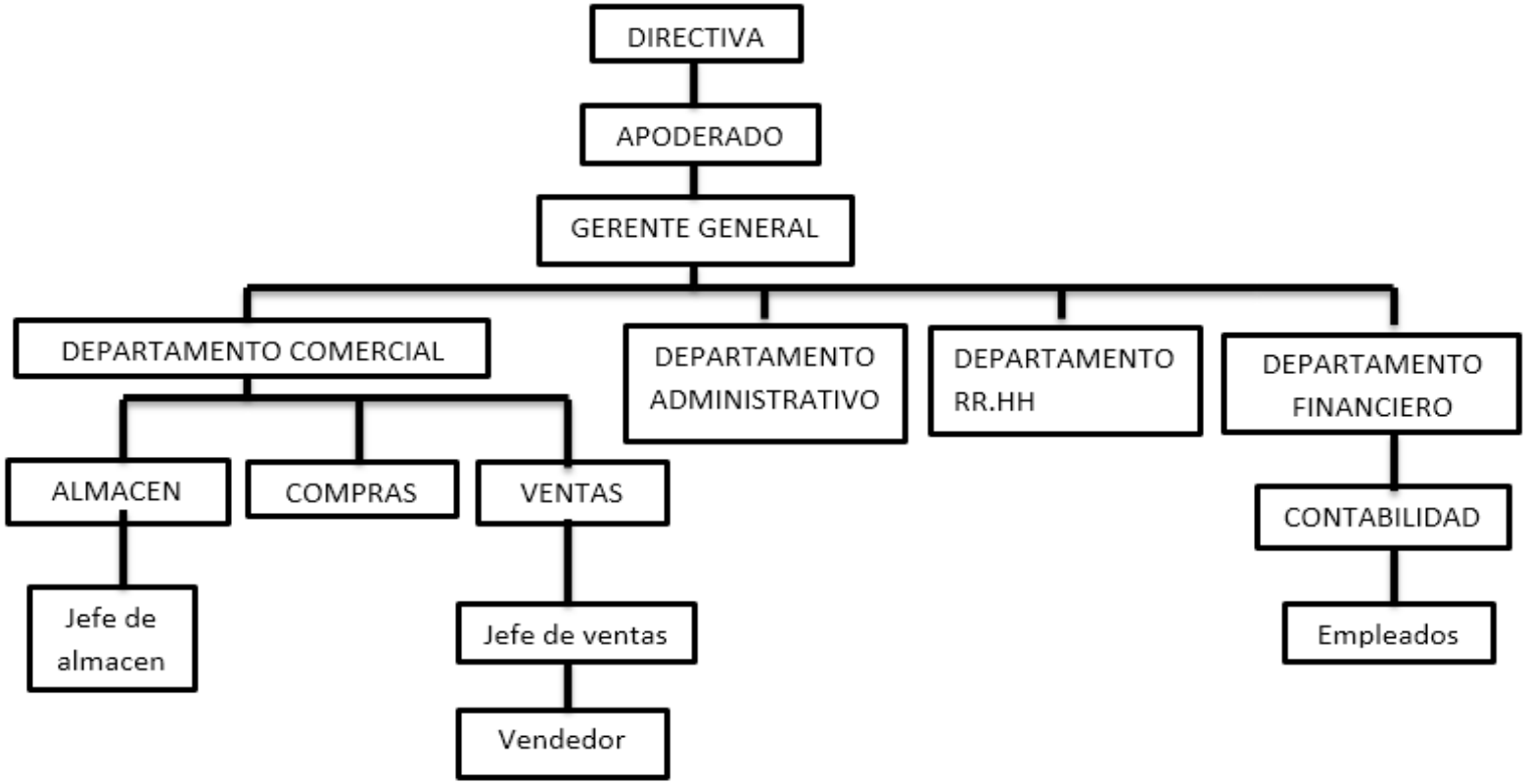
PLANIFICACION

Es la función que mediante el diagnostico de la realidad orienta la toma de decisiones conducentes al logro de las otras funciones y de la misión. Esta función es realizada por el dueño en coordinación con el Gerente General de la empresa.

LA ADMINISTRACION

Es una función que se sustenta, en otras, en forma de actividades especificas que sirven de apoyo a todo el proceso de la organización. Su órgano central es la dirección de Administracion, pero se manifiesta en todas las unidades jefes, Gerentes y Administrativos.

- **ORGANIGRAMAS Y JERARQUÍAS**



CAPITULO I: PLANEAMIENTO DE PROBLEMA

I PLANTEMIENTO DEL PROBLEMA

1.1. SITUACIÓN PROBLEMATICA

La librería y distribuidora Sánchez presta sus servicios a la sociedad el cual tiene como objetivo satisfacer las necesidades del cliente en base al desarrollo de la calidad. La empresa Sánchez tiene como visión Ser la empresa más exitosa en la venta de artículos escolares y de oficina. Actualmente se encuentra ubicada en la provincia de Chincha en la Calle Mariscal castilla 105.

En base a las actividades desarrolladas en la Librería y distribuidora Sánchez se han identificados dificultades en el departamento comercial el cual presentan un alto nivel de informalidad en cuanto a sus procesos, la dificultad que tiene el responsable de cada área dentro del departamento comercial en la empresa al momento de requerir información, la falta de precisión en cuanto a los ingresos y egresos que no permiten visualizar las evoluciones económicas, Otro factor es la perdida de información debido a que no existe el almacenamiento adecuado al momento de archivarlos.

Por ende en el presente proyecto se plantea el desarrollo un sistema computarizado para agilizar los procesos en el departamento comercial, tomando en cuenta el comportamiento de las áreas involucradas, las características, entre otros factores de importancia.

1.2. FORMULACIÓN DEL PROBLEMA

¿Desarrollar un sistema computarizado en el departamento comercial para agilizar los procesos de la librería distribuidora Sánchez?

1.3. JUSTIFICACIÓN E IMPORTANCIA

Producto de las problemáticas que presentan en la librería distribuidora Sánchez se planea desarrollar un sistema computarizado en el departamento comercial en el cual podrán agilizar sus procesos, obtener de manera eficaz y eficiente la disposición de la información a los responsables de dicho departamento, monitorear las actividades realizadas, a la vez disminuir las demoras en la atención al cliente, obtener la evolución económica.

CAPITULO II: MARCO TEORICO

II MARCO TEORICO

2.1 ANTECEDENTES

INTERNACIONAL

Autores

José Altaír Villavicencio

Lugar

Colombia

Año

Enero - 2014

Titulo

Software para el mejoramiento de los procesos de la librería y papelería latina

Objetivos

El objetivo de este proyecto es proporcionar la información necesaria para un mejor método de trabajo de los operarios y recepción de la mercancía utilizando las herramientas de ingeniería.

Conclusiones

Este sistema evita el congestionamiento al optimizar la distribución y mejorar las actividades realizadas por el operario. Así mismo, para obtener un mejor servicio y de alta calidad. Otros de los propósito de este proyecto es hallar un mejor método de trabajo a los operarios, determinando así el tiempo estándar de una actividad que se realice en la librería por medio de la observación y el cronometraje para así tener un tiempo exacto en este caso el área de atención al cliente.

Autores

Patricio Andrés Lerou Godas

Lugar

chile

Año

2007

Titulo

Sistema para control de inventario, venta y generación de datos comerciales de restaurante

Objetivos

Desarrollar un sistema para el control de inventario y generación de datos para el restaurante ok corral el cual contribuya al mejoramiento de sus actividades.

Conclusiones

Una vez finalizado el trabajo se puede concluir que el objetivo general, la creación de un sistema para control de inventario, venta y generación de datos comerciales se cumplió a cabalidad mediante el cumplimiento de los objetivos específicos planeados en un comienzo.

Mediante las metodologías realizadas a cada uno de los elementos involucrados se pudo verificar el éxito y aceptación de las pruebas individuales y globales, tanto para la base de datos como para la aplicación.

NACIONAL

Autor

- María Luisa Muñoz
- Alexandra Fuentes Arévalo

Lugar

Lima - Perú

Año

2012

Titulo

Software para la librería y papelería LiLi c.a

Objetivos

El objetivo es proporcionar la información necesaria para establecer un mejor método de trabajo y distribución de los productos

Conclusión

La importancia de la resolución de este problema radica en que una mejor distribución de los materiales trae como resultado la fluidez a la hora de brindar un servicio, además de crear un buen ambiente de trabajo

Autores:

- Martin morales berrenchea
- Karim moreno la rosa

Lugar

Lima - Perú

Año

2004

Titulo

Sistema de Gestión de Almacén de Productos Terminados

Objetivos

El objetivo es proporcionar un sistema que automatice los procesos de gestión del almacén acorde a los productos terminados.

Conclusión

En el desarrollo del proyecto se han conocido las diversas gestiones y las buenas prácticas aplicadas en la administración de los almacenes terminados. Así como también las diversas herramientas tecnológicas de aplicación logística como las balanzas digitales y los terminales portátiles de datos. La combinación de estos conocimientos y tecnologías permiten obtener resultados favorables para la organización.

REGIONAL

Autores

- Juana Gana Quiroz
- Marcelo Gerlach Velásquez

Lugar

Ica

Año 2013

Titulo

Desarrollo de un software comercial de la librería Dimeiggs s.a. y estudio de posible plan de expansión

Objetivos General

Tener, en el mediano plazo, un sistema de información integrado, oportuno y confiable, de interacción interna y externa, de acuerdo a las disponibilidades de recursos.

Objetivos Específicos

El objetivo de este trabajo es desarrollar un software comercial y un plan estratégico que identificara los factores necesarios para seleccionar las localizaciones y realizar una evaluación estratégica y económica de un local de Dimeiggs como una unidad de negocio independiente.

Conclusión:

Es el mejoramiento en las venta producto del software en conjunto del plan de marketing, introducir nuevos productos, entregar servicios complementarios y abrir nuevos locales que le otorguen ingresos hoy inexistentes. Para ello se determinará un estándar aplicable a todos los locales nuevos que decidirá la empresa abrir en el futuro.

2.2 BASES TEORICAS

ÁREAS INVOLUCRADAS EN EL NEGOCIO

- **Gerencia**

El Área de Gerencia es el área de Dirección responsable de la gestión municipal, encargado de planear, organizar, dirigir y supervisar las actividades administrativas de servicios, que tiene bajo su responsabilidad y como complemento a las funciones administrativas y ejecutivas de la alcaldía. Está a cargo de un funcionario de confianza- a tiempo completo, designado por el alcalde, de quien depende administrativa, funcional y jerárquicamente; y es de libre designación y remoción, conforme ley.

Funciones

La gestión organizacional es un proceso complejo, que consiste en la ejecución secuencial de pasos o fases, que constituyen el trabajo típico de un administrador o gerente. Esas tareas pueden expresarse en forma generalizada como funciones directivas, que se ejercen a diversos niveles:

Planificación

Consiste en establecer los objetivos de largo, mediano y corto plazo (o metas) de la organización, y en especificar los cursos de acción que se seguirán para conseguirlos. Debe haber coherencia entre los diversos niveles temporales de objetivos, los cuales por otra parte debe ser concretos, claros, y de ser posible, cuantificables, para poder luego hacer comparaciones con los resultados. Incluye también el análisis de los recursos necesarios, su adecuación y disponibilidad; y todo ello se debe concretar finalmente en planes, programas y presupuestos.

Dirección

Consiste en orientar (“dirigir”) los esfuerzos de todos los empleados de la organización, inclusive los directivos, hacia la obtención de las finalidades organizativas. La función directiva se relaciona con los objetivos permanentes, de largo plazo, de la organización; y con los cambios constantes del contexto con el que la organización está vitalmente relacionada. La función de dirección se ocupa también de la selección del personal que desempeñará los cargos diseñados, de su integración al

conjunto de la empresa, de la orientación de su trabajo, capacitación y motivación, estableciendo el sistema de liderazgo que resulte más adecuado, así como el esquema de sus remuneraciones y promociones, vale decir, de todo lo relacionado con la gestión de los llamados “recursos humanos”.

Control

Consiste en procurar que todo se haga según las previsiones, asegurando la obtención de los objetivos de la organización, mediante la comparación de los resultados reales con los resultados esperados, para definir el nivel de ajuste o de divergencia entre ambos, y emprender las acciones correctivas que reencaucen la situación. La función de control está, pues, estrechamente vinculada con la función de planificación. No se pueden controlar resultados sin previsiones previas, y no se pueden establecer nuevas metas sin controlar los resultados anteriores.

Estas funciones son secuenciales, se realizan periódicamente, en momentos significativos de la vida de la empresa. En el resto del tiempo, los directivos realizan las llamadas funciones continuas:

Análisis de problemas: En toda organización, constantemente se están produciendo problemas, incidentes y dificultades. Hay que detectarlos, analizarlos, buscar sus causas, establecer su importancia y prioridad, para buscar su solución e implementarla.

Toma de decisiones: Frente a los problemas u oportunidades que plantea el entorno, hay que plantear las diversas alternativas de cursos de acción posibles, valorarlas según diversos criterios, sopesar opiniones y consejos, y en definitiva elegir una, tomar la decisión y finalmente llevarla a cabo.

El proceso de toma de decisiones es considerado un aspecto central de la función directiva, y en una visión más metódica y sistematizada se lo considera integrado por las siguientes fases: Identificación del problema - Desarrollo de las alternativas Identificación de los criterios para decidir Ponderación de los criterios para decidir Evaluación de las alternativas - Selección de una alternativa -Implantación de la alternativa -Evaluación de la decisión.

Comunicación: Los directivos, para analizar problemas y tomar decisiones, necesitan mucha información proveniente de otros niveles de la organización, que pueden obtener si tienen adecuados canales de comunicación. Lo mismo ocurre cuando deben informar sobre las decisiones que toman, para crear bases de consenso y de encuadre disciplinario a los fines de su puesta en práctica.

❖ **Área de Almacén**

El Área de Almacén, está encargada del Resguardo de los materiales y suministros de la Municipalidad.

Funciones

1) Velar y coordinar por el cumplimiento del trabajo encomendado a la oficina.
2) Control de entradas y salidas del personal por medio de un libro.
3) Inspeccionar proyectos para velar por el buen uso de los materiales entregados.
4) Controlar que toda salida de materiales sea registrada por medio de una solicitud con las firmas correspondientes.
5) Revisar y firmar facturas de materiales entregados a la comunidad.
6) Planificar la entrega de materiales a los diversos proyectos en ejecución.
7) Mantener un stock variado de materiales indispensable.

- **Área de Compras**

Planificar, coordinar, y supervisar la ejecución del desempeño de todas aquellas tareas que se lleven acabo dentro del departamento de compras de la Librería Sanchez, así como también efectuar contactos con proveedores a nivel nacional e internacional.

Funciones:

1.) Realizar los diferentes planes referentes al departamento de compras en coordinación con el Gerente Comercial.

2.) Coordinar las actividades dentro del área de compras.

3.) Revisar y aprobar los diferentes contratos de compras.

4.) Preparar y revisar reuniones con los diferentes proveedores de la empresa.

5.) Establecer las metas del departamento de compras y verificar que estos se cumplan.

6.) Convocar a reuniones de trabajo a todo el personal del área de compras semanalmente.

7.) Trabajar en Coordinación con el Gerente General y el Jefe de Almacen con relación a los niveles de inventario.

- **Área de Ventas:**

El departamento de ventas es planear, ejecutar y dar seguimiento y control continuo a las actividades de ventas y existencias de Productos.

Funciones:

1.) Establecer metas y objetivos.

2.) Calcular la demanda.

3.) Pronosticar las ventas.

4.) Planear, organizar y dirigir las funciones del departamento.

5.) Determinar el tamaño y la estructura de la fuerza de ventas.

6.) Reclutamiento, selección y capacitación de los vendedores.

7.) Definir los estándares de desempeño.

RUP- RATIONAL UNIFIED PROCESS O PROCESO UNIFICADO DE RATIONAL

Es un proceso de ingeniería de software que suministra un enfoque para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta y de mayor calidad para satisfacer las necesidades de los usuarios que tienen un cumplimiento al final dentro de un limite de tiempo y presupuesto previsible. Es una metodología de desarrollo iterativo que es enfocada hacia " diagramas de los casos de uso, y manejo de los riesgos y el manejo de la arquitectura" como tal.

El RUP mejora la productividad del equipo ya que permite que cada miembro del grupo sin importar su responsabilidad específica pueda acceder a la misma base de datos incluyendo sus conocimientos. Esto hace que todos compartan el mismo lenguaje, la misma visión y el mismo proceso acerca de cómo desarrollar un software.

UML

Lenguaje Unificado de Modelado (LUM o UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

Es importante remarcar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

Se puede aplicar en el desarrollo de software gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional o RUP), pero no especifica en sí mismo qué metodología o proceso usar.

UML no puede compararse con la programación estructurada, pues UML significa Lenguaje Unificado de Modelado, no es programación, solo se diagrama la realidad de una utilización en un requerimiento. Mientras que, programación estructurada, es una forma de programar como lo es la orientación a objetos, sin embargo, la programación orientada a objetos viene siendo un complemento perfecto de UML, pero no por eso se toma UML sólo para lenguajes orientados a objetos.

UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.

- **Diagrama de Casos de Uso**

El Caso de Uso, es una técnica para capturar información de cómo un sistema o negocio trabaja, o de cómo se desea que trabaje. No pertenece estrictamente al enfoque orientado a objetos, es una técnica cuyo objetivo es capturar requisitos.

Actores

Un actor es una agrupación uniforme de personas, sistemas o máquinas que interactúan con el sistema que estamos construyendo de la misma forma. Por ejemplo, para una empresa que recibe pedidos en forma telefónica, todos los operadores que reciban pedidos y los ingresen en un sistema de ventas, si pueden hacer las mismas cosas con el sistema, son considerados un único actor.

Es importante tener clara la diferencia entre usuario y actor. Un actor es una clase de rol, mientras que un usuario es una persona que, cuando usa el sistema, asume un rol. De esta forma, un usuario puede acceder al sistema como distintos actores. La forma más simple de entender esto es pensar en perfiles de usuario de un sistema operativo. Una misma persona puede acceder al sistema con distintos perfiles, que le permiten hacer cosas distintas. Los perfiles son en este caso equivalentes a los actores.

Entre los actores tenemos:

Principales: Personas que no usan el sistema.

Secundarios: Personas que mantienen o administran el sistema.

Material Externo: Dispositivos, materiales imprescindibles que forman parte del ámbito de la aplicación y deben ser utilizados.

Otros Sistemas: Sistemas con los que el sistema interactúa.

La misma persona física puede interpretar varios papeles como actores distintos, el nombre del actor describe el papel desempeñado.

Relaciones de Casos de Uso

Las tres relaciones principales entre los casos de uso son soportadas por el estándar UML, el cual describe notación gráfica para esas relaciones. Veamos una revisión de ellas a continuación:

- **Inclusión (include o use)**

  Es una forma de interacción o creación, un caso de uso dado puede "incluir" otro caso de uso. El primer caso de uso a menudo depende del resultado del caso de uso incluido. Esto es útil para extraer comportamientos verdaderamente comunes desde múltiples casos de uso a una descripción individual, desde el caso de uso. El estándar de Lenguaje de Modelado Unificado de OMG define una notación gráfica para realizar diagramas de casos de uso, pero no el formato para describir casos de uso. Mucha gente sufre la equivocación pensando que un caso de uso es una notación gráfica (o es su descripción). Mientras la notación gráfica y las descripciones esto no sirve.

- **Extensión (Extend)**

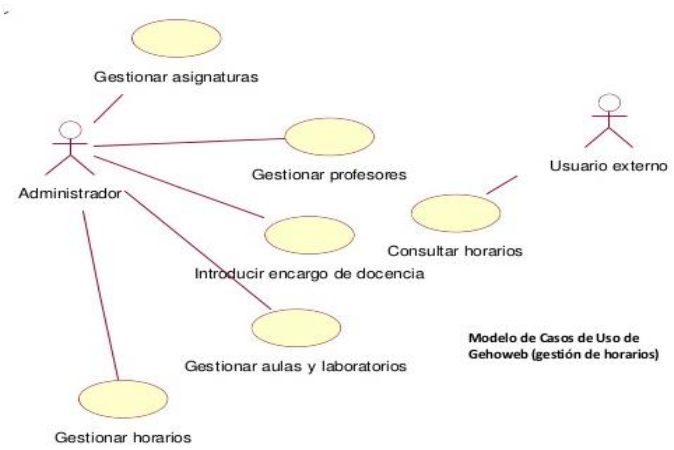
  Es otra forma de interacción, un caso de uso dado (la extensión) puede extender a otro. Esta relación indica que el comportamiento del caso de la extensión se utiliza en casos de uso, un caso de uso a otro caso siempre debe tener extensión o inclusión. El caso de uso extensión puede ser insertado en el caso de uso extendido bajo ciertas condiciones. La notación, es una flecha de punta abierta con línea discontinua, desde el caso de uso extensión al caso de uso extendido, con la etiqueta «extend». Esto puede ser útil para lidiar con casos especiales, o para acomodar nuevos requisitos durante el mantenimiento del sistema y su extensión.

  "La extensión, es el conjunto de objetos a los que se aplica un concepto. Los objetos de la extensión son los ejemplos o instancias de los conceptos.", documentan el comportamiento de un sistema desde el punto de vista de un usuario.

- **Generalización**

  La Generalización es la actividad de identificar elementos en común entre conceptos y definir las relaciones de una superclase (concepto general) y subclase (concepto especializado). Es una manera de construir clasificaciones taxonómicas entre conceptos que entonces se representan en jerarquías de clases. Las subclases conceptuales son conformes con las superclases conceptuales en cuanto a la intención y extensión."

  En la tercera forma de relaciones entre casos de uso, existe una relación generalización/especialización. Un caso de uso dado puede estar en una forma especializada de un caso de uso existente. La notación es una línea sólida terminada en un triángulo dibujado desde el caso de uso especializado al caso de uso general. Esto se asemeja al concepto orientado a objetos de sub-clases, en la práctica puede ser útil factorizar comportamientos comunes, restricciones al caso de uso general, describirlos una vez, y enfrentarse a los detalles excepcionales en los casos de uso especializados.



- **Diagrama de Clases**

Es el diagrama principal para el análisis y diseño. Un diagrama de Clases presenta las clases del sistema con sus relaciones estructurales y de herencia. La definición de clases incluye definiciones para atributos y operaciones. El modelo de Caso de Uso aporta información para establecer las clases, objetos, atributos y operaciones. El mundo real puede ser visto desde abstracciones diferentes (subjetividad).

Cada clase se representa en un rectángulo con tres comportamientos:

- Nombre de la Clase.
- Atributos de la Clase.
- Operaciones de la Clase.

Los atributos de una clase no deberían ser manipuladas directamente por el resto de objetos.

Por esta razón se crearon niveles de visibilidad para los elementos que son:

(-) Privado: Es el más fuerte, esta parte es totalmente invisible (excepto para clases friends en terminología C++)

(#) Protected: Los atributos/operaciones protegidos están visibles para las clases friends y para las clases derivadas de la original.

(+) Public: Los atributos/operaciones públicos son visibles a otras clases (cuando se trata de atributos se está transgrediendo el principio de encapsulación).

Relaciones entre Clases:

Son los enlaces entre objetos que se podrán representar entre las respectivas clases y sus formas de relación son:

Asociación y Agregación (vista como un caso particular de Asociación).

Asociación: La Asociación expresa una conexión bidireccional entre objetos. Una asociación es una abstracción de la relación existente en los enlaces entre los objetos. Puede determinarse por la especificación de multiplicidad (mínima y máxima).

- Uno y sólo uno.
- 0…1 Cero a uno.
- M…N Desde M hasta N (enteros naturales).
- 0…* Cero a Muchos.
- 1….* Uno o Muchos.

Agregación: En UML se proporciona una cascada, caracterización de la agregación. Esta relación puede ser caracterizada con precisión determinando las relaciones de comportamiento y estructura que existen entre el objeto agregado y cada uno de sus objetos componentes.

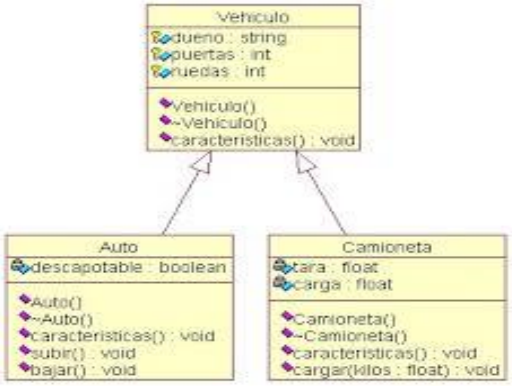
Ventajas y Desventajas del Diagrama de Clases

Ventajas

- Genera un código automáticamente.
- Propone soluciones a algunos errores.
- Representa las relaciones entre las clases de sistema.
- Se diseña los componentes de los sistemas.
- Se protegen los datos.
- Se posibilita una reducción de acoplamiento.
- Más fácil la comunicación entre los programadores, descubrimiento de fallas del sistema en el diseño Mejor diseño del sistema ofrece más documentación.

Desventajas

- Los diagramas de clases especifican qué clases hay y cómo están relacionadas, pero no cómo interactúan para alcanzar comportamientos particulares.
- El método tiende hacer muy lento.
- La instalación es muy costosa.



- **Diagrama de Secuencia**

El Diagrama de Secuencia representa la forma en cómo un Cliente (Actor) u Objetos (clases) se comunican entre sí en petición a un evento.

Esto implica recorrer toda la secuencia de llamadas, de donde se obtienen las responsabilidades claramente.

Dicho Diagrama puede ser obtenido de dos partes, desde el Diagrama Estático de Clases o el Caso de Uso (son diferentes).

Los Componentes de un Diagrama de Secuencia son:

- Un Objeto o Actor.
- Mensaje de un Objeto a otro Objeto.
- Mensaje de un objeto a sí mismo.

Elementos del Diagrama de Secuencia

- **Objeto/Actor,** el rectángulo representa una instancia de un Objeto en particular, y la línea punteada representa las llamadas a métodos del objeto.
- **Mensaje a otro Objeto,** se presenta por una flecha entre un objeto y otro, representa la llamada de un método (operación) de un objeto en particular.
- **Mensaje al Mismo Objeto, n**o solo llamadas a métodos de objetos externos pueden realizarse, también es posible visualizar llamadas a métodos desde el mismo objeto de estudio.

Características del Diagrama de Secuencia

- Los diagramas de secuencia muestran gráficamente las interacciones del actor y de las operaciones a quedan origen.
- Los diagramas de secuencia se preparan durante la fase de análisis de un ciclo de desarrollo.
- Su creación depende de la formulación previa de los casos de uso.
- El comportamiento del sistema es una descripción delo que hace, y no como lo hace.
- El diagrama de secuencia muestra un determinado escenario de un caso de uso, los eventos generados por actores externos, su orden y los eventos internos del sistema.
- A todos los sistemas se les trata como una caja negra, y se centran en los eventos que van de los actores a los sistemas.

- **Diagrama de Colaboración**

Son útiles en la fase exploratoria para identificar objetos.

La distribución de los objetos en el diagrama permite observar adecuadamente la interacción de un objeto con respecto a los demás. La estructura estática viene dada por los enlaces, la dinámica por el envío de mensajes por los mensajes.

¿Qué es una Colaboración?

Es una descripción de una colección de objetos que interactúan para implementar un cierto comportamiento dentro de un contexto. Describe una sociedad de objetos cooperantes unidos para realizar un cierto propósito. Una Colaboración contiene ranuras que son rellenadas por los objetos enlace en tiempo de ejecución. Una ranura de Colaboración se llama Rol porque describe el propósito de un objeto o un enlace dentro de la Colaboración.

Elementos del Diagrama de Colaboración

- Objetos o Roles
- Enlaces o comunicaciones
- Mensajes
- Anidamiento
- Iteración
- Bifurcación

Ventajas y Desventajas del Diagrama de Colaboración

Ventajas

- Permite elegir el orden en que pueden hacerse las cosas.
- Puede describir procesos o casos de uso.
- Muestra los aspectos dinámicos de un sistema.
- Establece las reglas de secuencia a seguir.
- Ayuda a un programador a desarrollar código a través de una descripción lógica de un proceso.

Desventajas

- La gran desventaja de los diagramas de colaboración es que no indican de forma explícita que los objetos ejecutan qué actividades ni tampoco la forma en que el servicio de mensajería trabaja entre ellos.

- **Diagrama de Actividades**

El Diagrama de Actividades es una especialización del Diagrama de Estado, organizado respecto de las acciones y usado para especificar.

- Un Método
- Un Caso de Uso
- Un Proceso de Negocio

Un Diagrama de Actividades es provechoso para entender el comportamiento de alto nivel de la ejecución de un sistema, sin profundizar en los detalles internos de los mensajes. Los parámetros de entrada y salida de una acción se pueden mostrar usando las relaciones de flujo que conectan la acción y un estado de flujo de objeto.

Elementos del Diagrama de Actividades

- Nombre diagrama
- Estado de Acción
- Transición
- Barras de Sincronización
- Nodo de decisión
- Inicio y Fin

Características del Diagrama de Actividades

- Un diagrama de actividades es provechoso para entender el comportamiento de alto nivel de la ejecución de un sistema, sin profundizar en los detalles internos de los mensajes. Los parámetros de entrada y salida de una acción se pueden mostrar usando las relaciones de flujo que conectan la acción y un estado de flujo de objeto.
- Un grafo de actividades contiene estados de actividad que representa la ejecución de una secuencia en un procedimiento, o el funcionamiento de una actividad en un flujo de trabajo. En vez de esperar un evento, como en un estado de espera normal, un estado de actividad espera la terminación de su cómputo.

Cuando la actividad termina, entonces la ejecución procede al siguiente estado de actividad dentro del diagrama.

- Un grafo de actividades puede contener también estados de acción, que son similares a los de actividad pero son atómicos y no permiten transiciones mientras están activos.

Los estados de acción se deben utilizar para las operaciones cortas de mantenimiento.

- Un diagrama de actividades puede contener bifurcaciones, así como divisiones de control en hilos concurrentes. Los hilos concurrentes representan actividades que se pueden realizar

concurrentemente por los diversos objetos o personas. La concurrencia se representa a partir de la agregación, en la cual cada objeto tiene su propio hilo. Las actividades concurrentes se pueden realizar simultáneamente o en cualquier orden.

- **Diagrama de Estado**

Muestra el conjunto de estados por los cuales pasa un objeto durante su vida en una aplicación, junto con los cambios que permiten pasar de un estado a otro. Los Diagramas de Estado representan autómatas de estados finitos, desde el proceso de vida de los estados y las transiciones.

Son útiles solo para los objetos con un comportamiento significativo, cada objeto está en un estado en cierto instante el cual está caracterizado parcialmente por los valores de alguno de los atributos del objeto.

Los Diagramas de Estado son autómatas, jerárquicos que permiten expresar concurrencia, sincronización y jerarquías de los objetos. Son gastos dirigidos y deterministas, esta transición es instantánea y se debe a la concurrencia de un objeto.

Partes que conforman el Diagrama de Estado

- **Estado**

Un estado se representa como una caja redondeada con el nombre del estado en su interior. Una transición se representa como una flecha desde el estado origen al estado destino. La caja de un estado puede tener 1 o 2 compartimentos. En el primer compartimento aparece el nombre del estado. El segundo compartimento es opcional, y en él pueden aparecer acciones de entrada, de salida y acciones internas.

- **Eventos**

Es una ocurrencia que puede causar la transición de un estado a otro de un objeto. Esta ocurrencia puede ser una de varias cosas:

- ✓ Condición que toma el valor de verdadero o falso.
- ✓ Recepción de una señal de otro objeto en el modelo.

- **Recepción de un mensaje**

  Paso de cierto período de tiempo, después de entrar al estado o de cierta hora y fecha particular. El nombre de un evento tiene alcance dentro del paquete en el cual está definido, no es local a la clase que lo nombre.

- **Envío de mensajes**

  Además de mostrar y transición de estados por medio de eventos, puede representarse el momento en el cual se envían mensajes a otros objetos. Esto se realiza mediante una línea punteada dirigida al diagrama de estados del objeto receptor del mensaje.

- **Transición simple**

  Una transición simple es una relación entre dos estados que indica que un objeto en el primer estado puede entrar al segundo estado y ejecutar ciertas operaciones, cuando un evento ocurre y si ciertas condiciones son satisfechas.

- **Transición interna**

  Es una transición que permanece en el mismo estado, en vez de involucrar dos estados distintos. Representa un evento que no causa cambio de estado. Se denota como una cadena adicional en el compartimiento de acciones del estado.

- **Acciones**

  Se puede especificar la solicitud de un servicio a otro objeto como consecuencia de la transición. Se puede especificar el ejecutar una acción como consecuencia de entrar, salir, estar en un estado, o por la ocurrencia de un evento.

- **Generalización de Estados**

Se puede reducir la complejidad de estos diagramas usando la generalización de estados. Se distingue así entre súper estado y sub estados. Un estado puede contener varios sub estados disjuntos. Los sub estados heredan las variables de estado y las transiciones externas. La agregación de estados es la composición de un estado a partir de varios estados independientes. La composición es concurrente por lo que el objeto estará en alguno de los estados de cada uno de los sub estados concurrentes. La destrucción de un objeto es efectiva cuando el flujo de control del autómata alcanza un estado final no anidado. La llegada a un estado final anidado implica la subida al súper estado asociado, no el fin del objeto.

- **Sub estados**

  Un estado puede descomponerse en sub estados, con transiciones entre ellos y conexiones al nivel superior. Las conexiones se ven al nivel inferior como estados de inicio o fin, los cuales se suponen conectados a las entradas y salidas del nivel inmediatamente superior.

- **Transacción Compleja**

  Una transición compleja relaciona tres o más estados en una transición de múltiples fuentes y/o múltiples destinos. Representa la subdivisión en threads del control del objeto o una sincronización. Se representa como una línea vertical de la cual salen o entran varias líneas de transición de estado.

- **Transición a estados anidados**

  Una transición de hacia un estado complejo (descrito mediante estados anidados) significa la entrada al estado inicial del sub diagrama. Las transiciones que salen del estado complejo se entienden como transiciones desde cada uno de los sub estados hacia afuera (a cualquier nivel de profundidad).

- **Transiciones temporizadas**

  Las esperas son actividades que tienen asociada cierta duración. La actividad de espera se interrumpe cuando el evento esperado tiene

lugar. Este evento desencadena una transición que permite salir del estado que alberga la actividad de espera. El flujo de control se transmite entonces a otro estado.

Ventajas y Desventajas del Diagrama de Estado

Ventajas

- El Diagrama de Estados tiene éxito en sistemas interactivos, ya que expresa la intención que tiene el actor (su usuario) al hacer uso del sistema.
- Como técnica de extracción de requerimiento permite que el analista se centre en las necesidades del usuario, qué espera éste lograr al utilizar el sistema, evitando que la gente especializada en informática dirija la funcionalidad del nuevo sistema basándose solamente en criterios tecnológicos.
- A su vez, durante la extracción, el analista se concentra en las tareas centrales del usuario describiendo por lo tanto los casos de uso que mayor valor aportan al negocio. Esto facilita luego la priorización del requerimiento.

Desventajas

- La inclusión de estas relaciones hace que los diagramas sean más difíciles de leer, sobre todo para los clientes.

HERRAMIENTAS DE SOFTWARE

❖ Microsoft Project

Microsoft Project (o MSP) es un software de administración de proyectos diseñado, desarrollado y comercializado por Microsoft para asistir a administradores de proyectos en el desarrollo de planes, asignación de recursos a tareas, dar seguimiento al progreso, administrar presupuesto y analizar cargas de trabajo.

Características de Microsoft Project

- **Ruta crítica:** Se realiza una vez que todos los datos están ingresados. De esta forma de obtienen datos relevantes como los tiempos de inicio y término más cercanos y más lejano de cada actividad.

- **Diagrama de Gantt:** Se muestra por default. Esta gráfica ilustra la programación de actividades en forma de calendario, en el que el eje horizontal representa el tiempo, y el vertical las actividades. Se señalan con barras los tiempos de inicio y término de la actividad, su duración y su secuencia. La actividad crítica se muestra en rojo, las otras en azul. Las actividades que tienen otras secundarias dentro se muestran en color negro.

- **Sobrecarga de recursos:** Significa que le estamos asignando a un recurso humano más tareas de las que puede realizar.

Esto lo podemos observar en la herramienta Gráfica de Recursos del menú.

- **Resumen de Proyecto:** Nos brinda diferentes tipos de información, como las fechas de inicio y término del proyecto en la parte superior, la duración, las horas totales de trabajo, los costos, el estado de las tareas y de los recursos.

- **Cálculo de costos:** Calcula los costos de los recursos y la mano de obra, una vez que los recursos son asignados a cada tarea. Hay dos tipos de reportes: el flujo de efectivo - es un reporte del gasto semanal- y el requerimiento de materiales.

- **Control de proyecto:** Cuando ya se han introducido todos los datos necesarios para realizar la ruta crítica, y se ha establecido el programa de proyecto como se desea, se puede salvar como línea base. Esto permitirá compararla con las modificaciones que se le vayan haciendo al proyecto.

Ventajas y Desventajas de Microsoft Project

Ventajas

- Mejorar la inteligencia empresarial

- Colaborar y coordinar con facilidad

- Administrar los recursos eficazmente
- Administrar el ciclo de vida del proyecto
- Administrar lo simple y lo complejo
- Mejorar constantemente los procesos
- Contrataciones estratégicas
- Obtener más beneficios de las inversiones tecnológicas existentes
- Recuperación real de la inversión

Desventajas

- No se puede medir ni la productividad
- Muy caro
- No se trata de un programa multiplataforma
- No cuenta tampoco con las herramientas básicas

❖ **El IBM Rational Rose**

El Rational Unified Process o Proceso Unificado de Racional. Es un proceso de ingeniería de software que suministra un enfoque para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que satisfaga la necesidad del usuario final dentro de un tiempo y presupuesto previsible. Es una

metodología de desarrollo iterativo enfocada hacia “los casos de uso, manejo de riesgos y el manejo de la arquitectura”.

El RUP mejora la productividad del equipo ya que permite que cada miembro del grupo sin importar su responsabilidad específica acceda a la misma base de datos de conocimiento.

Ciclo De Vida de IBM Rational Rose

En el ciclo de vida RUP veremos una implementación del desarrollo en espiral. Con el ciclo de vida se establecen tareas en fases e iteraciones. El RUP maneja el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable.

Las primeras iteraciones (en las fases de Inicio y Elaboración) se enfocan hacia la comprensión del problema y la tecnología, la delimitación del ámbito del proyecto, la eliminación de los riesgos críticos, y al establecimiento de una base de inicio.

Fases de IBM Rational Rose

a) Fase de inicio

Durante esta fase de inicio las iteraciones se centran con mayor énfasis en las actividades de modela miento de la empresa y en sus requerimientos.

b) Fase de elaboración

Durante esta fase de elaboración, las iteraciones se centran al desarrollo de la base del diseño, encierran más los flujos de trabajo de requerimientos, modelo de la organización, análisis, diseño y una parte de implementación orientada a la base de la construcción.

c) Fase de construcción

Durante esta fase de construcción, se lleva a cabo la construcción del producto por medio de una serie de iteraciones las cuales se seleccionan algunos Casos de Uso

se redefine su análisis y diseño y se procede a su implantación y pruebas. En esta fase se realiza una pequeña cascada para cada ciclo,

se realizan tantas iteraciones hasta que se termine la nueva implementación del producto.

d) Fase de transición

Durante esta fase de transición busca garantizar que se tiene un producto preparado para su entrega al usuario.

Características de IBM Rational Rose

- Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo).
- Pretende implementar las mejores prácticas en Ingeniería de Software.
- Desarrollo iterativo.
- Administración de requisitos.
- Uso de arquitectura basada en componentes.
- Control de cambios.
- Modelado visual del software.
- Verificación de la calidad del software.

El RUP es un producto de Rational (IBM). Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso).

❖ **Erwin Data Modeler**

Es un software de herramienta para el modelado de datos (datos de análisis de requisitos, la base de datos de diseño, etc.) del personalizados desarrollado sistemas de información, incluyendo bases de datos de los sistemas transaccionales y data marts. Motor de modelado de datos de ERwin se basa en la IDEF1X método, a pesar de que ahora es compatible con los diagramas que aparecen con ingeniería de la información notación también. La lógica funciona "ERwin" estilizada de una abreviatura de "Entidad de Relaciones" y "Windows" de Microsoft.

Características de Erwin Data Modeler

- **Modelado de datos lógico**: Modelos puramente lógicos pueden crearse, de la cual se pueden derivar modelos físicos. También se admiten combinaciones de modelos lógicos y físicos. Apoya el tipo de entidad y lógica nombres de atributos y descripciones, lógicos dominios y tipos de datos, así como relación de nombres.

- **Modelado de datos físico:** Se pueden crear modelos puramente físicos así como combinaciones de modelos lógicos y físicos. Apoya la denominación y descripción de tablas y columnas, tipos de datos definidos por el usuario, claves primarias, llaves foráneas, claves alternativas y el nombramiento y la definición de restricciones. También se incluye soporte para índices, vistas, procedimientos almacenados y desencadenadores.

- **Transformación de lógico a físico:** Incluye un diccionario de abreviatura llamado “Nombres de Editor de normas” y una asignación de tipo de datos lógico-a-RDBMS llamado “Tipo de datos estándares Editor”.

- **Microsoft SQL Server**

Microsoft SQL Server es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. Sus lenguajes para consultas son T-SQL y ANSI SQL. Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle, PostgreSQL o MySQL.

Características de Microsoft SQL Server

- Soporte de transacciones.
- Soporta procedimientos almacenados.
- Incluye también un entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
- Además permite administrar información de otros servidores de datos.

Este sistema incluye una versión reducida, llamada MSDE con el mismo motor de base de datos pero orientado a proyectos más pequeños, que en sus versiones 2005 y 2008 pasa a ser el SQL Express Edition, que se distribuye en forma gratuita.

Para el desarrollo de aplicaciones más complejas (tres o más capas), Microsoft SQL Server incluye interfaces de acceso para varias plataformas de desarrollo, entre ellas .NET, pero el servidor sólo está disponible para Sistemas Operativos.

SISTEMA OPERATIVO

En esta ocasión describiré el S.O en la cual se desenvolverá el proyecto a implantar:

- **Windows 7**

  Es la versión más reciente de Microsoft Windows, línea de sistemas operativos producida por Microsoft Corporation. Windows 7 incluye varias características nuevas, como mejoras en el reconocimiento de escritura a mano, soporte para discos duros virtuales, rendimiento mejorado en procesadores multinúcleo, mejor rendimiento de arranque, DirectAccess, y mejoras en el núcleo. Windows 7 añade soporte para sistemas que utilizan múltiples tarjetas gráficas de proveedores distintos (heterogéneos multi-adapter o Multi-GPU), una nueva versión de Windows Media Center y un gadget para él, y aplicaciones como Paint, Wordpad y la Calculadora rediseñadas. Se añadieron varios elementos al Panel de control, como un asistente para calibrar el color de la pantalla, un calibrador de texto ClearType, Solución de problemas, Ubicación y otros sensores, Administrador de credenciales, Iconos en el área de notificación, entre otros.

  La barra de tareas fue rediseñada, haciéndola más ancha, y los botones de las ventanas ya no traen texto, sino únicamente el icono de la aplicación. Estos cambios se hacen para mejorar el desempeño en sistemas de pantalla táctil.

LENGUAJE DE PROGRAMACIÓN

- **Visual C#**

El lenguaje Visual C# es actualmente uno de los lenguajes de programación más populares, ya que es un lenguaje de para el desarrollo de sistemas d el propósito general. En los últimos tiempos C y C++ han sido los lenguajes más utilizados en el desarrollo de aplicación una aplicación es. ambos lenguajes proporcionan al programador el nivel de abstracción preciso para abordar el desarrollo de cualquier aplicación por compleja que sea, así como mecanismos de bajo nivel para utilizar las característica más avanzadas de las plataformas sobre las que se desarrolla resalta largo comparado con otros lenguajes como Visual Basic, que ofrecen además de facilidad, cuando la flexibilidad de los desarrolladores de C y C++ requieren. La solución que Microsoft da a este problema des el lenguaje denominado C#. Se trata de un lenguaje moderno orientado a objetos que permite desarrollar una amplia gama de aplicaciones para la nueva plataforma Microsoft. Net, la cual se caracteriza por proporcionar utilidades y servicios para sacar un provecho total tanto de la informática como de las comunicaciones.

Definición

Microsoft.Net se trata de un entorno de desarrollo multilenguaje diseñado por Microsoft para simplificar la construcción, distribución y ejecución de aplicaciones para Internet. Tiene fundamentalmente tres componentes: una máquina virtual (CLR: Common Language Runtime= que procesa código escrito en un lenguaje intermedio)

MSIL: Microsoft Intermediante Language, una biblioteca de clases) biblioteca. NET Framework= y ASP.NET que proporciona los servicios necesarios para crear aplicaciones Web.

Visual C# es uno de los lenguajes de programación de alto nivel que pertenecen al paquete .NET otros lenguajes son Visual Basic, C/C++. Con él se pueden escribir tanto programas convencionales como para Internet.

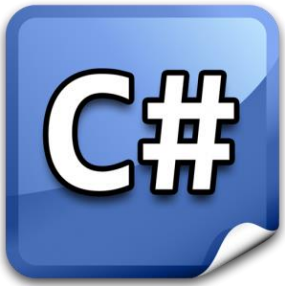
El paquete .NET incluye un compilador (programa traductor= de C# que produce un código escrito en un lenguaje intermedio, común para todos los lenguajes de dicha plataforma, que será el que la máquina virtual ejecutará) esto es, cada lenguaje de la plataforma tiene su compilador que produce código correspondiente a un único lenguaje: MSIL.

Por lo tanto, MSIL es un lenguaje máquina que no es específico de ningún procesador, sino de la máquina virtual de .NET En realidad se trata de un lenguaje de más alto nivel que otros lenguajes máquina: trata directamente con objetos y tiene instrucciones para cargarlos, guardarlos, iniciarlos, invocar a sus métodos.

Así como para realizar operaciones aritméticas y lógicas, para controlar el flujo de ejecución, etc. A su vez, la máquina virtual posee un recolector de basura (para eliminar los objetos cuando no estén referenciados) y proporciona traductores del lenguaje intermedio a código nativo para cada arquitectura soportada; se trata de compiladores JIT (Just in Time: al instante).

Una característica importante del por que utilizamos el lenguaje C#, es que es un lenguaje de programación orientado a objetos (POO).

Además es fácil de aprender. Tiene un tamaño pequeño que favorece el desarrollo y reduce las posibilidades de cometer errores; a la vez es potente y flexible.



REDES

¿Qué es una red?

Una red, es aquella conexión entre dos o más computadoras, las cuales comparten algunos elementos del hardware, como las impresoras o el CD-ROM, al igual que información, como diversos archivos de la organización. Al igual que ciertos servicios comunes, como salas de chat y mensajería instantánea, correos electrónicos, etc.

Existen diversas tecnologías de red, siendo las más conocidas las de tipo PPP, la HDLC, ETHERNET, TOKEN RING, entre otras.

Con respecto a los estándares de red, tenemos el IEEE 802.3, 1EEE 802.5, IEEE 802.11, IEEE 802.15. El primero se utiliza en el ETHERNET, el segundo en el TOKEN RING, el tercero en el sistema Wi-FI y el último en el BLUETOOTH.

Ahora, una red, puede llegar a abarcar distintas distancias, por lo que cada una tiene una categorización independiente. Tenemos la PAN, que es una red de área personal que abarca hasta 10 metros cuadrados.

Por otra parte tenemos la LAN, que es la más habitual de todas. Ya que se utiliza muchísimo en las empresas. Se le llama área de red local y esta abarca hasta un kilómetro cuadrado. Asimismo, existe la MAN o área de red metropolitana, que llega a abarcar hasta diez kilómetros cuadrados. Por último, existe la red WAN o red de área amplia, la cual se utiliza en todo un país, continente, etc.

Por último, podemos categorizar a una red, según las direcciones que tiene, para compartir la información o transmitirla. Esta aquella en la cual, una terminal transmite y la otra recibe. Por otra parte, esta aquella red que permite que una computadora transmita información y las otras reciban y por último, esta aquella red que permite que varias envíen y reciban información de manera simultánea.

El uso más común de una red lo podemos ver en las oficinas, en donde varios usuarios comparten recursos como una impresora desde sus computadoras. En la actualidad es frecuente incluso que las personas configuren redes internas en el hogar para compartir los recursos informáticos con la familia, como por ejemplo el acceso a Internet y el uso de impresoras y fax.

Características de una red

- **Compartición de archivos:** Fue la razón principal para tener una red. Para que se cumpla se requiere de un directorio compartido que pueda ser accesado por muchos usuarios de la red, junto a toda la lógica asociada para que más de una persona no realice cambios conflictivos a un archivo al mismo tiempo.

- **Compartición de impresoras:** Con esto reducimos el número de impresoras en la organización. Se hace necesario el uso de colas de impresión para que las impresiones se lleven a cabo y de forma automática enviar los trabajos en espera en dicha cola.

- **Servicios de aplicación:** Así como se pueden compartir archivos o carpetas en una red, se pueden compartir aplicaciones, las más comunes son aplicativos de contabilidad. Si se requiere por ejemplo de instalar algún programa en diversas computadoras de la red, en lugar de ir colocando el CD-ROM en cada una, se puede tener una carpeta con el contenido del mismo y ejecutar el instalador desde cada equipo.

- **Correo electrónico:** Es un recurso bastante valioso y que incluso muchas organizaciones no lo aprovechan al máximo. No solamente es útil para las comunicaciones internas sino también para las externas.

- **Acceso remoto:** Se usa principalmente para acceder desde el exterior a los recursos de la red interna. Los usuarios la utilizan para ver sus archivos, correo electrónico ya sea que se encuentren de viaje, desde su hogar, etc.

INTERNET

Internet es un conjunto descentralizado de redes de comunicación interconectadas que utilizan la familia de protocolos TCP/IP, garantizando que las redes físicas heterogéneas que la componen funcionen como una red lógica única, de alcance mundial. Sus orígenes se remontan a 1969, cuando se estableció la primera conexión de computadoras, conocida como ARPANET, entre tres universidades en California y una en Utah, Estados Unidos.

Uno de los servicios que más éxito ha tenido en Internet ha sido la World Wide Web (WWW, o "la Web"), hasta tal punto que es habitual la confusión entre ambos términos.

La WWW es un conjunto de protocolos que permite, de forma sencilla, la consulta remota de archivos de hipertexto. Ésta fue un desarrollo posterior (1990) y utiliza Internet como medio de transmisión.

CAPITULO III: OBJETIVOS

III OBJETIVOS

3.1 OBJETIVO GENERAL

Desarrollar un sistema computarizado en el departamento comercial para automatizar las actividades, garantizando la gestión y seguridad de la información para servir de apoyo a la toma de decisiones dentro de la empresa.

3.2 OBJETIVOS ESPECÍFICOS

- ✓ Apoyo a los Procesos de la empresa
- ✓ Control de ventas realizadas
- ✓ Desarrollar estrategias para la empresa
- ✓ Rapidez al requerir la información
- ✓ Restringir el acceso por niveles.
- ✓ Optima administración de las actividades
- ✓ Aplicación rápida, sencilla y segura.
- ✓ Desarrollar competencias para crear, liderar, dinamizar, evaluar y acompañar.
- ✓ Desplegar información de la empresa

CAPITULO IV: HIPOTESIS Y VARIABLES

IV HIPOTESIS Y VARIABLES

4.1 HIPÓTESIS GENERAL

El factor de mayor predisposición es que la librería y distribuidora Sánchez carece del control de actividades en cuanto al manejo de datos en el departamento comercial.

4.2 VARIABLES

- **VARIABLES INDEPENDIENTE**

  Sistema Computarizado

- **VARIABLE DEPENDIENTE**

  Administración del Departamente Comercial

4.3 OPERACIONALIZACIÓN DE VARIABLES

✓ Variable Independiente

VARIABLE	DEFINICIÓN	TIPO	DIMENCIÓN	INDICADORES
Sistema computarizado	El sistema computarizado tiene una enorme importancia en el incremento de la capacidad organizacional, que nos permita obtener ventajas competitivas el cual ayudara al personal de la empresa a analizar los problemas. Mejorando las posibles deficiencias que puedan existir en el proceso de la en la empresa	Independiente		• Herramientas tecnológicas • Organización en la información de la empresa
			Restringir el acceso por niveles	• Registro de los usuarios con acceso • Asignación de permisos de forma personalizada
			Optimizar la administración de las actividades	• Formulario de registro • Cambio de conductas • Programar los Eventos futuros

VARIABLE	DEFINICIÓN	TIPO	DIMENCIÓN	INDICADORES
Administración departamento comercial	Su objetivo es evaluar la información y disponibilidad que se maneja dentro del departamento comercial luego desarrollar las posibles alternativas y finalmente convertirla en acción a la decisión seleccionada	Dependiente	Apoyo a los procesos de la empresa	• Organigrama de la empresa • Planeación estratégica • Interacción del cliente con la empresa
			Estrategias para una organización.	• Atender la necesidades del cliente • Cooperación Técnica • Formulación de marco estratégico
			Desarrollo de competencias	• Habilidad de evaluar los procesos • Biblioteca de recursos • Unidades de conocimiento

CAPITULO V: ESTRATEGIA METODOLOGICA

V ESTRATEGIA METODOLOGICA

5.1 TIPO DE INVESTIGACIÓN

Por su finalidad su método de investigación documental y campo (Teórico y práctico). Por su temporalidad es trasversal o sincrónica

Dicho método se concentró en la recopilación de la información de la empresa Sánchez en forma fundamental a la dificultad que presentan al ejecutar sus procesos.

Respectivamente la recolección se realizó englobando las áreas de ventas facturación y almacén. En el que encontramos puntos de deficiencia (falta de organización, no contar con sistema de almacenamiento de la información de la empresa, carece de interacción con los clientes) que son dichas fallas el cual dificultan la toma de decisiones de la empresa.

5.2 DISEÑO DE INVESTIGACIÓN

El tipo de diseño es no experimental. Este diseño se emplea en situaciones en las cuales es difícil o casi imposible el control de variables que pueden afectar el trabajo investigado.
El cual consiste en ser descriptivo simple. Por las siguientes razones.

I. Se puede manipular intencionalmente la variable independiente
II. Se puede medir de manera confiable el efecto que tiene la variable independiente sobre la variable dependiente.
III. La variación de la variable dependiente se debe únicamente a la manipulación de la variable independiente.

5.3 POBLACIÓN Y MUESTRA

POBLACION

La población motivo de esta investigación está conformada por el total de 30 personas entre trabajadores y personas externas, habiendo sido seleccionadas por la naturaleza de su actividad y por su interés en aplicar la técnica de evaluación del desempeño.

MUESTRA

La muestra utilizada de la fuerza laboral en la presente investigación, está conformada por el personal de la Librería Sánchez SRL, siendo dicha muestra el orden de **5** trabajadores, comprendiendo también a gerentes, clasificados en tres categorías, profesional, técnico, auxiliar, utilizado preferentemente a personas que tenían conocimiento de dicha problemática e interés por participar

Dónde:

Tamaño de la muestra: n

Tamaño de la población: N

Probabilidad de éxito: q

Margen de error: E

Nivel de confianza requerido: Z

Reemplazando:

n=30

Cuando la población es pequeña, la muestra obtenida mediante esta fórmula es demasiado grande, en estos casos se debe aplicar la siguiente formula correctora:

$$nh = Nh(n)/N$$

5.4 TÉCNICAS E INSTRUMENTOS DE RECOLECCIÓN DE INFORMACIÓN

- **Técnicas de campo**

  Se hiso el uso de entrevistas para el recojo de la información pertinente relacionado a las áreas en el que se desempeñan las actividades.

- **Procedimientos para la recolección de datos**

  Para la ejecución de la primera actividad haremos uso del método histórico tendencial, el mismo que está vinculado al conocimiento de las distintas etapas del surgimiento desarrollo y evolución del sistema

  Para desarrollar la segunda actividad, haremos uso del método empírico que permitirá recoger toda la información empírica respecto a la opinión de los responsables pre y post a la implantación del sistema computarizado para la administración del departamento comercial.

  La tercera actividad, está referida al computarizado para la administración del departamento comercial para el área de compra, venta y Existencias.

5.5 TÉCNICAS DE ANÁLISIS E INTERPRETACIÓN DE DATOS

- **Entrevista**

  Las entrevistas se utilizan para recopilar información en forma verbal, a través de preguntas que propone el grupo de investigación. Quienes responden pueden ser los jefes o empleados de cada área, en otras palabras, la entrevista es un intercambio de información que se efectúa cara a cara. Es un canal de comunicación entre el entrevistador y los usuarios. La entrevista es la técnica más significativa y productiva de que dispone el grupo investigador para recabar datos. Por otra parte, la entrevista ofrece al investigador una excelente oportunidad para establecer una conversación con el personal de trabajo, lo cual es fundamental en el transcurso del tema de investigación.

Encuesta

Herramienta para recolectar información mediante la elaboración de un cuestionario sobre temas relacionados a la investigación. Al hacer el cuestionario hay que formular preguntas que revelen realmente la información deseada.

PLANIFICACIÓN DEL PROYECTO

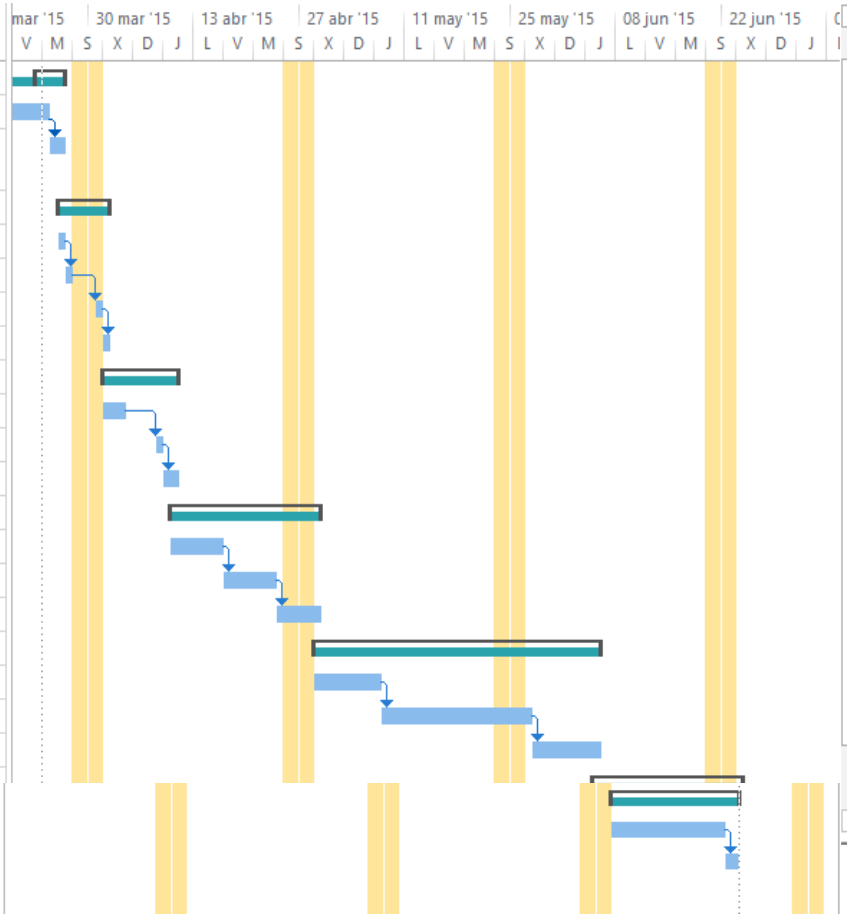
➢ PLANIFICACION DE PROYECTO

ALCANCE DEL SISTEMA

- ✓ **Mantenimiento**
  - Gestion de Categorias.
  - Gestión de Clientes.
  - Gestión de Empresas.
  - Gestión de Proveedores.
  - Gestión de Trabajador.
  - Gestión de Locales.
  - Gestión de Servicios.
  - Interface.
  - Lineas.
  - Gestion de Marcas.
  - Persona.
- ✓ **Procesos**
  - Interfaz principal.
  - Inicio de sesión.
  - Amortizaciones.
  - Compra.
  - Venta.
- ✓ **Reportes**
  - Reportes de Existencias.
  - Reporte de Trabajadores.
  - Reporte de Ventas.
  - Reporte de créditos.

➢ CRONOGRAMA DE ACTIVIDADES DEL PROYECTO

Nombre de tarea	Duración	Comienzo	Fin
IDENTIFICACIÓN DEL PROBLEMA	**3 días**	**lun 23/03/15**	**jue 26/03/15**
Análisis de Datos obtenidos en la visita	1 día	vie 20/03/15	mar 24/03/15
Formulación de Problema encontrado (2da visita)	2 días	mié 25/03/15	jue 26/03/15
DETERMINACIÓN DE REQUERIMIENTOS	**4 días**	**jue 26/03/15**	**mié 01/04/15**
Generación de formatos	1 día	jue 26/03/15	jue 26/03/15
Análisis de datos obtenidos	1 día	vie 27/03/15	vie 27/03/15
Creación del acta de requerimientos	1 día	mar 31/03/15	mar 31/03/15
Aprobación del acta de requerimientos	1 día	mié 01/04/15	mié 01/04/15
ANÁLISIS DE SISTEMAS	**6 días**	**mié 01/04/15**	**vie 10/04/15**
Creación de Diagramas de Caso de Uso	3 días	mié 01/04/15	vie 03/04/15
Modelamiento de BD	1 día	mié 08/04/15	mié 08/04/15
Creación de otros Diagramas de UML	2 días	jue 09/04/15	vie 10/04/15
DISEÑO DE SISTEMAS	**8 días**	**vie 10/04/15**	**mié 29/04/15**
Diseño de la Base de Datos	3 días	vie 10/04/15	jue 16/04/15
Diseño de Formularios	3 días	vie 17/04/15	jue 23/04/15
Diseño de Formatos de Reportes	2 días	vie 24/04/15	mié 29/04/15
CODIFICACIÓN	**18 días**	**mié 29/04/15**	**vie 05/06/15**
Codificación de la BD	5 días	mié 29/04/15	jue 07/05/15
Codificación de los formularios	8 días	vie 08/05/15	mié 27/05/15
Codificación de los reportes	5 días	jue 28/05/15	vie 05/06/15
PRUEBA	**8 días**	**vie 05/06/15**	**mié 24/06/15**
IMPLEMENTACIÓN	**9 días**	**mié 24/06/15**	**vie 10/07/15**
Capacitación en uso del software	7 días	mié 24/06/15	mié 08/07/15
Revision	2 días	jue 09/07/15	vie 10/07/15



➢ FACTORES CRITICOS DE ÉXITO

- ✓ Investigar sobre sistemas implantados con anterioridad.
- ✓ Definir las iteraciones y las funcionalidades que formarán parte del alcance del sistema a implementar.
- ✓ Definir los estándares de desarrollo y documentación necesaria para el inicio de la ejecución del proyecto.
- ✓ Interés de parte de la alta dirección de la municipalidad y de parte del equipo de desarrollo.

➢ RESTRICCIONES

- ✓ El Sistema Computarizado para la administración del área comercial de la Librería Distribuidora Sanchez involucrará los módulos de Compras, Ventas, Amortizaciones, consultas y reportes.
- ✓ Los usuarios que se autentiquen en el sistema solo tendrán acceso a ciertas operaciones, según los roles y permisos asignados.

➢ ANALISIS DE LA FACTIBILIDAD

FACTIBILIDAD TECNICA

- ✓ El equipo de desarrollo se encuentra capacitado en cada una de las herramientas utilizadas para el desarrollo del sistema.
- ✓ El sistema contará con interfaces amigables, la cual permitirá al usuario interactuar con facilidad.
- ✓ Actualmente la empresa Sánchez cuenta con la arquitectura tecnológica necesaria para la implementación de proyecto.

FACTIBILIDAD OPERATIVA

- ✓ Se cuenta con el apoyo del personal correspondiente de la Librería distribuidora Sánchez, para el desarrollo del proyecto proporcionando la información necesaria.
- ✓ Se capacitará a los usuarios finales.
- ✓ Los usuarios finales del sistema cuentan con conocimientos básicos de informática.
- ✓ El sistema agilizará los procesos involucrados.

FACTIBILIDAD ECONÓMICA

El estudio de factibilidad económica involucra las etapas de desarrollo e implantación del proyecto. Para estas etapas será necesario contar con recursos humanos, hardware, software, materiales y servicios.

Estos recursos significan un costo para el proyecto el cual se ha divido en:

- ✓ Costo de inversión
- ✓ Costo de desarrollo
- ✓ Costo de operación

❖ COSTO DE INVERSION

Los costos de inversión son aquellos gastos que se realizan al inicio del proyecto, se refieren a la adquisición de equipo de cómputo y programas tecnológicos necesarios para el desarrollo del sistema. Estos se han clasificado en costo de hardware y costo de software.

COSTO DE HARDWARE

HARDWARE	CANTIDAD	PRECIO UNITARIO(S/.)	SUBTOTAL (S/.)
PC SERVIDOR	1	2200	2200.00
		TOTAL	2200.00

COSTO DE SOFTWARE

SOFTWARE DE DESARROLLO	SUBTOTAL (S/.)
Visual Studio 2010	600.00
SQL Server 2008 r2	5400.00
Windows Server 2008 Standard	2850.00
CA Erwin Data Modeler Community Edition	300.00
TOTAL	**9150.00**

En resumen los costos de inversión se detallan en la siguiente tabla:

COSTO DE INVERSIÓN	SUBTOTAL
Costo Hardware	2200.00
Costo Software	9150.00
TOTAL	**11140.00**

❖ COSTO DE DESARROLLO

Estos costos son los que se generan durante la implementación del proyecto. Estos costos se han clasificado en costos de recursos humanos, recursos materiales, servicios.

RECURSOS HUMANOS

CANTIDAD (UNID.)	FUNCION	SUELDO MENSUAL	TIEMPO (MESES)	SUBTOTAL (S/.)
2	Analista-Programador	0.00	9	0.00
1	Diseñador	0.00	9	0.00
			TOTAL	**0.00**

RECURSOS MATERIALES

Estos son los materiales de escritorio utilizados para la documentación del proyecto.

ITEM	CANTIDAD	SUBTOTAL (S/.)
CD	15 unidades	15.00
DVD	15 unidades	22.50
Papel Bond A4	03 Millares	76.50
Lapiceros	05 unidades	5.00
Folders	24 unidades	12.00
Sobre Manila A4	10 unidades	5.00
	TOTAL	**136.00**

SERVICIOS

Estos son los costos generados por los servicios utilizados durante la implementación del proyecto.

ITEM	SUBTOTAL (S/.)
Movilidad	100.00
Telecomunicaciones	50.00
Fotocopias	50.00
Impresiones	200.00
Espiralados	20.00
Encuadernación	35.00
Internet	300.00
TOTAL	**755.00**

En resumen los costos de desarrollo se detallan en la siguiente tabla:

DESCRIPCION	SUBTOTAL
Recursos Humanos	0.00
Recursos materiales	146.00
Servicios	994.00
TOTAL	**1141.00**

- **COSTO DE OPERACIÓN**

  Los costos de operación involucran los costos de mantenimiento de hardware, mantenimiento de software, recursos humanos y depreciación.

MANTENIMIENTO DE HARDWARE

Está definido por los gastos generados por la operación mantenimiento y soporte de los equipos de desarrollo. Para esto se ha estimado un monto anual de S/.1200.00.

MANTENIMIENTO DE SOFTWARE

Está definido por los gastos generados para el mantenimiento y actualización del software a utilizar durante el desarrollo y puesta en marcha del sistema.

Para el mantenimiento y actualización del software se tendrá un costo de S/. 00.00, ya que el equipo de desarrollo se encargará de dicha función.

RECURSOS HUMANOS

Está definido por el personal encargado de dar soporte al sistema, además del software y hardware utilizado para el desarrollo e implantación del proyecto.

Los costos generados por recursos humanos será de S/ 00.00 nuevos soles ya que los responsables del desarrollo del proyecto serán los encargados de cubrir el mantenimiento y soporte del sistema.

DEPRECIACIÓN

CONCEPTO	INVERSION INICIAL (S/.)	TASA DE DEPRECIACION (%)	DEPRECIACION POR AñO(S/.)
HARDWARE	3200.00	20	500.00
		TOTAL	500.00

Los costos de operación quedan resumidos en la siguiente tabla:

COSTO	SUBTOTAL(S/.)
Mantenimiento de hardware	1300.00
Mantenimiento de software	0.00
Recursos humanos	0.00
Depreciación	500
TOTAL	**1800.00**

El resumen de los costos se muestra en la siguiente tabla:

COSTO	SUBTOTAL(S/.)
COSTO DE INVERSION	11140.00
COSTO DE DESARROLLO	0.00
COSTO DE OPERACIÓN	1800.00
TOTAL	12940.00

ANÁLISIS DE RENTABILIDAD

Inversión	50 días
Costo del plan de marketing	S/. 12.940,00
Inversión total	S/. 12.940,00

Valores cálculos TIR	1 año
Inversión	- s/ 12.940,00
Tasa	20 %
Beneficio Obtenido	S/. 20.000,00
TIR	12%

Resultado	1 año
Total Beneficios	S/. 20.000,00

FCF(caja libre)	S/. 7060.00

ROI-1 AÑO	54.55 %
VAN-1 AÑO	1.027.12

TIR-1AÑO	12%
PRI	7.764

Formula	1 año
FCF	Es total beneficio-total inversión
ROI	Es (total de beneficio –total inversión)/inversión
TIR	Es la inversión, la tasa, el beneficio
VAN	Es (la tasa, beneficio)+ la inversión
PERIODO DE RECUPERACÓN DE LA INVERSIÓN (PRI)	Es (inversión total /total beneficio) * 12 meses

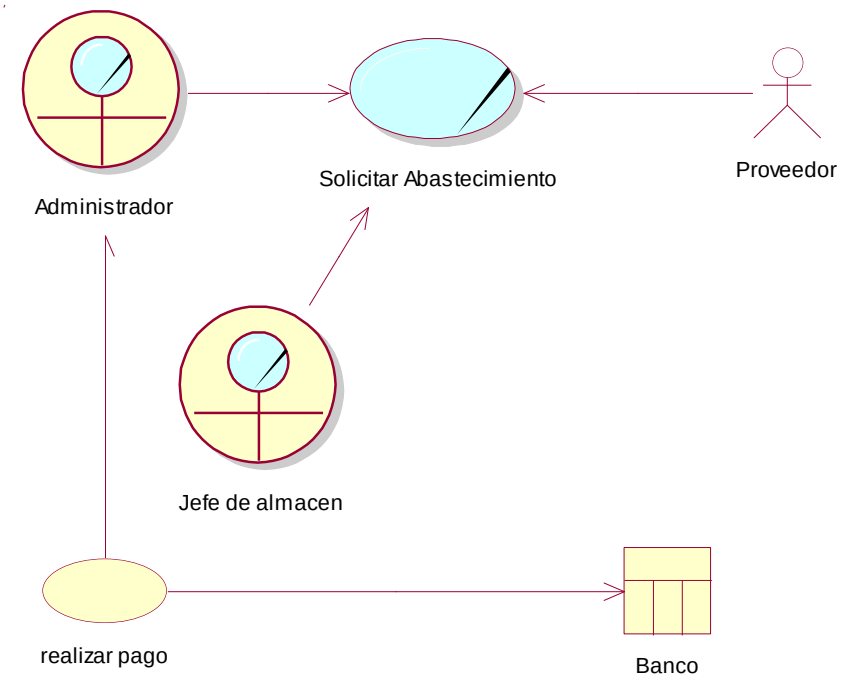
CAPITULO VI: PRESENTACION, INTERPRETACIÓN Y DISCUSIÓN DE RESULTADOS

IV PRESENTACION, INTERPRETACION Y DISCUCION DE RESULTADOS

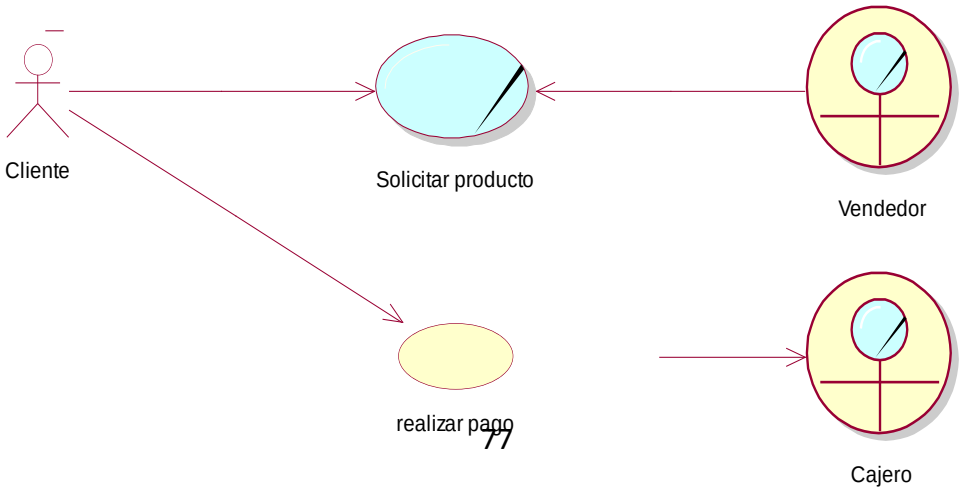
DISEÑO – DESARROLLO DEL SISTEMA

DIAGRAMA CASOS DE USO DEL NEGOCIO – ACTUAL.

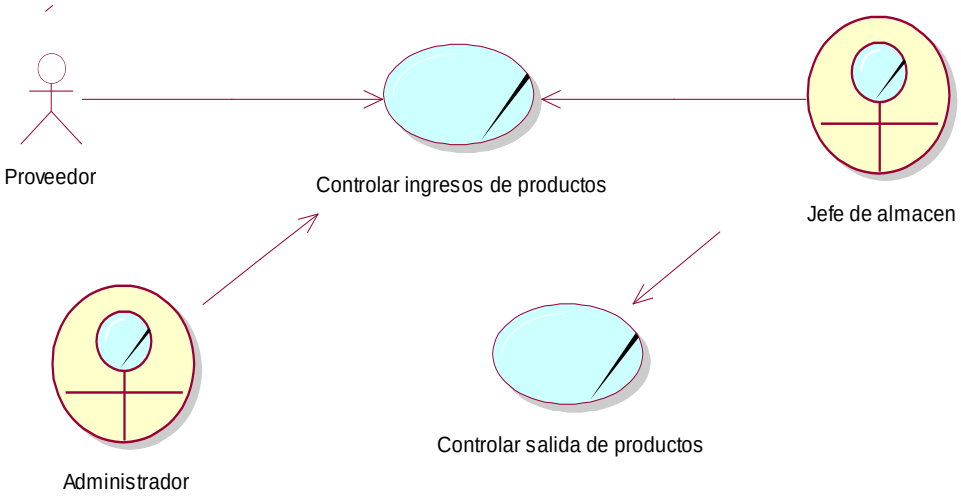
COMPRA



VENTA



KARDEX (CONTROL DE PRODUCTOS)



AMORTIZACIONES (CONTROL DE PAGOS)

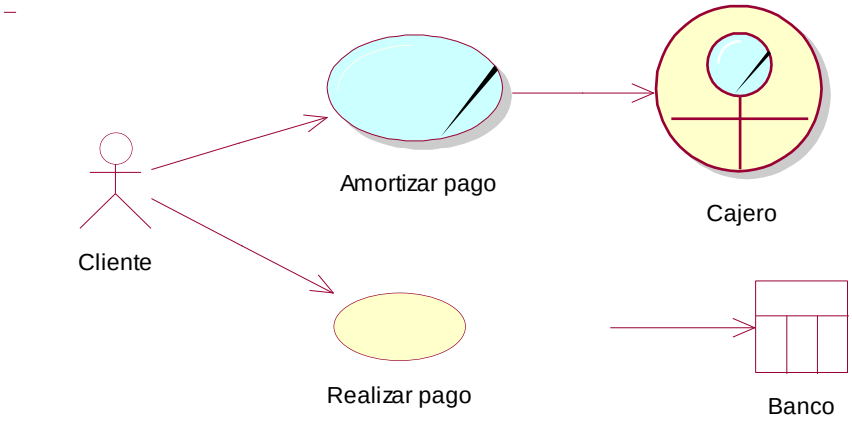
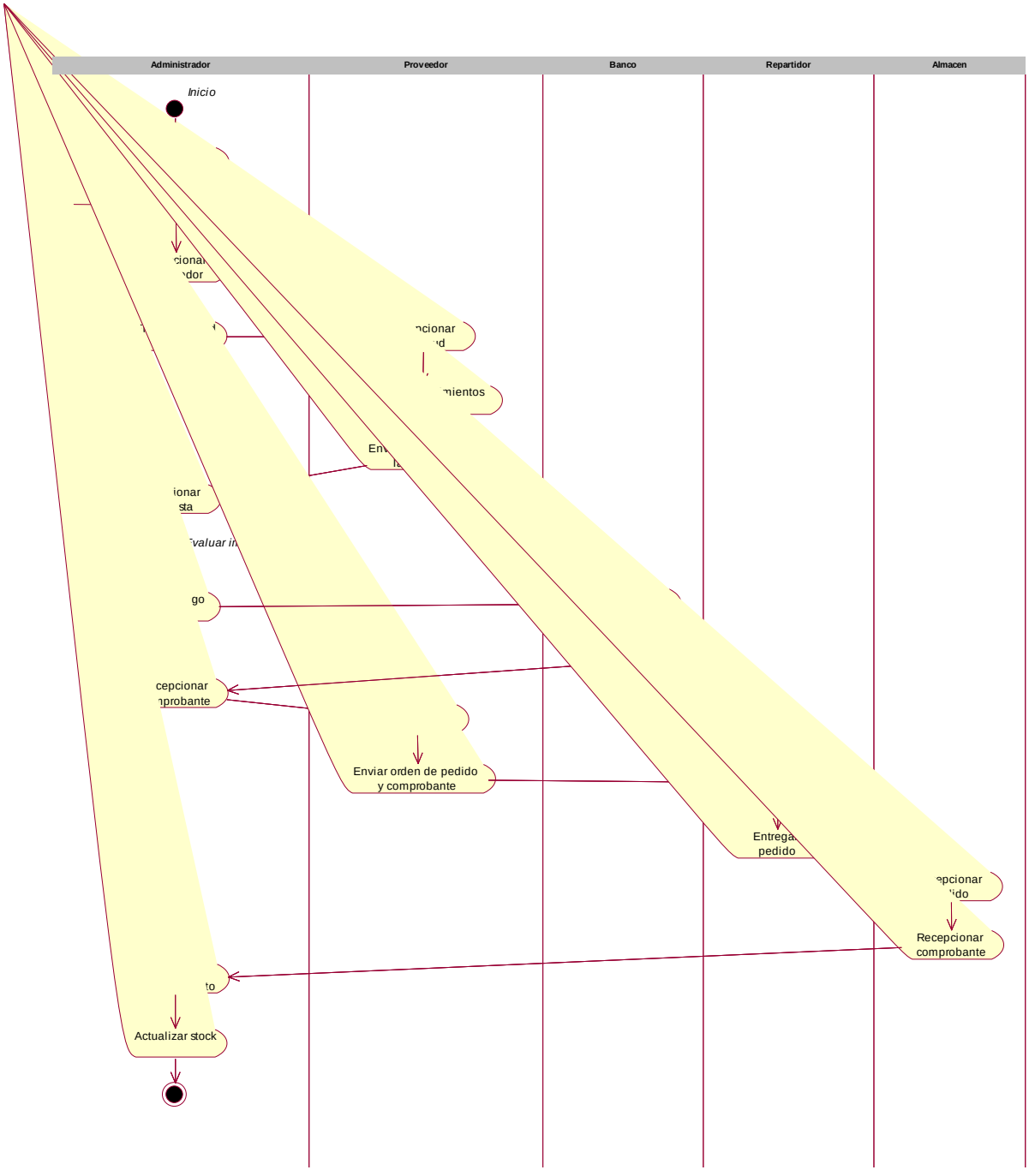
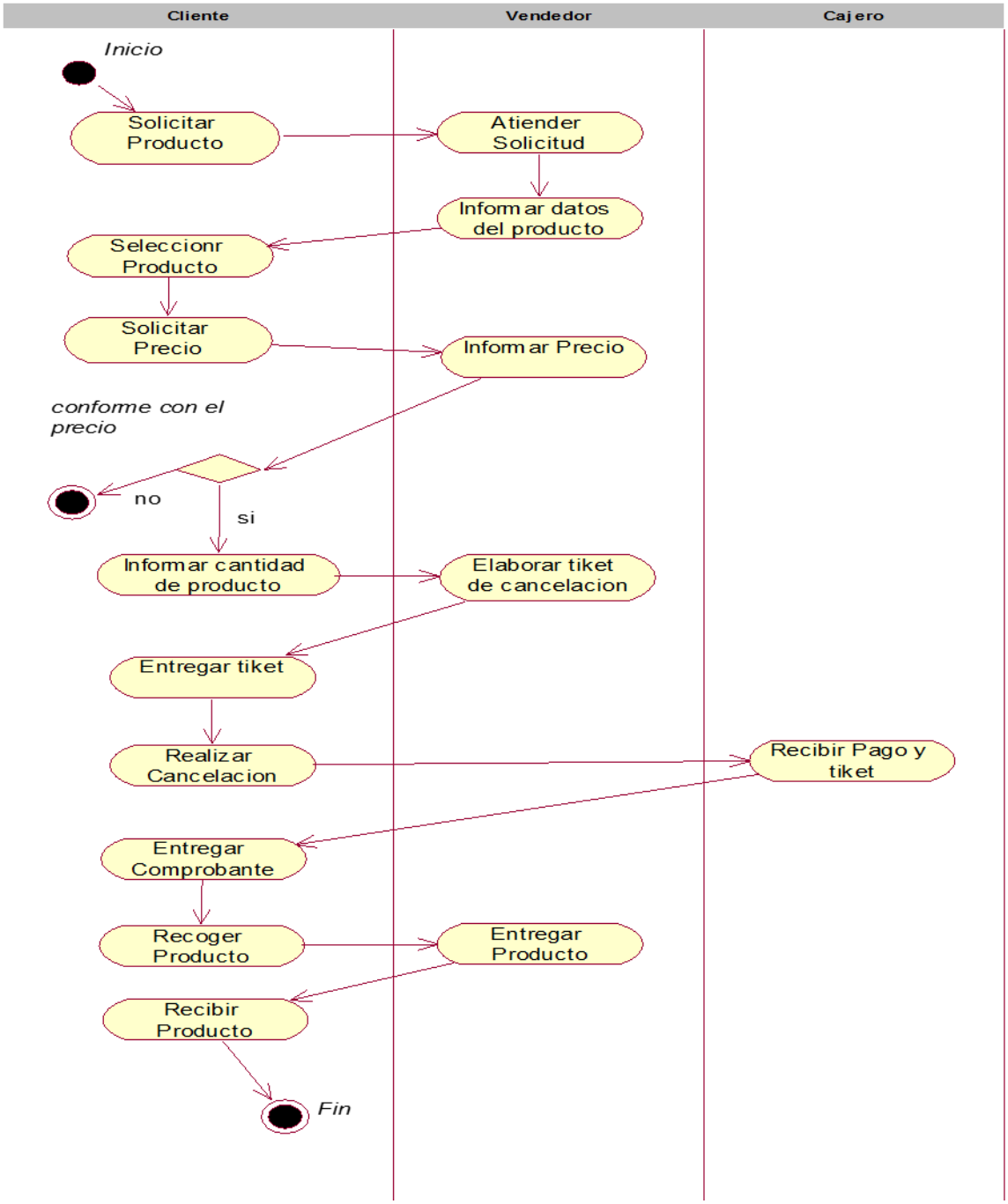


DIAGRAMA DE ACTIVIDADES DEL NEGOCIO.

COMPRA

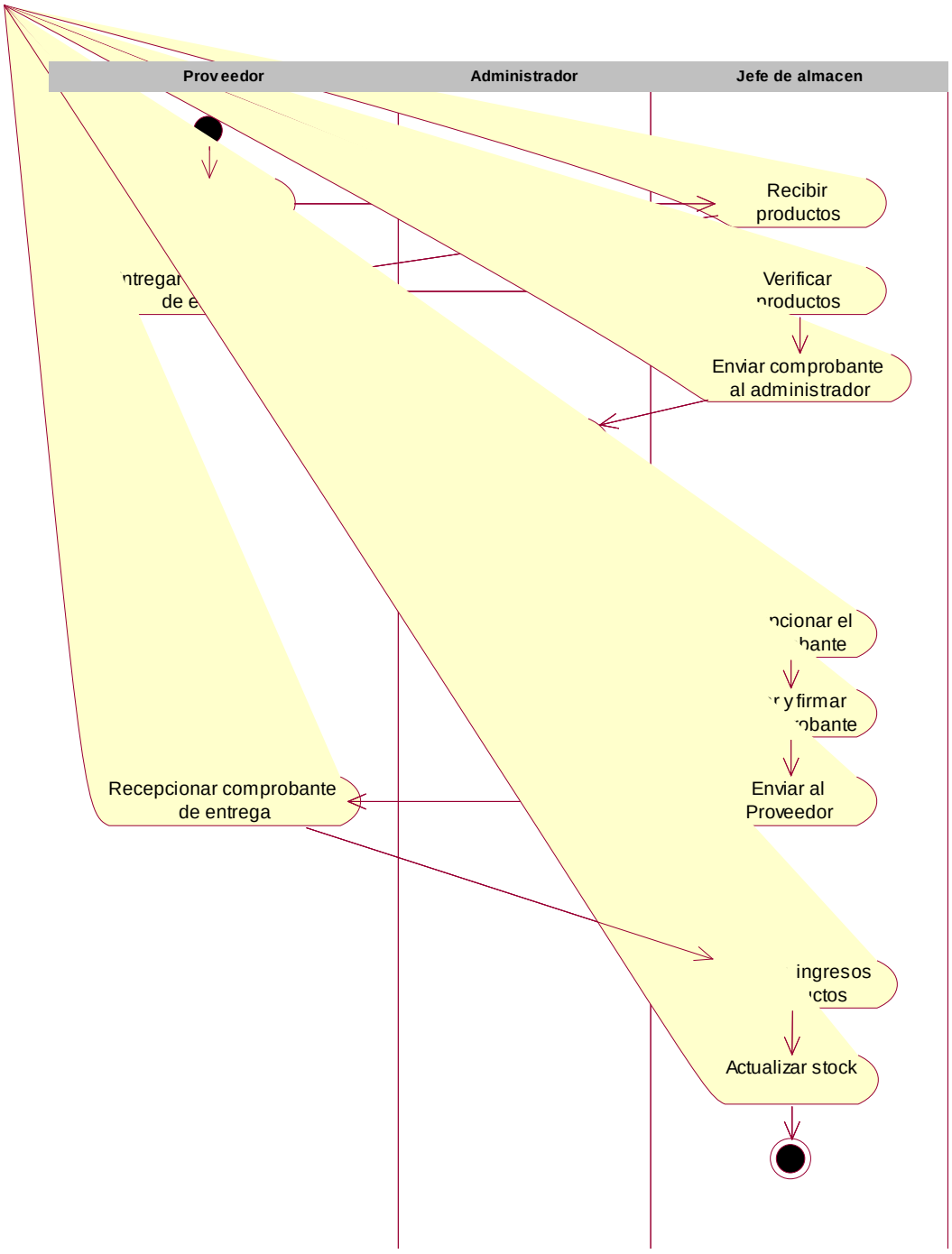


VENTA

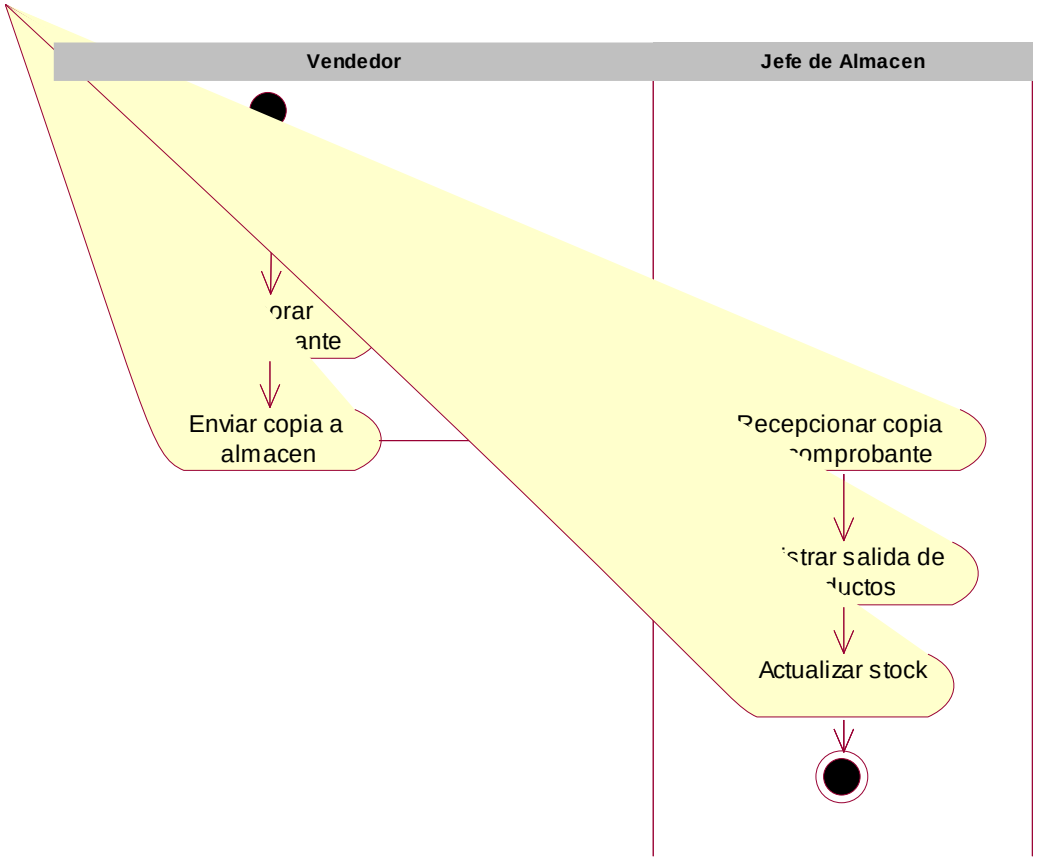

Cliente
Vendedor
Cajero
Inicio
Solicitar Producto
Atiender Solicitud
Informar datos del producto
Seleccionr Producto
Solicitar Precio
Informar Precio
conforme con el precio
no
si
Informar cantidad de producto
Elaborar tiket de cancelacion
Entregar tiket
Realizar Cancelacion
Recibir Pago y tiket
Entregar Comprobante
Recoger Producto
Entregar Producto
Recibir Producto
Fin

KARDEX (CONTROL DE PRODUCTOS)

CONTROLAR INGRESO DE PRODUCTOS



CONTROLAR SALIDA DE PRODUCTOS



AMORTIZACIONES (CONTROL DE PAGOS)

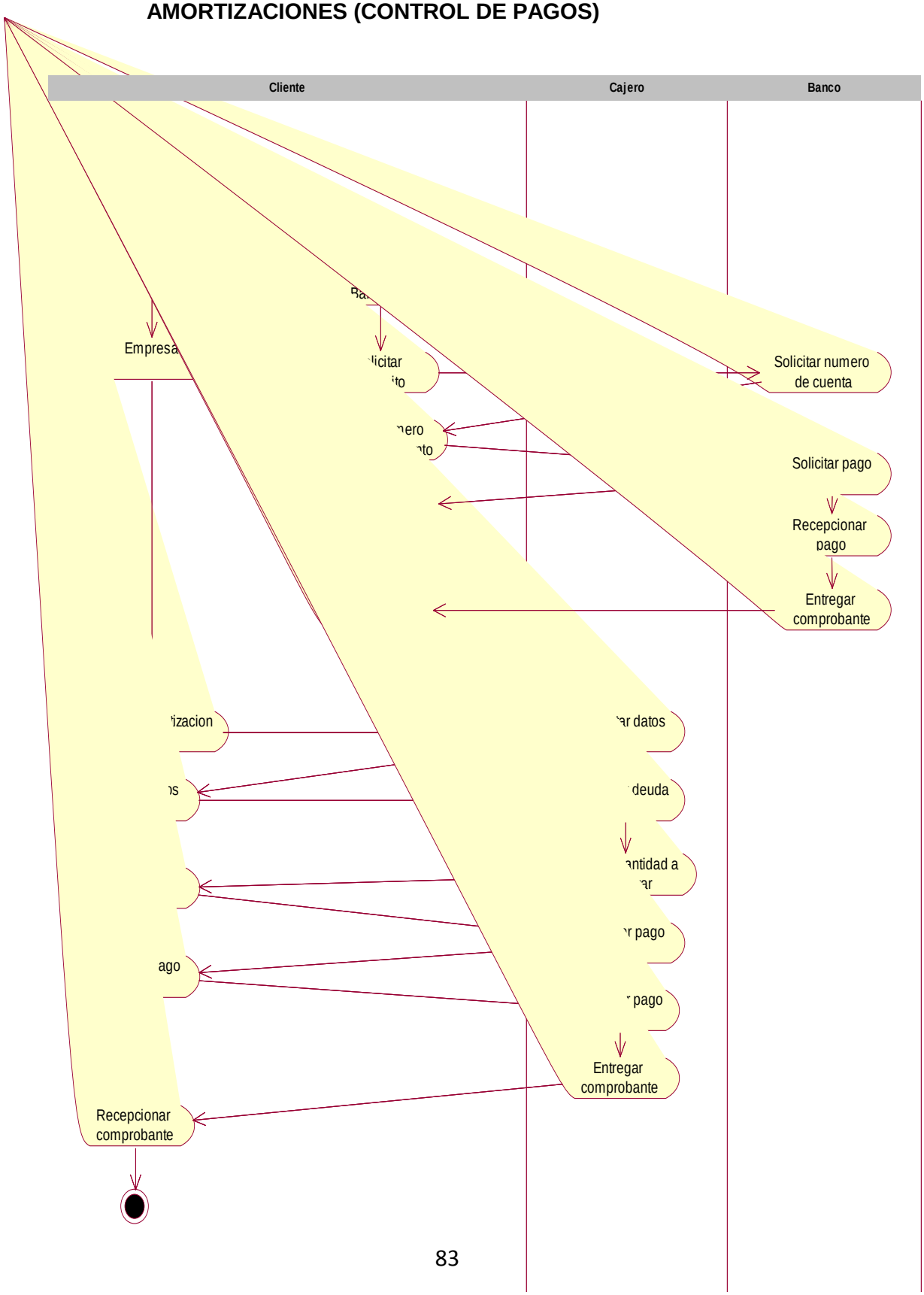
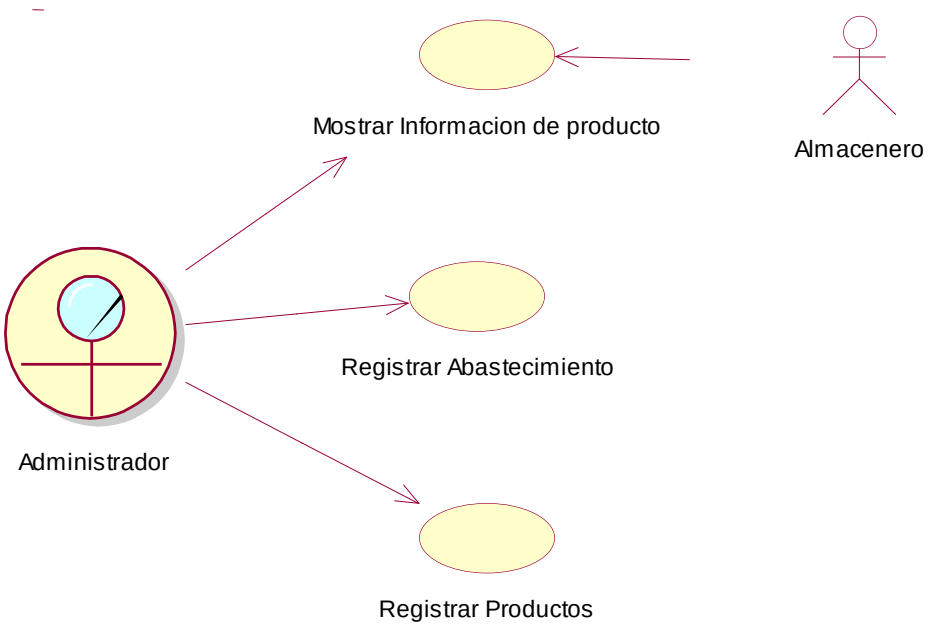
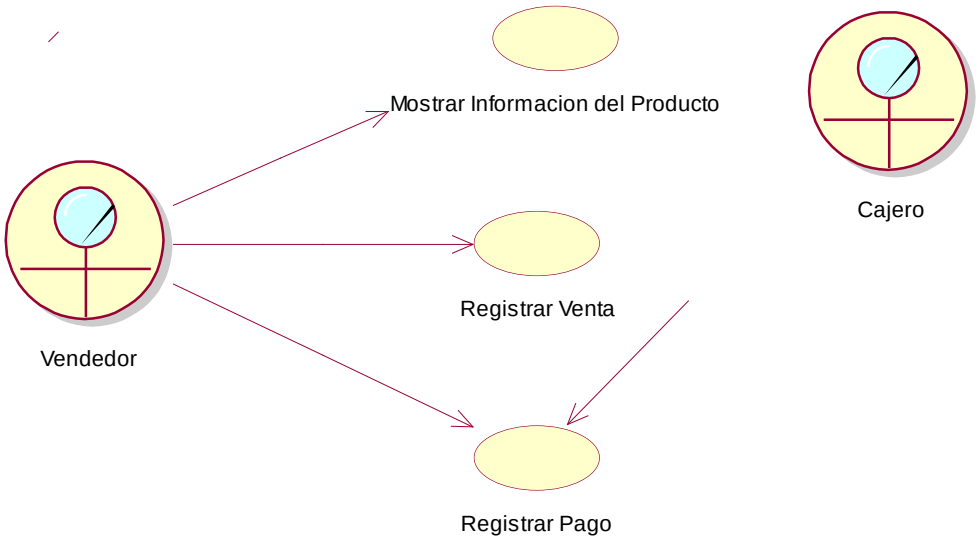


DIAGRAMA DE CASOS DE USO DEL SOFTWARE

COMPRA

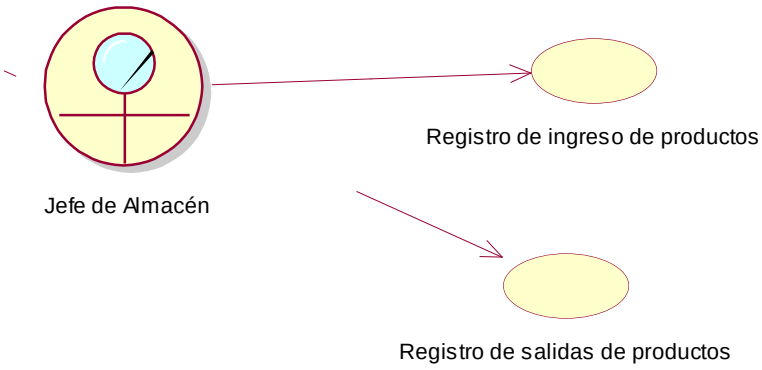


VENTA

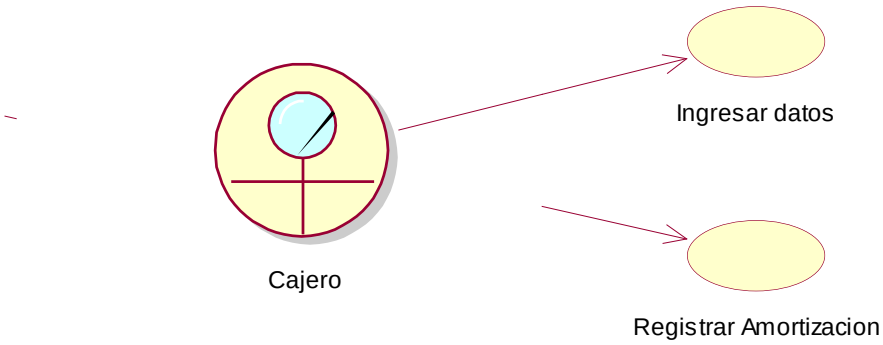


KARDEX (CONTROL DE PRODUCTOS)

CONTROLAR INGRESO/SALIDA DE PRODUCTO



AMORTIZACIONES (CONTROL DE PAGOS)



PLANTILLA DE ESPECIFICACIÓN:

COMPRA

Generar informe

Nombre	Mostrar información del Stock
Actor	Jefe de Almacén
Pre Condición	Producto registrados
Post Condición	Jefe de Almacén entrega informe

Flujo Básico

1. Jefe de Almacén ingresa al sistema
2. indica generar informe
3. El sistema muestra formulario de "Informe de Producto"
4. Se busca datos del producto
5. Ingresa los datos
6. Almacenero indica guardar e imprimir

Registrar Abastecimiento

Nombre	Registrar Abastecimiento
Actor	Administrador
Pre Condición	
Post Condición	

Flujo Básico

1. Administrador ingresa al sistema
2. indica registrar compra
3. Es sistema muestra formulario registrar Compra
4. Administrador ingresa datos de la compra
5. Sistema genera código de registro
6. Administrador indica registrar y guardar

REGISTRAR PRODUCTO

Nombre	Registrar Producto
Actor	Administrador
Pre Condición	
Post Condición	Producto registrado

Flujo Básico

1. Administrador ingresa al sistema
2. indica registrar producto
3. El sistema muestra formulario registrar producto
4. Administrador ingresa datos del producto
5. Sistema genera código del producto
6. Administrador indica registrar y guardar

VENTA

MOSTRAR INFORMACION PRODUCTO

Nombre	Mostrar información del producto
Actor	Vendedor
Pre Condición	Producto registrado
Post Condición	

Flujo Básico

1. vendedor ingresa al sistema
2. indica informe del producto
3. Es sistema muestra formulario informe de producto
4. Vendedor busca datos de producto
5. Lee los datos del producto
6. Vendedor indica salir

REGISTRAR VENTA

Nombre	Registrar datos de la venta
Actor	Vendedor
Pre Condición	producto registrado
Post Condición	Venta registrada

Flujo Básico

1. El vendedor ingresa al sistema
2. Indica registrar venta
3. El sistema muestra formulario registrar ventas
4. Formulario muestra datos del cliente
5. Se busca datos del producto para asignar a la venta
6. Formulario muestra datos del producto
7. Vendedor ingresa datos de la venta
8. Sistema genera código de venta
9. Se indica registrar y guardar

REGISTRAR PAGO

Nombre	Registrar Pago
Actor	Cajero
Pre Condición	Venta registrada
Post Condición	Pago registrado

Flujo Básico

1. Vendedor ingresa al sistema
2. Indica registrar pago
3. Sistema muestra formulario registrar pago
4. Se busca datos de la venta
5. Formulario muestra datos de la venta

6. **Vendedor ingresa datos del pago**
7. **Sistema genera código de pago**
8. **Se indica guardar y registrar**

KARDEX

REGISTRO DE INGRESO DE PRODUCTOS

Nombre	Registro de Ingreso de Productos
Actor	Jefe de Almacén
Pre Condición	
Post Condición	

Flujo Básico

1. **Jefe de Almacén ingresa al sistema**
2. **Indica registrar Ingreso**
3. **Sistema muestra formulario Registro de Ingreso de Productos**
4. **Jefe de Almacén ingresa datos de la entrada**
5. **Sistema generar código de ingreso**
6. **Jefe de Almacén indica registrar y guardar**

REGISTRO DE SALIDA DE PRODUCTOS

Nombre	Registro de Salida de Productos
Actor	
Pre Condición	producto registrado
Post Condición	

Flujo Básico

1. **Jefe de Almacén ingresa al sistema**
2. **Indica registrar Ingreso**
3. **Sistema muestra formulario Registro de Salidas de Productos**
4. **Jefe de Almacén ingresa datos de la salida**
5. **Sistema generar código de Salida**
6. **Jefe de Almacén indica registrar y guardar**

AMORTIZACIÓN

REGISTRAR AMORTIZACIÓN

Nombre	Registrar Amortización
Actor	Cajero
Pre Condición	Cliente – producto registrado
Post Condición	Pedido registrado

Flujo Básico

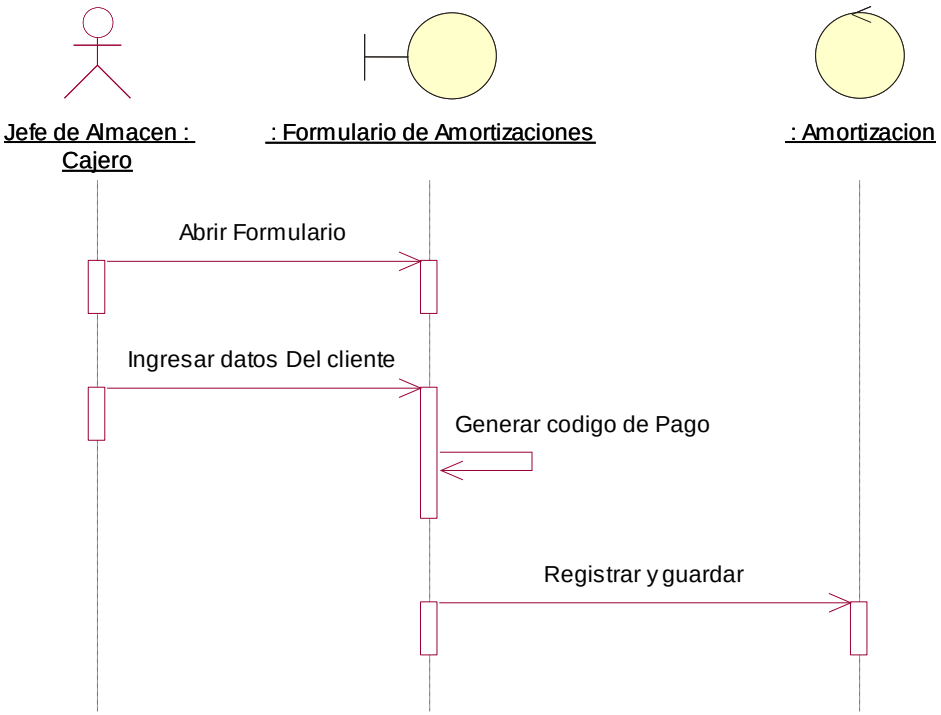
1. Cajero ingresa al sistema
2. Indica registrar Amortizacion
3. Sistema muestra formulario de Amortizacion (Pagos)
4. Se busca datos de la venta
5. Formulario muestra datos de la venta
6. Vendedor ingresa datos del pago
7. Sistema genera código de pago
8. Se indica guardar y registrar

DIAGRAMA DE INTERACCIÓN POR CADA CASO DE USO

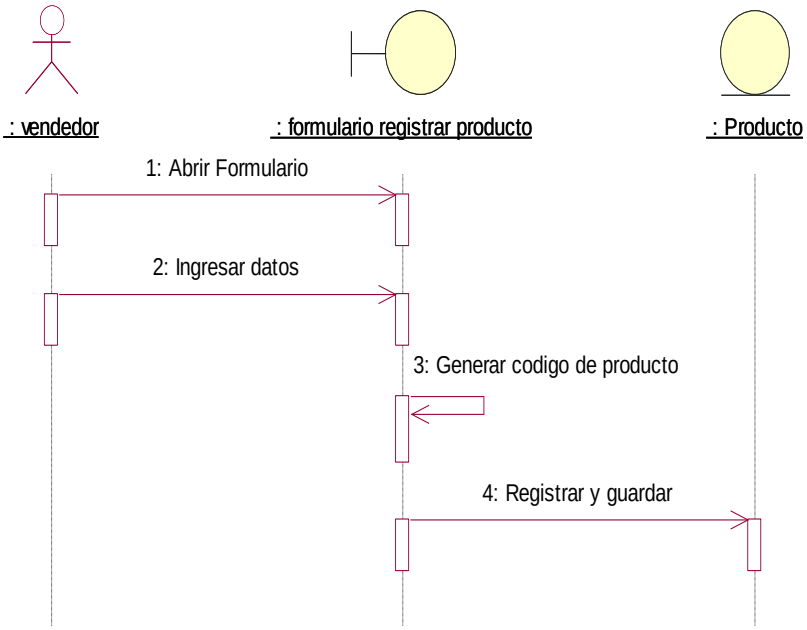
DIAGRAMA DE SECUENCIA

COMPRA

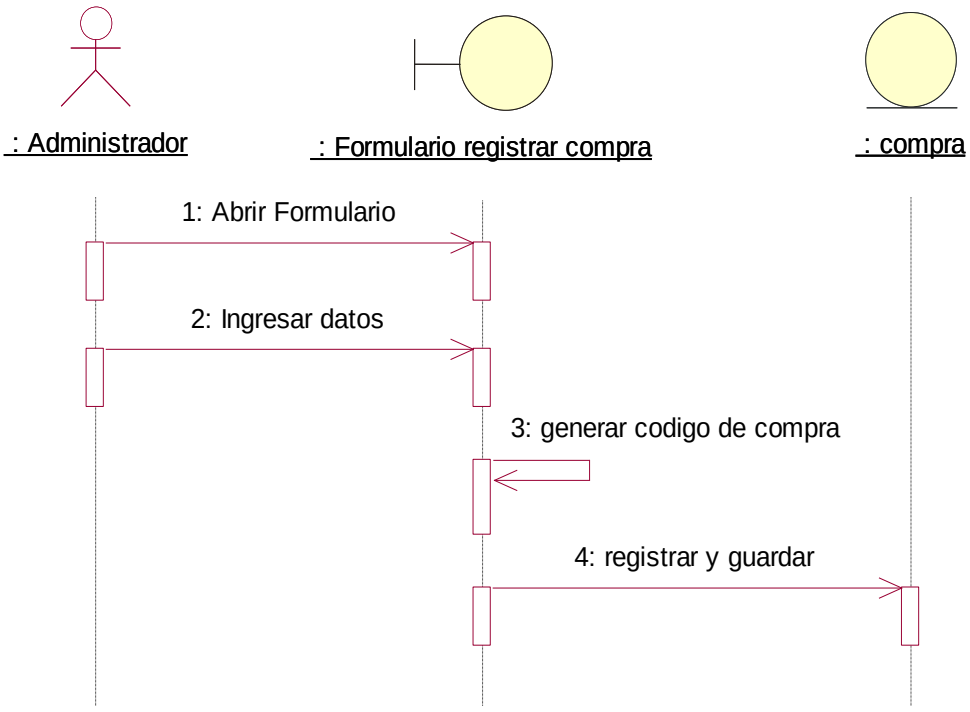
GENERAR INFORME



REGISTRAR PRODUCTOS

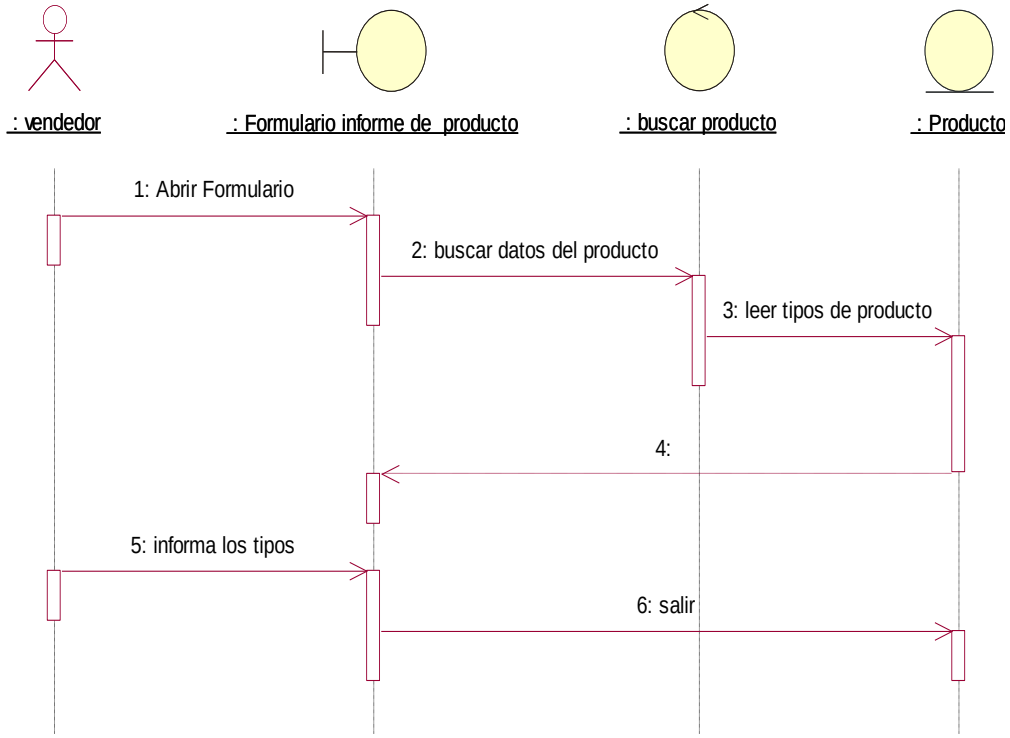


REGISTRAR ABASTECIMIENTO

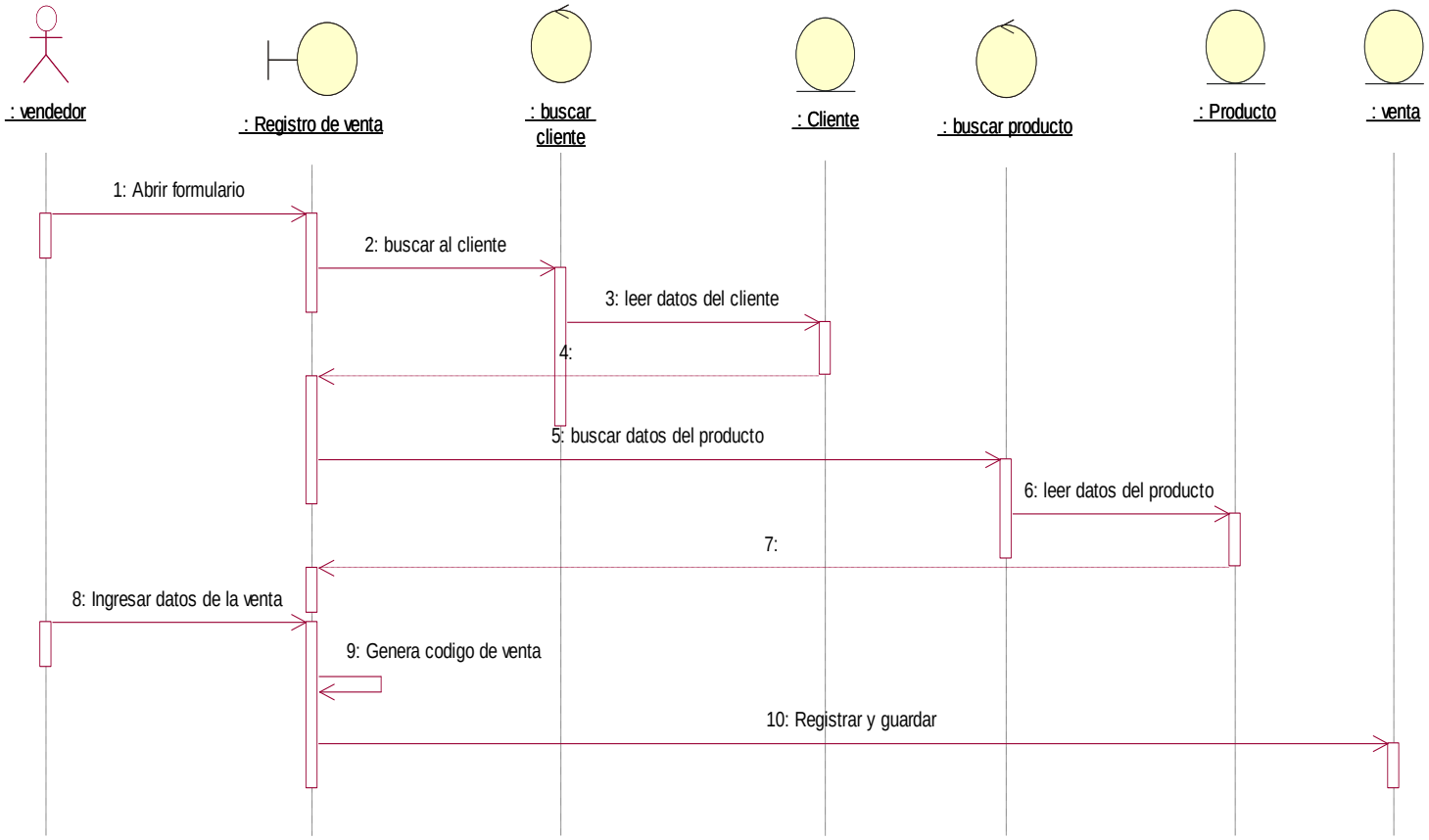


VENTA

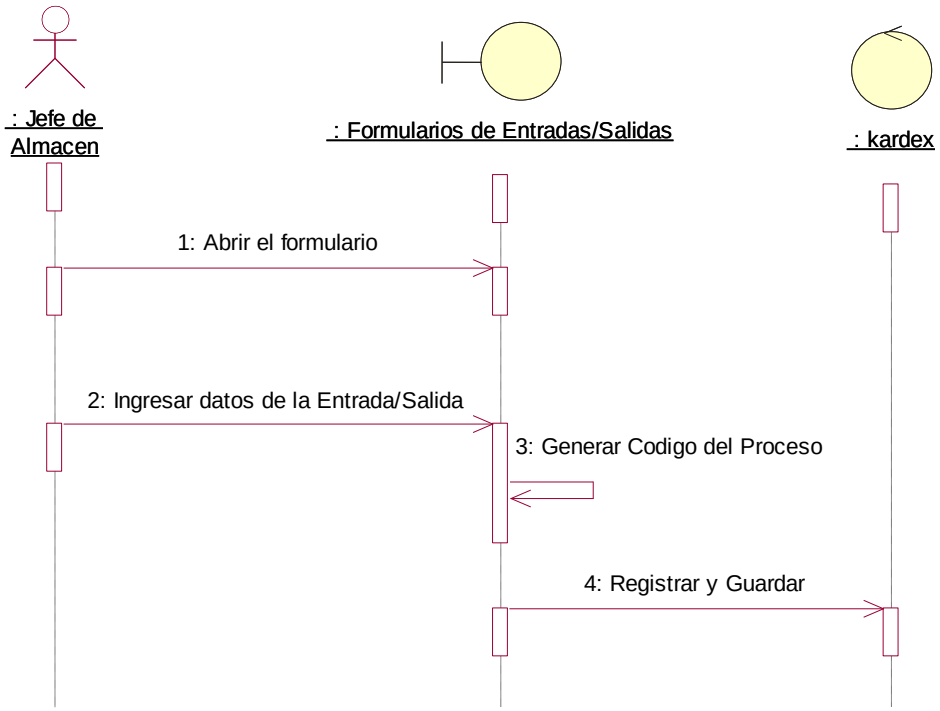
INFORMAR TIPOS DE PRODUCTO



REGISTRAR VENTA



KARDEX



AMORTIZACIONES

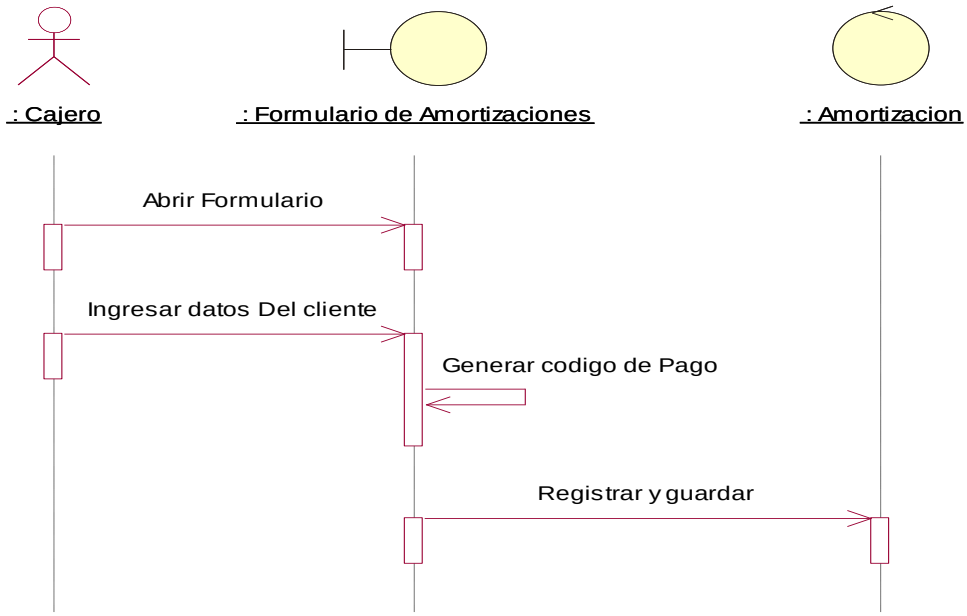
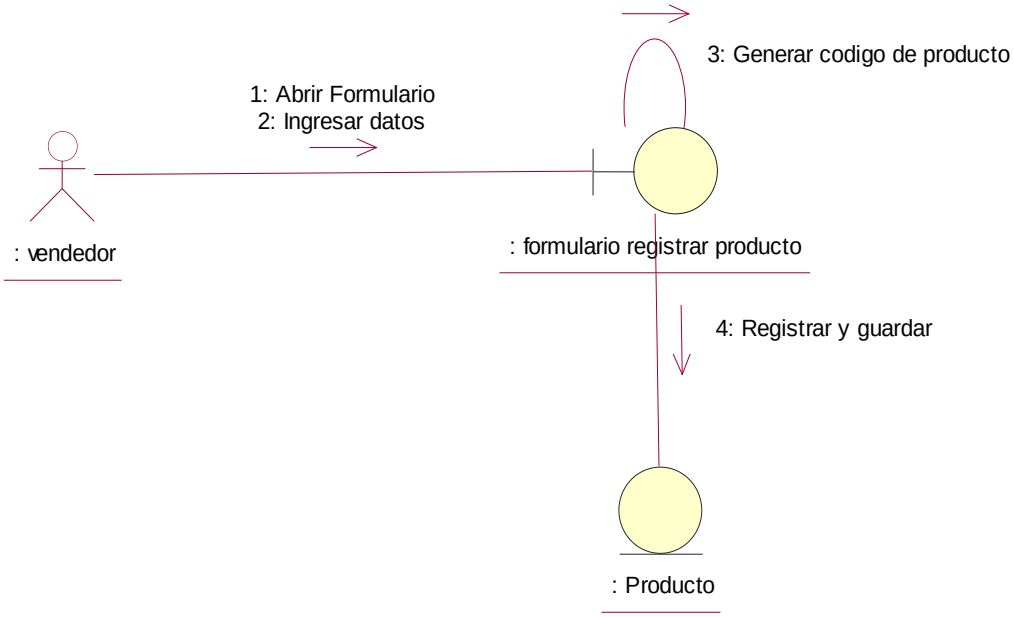


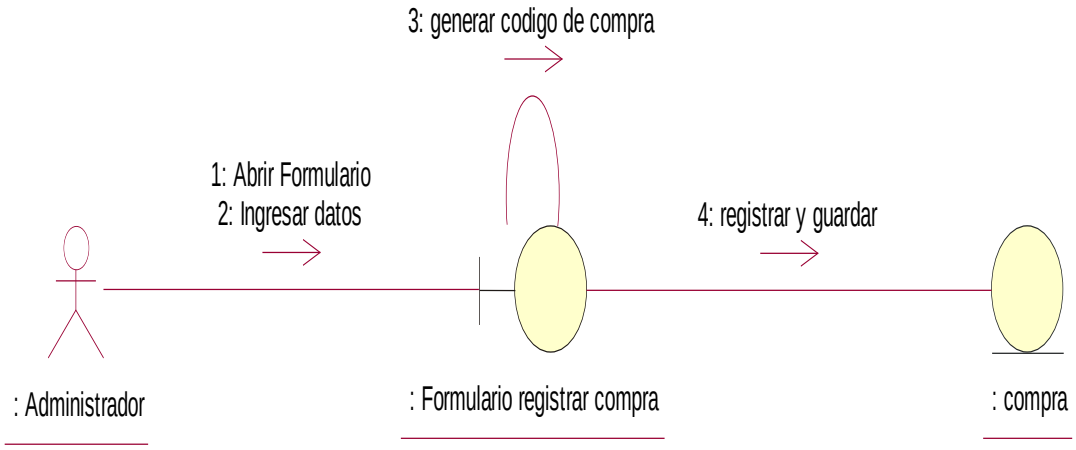
DIAGRAMA DE COLABORACIÓN

COMPRA

REGISTRAR PRODUCTO

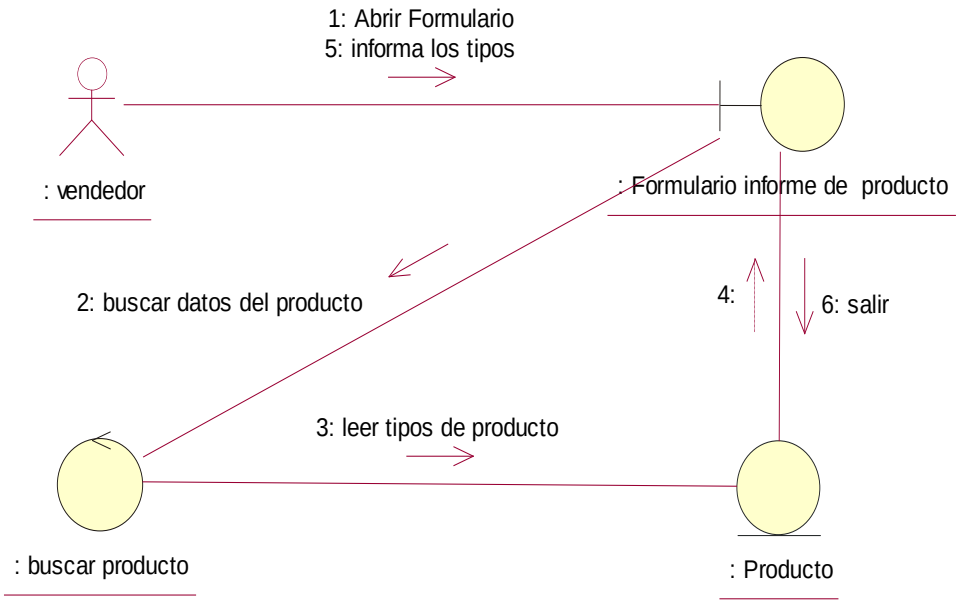


REGISTRAR COMPRA



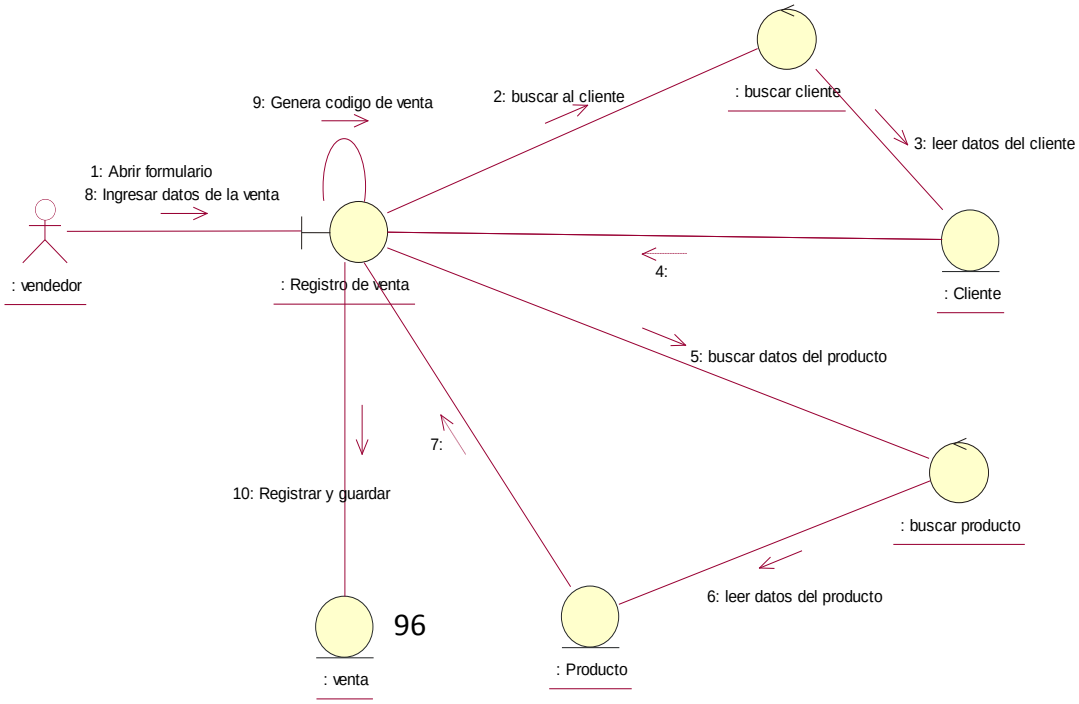
VENTA

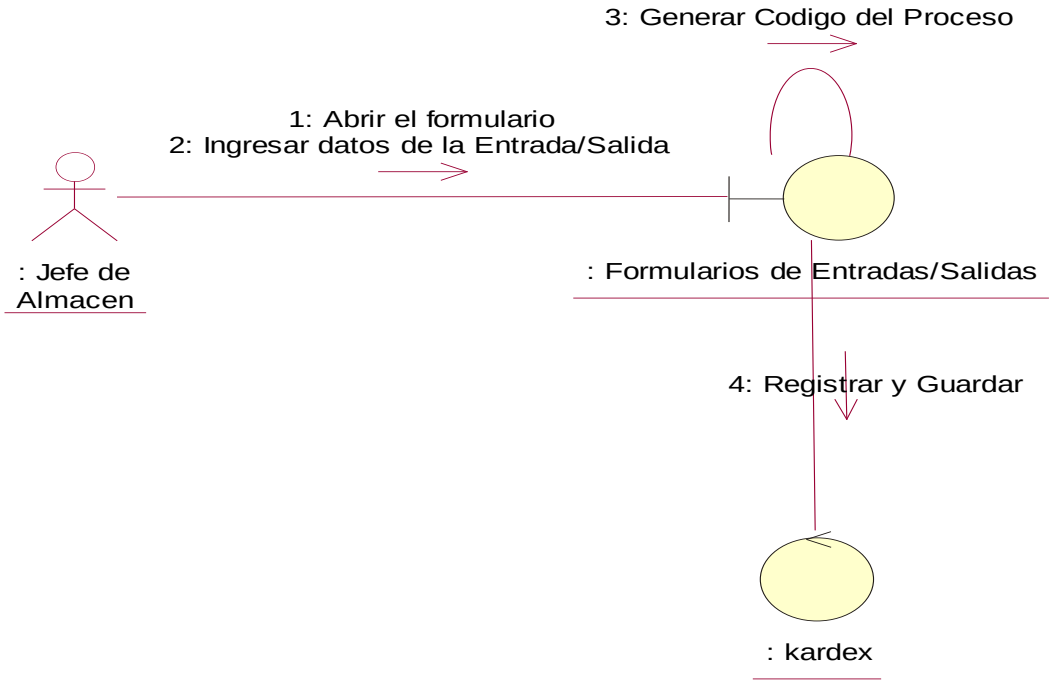
INFORMAR TIPOS DE PRODUCTO



REGISTRO DE VENTAS

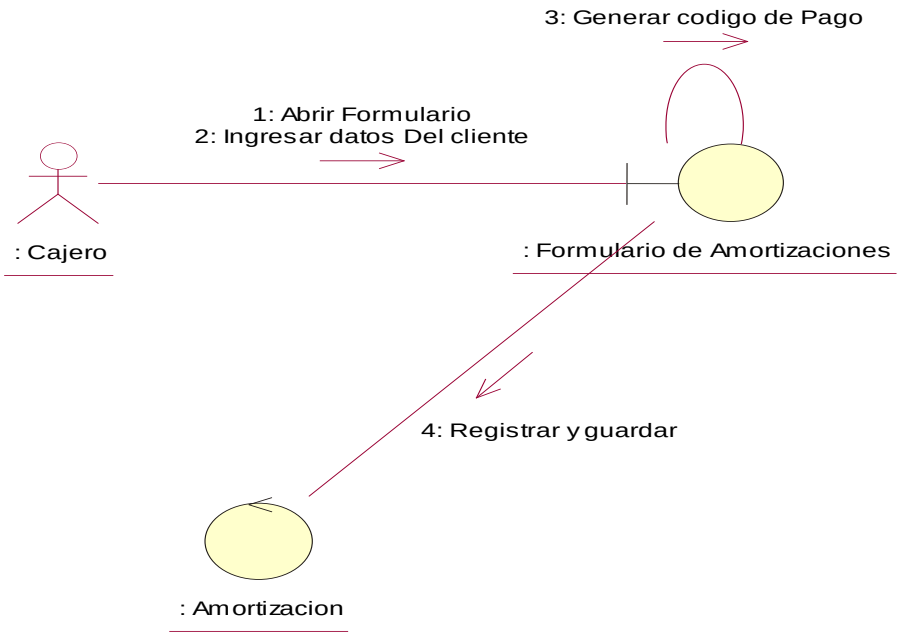
KARDEX



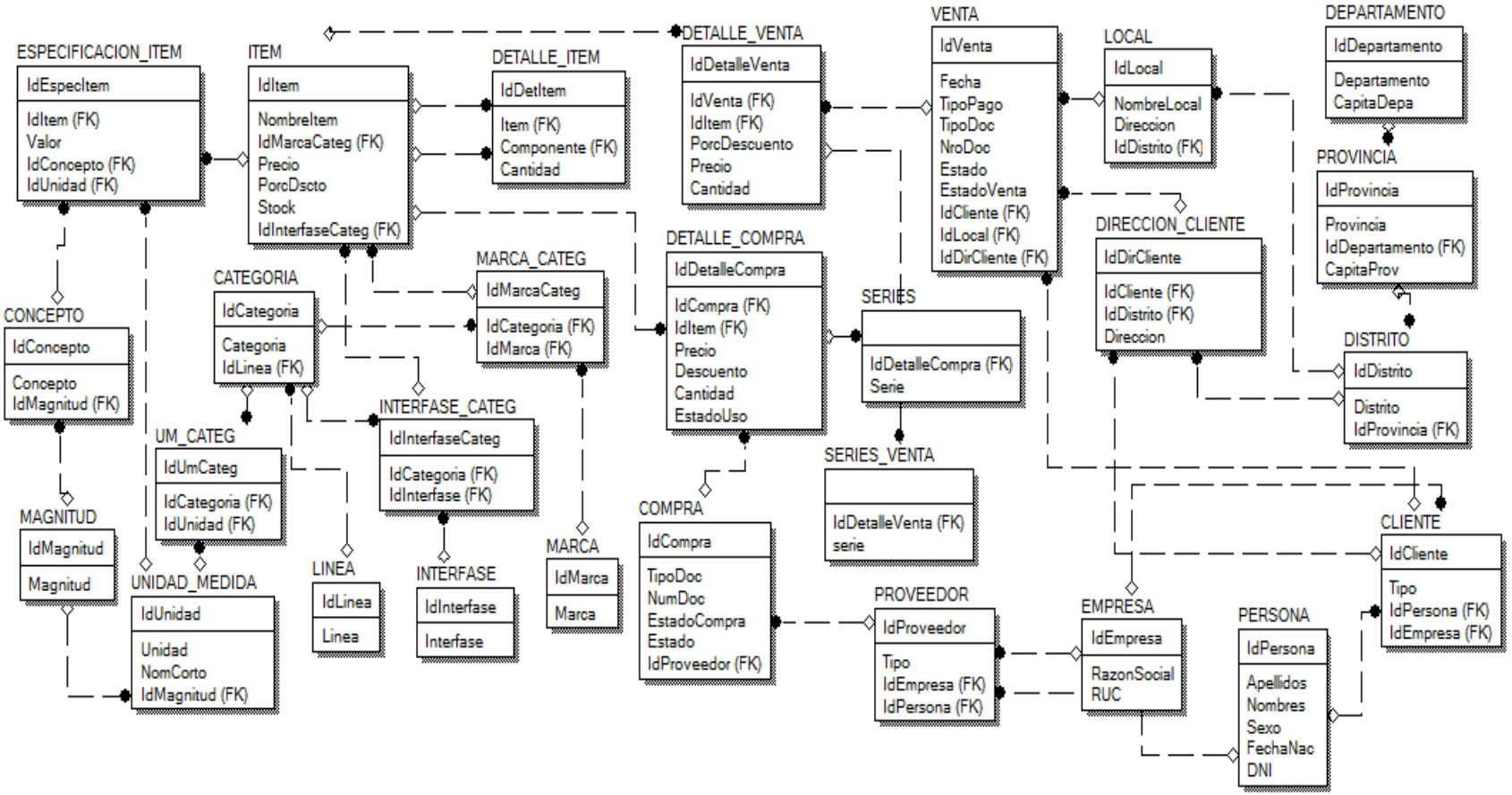


AMORTIZACIÓN

REGISTRAR AMORTIZACIÓN



MODELADO DE LA BASE DE DATO

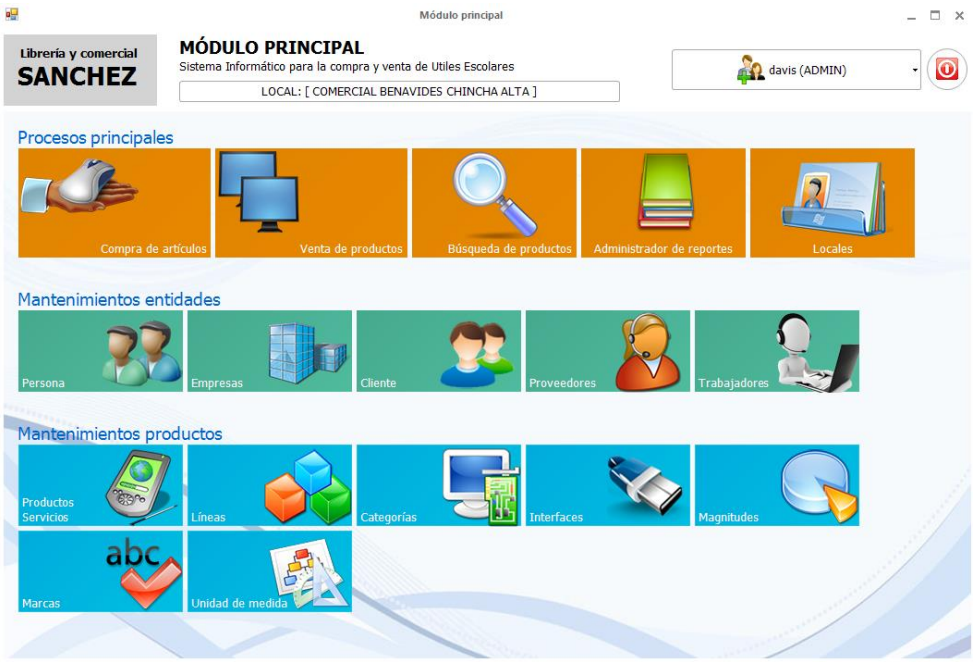


FASE DE DISEÑO DE FORMULARIOS

INGRESO AL SISTEMA

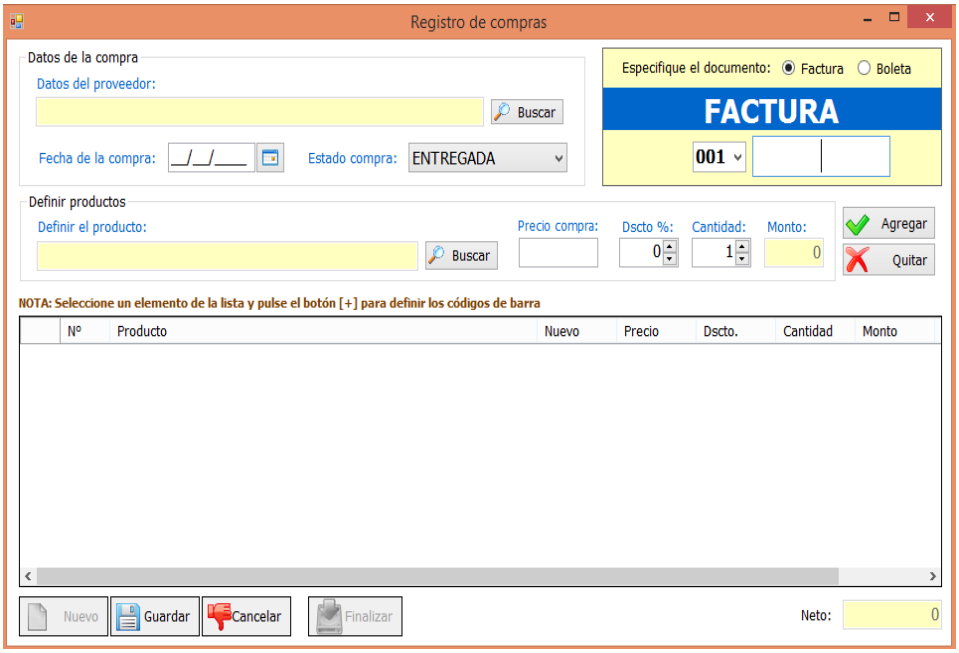
- PROTOTIPO- ACCESO DE SEGURIDAD

- MENÚ PRINCIPAL

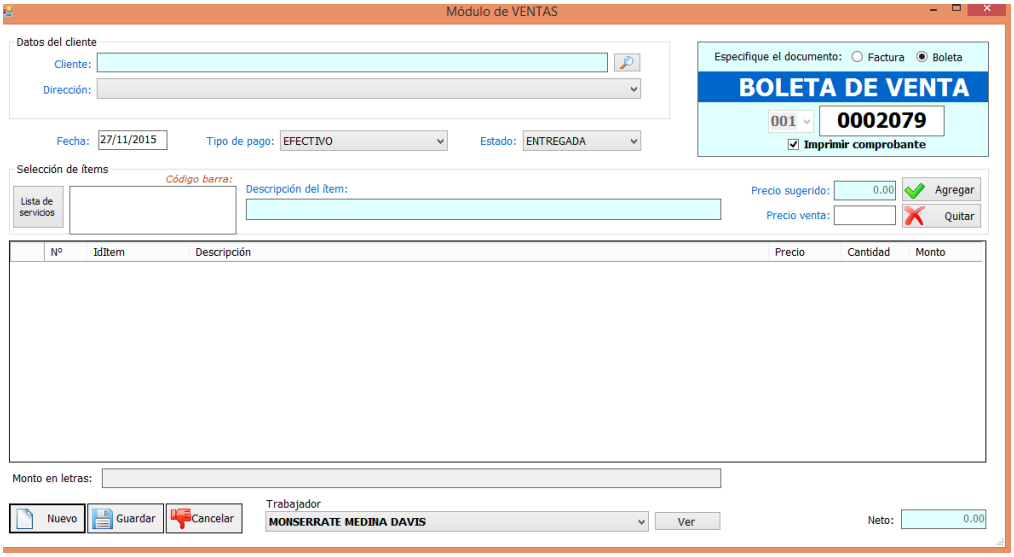


PROTOTIPO DE PROCESOS:

- COMPRA DE ARTÍCULOS:

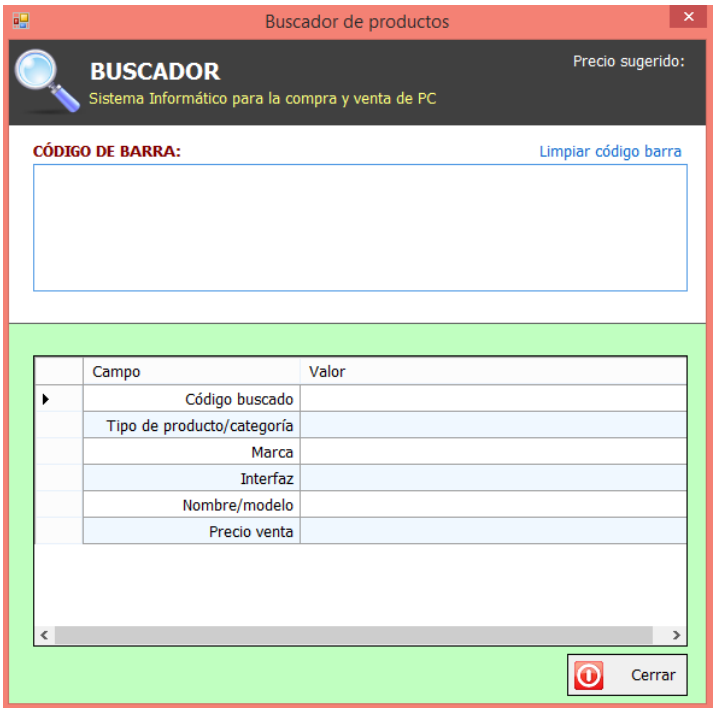


- VENTA DE PRODUCTOS



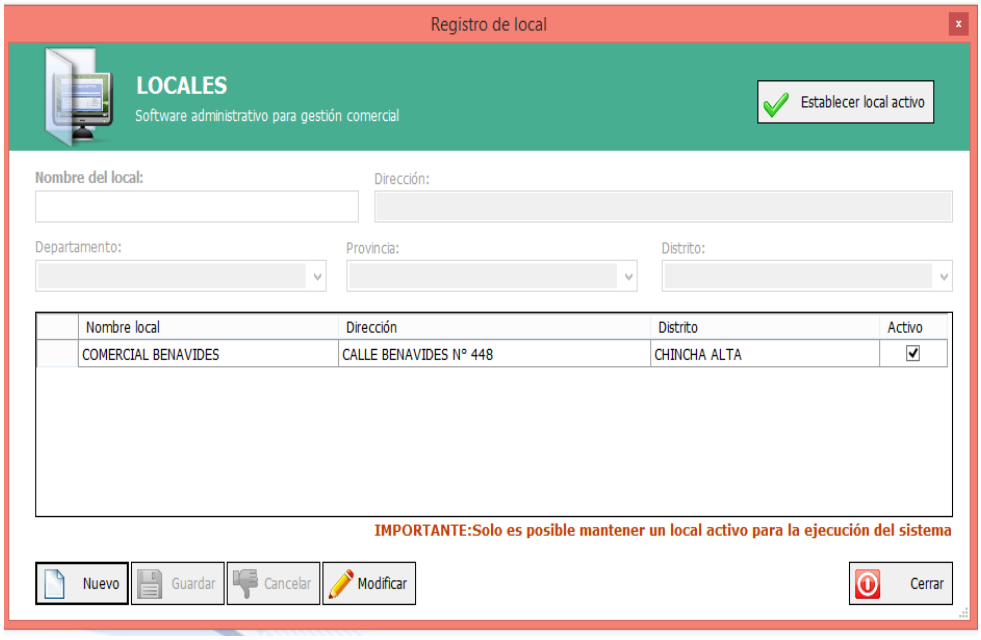
- AMORTIZACIONES

- BÚSQUEDA DE PRODUCTOS



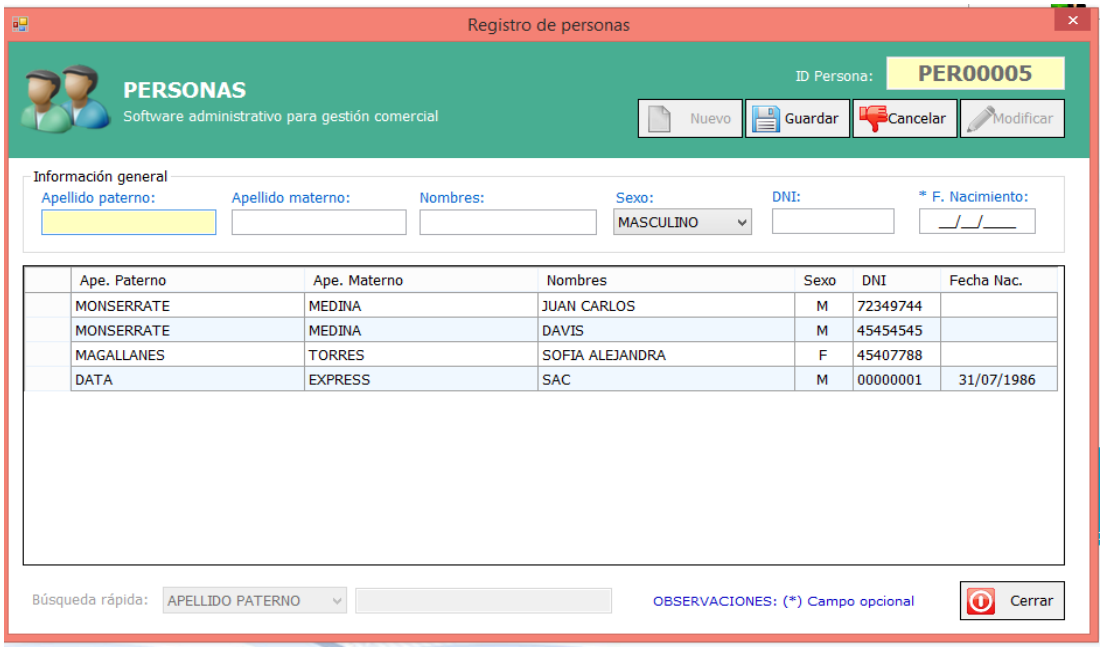
- ADMINISTRADOR DE REPORTES

- LOCALES

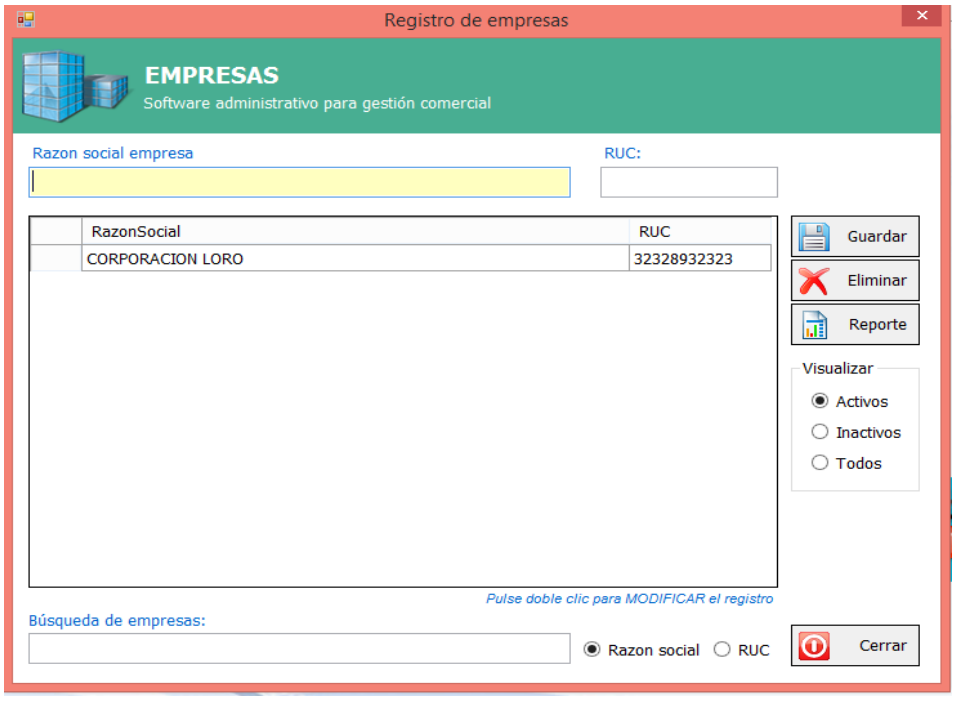


PROTOTIPO DE MANTENIMIENTOS DE ENTIDADES:

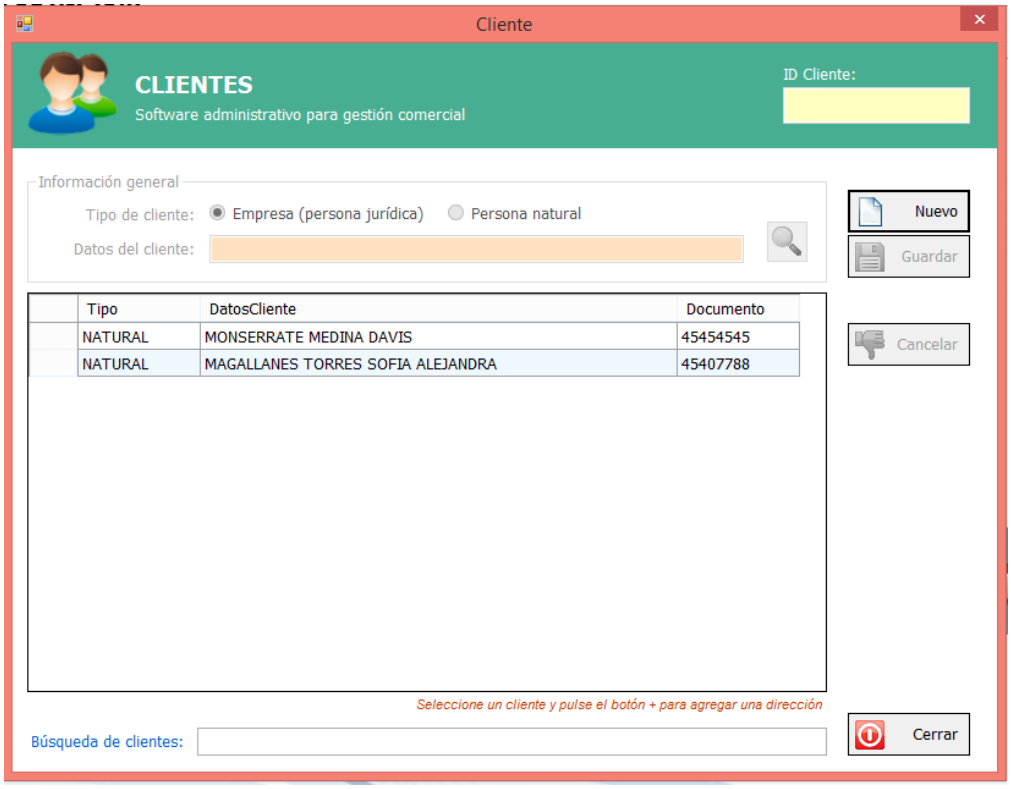
- PERSONA



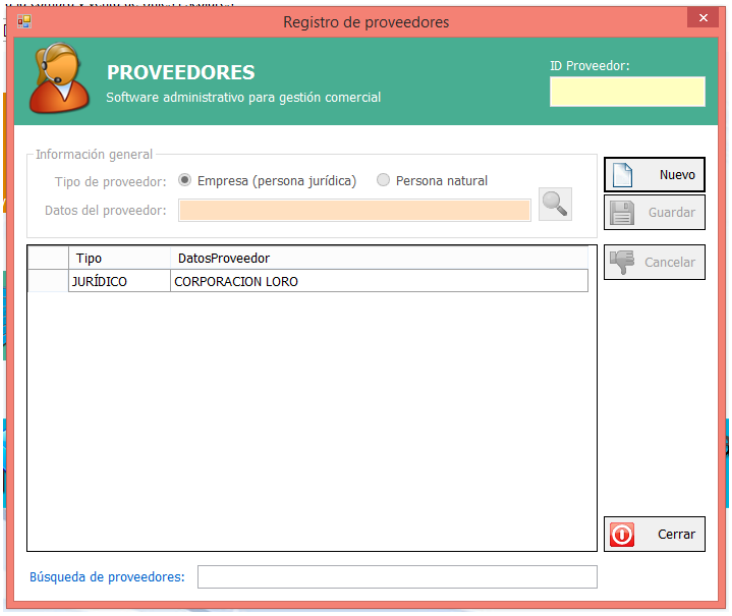
- EMPRESAS



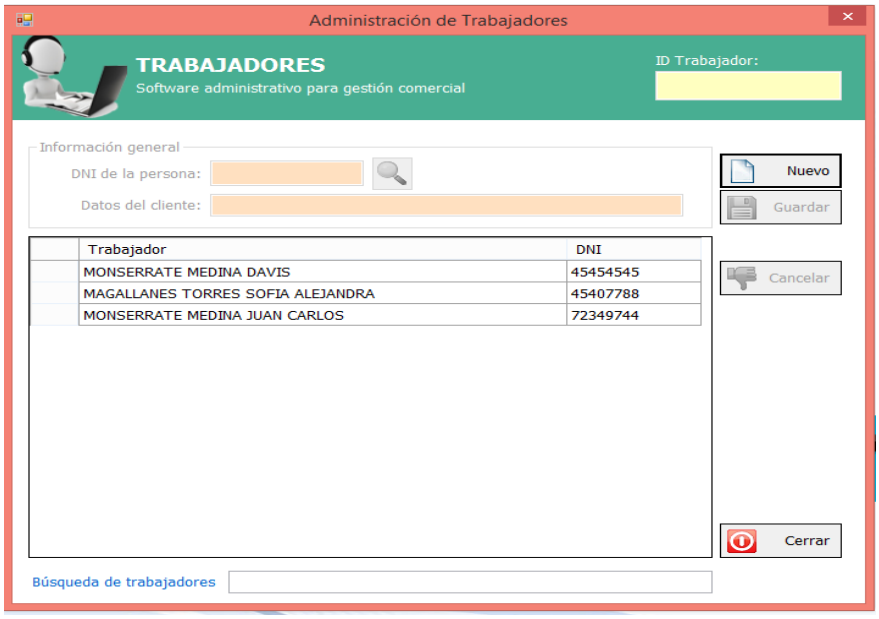
- CLIENTES



- PROVEEDORES

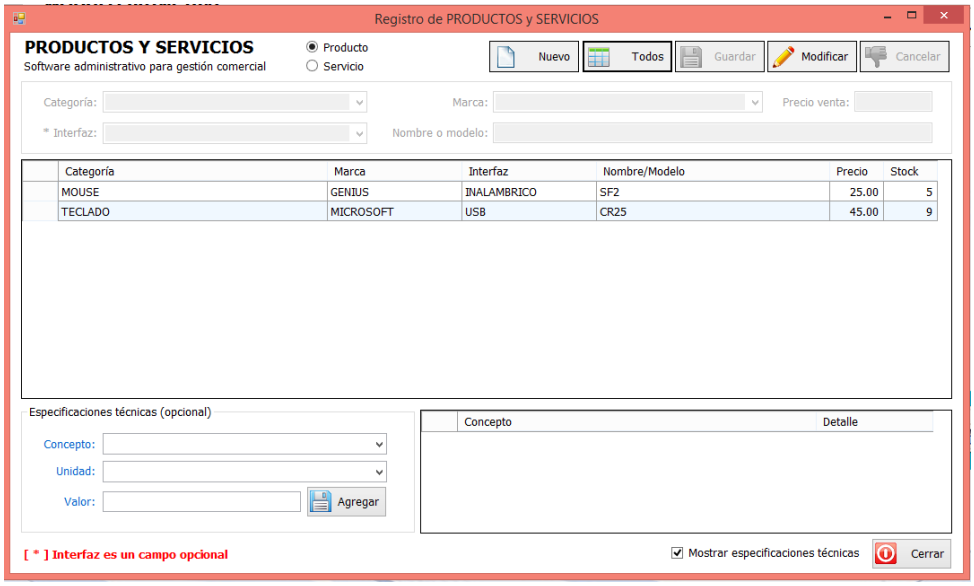


- TRABAJADORES



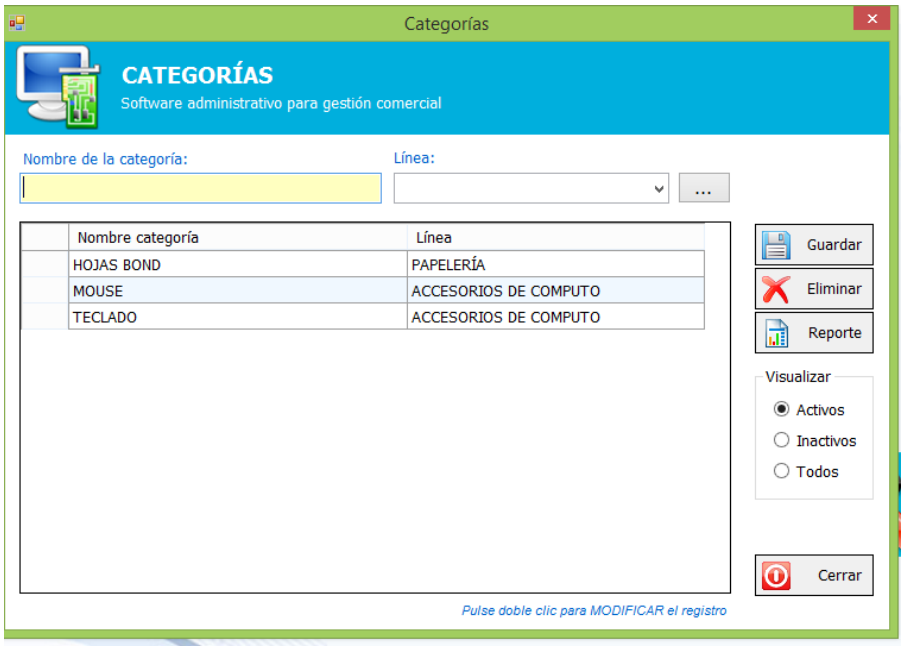
PROTOTIPO DE MANTENIMIENTO DE PRODUCTOS:

- PRODUCTOS Y SERVICIOS

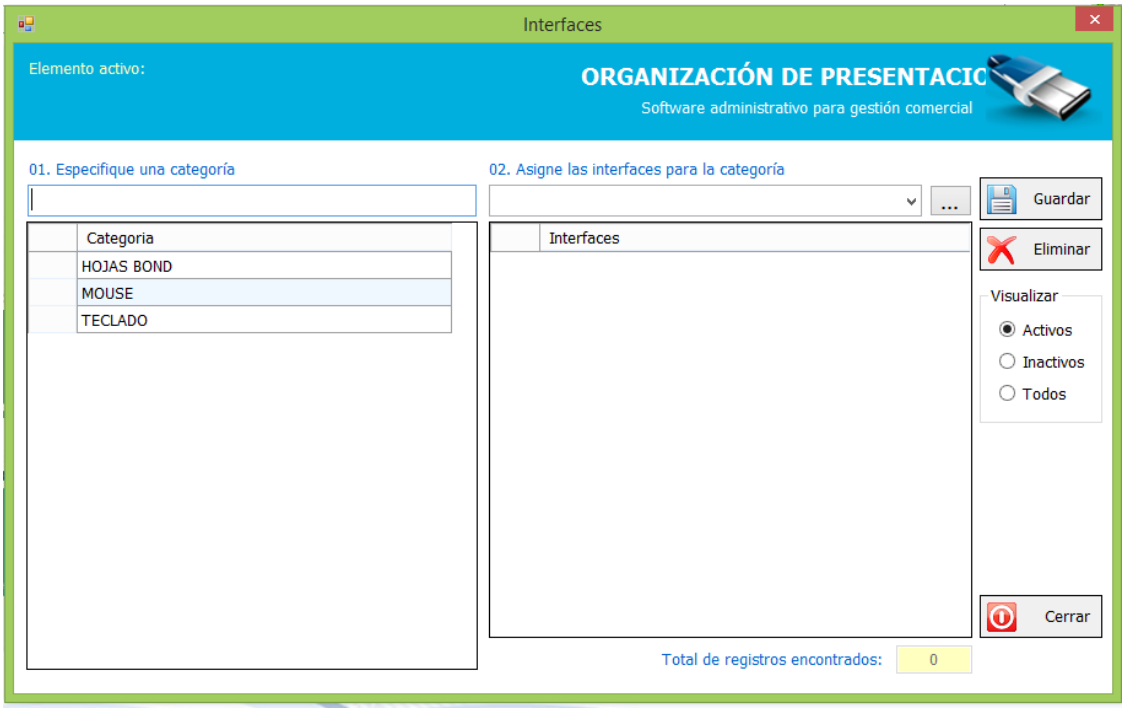


- LÍNEAS

- CATEGORÍAS

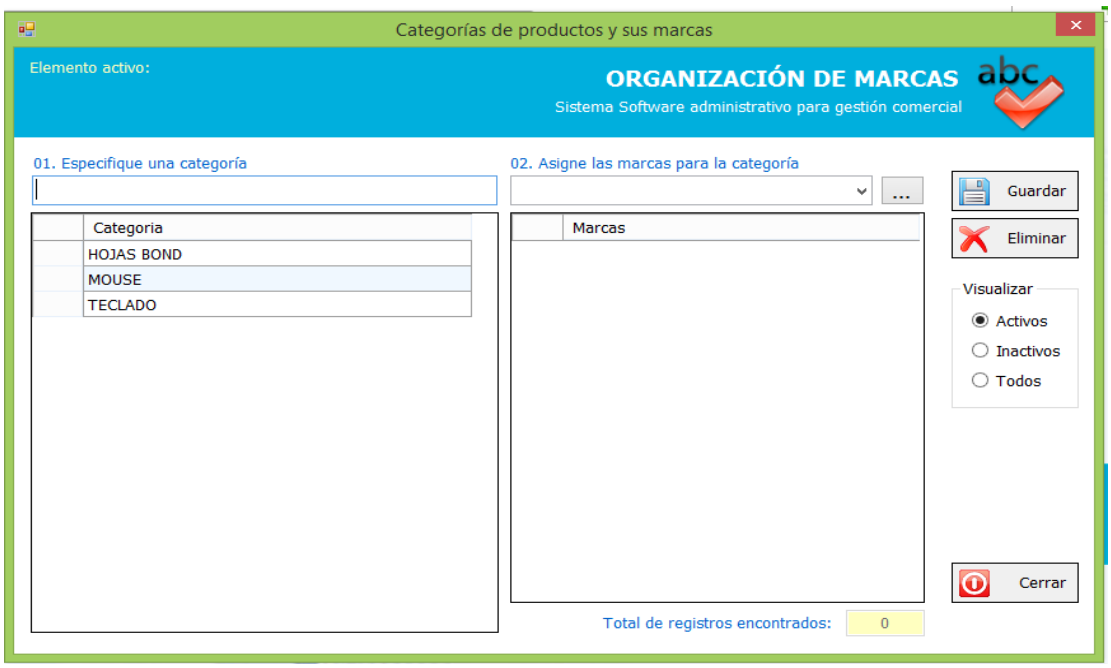


- INTERFACES

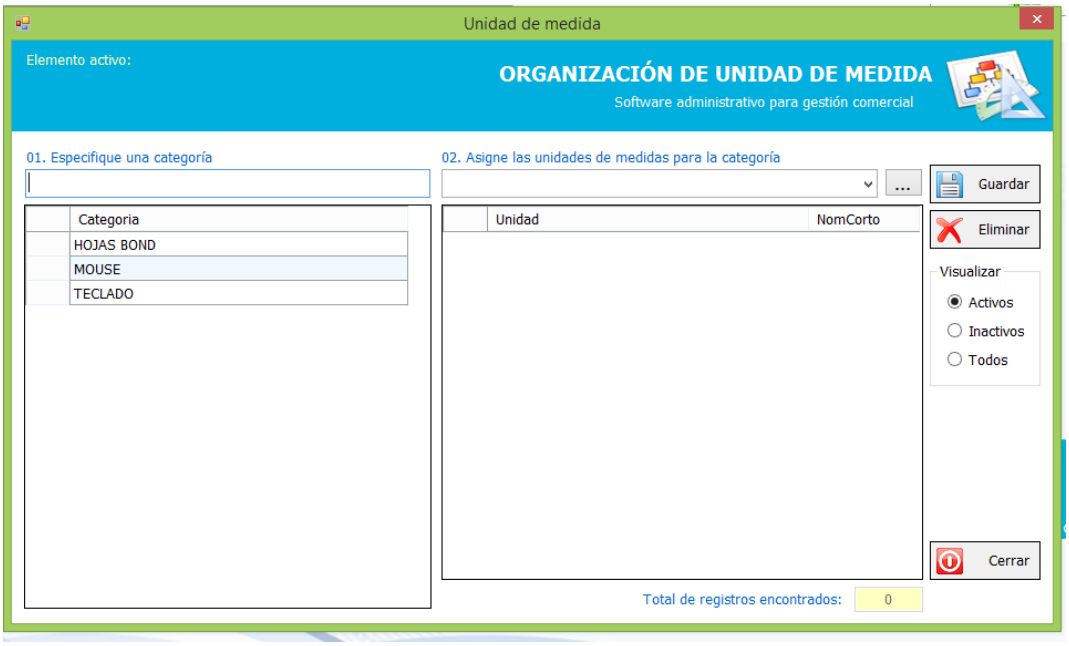


- MAGNITUDES

- MARCAS



- UNIDAD DE MEDIDA



FASE DE PROGRAMACIÓN DEL SISTEMA

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Procesos
{
   public partial class frmCompra : Form
   {
      Compra objCompra = new Compra();

      double precio = 0, dscto = 0, cantidad = 0, monto = 0;
      string idItem = "", idcompra = "", tmp = "";

      //Declaración de foráneos
      string idProveedor = "";
      char tdoc = 'F', EstEntrega = 'E';

      public frmCompra()
      {
         InitializeComponent();
      }

      private void btnProveedor_Click(object sender, EventArgs e)
      {
         Mantenimientos.frmProveedores f = new Mantenimientos.frmProveedores();
         f.ShowDialog();
      }

      private void btnCalendario_Click(object sender, EventArgs e)
      {
         Calendario.Visible = !Calendario.Visible;
      }
```

```
        private void Calendario_DateChanged(object sender, DateRangeEventArgs e)
        {
            txtFechaCompra.Text = Calendario.SelectionStart.ToShortDateString();
        }

        private void Calendario_Leave(object sender, EventArgs e)
        {
            Calendario.Visible = false;
        }

        private void Calendario_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (e.KeyChar == Convert.ToChar(Keys.Escape) || e.KeyChar ==
Convert.ToChar(Keys.Enter))
            {
                Calendario.Visible = false;
            }
        }

        private void grid_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (grid.Rows.Count > 0)
            {
                if (e.KeyChar == '+')
                {
                    if (grid.CurrentRow.Cells[8].Value.ToString() == string.Empty)
                    {
                        Sistema.informar("Guarde este registro de compra para añadir los
códigos");
                    }
                    else
                    {
                        frmRegistroSerie f = new frmRegistroSerie();

                        f.txtProducto.Text = grid.CurrentRow.Cells[2].Value.ToString();
                        f.txtCantidad.Text = grid.CurrentRow.Cells[6].Value.ToString();

                        for (int i = 1; i <=
Convert.ToInt32(grid.CurrentRow.Cells[6].Value.ToString()); i++)
                        {
                            f.gridCodigos.Rows.Add(i.ToString(),
grid.CurrentRow.Cells[8].Value.ToString(), "");
                        }
```

```
                f.ShowDialog();
            }
        }
    }
}

private void optFactura_CheckedChanged(object sender, EventArgs e)
{
    tdoc = 'F';
    lblTipoDoc.Text = "FACTURA";
}

private void optBoleta_CheckedChanged(object sender, EventArgs e)
{
    tdoc = 'B';
    lblTipoDoc.Text = "BOLETA DE VENTA";
}

private void btnBuscarProv_Click(object sender, EventArgs e)
{
    Mantenimientos.frmProveedores f = new Mantenimientos.frmProveedores();
    f._Sub = true;

    if (f.ShowDialog() == DialogResult.OK)
    {
        if (f.grid.CurrentRow != null)
        {
            idProveedor = f.grid.CurrentRow.Cells[0].Value.ToString();
            txtProveedor.Text = f.grid.CurrentRow.Cells[2].Value.ToString();
            txtFechaCompra.Focus();
        }
        else
        {
            txtProveedor.Clear();
        }
    }
}

private void AdControles(bool sw)
{
    BaseTipoDoc.Enabled = sw;
    gDatos.Enabled = sw;
    gProductos.Enabled = sw;
```

```
            btnAgregar.Enabled = sw;
            btnQuitar.Enabled = sw;
            lblAviso.Enabled = sw;
            grid.Enabled = sw;
            btnNuevo.Enabled = !sw;
            btnGuardar.Enabled = sw;
            btnCancelar.Enabled = sw;
        }

        private void Limpiar()
        {
            optFactura.Checked = true;
            cboSerie.SelectedIndex = 0;
            txtNumeroDoc.Clear();
            txtProveedor.Clear();
            idProveedor = "";
            txtFechaCompra.Clear();
            Calendario.Visible = false;
            cboEstadoCompra.SelectedIndex = 2;
            txtProducto.Clear();
            chkNuevo.Checked = true;
            txtPrecio.Clear();
            txtDscto.Value = 0;
            txtCantidad.Value = 1;
            txtMonto.Text = "0";
            grid.Rows.Clear();
            txtNeto.Text = "0";
            idItem = "";
            precio = 0; dscto = 0; cantidad = 0; monto = 0;
        }

        private void frmCompra_Load(object sender, EventArgs e)
        {
            toolTip.SetToolTip(txtFechaCompra,"dd/mm/aaaa");
            toolTip.SetToolTip(btnFinalizar,"Pulse este botón para iniciar otro proceso de
compra");
            toolTip.SetToolTip(txtPrecio,"Precio de compra del producto");
            //toolTip.SetToolTip();

            cboEstadoCompra.SelectedIndex = 2;
            EstEntrega = 'E';

            grid.Columns[1].Visible = false;
            //grid.Columns[8].Visible = false;
```

```
    this.AdControles(false);
}

private void txtPrecio_KeyPress(object sender, KeyPressEventArgs e)
{
    Sistema.soloNumeros(sender, e, 'D');
}

private void txtDescuento_KeyPress(object sender, KeyPressEventArgs e)
{
    Sistema.soloNumeros(sender, e, 'D');
}

private void CalcularMonto()
{
    if (txtPrecio.Text.Trim() != string.Empty)
    {
        precio = Convert.ToDouble(txtPrecio.Text);
        dscto = Convert.ToDouble(txtDscto.Value);
        cantidad = Convert.ToDouble(txtCantidad.Value);
        monto = precio * cantidad - ((precio * cantidad) * dscto / 100);
        txtMonto.Text = monto.ToString();
    }
}

private void txtCantidad_ValueChanged(object sender, EventArgs e)
{
    this.CalcularMonto();
}

private void txtDscto_ValueChanged(object sender, EventArgs e)
{
    this.CalcularMonto();
}

private void btnBuscarProd_Click(object sender, EventArgs e)
{
    frmItem f = new frmItem();
    f._Sub = true;

    if (f.ShowDialog() == DialogResult.OK)
    {
        if (f.gridItems.CurrentRow != null)
        {
```

```
                idItem = f.gridItems.CurrentRow.Cells[0].Value.ToString();
                txtProducto.Text = f.gridItems.CurrentRow.Cells[1].Value.ToString() + " " +
f.gridItems.CurrentRow.Cells[2].Value.ToString() + " " +
f.gridItems.CurrentRow.Cells[3].Value.ToString() + " " +
f.gridItems.CurrentRow.Cells[4].Value.ToString();
                txtPrecio.Focus();
            }
            else
            {
                idItem = "";
                txtProducto.Clear();
            }
        }
    }

    private void txtPrecio_TextChanged(object sender, EventArgs e)
    {
        this.CalcularMonto();
    }

    private void CalcularNeto()
    {
        double neto = 0;

        if (grid.Rows.Count > 0)
        {
            for (int i = 0; i < grid.Rows.Count; i++)
            {
                neto += Convert.ToDouble(grid.Rows[i].Cells[7].Value);
            }
        }

        txtNeto.Text = neto.ToString("###,###.00");
    }

    private void btnAgregar_Click(object sender, EventArgs e)
    {
        if (txtProducto.Text.Trim() == string.Empty || txtPrecio.Text.Trim() == string.Empty)
        {
            Sistema.advertir("Debe completar la información solicitada antes de agregar el
item");
            btnBuscarProd.Focus();
        }
        else
```

```
        {
            grid.Rows.Add((grid.Rows.Count + 1 ).ToString(), idItem, txtProducto.Text,
chkNuevo.Checked, txtPrecio.Text.ToString(), txtDscto.Value.ToString(),
txtCantidad.Value.ToString(), txtMonto.Text.ToString(),"");
            idItem = "";
            txtProducto.Clear();
            chkNuevo.Checked = true;
            txtPrecio.Clear();
            txtDscto.Value = 0;
            txtCantidad.Value = 1;
            txtMonto.Text = "0";
            grid.ClearSelection();

            this.CalcularNeto();
        }
    }

    private void btnQuitar_Click(object sender, EventArgs e)
    {
        if (grid.Rows.Count > 0)
        {
            if (Sistema.preguntar("¿Desea remover: \n" +
grid.CurrentRow.Cells[2].Value.ToString() + "?") == DialogResult.Yes)
            {
                grid.Rows.RemoveAt(grid.CurrentRow.Index);

                //Reordenar numeración...
                for (int i = 0; i < grid.Rows.Count; i++)
                {
                    grid.Rows[i].Cells[0].Value = i + 1;
                }

                this.CalcularNeto();
                grid.ClearSelection();
            }
        }
        else
        {
            Sistema.informar("No existen elementos para quitar");
        }
    }

    private void btnGuardar_Click(object sender, EventArgs e)
    {
```

```
        if (txtNumeroDoc.Text.Trim() == string.Empty)
        {
            Sistema.informar("Debe ingresar el número del documento");
            txtNumeroDoc.Focus();
            return;
        }

        if (idProveedor == string.Empty)
        {
            Sistema.informar("Debe seleccionar un proveedor para esta compra");
            btnBuscarProv.Focus();
            return;
        }

        if (txtFechaCompra.Text.Trim().Length != 10)
        {
            Sistema.informar("Escriba la fecha de compra dd/mm/aaaa");
            txtFechaCompra.Focus();
            return;
        }

        if (grid.Rows.Count == 0)
        {
            Sistema.informar("Debe indicar al menos un producto para esta compra");
            btnBuscarProd.Focus();
            return;
        }

        if (Sistema.preguntar("¿Está seguro de registrar esta compra?") ==
DialogResult.Yes)
        {
            idcompra =
objCompra.Registrar(idProveedor,Convert.ToDateTime(txtFechaCompra.Text),tdoc,cboSe
rie.Text + "-"+txtNumeroDoc.Text.Trim(),EstEntrega);

            if (idcompra.Trim() != string.Empty)
            {
                //Ahora vamos a enviar cada detalle
                for (int i = 0; i < grid.Rows.Count; i++)
                {
                    tmp = objCompra.RegistrarDetalle(idcompra,
grid.Rows[i].Cells[1].Value.ToString(), Convert.ToDouble(grid.Rows[i].Cells[4].Value),
Convert.ToDouble(grid.Rows[i].Cells[6].Value));
                    grid.Rows[i].Cells[8].Value = tmp;
```

```
                }

                btnGuardar.Enabled = false;
                btnCancelar.Enabled = false;
                btnFinalizar.Enabled = true;
                grid.ClearSelection();
                Sistema.informar("La compra ha sido registrada con éxito\n ahora podrá
añadir los código de barra");
            }
            else
            {
                Sistema.error("No se ha podido completar el proceso");
            }
        }
    }

    private void btnNuevo_Click(object sender, EventArgs e)
    {
        this.AdControles(true);
        this.Limpiar();
        txtNumeroDoc.Focus();
    }

    private void btnCancelar_Click(object sender, EventArgs e)
    {
        if (Sistema.preguntar("¿Está seguro de cancelar el proceso de compra?") ==
DialogResult.Yes)
        {
            this.Limpiar();
            this.AdControles(false);
        }
    }

    private void GenerarNumDoc()
    {
        if (txtNumeroDoc.Text.Trim() != string.Empty)
        {
            string tmp = "000000" + txtNumeroDoc.Text.Trim();
            txtNumeroDoc.Text = tmp.Substring(tmp.Length - 7, 7);
        }
    }

    private void txtNumeroDoc_KeyPress(object sender, KeyPressEventArgs e)
    {
```

```
            Sistema.soloNumeros(sender, e, 'E');
            if (e.KeyChar == Convert.ToChar(Keys.Enter))
            {
                this.GenerarNumDoc();
                btnBuscarProd.Focus();
            }
        }

        private void txtNumeroDoc_Leave(object sender, EventArgs e)
        {
            this.GenerarNumDoc();
        }

        private void cboEstadoCompra_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (cboEstadoCompra.SelectedIndex == 0)
                EstEntrega = 'P';
            else if (cboEstadoCompra.SelectedIndex == 1)
                EstEntrega = 'K';
            else
                EstEntrega = 'E';
        }

        private void btnFinalizar_Click(object sender, EventArgs e)
        {
            this.Limpiar();
            this.AdControles(false);
            btnFinalizar.Enabled = false;
        }

        private void gDatos_Enter(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
```

```
using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Procesos
{
   public partial class frmVenta : Form
   {
      Moneda objMoneda = new Moneda();
      Venta objVenta = new Venta();
      Trabajador objTrab = new Trabajador();

      DireccionCliente objDir = new DireccionCliente();
      DataTable tItems = new DataTable();

      string idVenta = "", idDetalle = "", idCliente = "", idDirCliente = "", iditem = "", serie =
"", docCliente = "", numeroDoc = "";
      bool existeSerie = false, existeItem = false;
      int nFilas, i, indiceFila;  //nFila cantidad de filas del GRID, i se utiliza para el bucle
búsquedas, indiceFila indica la fila para actualizar la cantidad
      double monto = 0, subtotal = 0, igv = 0, neto = 0;
      char    tPago = 'E', EmiteComprobante = '1';
      Nullable<char> tdoc = 'B';

      public frmVenta()
      {
         InitializeComponent();
      }

      private void frmVenta_Load(object sender, EventArgs e)
      {
         txtFecha.Text = DateTime.Now.ToShortDateString();
         cboTipoPago.SelectedIndex = 0;
         cboEstadoCompra.SelectedIndex = 2;
         txtCodigo.Font = new Font("barcode font", 36);

         grid.AlternatingRowsDefaultCellStyle.BackColor = Color.AliceBlue;
         actualizarNumDoc("B");

         //Cargando datos de los trabajadores
         this.actualizarListaTrabajadores();
      }

      private void actualizarNumDoc(string _Tipo)
```

```
    {
        numeroDoc = objVenta.generarNumDoc(_Tipo).Trim();
        cboSerie.Text = numeroDoc.Substring(0, 3);
        txtNumeroDoc.Text = numeroDoc.Substring(3, 7);
    }

    public void actualizarListaTrabajadores()
    {
        cboTrabajadores.DataSource = objTrab.listarTrabajadores();
        cboTrabajadores.DisplayMember = "Trabajador";
        cboTrabajadores.ValueMember = "idtrabajador";
    }

    private void btnMasClientes_Click(object sender, EventArgs e)
    {
        Mantenimientos.frmCliente f = new Mantenimientos.frmCliente();
        f._Sub = true;


        if (f.ShowDialog() == DialogResult.OK)
        {
            if (f.grid.CurrentRow != null)
            {
                //Si en caso se emite una factura se tiene que comprobar que el cliente es
natural
                if (f.grid.CurrentRow.Cells[1].Value.ToString() == "NATURAL" &&
optFactura.Checked)
                {
                    Sistema.informar("Debe seleccionar un cliente de tipo JURÍDICO para
FACTURA");
                }
                else
                {
                    cboDireccion.Text = "";

                    idCliente = f.grid.CurrentRow.Cells[0].Value.ToString();
                    docCliente = f.grid.CurrentRow.Cells[3].Value.ToString();
                    txtCliente.Text = f.grid.CurrentRow.Cells[2].Value.ToString();

                    cboDireccion.DataSource =
objDir.ListarDirJunta(f.grid.CurrentRow.Cells[0].Value.ToString());
                    cboDireccion.DisplayMember = "DIRECCION";
                    cboDireccion.ValueMember = "IdDirCliente";
```

```
                //Cargando datos de la dirección del cliente
                if (cboDireccion.Items.Count > 0)
                {
                    cboDireccion.SelectedItem = 0;
                    idDirCliente = cboDireccion.SelectedValue.ToString();
                }
                else
                {
                    idDirCliente = "";
                }

                txtCodigo.Focus();
            }
        }
    }
}

private void optFactura_CheckedChanged(object sender, EventArgs e)
{
    chkImprimir.Enabled = false;
    chkImprimir.Checked = true;
    tdoc = 'F';
    lblTipoDoc.Text = "FACTURA";
    txtNumeroDoc.Text = objVenta.generarNumDoc("F");
    Franja.BackColor = Color.FromName("DarkRed");
    actualizarNumDoc("F");

    lblSubTotal.Visible = true;
    txtSubTotal.Visible = true;
    lblIgv.Visible = true;
    txtIGV.Visible = true;

    //Vamos a reiniciar los datos del cliente activo
    txtCliente.Clear();
    cboDireccion.DataSource = objDir.ListarDirJunta("");

    this.calcularTotales();
}

private void optBoleta_CheckedChanged(object sender, EventArgs e)
{
    chkImprimir.Enabled = true;
    tdoc = 'B';
    lblTipoDoc.Text = "BOLETA DE VENTA";
```

```
    txtNumeroDoc.Text = objVenta.generarNumDoc("B");
    Franja.BackColor = Color.FromName("HotTrack");
    actualizarNumDoc("B");

    lblSubTotal.Visible = false;
    txtSubTotal.Visible = false;
    lblIgv.Visible = false;
    txtIGV.Visible = false;

    this.calcularTotales();
}

private void txtCodigo_KeyPress(object sender, KeyPressEventArgs e)
{
    if (e.KeyChar == Convert.ToChar(Keys.Enter))
    {
        tItems = objVenta.BuscarSerie(txtCodigo.Text.Trim());

        if (tItems.Rows.Count == 0)
        {
            //No se encontró el producto con este código de barra
            txtDescripcion.Clear();
            txtPrecioSugerido.Text = "0.00";
            txtPrecioVenta.Text = "0.00";

            Sistema.advertir("No existe ningún producto con este código");
            txtCodigo.Clear();
            iditem = "";
            serie = "";
            txtCodigo.Focus();
        }
        else
        {
            //Si se encontró el producto pero ahora debemos determinar si hay en stock

            if (Convert.ToInt32(tItems.Rows[0][4]) == 0)
            {

                iditem = ""; //IdItem
                serie = ""; //Serie
                txtDescripcion.Clear();
                txtPrecioSugerido.Clear();
                txtPrecioVenta.Clear();
```

```
                Sistema.informar("El siguiente producto:\n\n" + tItems.Rows[0][2].ToString()
+ "\n\nya ha sido vendido o se ha agotado");

                txtCodigo.Clear();
                txtCodigo.Focus();
            }
            else
            {
                txtCodigo.Clear();
                iditem = tItems.Rows[0][0].ToString(); //IdItem
                serie = tItems.Rows[0][1].ToString(); //Serie
                txtDescripcion.Text = tItems.Rows[0][2].ToString(); //Item(descripción)
                txtPrecioSugerido.Text = tItems.Rows[0][3].ToString(); //Precio venta
                txtPrecioVenta.Text = txtPrecioSugerido.Text;

                txtPrecioVenta.Focus();
            }
        }
    }
}

    private void AgregarDetalleVenta()
    {
        //Agregamos el producto a la lista
        monto = Convert.ToDouble(txtPrecioVenta.Text);
        grid.Rows.Add((grid.Rows.Count + 1).ToString(), iditem, "",
txtDescripcion.Text.Trim(), txtPrecioVenta.Text.Trim(),"1", monto.ToString("######.00"));

        gridSeries.Rows.Add(serie,iditem);
    }

    private void btnAgregar_Click(object sender, EventArgs e)
    {
        if (iditem == string.Empty || txtDescripcion.Text.Trim() == string.Empty ||
txtPrecioVenta.Text.Trim() == string.Empty)
        {
            Sistema.informar("Debe especificar un ítem y el precio de venta");
            txtCodigo.Focus();
        }
        else
        {
            //Algoritmo para agregar productos al detalle de la venta
```

```
            //1. Primero debemos verificar si el GRID tiene datos antes de iniciar la
validación
            i = 0;
            existeSerie = false;
            existeItem = false;

            if (grid.Rows.Count > 0)
            {
               while (i < gridSeries.Rows.Count && !existeSerie)
               {
                  if (gridSeries.Rows[i].Cells[0].Value.ToString() == serie)
                  {
                     existeSerie = true;
                  }
                  i++;
               }

               //Hasta este punto ya sabemos si existe o no la serie, es hora de tomar la
segunda decisión
               //Si la serie existe es proque el producto ya se agregó así que no se hace
nada, de lo
               //contrario tenemos que verificar si existe el ITEM para incrementar la
cantidad
               if (!existeSerie)
               {

                  //Como la serie no existe lo que hacemos es buscar el item
                  existeItem = false;
                  i = 0;

                  while (i < grid.Rows.Count && !existeItem)
                  {
                     if (grid.Rows[i].Cells[1].Value.ToString() == iditem)
                     {
                        indiceFila = i;
                        existeItem = true;
                     }
                     i++;
                  }

                  //Validamos la existencia del item
                  if (existeItem)
                  {
                     //Se tiene que actualizar la cantidad de productos
```

```
                    //Actualizamos la cantidad
                    grid.Rows[indiceFila].Cells[5].Value =
Convert.ToInt32(grid.Rows[indiceFila].Cells[5].Value) + 1;
                    //Calculamos el nuevo monto
                    monto = Convert.ToInt32(grid.Rows[indiceFila].Cells[5].Value) *
Convert.ToDouble(grid.Rows[indiceFila].Cells[4].Value);
                    //Mostramos el monto en el GRID
                    grid.Rows[indiceFila].Cells[6].Value = monto.ToString("######.00");
                    //Añadimos una nueva serie
                    gridSeries.Rows.Add(serie, iditem);

                }
                else
                {
                    //Se agrega un nuevo elemento que comparte el mismo item
                    this.AgregarDetalleVenta();
                }
            }
            else
            {
                //Antes de actualizar la cantidad se debe determinar de que fila es el item...
                existeItem = false;
                i = 0;

                while (i < grid.Rows.Count && !existeItem)
                {
                    if (grid.Rows[i].Cells[1].Value.ToString() == iditem)
                    {
                        indiceFila = i;
                        existeItem = true;
                    }
                    i++;
                }

                //Se agregará un producto similar, que comparten la misma serie o código
de barra
                //Actualizamos la cantidad
                grid.Rows[indiceFila].Cells[5].Value =
Convert.ToInt32(grid.Rows[indiceFila].Cells[5].Value) + 1;
                //Calculamos el nuevo monto
                monto = Convert.ToInt32(grid.Rows[indiceFila].Cells[5].Value) *
Convert.ToDouble(grid.Rows[indiceFila].Cells[4].Value);
                //Mostramos el monto en el GRID
                grid.Rows[indiceFila].Cells[6].Value = monto.ToString("######.00");
```

```
                //Añadimos una nueva serie
            }
        }
        else
        {
            //Si es el primer elemento se agrega sin ninguna validación...
            this.AgregarDetalleVenta();
        }

        //Se reinicia el item
        txtCodigo.Clear();
        txtDescripcion.Clear();
        txtPrecioSugerido.Text = "0.00";
        txtPrecioVenta.Text = "0.00";
        iditem = "";
        serie = "";
        grid.ClearSelection();

        //Generamos
        this.calcularTotales();

        txtCodigo.Focus();

    }
}

private void calcularTotales()
{
    if (grid.Rows.Count > 0)
    {
        subtotal = 0;
        igv = 0;
        neto = 0;

        for (int i = 0; i < grid.Rows.Count; i++)
        {
            neto += Convert.ToDouble(grid.Rows[i].Cells[6].Value);
        }

        if (optBoleta.Checked)
        {
            lblSubTotal.Visible = false;
            txtSubTotal.Visible = false;
            lblIgv.Visible = false;
```

```
            txtIGV.Visible = false;
        }

        if (optFactura.Checked)
        {
            lblSubTotal.Visible = true;
            txtSubTotal.Visible = true;
            lblIgv.Visible = true;
            txtIGV.Visible = true;

            subtotal = neto / 1.18;
            igv = subtotal * 0.18;
        }

        txtSubTotal.Text = subtotal.ToString("######.00");
        txtIGV.Text = igv.ToString("######.00");
        txtNeto.Text = neto.ToString("######.00");

        txtMontoLetras.Text = objMoneda.enletras(neto.ToString());
    }
}

private void txtPrecioVenta_KeyPress(object sender, KeyPressEventArgs e)
{
    Sistema.soloNumeros(sender, e, 'D');
    if (e.KeyChar == Convert.ToChar(Keys.Enter))
    {
        btnAgregar.Focus();
    }
}

private void btnQuitar_Click(object sender, EventArgs e)
{
    this.calcularTotales();
}


private string mesEnLetras(int numero)
{
    string rpta = "";

    switch (numero)
    {
        case 1: rpta = "ENERO"; break;
```

```
            case 2: rpta = "FEBRERO"; break;
            case 3: rpta = "MARZO"; break;
            case 4: rpta = "ABRIL"; break;
            case 5: rpta = "MAYO"; break;
            case 6: rpta = "JUNIO"; break;
            case 7: rpta = "JULIO"; break;
            case 8: rpta = "AGOSTO"; break;
            case 9: rpta = "SETIEMBRE"; break;
            case 10: rpta = "OCTUBRE"; break;
            case 11: rpta = "NOVIEMBRE"; break;
            case 12: rpta = "DICIEMBRE"; break;
            default: rpta = "INCORRECTO"; break;
        }

        return rpta;
    }

    private void btnGuardar_Click(object sender, EventArgs e)
    {
        //Validación...
        if (txtCliente.Text.Trim().Length == 0)
        {
            Sistema.informar("Es obligatorio especificar el cliente para el proceso de
compra");
            btnMasClientes.Focus();
            return;
        }

        if (optFactura.Checked && cboDireccion.Items.Count == 0)
        {
            Sistema.informar("Las ventas con FACTURA requieren la dirección del cliente");
            btnMasClientes.Focus();
            return;
        }

        if (txtFecha.Text.Trim().Length != 10)
        {
            Sistema.informar("No ha especificado una fecha correctamente");
            txtFecha.Focus();
            return;
        }

        if (grid.Rows.Count == 0)
        {
```

```
            Sistema.advertir("No ha especificado ningún ítem para esta compra");
            txtCodigo.Focus();
        }
        else
        {
            //Debemos saber si es un documento libre = null o si es Boleta o Factura, B-F
            if (!chkImprimir.Checked) { tdoc = null; }

            idVenta = objVenta.RegistrarVenta(idCliente, Sistema._IdLocal, idDirCliente,
Convert.ToDateTime(txtFecha.Text),
Convert.ToDateTime(DateTime.Now.ToShortTimeString()), tPago, tdoc,
cboSerie.Text.Trim() + txtNumeroDoc.Text.Trim(),'E',
cboTrabajadores.SelectedValue.ToString(), EmiteComprobante);

            if (idVenta.Trim() != string.Empty)
            {
                //Ahora guardamos todos los items y les asignamos su iddetalle
                for (int i = 0; i < grid.Rows.Count; i++)
                {
                    idDetalle = objVenta.RegistrarDetalle(idVenta,
grid.Rows[i].Cells[1].Value.ToString(),
Convert.ToDecimal(grid.Rows[i].Cells[4].Value.ToString().Trim()),
Convert.ToDecimal(grid.Rows[i].Cells[5].Value.ToString().Trim()));
                    grid.Rows[i].Cells[2].Value = idDetalle;

                    //Vamos a guardar las series
                    for (int j = 0; j < gridSeries.Rows.Count; j++)
                    {
                        if (grid.Rows[i].Cells[1].Value.ToString().Trim() ==
gridSeries.Rows[j].Cells[1].Value.ToString().Trim())
                        {
                            objVenta.RegistrarSeries(idDetalle,
gridSeries.Rows[j].Cells[0].Value.ToString());
                        }
                    }
                }

                //Se va a incrementar el numero del comprobante solo si está marcado el
checkbox imprimir comprobante
                //esto no se cumplirá cuando sea una boleta y no se entregue el comprobante
                if (chkImprimir.Checked)
                {
                    objVenta.incrementarDoc(tdoc);
                }
```

```
            Sistema.informar("Venta terminada correctamente");

            //Solo se genera el reporte dependiendo de si se seleccionó la emisión de
reportes
            if (EmiteComprobante == '1')
            {
               //Instancia del formulario...
               Reportes.frmReporte formulario = new Reportes.frmReporte();

               CrystalDecisions.CrystalReports.Engine.ReportClass reporte;

               if (tdoc == 'B')
               {
                  reporte = new Reportes.rptVenta();
               }
               else
               {
                  reporte = new Reportes.rptVentaFactura();
               }

               reporte.SetDataSource(objVenta.generarBoleta(idVenta));
               reporte.Refresh();

               reporte.SetParameterValue("cliente", txtCliente.Text);
               reporte.SetParameterValue("direccion", cboDireccion.Text);
               reporte.SetParameterValue("mesletras",
mesEnLetras(DateTime.Now.Month));
               reporte.SetParameterValue("dia", DateTime.Now.Day.ToString());
               reporte.SetParameterValue("anio", DateTime.Now.Year.ToString());

               if (tdoc == 'B')
               {
                  reporte.SetParameterValue("mes", DateTime.Now.Month.ToString());
                  reporte.SetParameterValue("dni", docCliente);
               }

               if (tdoc == 'F')
               {
                  reporte.SetParameterValue("ruc", docCliente);
                  reporte.SetParameterValue("subtotal", txtSubTotal.Text);
                  reporte.SetParameterValue("igv", txtIGV.Text);
                  reporte.SetParameterValue("neto", txtNeto.Text);
```

```
                    reporte.SetParameterValue("restofecha",
mesEnLetras(DateTime.Now.Month) + " del " + DateTime.Now.Year.ToString());
                    reporte.SetParameterValue("montoletras", txtMontoLetras.Text);
                }

                //Definiendo impresora para mandar el reporte
                reporte.PrintOptions.PrinterName = Sistema._NombreImpresora;
                reporte.PrintToPrinter(1, false, 0, 0);
            }

            this.ReiniciarCampos();
        }
    }
}

private void ReiniciarCampos()
{
    txtCliente.Clear();
    cboDireccion.DataSource = null;

    grid.Rows.Clear();
    gridSeries.Rows.Clear();
    chkImprimir.Checked = true;

    idVenta = "";
    idDetalle = "";
    idCliente = "";
    idDirCliente = "";
    iditem = "";
    serie = "";

    monto = 0;
    subtotal = 0;
    igv = 0;
    neto = 0;

    tdoc = 'B';
    tPago = 'E';
    EmiteComprobante = '1';


    cboSerie.Text = "001";
    optBoleta.Checked = true;
    actualizarNumDoc("B");
```

```
            txtMontoLetras.Clear();
        }

        private void btnCancelar_Click(object sender, EventArgs e)
        {
            if (Sistema.preguntar("¿Está seguro de cancelar el proceso de venta?") ==
DialogResult.Yes)
            {
                this.ReiniciarCampos();
            }
        }

        private void btnNuevo_Click(object sender, EventArgs e)
        {
            this.ReiniciarCampos();
        }

        private void chkImprimir_CheckedChanged(object sender, EventArgs e)
        {
            cboSerie.Visible = chkImprimir.Checked;
            txtNumeroDoc.Visible = chkImprimir.Checked;

            if (chkImprimir.Checked)
            {
                if (optFactura.Checked)
                    lblTipoDoc.Text = "FACTURA";
                else
                    lblTipoDoc.Text = "BOLETA DE VENTA";

                EmiteComprobante = '1';
            }
            else
            {
                lblTipoDoc.Text = "VENTA LIBRE";
                EmiteComprobante = '0';
            }
        }

        private void btnVerTrabajadores_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmTrabajador f = new Mantenimientos.frmTrabajador();
            f.ShowDialog();

            //Actualizando la lista
```

```
            this.actualizarListaTrabajadores();
        }

        private void btnListar_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmListaServicios f = new Mantenimientos.frmListaServicios();

            if (f.ShowDialog() == DialogResult.OK)
            {
                iditem = f.grid.CurrentRow.Cells[0].Value.ToString();
                serie = iditem; //Los servicios no tienen un número de serie por lo tanto reciben
el valor del ID (PK)
                txtDescripcion.Text = f.grid.CurrentRow.Cells[1].Value.ToString();
                txtPrecioSugerido.Text = f.grid.CurrentRow.Cells[2].Value.ToString();
                txtPrecioVenta.Text = f.grid.CurrentRow.Cells[2].Value.ToString();

                //Sistema.informar(iditem);

                //grid.Rows.Add((grid.Rows.Count + 1).ToString(),
f.grid.CurrentRow.Cells[0].Value.ToString(), "", f.grid.CurrentRow.Cells[1].Value.ToString(),
f.grid.CurrentRow.Cells[2].Value.ToString(), "1",
f.grid.CurrentRow.Cells[2].Value.ToString());
                this.calcularTotales();
            }
        }

        private void cboTipoPago_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (cboTipoPago.SelectedIndex == 0)
                tPago = 'E';
            else
                tPago = 'C';
        }

        private void gCliente_Enter(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```

```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Procesos
{
    public partial class frmAmoritzaciones : Form
    {
        public frmAmoritzaciones()
        {
            InitializeComponent();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void frmAmoritzaciones_Load(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER
{
    public partial class frmPrincipal : DevComponents.DotNetBar.Metro.MetroForm
    {
```

```
        Locales objLocales = new Locales();
        DataTable tLocal = new DataTable();
        Configuraciones objConfig = new Configuraciones();
        DataTable dtConfig = new DataTable();
        bool PermisoProcesos = false;

        public frmPrincipal()
        {
            InitializeComponent();
        }

        private void frmPrincipal_FormClosing(object sender, FormClosingEventArgs e)
        {
            Application.Exit();
        }

        private void VerificarLocal()
        {
            tLocal = objLocales.VerLocalActivo();

            if (tLocal.Rows.Count > 0)
            {
                PermisoProcesos = true;
                Sistema._IdLocal = tLocal.Rows[0][0].ToString();
                Sistema._NombreLocal = tLocal.Rows[0][1].ToString() + " " +
tLocal.Rows[0][2].ToString().Trim();
                btnLocal.Text = "LOCAL: [ " + Sistema._NombreLocal  + " ]";
            }
            else
            {
                PermisoProcesos = false;
                Sistema._IdLocal = "";
                Sistema._NombreLocal = "";
                btnLocal.Text = "AUN NO SE HA DEFINIDO UN LOCAL (CLIC AQUÍ PARA
REGISTRAR)";
            }
        }

        private void frmPrincipal_Load(object sender, EventArgs e)
        {
            //Ajustando a la resolución de pantalla
            metro.Height = this.Height;
            metro.Width = this.Width;
```

```
            btnUsuario.Location = new Point(this.Width - (btnUsuario.Width +
70),btnUsuario.Location.Y);
            btnCerrar.Location = new Point(this.Width - (btnCerrar.Width + 20),
btnCerrar.Location.Y);

            string nivelAcceso = "(USUARIO)";
            if (Sistema._Administrador == true) { nivelAcceso = "(ADMIN)"; }
            btnUsuario.Text = Sistema._NombreUsuario + " " + nivelAcceso;

            //Definiendo impresora activa
            //Vamos a cargar toda la configuración en el datatable
            dtConfig = objConfig.listarConfiguraciones();

            //Mostrar el nombre de la impresora
            Sistema._NombreImpresora = dtConfig.Rows[2][2].ToString().Trim();

            this.VerificarLocal();
        }

        private void frmPrincipal_Resize(object sender, EventArgs e)
        {
            metro.Height = this.Height;
            metro.Width = this.Width;
            btnUsuario.Location = new Point(this.Width - (btnUsuario.Width + 70),
btnUsuario.Location.Y);
            btnCerrar.Location = new Point(this.Width - (btnCerrar.Width + 20),
btnCerrar.Location.Y);
        }

        private void btnPersona_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmPersona f = new Mantenimientos.frmPersona();
            f.Show();
        }

        private void btnCliente_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmCliente f = new Mantenimientos.frmCliente();
            f.Show();
        }

        private void btnEmpresas_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmEmpresa f = new Mantenimientos.frmEmpresa();
```

```
            f.Show();
        }

        private void btnProveedores_Click(object sender, EventArgs e)
        {
            if (Sistema._Administrador)
            {
                Mantenimientos.frmProveedores f = new Mantenimientos.frmProveedores();
                f.Show();
            }
            else
            {
                this.AvisoNivelAcceso();
            }
        }

        private void btnCategorias_Click(object sender, EventArgs e)
        {
            if (Sistema._Administrador)
            {
                Mantenimientos.frmCategoria f = new Mantenimientos.frmCategoria();
                f.Show();
            }
            else
            {
                this.AvisoNivelAcceso();
            }
        }

        private void btnInterfaces_Click(object sender, EventArgs e)
        {
            if (Sistema._Administrador)
            {
                Mantenimientos.frmInterfazCategorias f = new
Mantenimientos.frmInterfazCategorias();
                f.Show();
            }
            else
            {
                this.AvisoNivelAcceso();
            }
        }

        private void btnLineas_Click(object sender, EventArgs e)
```

```
    {
        if (Sistema._Administrador)
        {
            Mantenimientos.frmLinea f = new Mantenimientos.frmLinea();
            f.Show();
        }
        else
        {
            this.AvisoNivelAcceso();
        }
    }

    private void btnMagnitudes_Click(object sender, EventArgs e)
    {
        if (Sistema._Administrador)
        {
            Mantenimientos.frmMagnitud f = new Mantenimientos.frmMagnitud();
            f.Show();
        }
        else
        {
            this.AvisoNivelAcceso();
        }
    }

    private void btnMarcas_Click(object sender, EventArgs e)
    {
        if (Sistema._Administrador)
        {
            Mantenimientos.frmMarcaCategorias f = new
Mantenimientos.frmMarcaCategorias();
            f.Show();
        }
        else
        {
            this.AvisoNivelAcceso();
        }
    }

    private void btnUnidad_Click(object sender, EventArgs e)
    {
        if (Sistema._Administrador)
        {
```

```
            Mantenimientos.frmUnidadMedidaCategorias f = new
Mantenimientos.frmUnidadMedidaCategorias();
            f.Show();
         }
         else
         {
            this.AvisoNivelAcceso();
         }
      }

      private void btnProductos_Click(object sender, EventArgs e)
      {
         Procesos.frmItem f = new Procesos.frmItem();
         f.Show();
      }

      private void buttonItem1_Click(object sender, EventArgs e)
      {
         if (Sistema.preguntar("¿Está seguro de abandonar el sistema?") ==
DialogResult.Yes)
         {
            Application.Exit();
         }
      }

      private void buttonItem2_Click(object sender, EventArgs e)
      {
         Procesos.frmCambiaClave f = new Procesos.frmCambiaClave();
         f.ShowDialog();
      }

      private void buttonItem3_Click(object sender, EventArgs e)
      {
         if (Sistema._Administrador)
         {
            Procesos.frmRegistraUsuario f = new Procesos.frmRegistraUsuario();
            f.ShowDialog();
         }
         else
         {
            this.AvisoNivelAcceso();
         }
      }
```

```
private void btnCompra_Click(object sender, EventArgs e)
{
    if (Sistema._Administrador)
    {
        if (PermisoProcesos)
        {
            Procesos.frmCompra f = new Procesos.frmCompra();
            f.Show();
        }
        else
        {
            Sistema.advertir("No se ha definido un local, imposible continuar");
        }
    }
    else
    {
        this.AvisoNivelAcceso();
    }
}

private void btnBuscarProducto_Click(object sender, EventArgs e)
{
    if (PermisoProcesos)
    {
        Procesos.frmBuscadorItem f = new Procesos.frmBuscadorItem();
        f.ShowDialog();
    }
    else
    {
        Sistema.advertir("No se ha definido un local, imposible continuar");
    }
}

private void btnVenta_Click(object sender, EventArgs e)
{
    if (PermisoProcesos)
    {
        Procesos.frmVenta f = new Procesos.frmVenta();
        f.ShowDialog();
    }
    else
    {
        Sistema.advertir("No se ha definido un local, imposible continuar");
    }
```

```
    }

    private void btnCerrar_Click(object sender, EventArgs e)
    {
        if (Sistema.preguntar("¿Está seguro de abandonar la aplicación?") ==
DialogResult.Yes)
        {
            Application.Exit();
        }
    }

    private void btnLocal_Click(object sender, EventArgs e)
    {
        Mantenimientos.frmLocal f = new Mantenimientos.frmLocal();
        f.ShowDialog();

        this.VerificarLocal();
    }

    private void btnReporte_Click(object sender, EventArgs e)
    {
        if (PermisoProcesos)
        {
            Reportes.frmAdminReportes f = new Reportes.frmAdminReportes();
            f.ShowDialog();
        }
        else
        {
            Sistema.advertir("No se ha definido un local, imposible continuar");
        }
    }

    private void btnLocales_Click(object sender, EventArgs e)
    {
        if (Sistema._Administrador)
        {
            Mantenimientos.frmLocal f = new Mantenimientos.frmLocal();
            f.ShowDialog();
            this.VerificarLocal();
        }
        else
        {
            this.AvisoNivelAcceso();
        }
```

```
        }

        private void AvisoNivelAcceso()
        {
            Sistema.advertir("La cuenta de acceso actual no le permite ingresar a este
módulo\nInicie sesión con una cuenta administrador para proceder");
        }

        private void btnTrabajadores_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmTrabajador f = new Mantenimientos.frmTrabajador();
            f.Show();
        }

        private void btnConfig_Click(object sender, EventArgs e)
        {
            Procesos.frmConfiguración f = new Procesos.frmConfiguración();
            f.ShowDialog();
        }

        private void btnControl_Click(object sender, EventArgs e)
        {
            Procesos.frmAmoritzaciones f = new Procesos.frmAmoritzaciones();
            f.Show();
        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Procesos
{
    public partial class frmItem : Form
    {
        Item objItem = new Item();
```

```
Categorias objCategoria = new Categorias();
MarcaCategoria objMarcaCat = new MarcaCategoria();
Concepto objConcepto = new Concepto();
UnidadMedida objUnidadMedida = new UnidadMedida();
InterfazCategoria objInterfazCat = new InterfazCategoria();

DataView dv;
DataTable tItems = new DataTable();

string iditem = "", idInterface = "", cadenaFiltro = "";
public bool _Sub = false;
bool ready = false;

public frmItem()
{
   InitializeComponent();
}

private void btnCerrar_Click(object sender, EventArgs e)
{
   this.Close();
}

private void CargarCategorias()
{
   cboCategoria.DataSource = objCategoria.ListarCategorias(1);
   cboCategoria.DisplayMember = "Categoria";
   cboCategoria.ValueMember = "IdCategoria";
   cboCategoria.Text = "";
}

private void CargarConcepto()
{
   cboConcepto.DataSource = objConcepto.ListarConceptos(1);
   cboConcepto.DisplayMember = "Concepto";
   cboConcepto.ValueMember = "IdConcepto";
   cboConcepto.Text = "";
}

private void CargarUnidadMedida()
{
   cboUnidad.DataSource = objUnidadMedida.ListarUnidadMedida(1);
   cboUnidad.DisplayMember = "Unidad";
   cboUnidad.ValueMember = "IdUnidad";
```

```
        cboUnidad.Text = "";
    }

    private void Limpiar()
    {
        txtNombreItem.Clear();
        cboCategoria.Text = "";
        txtPrecio.Clear();
        cboMarca.Text = "";
        cboInterfaz.Text = "";
        idInterface = "";
        cboConcepto.Text = "";
        cboUnidad.Text = "";
        txtValor.Clear();
        btnGuardar.Text = "Guardar";
    }

    private void AdControles(bool sw)
    {
        gDatosItem.Enabled = sw;
        gEspecificaciones.Enabled = !sw;
        btnNuevo.Enabled = !sw;
        btnTodos.Enabled = !sw;
        btnGuardar.Enabled = sw;
        btnModificar.Enabled = !sw;
        btnCancelar.Enabled = sw;

        //optProducto.Enabled = sw;
        //optServicio.Enabled = sw;

        optProducto.Visible = !sw;
        optServicio.Visible = !sw;
    }

    private void frmItem_Load(object sender, EventArgs e)
    {
        if (_Sub) { btnExportar.Visible = true; }

        this.CargarCategorias();
        this.CargarConcepto();
        this.CargarUnidadMedida();

        tItems = objItem.ListarItems();
        gridItems.DataSource = tItems;
```

```
            gridEspecificaciones.DataSource = objItem.EspecificacionesItem("");
            gridEspecificaciones.Columns[0].Width = 400;
            gridEspecificaciones.Columns[1].Width = 130;
            gridEspecificaciones.Columns[1].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleRight;
            gridEspecificaciones.Columns[1].HeaderText = "Detalle";
            gridEspecificaciones.ClearSelection();

            gridItems.Columns[0].Visible = false;
            gridItems.Columns[1].Width = 300;
            gridItems.Columns[2].Width = 150;
            gridItems.Columns[3].Width = 150;
            gridItems.Columns[4].Width = 260;

            gridItems.Columns[1].HeaderText = "Categoría";
            gridItems.Columns[2].HeaderText = "Marca";
            gridItems.Columns[3].HeaderText = "Interfaz";
            gridItems.Columns[4].HeaderText = "Nombre/Modelo";

            gridItems.Columns[5].Width = 60;
            gridItems.Columns[6].Width = 60;
            gridItems.Columns[5].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleRight;
            gridItems.Columns[6].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleRight;
            gridItems.AlternatingRowsDefaultCellStyle.BackColor = Color.AliceBlue;
            gridItems.ClearSelection();

            this.Limpiar();
            this.AdControles(false);

            ready = true;
            iditem = "";
        }

        private void cargarMarcas()
        {
            cboMarca.DataSource =
objMarcaCat.VerMarcas(cboCategoria.SelectedValue.ToString());
            cboMarca.DisplayMember = "Marca";
            cboMarca.ValueMember = "IdMarcaCateg";
            cboMarca.Text = "";
        }
```

```
        private void cargarInterfaces()
        {
            cboInterfaz.DataSource =
objInterfazCat.VerInterfaces(cboCategoria.SelectedValue.ToString());
            cboInterfaz.DisplayMember = "Interfase";
            cboInterfaz.ValueMember = "IdInterfaseCateg";
            cboInterfaz.Text = "";
        }

        private void cboCategoria_SelectedIndexChanged(object sender, EventArgs e)
        {
            this.cargarMarcas();
            this.cargarInterfaces();

            if (cboInterfaz.Items.Count == 0) { idInterface = ""; }

            if (ready)
            {
                dv = tItems.DefaultView;
                dv.RowFilter = "Categoria LIKE '" + cboCategoria.Text.Trim() + "%'";
                gridItems.ClearSelection();
            }

            iditem = "";
        }

        private void txtPrecio_KeyPress(object sender, KeyPressEventArgs e)
        {
            Sistema.soloNumeros(sender, e, 'D');
        }

        private void txtStock_KeyPress(object sender, KeyPressEventArgs e)
        {
            Sistema.soloNumeros(sender, e, 'E');
        }

        private void txtNombreItem_Enter(object sender, EventArgs e)
        {
            txtNombreItem.BackColor = Color.FromArgb(255, 255, 192);
        }

        private void cboCategoria_Enter(object sender, EventArgs e)
        {
```

```
    cboCategoria.BackColor = Color.FromArgb(255, 255, 192);
}

private void cboMarca_Enter(object sender, EventArgs e)
{
    cboMarca.BackColor = Color.FromArgb(255, 255, 192);
}

private void cboInterfaz_Enter(object sender, EventArgs e)
{
    cboInterfaz.BackColor = Color.FromArgb(255, 255, 192);
}

private void txtPrecio_Enter(object sender, EventArgs e)
{
    txtPrecio.BackColor = Color.FromArgb(255, 255, 192);
}

private void cboConcepto_Enter(object sender, EventArgs e)
{
    cboConcepto.BackColor = Color.FromArgb(255, 255, 192);
}

private void cboUnidad_Enter(object sender, EventArgs e)
{
    cboUnidad.BackColor = Color.FromArgb(255, 255, 192);
}

private void txtValor_Enter(object sender, EventArgs e)
{
    txtValor.BackColor = Color.FromArgb(255, 255, 192);
}

private void txtNombreItem_Leave(object sender, EventArgs e)
{
    txtNombreItem.BackColor = Color.White;
}

private void cboMarca_Leave(object sender, EventArgs e)
{
    cboMarca.BackColor = Color.White;
}

private void cboCategoria_Leave(object sender, EventArgs e)
```

```
    {
        cboCategoria.BackColor = Color.White;
    }

    private void cboInterfaz_Leave(object sender, EventArgs e)
    {
        cboInterfaz.BackColor = Color.White;
    }

    private void txtPrecio_Leave(object sender, EventArgs e)
    {
        txtPrecio.BackColor = Color.White;
    }

    private void cboConcepto_Leave(object sender, EventArgs e)
    {
        cboConcepto.BackColor = Color.White;
    }

    private void cboUnidad_Leave(object sender, EventArgs e)
    {
        cboUnidad.BackColor = Color.White;
    }

    private void txtValor_Leave(object sender, EventArgs e)
    {
        txtValor.BackColor = Color.White;
    }

    private void btnGuardar_Click(object sender, EventArgs e)
    {
        if (cboCategoria.Text.Trim() == string.Empty)
        {
            Sistema.informar("Selecciona la categoría/tipo de item a registrar");
            cboCategoria.Focus();
            return;
        }
cboMarca.Text.Trim() == string.Empty)
            {
                Sistema.informar("Debe especificar la marca para continuar");
                cboMarca.Focus();
                return;
            }
        }
```

```
        if (txtPrecio.Text.Trim() == string.Empty)
        {
            Sistema.informar("Debe especificar el precio");
            txtPrecio.Focus();
            return;
        }

        if (optProducto.Checked)
        {
            if (txtNombreItem.Text.Trim() == string.Empty)
            {
                Sistema.informar("Escriba el nombre o modelo del item");
                txtNombreItem.Focus();
                return;
            }
        }

        //Para mejorar la exactitud de la validación
        if (btnGuardar.Text.Trim() == "Guardar") { iditem = ""; }

        //Primero tenemos que definir si se está grabando un producto o un servicio
        bool resultado;

        if (optProducto.Checked)
        {
            resultado = objItem.EnviarItem(iditem, txtNombreItem.Text.Trim(),
cboMarca.SelectedValue.ToString(), Convert.ToDouble(txtPrecio.Text), idInterface);
        }
        else
        {
            //Servicio
            resultado = objItem.EnviarItem(iditem, txtNombreItem.Text.Trim(),
cboMarca.SelectedValue.ToString(), Convert.ToDouble(txtPrecio.Text), idInterface);
        }

        if (resultado)
        {
            this.Limpiar();
            this.AdControles(false);
            tItems = objItem.ListarItems();
            gridItems.DataSource = tItems;
            gridItems.ClearSelection();
            iditem = "";
```

```
            }
            else
            {
                Sistema.error("No se ha podido completar el proceso, verifique los datos
ingresados");
            }
        }

        private void lblCategoria_MouseEnter(object sender, EventArgs e)
        {
            lblCategoria.Font = new Font(lblCategoria.Font, FontStyle.Underline);
        }

        private void lblMarca_MouseEnter(object sender, EventArgs e)
        {
            lblMarca.Font = new Font(lblMarca.Font, FontStyle.Underline);
        }

        private void lblInterfaz_MouseEnter(object sender, EventArgs e)
        {
            lblInterfaz.Font = new Font(lblInterfaz.Font,FontStyle.Underline);
        }

        private void lblUnidad_MouseEnter(object sender, EventArgs e)
        {
            lblUnidad.Font = new Font(lblUnidad.Font,FontStyle.Underline);
        }

        private void lblConcepto_MouseEnter(object sender, EventArgs e)
        {
            lblConcepto.Font = new Font(lblConcepto.Font,FontStyle.Underline);
        }

        private void lblCategoria_MouseLeave(object sender, EventArgs e)
        {
            lblCategoria.Font = new Font(lblCategoria.Font, FontStyle.Regular);
        }

        private void lblMarca_MouseLeave(object sender, EventArgs e)
        {
            lblMarca.Font = new Font(lblMarca.Font,FontStyle.Regular);
        }

        private void lblInterfaz_MouseLeave(object sender, EventArgs e)
```

```
{
    lblInterfaz.Font = new Font(lblInterfaz.Font,FontStyle.Regular);
}

private void lblConcepto_MouseLeave(object sender, EventArgs e)
{
    lblConcepto.Font = new Font(lblConcepto.Font,FontStyle.Regular);
}

private void lblUnidad_MouseLeave(object sender, EventArgs e)
{
    lblUnidad.Font = new Font(lblUnidad.Font,FontStyle.Regular);
}

private void cboInterfaz_SelectedIndexChanged(object sender, EventArgs e)
{
    dv = tItems.DefaultView;
    cadenaFiltro = "Interfase LIKE '" + cboInterfaz.Text.Trim() + "%'";

    if (cboMarca.Text.Trim() != string.Empty)
    {
        cadenaFiltro += " and Marca LIKE '" + cboMarca.Text.Trim() + "%'";
    }

    if (txtNombreItem.Text.Trim() != string.Empty)
    {
        cadenaFiltro += " and NombreItem LIKE '%" + txtNombreItem.Text.Trim() + "%'";
    }

    idInterface = cboInterfaz.SelectedValue.ToString();
    dv.RowFilter = cadenaFiltro;
    gridItems.ClearSelection();
    iditem = "";
}

private void btnNuevo_Click(object sender, EventArgs e)
{
    this.AdControles(true);
    this.Limpiar();
    gridEspecificaciones.DataSource = objItem.EspecificacionesItem("");
    gridItems.ClearSelection();
    gridEspecificaciones.ClearSelection();
}
```

```
        private void btnCancelar_Click(object sender, EventArgs e)
        {
            if (Sistema.preguntar("¿Está seguro de cancelar el proceso?") ==
DialogResult.Yes)
            {
                tItems = objItem.ListarItems();
                gridItems.DataSource = tItems;

                gridEspecificaciones.DataSource = objItem.EspecificacionesItem("");
                gridEspecificaciones.ClearSelection();
                gridItems.ClearSelection();
                this.Limpiar();
                this.AdControles(false);
            }
        }

        private void lblCategoria_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmCategoria f = new Mantenimientos.frmCategoria();
            f.ShowDialog();
            this.CargarCategorias();
        }

        private void lblMarca_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmMarcaCategorias f = new
Mantenimientos.frmMarcaCategorias();
            f.ShowDialog();
            this.cargarMarcas();
        }

        private void lblInterfaz_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmInterfazCategorias f = new
Mantenimientos.frmInterfazCategorias();
            f.ShowDialog();
            this.cargarInterfaces();
        }

        private void lblConcepto_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmConcepto f = new Mantenimientos.frmConcepto();
            f.ShowDialog();
            this.CargarConcepto();
```

```
    }

    private void cboMarca_SelectedIndexChanged(object sender, EventArgs e)
    {
        dv = tItems.DefaultView;
        cadenaFiltro = "Marca LIKE '" + cboMarca.Text.Trim() + "%'";

        if (cboInterfaz.Text.Trim() !=string.Empty)
        {
            cadenaFiltro += " and Interfase LIKE '" + cboInterfaz.Text.Trim() + "%'";
        }


        if (txtNombreItem.Text.Trim() != string.Empty)
        {
            cadenaFiltro += " and NombreItem LIKE '%" + txtNombreItem.Text.Trim() + "%'";
        }

        dv.RowFilter = cadenaFiltro;
        gridItems.ClearSelection();
        iditem = "";
    }

    private void txtNombreItem_TextChanged(object sender, EventArgs e)
    {
        dv = tItems.DefaultView;

        cadenaFiltro = "Categoria LIKE '" + cboCategoria.Text.Trim() + "%' and
NombreItem LIKE '%" + txtNombreItem.Text.Trim() + "%'";

        if (cboMarca.Text.Trim() != string.Empty)
        {
            cadenaFiltro += " and Marca LIKE '" + cboMarca.Text.Trim() + "%'";
        }

        if (cboInterfaz.Text.Trim() != string.Empty)
        {
            cadenaFiltro += " and Interfase LIKE '" + cboInterfaz.Text.Trim() + "%'";
        }

        dv.RowFilter = cadenaFiltro;
        gridItems.ClearSelection();
        iditem = "";
    }
```

```
    private void btnTodos_Click(object sender, EventArgs e)
    {
        txtNombreItem.Clear();
        cboInterfaz.Text = "";
        txtPrecio.Clear();
        cboMarca.Text = "";
        cboCategoria.Text = "";

        tItems = objItem.ListarItems();
        gridItems.DataSource = tItems;
        gridItems.ClearSelection();
        iditem = "";
    }

    private void gridItems_CellClick(object sender, DataGridViewCellEventArgs e)
    {
        if (gridItems.Rows.Count > 0)
        {
            gridEspecificaciones.DataSource =
objItem.EspecificacionesItem(gridItems.CurrentRow.Cells[0].Value.ToString());
            gridEspecificaciones.ClearSelection();
        }
    }

    private void btnAgregar_Click(object sender, EventArgs e)
    {
        if (cboConcepto.Text.Trim() == string.Empty || cboUnidad.Text.Trim() ==
string.Empty || txtValor.Text.Trim() == string.Empty)
        {
            Sistema.informar("Debe especificar el CONCEPTO > UNIDAD > VALOR para
continuar");
            cboConcepto.Focus();
        }
        else
        {
            if (Sistema.preguntar("¿Está seguro de continuar?") == DialogResult.Yes)
            {
                if
(objItem.AgregarEspecificaciones(gridItems.CurrentRow.Cells[0].Value.ToString(),txtValor.
Text,cboConcepto.SelectedValue.ToString(),cboUnidad.SelectedValue.ToString()))
                {
                    Sistema.informar("Se ha guardado una especificación correctamente");
```

```
                gridEspecificaciones.DataSource =
objItem.EspecificacionesItem(gridItems.CurrentRow.Cells[0].Value.ToString());
                gridEspecificaciones.ClearSelection();
                cboConcepto.Text = "";
                cboUnidad.Text = "";
                txtValor.Clear();
            }
        }
    }
}

private void chkMostrar_CheckedChanged(object sender, EventArgs e)
{
    gEspecificaciones.Visible = chkMostrar.Checked;
    gridEspecificaciones.Visible = chkMostrar.Checked;

    if (chkMostrar.Checked)
        gridItems.Size = new Size(gridItems.Width, 267);
    else
        gridItems.Size = new Size(gridItems.Width, 416);
}

private void btnModificar_Click(object sender, EventArgs e)
{
    if (gridItems.Rows.Count == 0)
    {
        Sistema.informar("No existen registros para modificar");
    }
    else
    {
        if (iditem == string.Empty)
        {
            Sistema.informar("Seleccione un elemento de la lista");
        }
        else
        {
            cboCategoria.Text = gridItems.CurrentRow.Cells[1].Value.ToString();
            cboMarca.Text = gridItems.CurrentRow.Cells[2].Value.ToString();
            cboInterfaz.Text = gridItems.CurrentRow.Cells[3].Value.ToString();
            txtNombreItem.Text = gridItems.CurrentRow.Cells[4].Value.ToString();
            txtPrecio.Text = gridItems.CurrentRow.Cells[5].Value.ToString();

            //Se vuelve a cargar el valor en la variable "iditem", porque al hacer un
cambio en los controles de la parte superior
```

```
                //en el evento SelectedIndexChanged reinicia el valor de dicha variable y no
se puede completar el proceso modificar
                iditem = gridItems.CurrentRow.Cells[0].Value.ToString();

                this.AdControles(true);
                btnGuardar.Text = "Actualizar";
            }
        }
    }

    private void gridItems_CellEnter(object sender, DataGridViewCellEventArgs e)
    {
        iditem = gridItems.CurrentRow.Cells[0].Value.ToString();
    }

    private void lblUnidad_Click(object sender, EventArgs e)
    {
        Mantenimientos.frmUnidadMedida f = new Mantenimientos.frmUnidadMedida();
        f.ShowDialog();
        this.CargarUnidadMedida();
    }

    private void esProducto(bool sw)
    {
        //lblMarca.Enabled = sw;
        //cboMarca.Enabled = sw;
        //lblInterfaz.Enabled = sw;
        //cboInterfaz.Enabled = sw;
    }

    private void optServicio_CheckedChanged(object sender, EventArgs e)
    {
        this.esProducto(false);
        cboCategoria.DataSource = objItem.ListarItemServicios();
        cboCategoria.DisplayMember = "Categoria";
        cboCategoria.ValueMember = "IdCategoria";
    }

    private void optProducto_CheckedChanged(object sender, EventArgs e)
    {
        this.esProducto(true);
        this.CargarCategorias();
    }
}
```

```
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Procesos
{
   public partial class frmConfiguración : Form
   {
      DataTable dtConfig = new DataTable();
      Configuraciones objConfig = new Configuraciones();
      string numBoleta = "", numFactura = "";

      public frmConfiguración()
      {
         InitializeComponent();
      }

      private void btnCerrar_Click(object sender, EventArgs e)
      {
         this.Close();
      }

      private void txtSerieB_KeyPress(object sender, KeyPressEventArgs e)
      {
         Sistema.soloNumeros(sender, e, 'E');
      }

      private void txtNumBoleta_KeyPress(object sender, KeyPressEventArgs e)
      {
         Sistema.soloNumeros(sender, e, 'E');
      }

      private void txtSerieF_KeyPress(object sender, KeyPressEventArgs e)
      {
         Sistema.soloNumeros(sender, e, 'E');
```

```
    }

    private void txtNumFactura_KeyPress(object sender, KeyPressEventArgs e)
    {
        Sistema.soloNumeros(sender, e, 'E');
    }

    private void frmConfiguración_Load(object sender, EventArgs e)
    {
        //Vamos a cargar toda la configuración en el datatable
        dtConfig = objConfig.listarConfiguraciones();

        //Devolviendo las configuraciones

        //Boleta
        numBoleta = dtConfig.Rows[0][2].ToString().Trim();
        if (numBoleta.Length == 10)
        {
            txtSerieB.Text = numBoleta.Substring(0,3);
            txtNumBoleta.Text = numBoleta.Substring(3,7);
        }

        //Factura
        numFactura = dtConfig.Rows[1][2].ToString().Trim();
        if (numFactura.Length == 10)
        {
            txtSerieF.Text = numFactura.Substring(0,3);
            txtNumFactura.Text = numFactura.Substring(3,7);
        }

        //Cargando impresora en la lista
        //Cargando lista de impresoras en la lista
        lstImpresoras.Items.Clear();

        foreach (string nombreImpresora in
System.Drawing.Printing.PrinterSettings.InstalledPrinters)
        {
            lstImpresoras.Items.Add(nombreImpresora);
        }

        lstImpresoras.ClearSelected();

        //Mostrar el nombre de la impresora
        txtImpresora.Text = dtConfig.Rows[2][2].ToString().Trim();
```

```
    }

    private void btnGuardarBoleta_Click(object sender, EventArgs e)
    {
        if (txtSerieB.Text.Trim().Length != 3)
        {
            Sistema.informar("El número de serie debe poseer 3 dígitos, Ejem.: 001, 002,
003");
            txtSerieB.Focus();
        }
        else
        {
            if (txtNumBoleta.Text.Trim().Length != 7)
            {
                Sistema.informar("El número de la Boleta debe tener 7 dígitos, Ejem.:
0000105");
                txtNumBoleta.Focus();
            }
            else
            {
                if (objConfig.registrarConfiguracion("CNF00001",txtSerieB.Text.Trim() +
txtNumBoleta.Text.Trim()))
                {
                    txtSerieB.Enabled = false;
                    txtNumBoleta.Enabled = false;
                    Sistema.informar("Se registró correctamente la serie y número que servirán
como punto inicial de referencia para la emisión de BOLETAS");
                }
            }
        }
    }

    private void btnGuardarFactura_Click(object sender, EventArgs e)
    {
        if (txtSerieF.Text.Trim().Length != 3)
        {
            Sistema.informar("El número de serie debe poseer 3 dígitos, Ejem.: 001, 002,
003");
            txtSerieF.Focus();
        }
        else
        {
            if (txtNumFactura.Text.Trim().Length != 7)
            {
```

```
                Sistema.informar("El número de la Factura debe tener 7 dígitos, Ejem.:
0000105");
                txtNumFactura.Focus();
            }
            else
            {
                if (objConfig.registrarConfiguracion("CNF00002", txtSerieF.Text.Trim() +
txtNumFactura.Text.Trim()))
                {
                    txtSerieF.Enabled = false;
                    txtNumFactura.Enabled = false;
                    Sistema.informar("Se registró correctamente la serie y número que servirán
como punto inicial de referencia para la emisión de FACTURAS");
                }
            }
        }
    }

    private void btnGuardarImpresora_Click(object sender, EventArgs e)
    {
        if (lstImpresoras.Items.Count == 0)
        {
            Sistema.informar("No existe ninguna impresora en la lista para configurar");
        }
        else
        {
            if (lstImpresoras.SelectedIndex == -1)
            {
                Sistema.informar("Debe seleccionar una impresora de la lista para
continuar");
            }
            else
            {
                if (Sistema.preguntar("¿Está seguro de asignar la impresora: " +
lstImpresoras.Text + " como predeterminada para la emisión de los comprobantes?") ==
DialogResult.Yes)
                {
                    if (objConfig.registrarConfiguracion("CNF00003",
lstImpresoras.Text.Trim()))
                    {
                        Sistema.informar("Se ha guardado la configuración correctamente");
                        txtImpresora.Text = lstImpresoras.Text;
                        txtImpresora.Enabled = false;
                    }
```

```
                }
            }
        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Procesos
{
    public partial class frmCambiaClave : Form
    {
        Usuario objUsuario = new Usuario();
        string clave1, clave2, clave3;

        public frmCambiaClave()
        {
            InitializeComponent();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void btnActualizar_Click(object sender, EventArgs e)
        {
            clave1 = txtClave1.Text.Trim();
            clave2 = txtClave2.Text.Trim();
            clave3 = txtClave3.Text.Trim();

            if (clave1 == string.Empty || clave2 == string.Empty || clave3 == string.Empty)
            {
                Sistema.informar("Debe escribir las 3 contraseñas solicitadas");
```

```
            txtClave1.Focus();
        }
        else
        {
            if (clave2 != clave3)
            {
                Sistema.informar("Las nuevas contraseñas no coinciden");
                txtClave2.Focus();
            }
            else
            {
                if (clave1 != Sistema._Clave)
                {
                    Sistema.advertir("La contraseña original no es correcta");
                    txtClave1.Focus();
                }
                else
                {
                    //Actualizamos
                    if (objUsuario.ActualizarClave(Sistema._IdUsuario, clave2))
                    {
                        Sistema._Clave = txtClave2.Text.Trim();
                        Sistema.informar("Contraseña actualizada correctamente");
                        this.Close();
                    }
                }
            }
        }
    }

    private void groupBox1_Enter(object sender, EventArgs e)
    {

    }
  }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
```

```
using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Mantenimientos
{
    public partial class frmProveedores : Form
    {
        DataTable tProveedor = new DataTable();
        DataView dv;
        Proveedor objproveedor = new Proveedor();
        string idEmpresa = "", idPersona = "";
        char tipo = 'J';

        public bool _Sub = false;

        public frmProveedores()
        {
            InitializeComponent();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void Limpiar()
        {
            optEmpresa.Checked = true;
            txtID.Clear();
            txtProveedor.Clear();
            idEmpresa = "";
            idPersona = "";
        }

        private void AdControles(bool sw)
        {
            gInfo.Enabled = sw;
            grid.Enabled = !sw;
            btnNuevo.Enabled = !sw;
            btnGuardar.Enabled = sw;
            btnCancelar.Enabled = sw;
        }
```

```
    private void frmProveedores_Load(object sender, EventArgs e)
    {
        if (_Sub) { btnExportar.Visible = true; }

        tProveedor = objproveedor.ListarProveedores();
        grid.DataSource = tProveedor;

        grid.Columns[0].Visible = false;
        grid.Columns[1].Width = 100;
        grid.Columns[2].Width = 410;

        this.Limpiar();
        this.AdControles(false);

        grid.ClearSelection();
    }

    private void btnNuevo_Click(object sender, EventArgs e)
    {
        this.Limpiar();
        this.AdControles(true);
        grid.ClearSelection();
        txtID.Text = objproveedor.GenerarID();
    }

    private void btnCancelar_Click(object sender, EventArgs e)
    {
        if (Sistema.preguntar("¿Está seguro de cancelar el proceso?") ==
DialogResult.Yes)
        {
            this.Limpiar();
            this.AdControles(false);
        }
    }

    private void btnBuscarProveedor_Click(object sender, EventArgs e)
    {
        if (optEmpresa.Checked)
        {
            frmEmpresa f = new frmEmpresa();
            f._Sub = true;

            if (f.ShowDialog() == DialogResult.OK)
            {
```

```
                if (f.grid.CurrentRow != null)
                {
                    idPersona = "";
                    idEmpresa = f.grid.CurrentRow.Cells[0].Value.ToString();
                    txtProveedor.Text = f.grid.CurrentRow.Cells[1].Value.ToString();
                }
                else
                {
                    Sistema.informar("No seleccionó ningún elemento");
                }
            }
            else
            {
                idPersona = "";
                idEmpresa = "";
                txtProveedor.Clear();
            }
        }

        if (optPersona.Checked)
        {
            frmPersona f = new frmPersona();
            f._Sub = true;

            if (f.ShowDialog() == DialogResult.OK)
            {
                if (f.grid.CurrentRow != null)
                {
                    idEmpresa = "";
                    idPersona = f.grid.CurrentRow.Cells[0].Value.ToString();
                    txtProveedor.Text = f.grid.CurrentRow.Cells[1].Value.ToString() + " " +
f.grid.CurrentRow.Cells[2].Value.ToString() + " " +
f.grid.CurrentRow.Cells[3].Value.ToString();
                }
                else
                {
                    Sistema.informar("No seleccionó ningún elemento");
                }
            }
            else
            {
                idPersona = "";
                idEmpresa = "";
                txtProveedor.Clear();
```

```
            }
        }
    }

    private void btnGuardar_Click(object sender, EventArgs e)
    {
        if (idEmpresa == string.Empty && idPersona == string.Empty)
        {
            Sistema.informar("Debe especificar una empresa o persona como nuevo
proveedor");
        }
        else
        {
            if (Sistema.preguntar("¿Está seguro de registrar este proveedor?") ==
DialogResult.Yes)
            {
                if (objproveedor.Insertar(tipo, idEmpresa, idPersona))
                {
                    tProveedor = objproveedor.ListarProveedores();
                    grid.DataSource = tProveedor;

                    this.Limpiar();
                    this.AdControles(false);

                    grid.ClearSelection();
                }
            }
        }
    }

    private void optEmpresa_CheckedChanged(object sender, EventArgs e)
    {
        tipo = 'J';

        txtProveedor.Clear();
        idEmpresa = "";
        idPersona = "";
    }

    private void optPersona_CheckedChanged(object sender, EventArgs e)
    {
        tipo = 'N';

        txtProveedor.Clear();
```

```
            idEmpresa = "";
            idPersona = "";
        }

        private void txtBusqueda_TextChanged(object sender, EventArgs e)
        {
            dv = tProveedor.DefaultView;
            dv.RowFilter = "DatosProveedor LIKE '%" + txtBusqueda.Text.Trim() + "%'";
        }

        private void txtBusqueda_Enter(object sender, EventArgs e)
        {
            grid.ClearSelection();
        }

        private void txtBusqueda_KeyUp(object sender, KeyEventArgs e)
        {
            if (txtBusqueda.Text.Trim() == string.Empty) { grid.ClearSelection(); }
        }

        private void btnExportar_Click(object sender, EventArgs e)
        {

        }

        private void txtProveedor_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Mantenimientos
```

```
{
    public partial class frmLocal : Form
    {
        Peru objPeru = new Peru();
        Locales objLocales = new Locales();
        string idLocal = "", idDistrito = "";

        public frmLocal()
        {
            InitializeComponent();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void Limpiar()
        {
            txtNombre.Clear();
            txtDireccion.Clear();
            cboDepartamento.Text = "";
            cboProvincia.Text = "";
            cboDistrito.Text = "";
            grid.ClearSelection();
            idLocal = "";
        }

        private void AdControles(bool sw)
        {
            lblNombre.Enabled = sw;
            txtNombre.Enabled = sw;
            lblDireccion.Enabled = sw;
            txtDireccion.Enabled = sw;
            lblDepartamento.Enabled = sw;
            cboDepartamento.Enabled = sw;
            lblProvincia.Enabled = sw;
            cboProvincia.Enabled = sw;
            lblDistrito.Enabled = sw;
            cboDistrito.Enabled = sw;
            grid.Enabled = !sw;

            btnNuevo.Enabled = !sw;
            btnEstablecer.Enabled = !sw;
```

```
            btnGuardar.Enabled = sw;
            btnCancelar.Enabled = sw;
            btnModificar.Enabled = !sw;
        }

        private void frmLocal_Load(object sender, EventArgs e)
        {
            cboDepartamento.DataSource = objPeru.ListarDepartamentos();
            cboDepartamento.DisplayMember = "Departamento";
            cboDepartamento.ValueMember = "IdDepartamento";
            cboDepartamento.Text = "";

            grid.DataSource = objLocales.ListarLocales();
            grid.Columns[0].Visible = false;
            grid.Columns[3].Visible = false;
            grid.Columns[4].Visible = false;

            grid.Columns[1].HeaderText = "Nombre local";
            grid.Columns[2].HeaderText = "Dirección";
            grid.Columns[5].HeaderText = "Distrito";
            grid.Columns[6].HeaderText = "Activo";

            grid.Columns[1].Width = 250;
            grid.Columns[2].Width = 300;
            grid.Columns[5].Width = 220;
            grid.Columns[6].Width = 65;

            this.Limpiar();
            this.AdControles(false);
        }

        private void cboDepartamento_SelectedIndexChanged(object sender, EventArgs e)
        {
            cboProvincia.DataSource =
objPeru.ListaProvincias(cboDepartamento.SelectedValue.ToString());
            cboProvincia.DisplayMember = "Provincia";
            cboProvincia.ValueMember = "IdProvincia";
            cboProvincia.Text = "";
        }

        private void cboProvincia_SelectedIndexChanged(object sender, EventArgs e)
        {
            cboDistrito.DataSource =
objPeru.ListarDistritos(cboProvincia.SelectedValue.ToString());
```

```
    cboDistrito.DisplayMember = "Distrito";
    cboDistrito.ValueMember = "IdDistrito";
    cboDistrito.Text = "";
}

private void txtNombre_Enter(object sender, EventArgs e)
{
    txtNombre.BackColor = Color.FromArgb(255, 255, 192);
}

private void txtDireccion_Enter(object sender, EventArgs e)
{
    txtDireccion.BackColor = Color.FromArgb(255, 255, 192);
}

private void cboDepartamento_Enter(object sender, EventArgs e)
{
    cboDepartamento.BackColor = Color.FromArgb(255, 255, 192);
}

private void cboProvincia_Enter(object sender, EventArgs e)
{
    cboProvincia.BackColor = Color.FromArgb(255, 255, 192);
}

private void cboDistrito_Enter(object sender, EventArgs e)
{
    cboDistrito.BackColor = Color.FromArgb(255, 255, 192);
}

private void txtNombre_Leave(object sender, EventArgs e)
{
    txtNombre.BackColor = Color.White;
}

private void txtDireccion_Leave(object sender, EventArgs e)
{
    txtDireccion.BackColor = Color.White;
}

private void cboDepartamento_Leave(object sender, EventArgs e)
{
    cboDepartamento.BackColor = Color.White;
}
```

```
    private void cboProvincia_Leave(object sender, EventArgs e)
    {
        cboProvincia.BackColor = Color.White;
    }

    private void cboDistrito_Leave(object sender, EventArgs e)
    {
        cboDistrito.BackColor = Color.White;
    }

    private void btnNuevo_Click(object sender, EventArgs e)
    {
        this.AdControles(true);
        this.Limpiar();
    }

    private void btnCancelar_Click(object sender, EventArgs e)
    {
        if (Sistema.preguntar("¿Está seguro de cancelar?") == DialogResult.Yes)
        {
            this.Limpiar();
            this.AdControles(false);
        }
    }

    private void btnGuardar_Click(object sender, EventArgs e)
    {
        if (txtNombre.Text.Trim() == string.Empty)
        {
            Sistema.informar("Especifique el nombre del local");
            txtNombre.Focus();
        }
        else
        {
            if (Sistema.preguntar("¿Está seguro de agregar un nuevo local?") ==
DialogResult.Yes)
            {
                if (cboDistrito.Text.Trim() == string.Empty) { idDistrito = ""; }

                if (objLocales.EnviarLocales(idLocal, txtNombre.Text, idDistrito,
txtDireccion.Text))
                {
                    grid.DataSource = objLocales.ListarLocales();
```

```
                this.Limpiar();
                this.AdControles(false);
            }
        }
    }
}

private void btnEstablecer_Click(object sender, EventArgs e)
{
    if (grid.Rows.Count == 0)
    {
        Sistema.advertir("No existen elementos para activar");
    }
    else
    {
        if (idLocal.Trim() == string.Empty)
        {
            Sistema.advertir("Seleccione un elemento de la lista");
        }
        else
        {
            if (Sistema.preguntar("¿Está seguro de definir como local activo a:\n" +
grid.CurrentRow.Cells[1].Value.ToString() + "?") == DialogResult.Yes)
            {
                if (objLocales.Activar(idLocal))
                {
                    grid.DataSource = objLocales.ListarLocales();
                    this.Limpiar();
                }
            }
        }
    }
}

private void grid_CellEnter(object sender, DataGridViewCellEventArgs e)
{
    idLocal = grid.CurrentRow.Cells[0].Value.ToString();
}

private void btnModificar_Click(object sender, EventArgs e)
{
    if (grid.Rows.Count == 0)
    {
        Sistema.advertir("No existen elementos para activar");
```

```
            }
            else
            {
                if (idLocal.Trim() == string.Empty)
                {
                    Sistema.advertir("Seleccione un elemento de la lista");
                }
                else
                {
                    idLocal = grid.CurrentRow.Cells[0].Value.ToString();
                    txtNombre.Text = grid.CurrentRow.Cells[1].Value.ToString();
                    txtDireccion.Text = grid.CurrentRow.Cells[2].Value.ToString();
                    cboDepartamento.Text = grid.CurrentRow.Cells[3].Value.ToString();
                    cboProvincia.Text = grid.CurrentRow.Cells[4].Value.ToString();
                    cboDistrito.Text = grid.CurrentRow.Cells[5].Value.ToString();

                    this.AdControles(true);
                    txtNombre.Focus();
                    txtNombre.SelectionStart = txtNombre.Text.Trim().Length;
                }
            }
        }

        private void cboDistrito_SelectedIndexChanged(object sender, EventArgs e)
        {
            idDistrito = cboDistrito.SelectedValue.ToString();
        }

        private void txtDireccion_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
```

```
using BOL;

namespace DESIGNER.Mantenimientos
{
    public partial class frmPersona : Form
    {
        Persona objPersona = new Persona();
        DataTable tPersona = new DataTable();
        string IdPersona = "";

        public bool _Sub = false;

        public frmPersona()
        {
            InitializeComponent();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void Limpiar()
        {
            txtID.Clear();
            txtApellidoPat.Clear();
            txtApellidoMat.Clear();
            txtNombres.Clear();
            cboSexo.SelectedIndex = 0;
            txtFecha.Clear();
            txtDNI.Clear();
            grid.ClearSelection();

            IdPersona = "";
        }

        private void AdControles(bool sw)
        {
            gInfo.Enabled = sw;
            btnNuevo.Enabled = !sw;
            btnGuardar.Enabled = sw;
            btnCancelar.Enabled = sw;
            btnModificar.Enabled = !sw;
            grid.Enabled = !sw;
```

```
            lblFiltro.Enabled = !sw;
            cboTipoFiltro.Enabled = !sw;
            txtValorFiltro.Enabled = !sw;
        }

        private void frmPersona_Load(object sender, EventArgs e)
        {
            if (_Sub) { btnExportar.Visible = true; }

            toolTip.SetToolTip(txtFecha, "Este campo es opcional");

            cboTipoFiltro.Items.Add("APELLIDO PATERNO");
            cboTipoFiltro.Items.Add("APELLIDO MATERNO");
            cboTipoFiltro.Items.Add("NOMBRES");
            cboTipoFiltro.Items.Add("DNI");
            cboTipoFiltro.DropDownStyle = ComboBoxStyle.DropDownList;

            tPersona = objPersona.ListarPersonas();
            grid.DataSource = tPersona;
            grid.Columns[0].Visible = false;

            grid.Columns[1].HeaderText = "Ape. Paterno";
            grid.Columns[2].HeaderText = "Ape. Materno";
            grid.Columns[3].HeaderText = "Nombres";
            grid.Columns[4].HeaderText = "Sexo";
            grid.Columns[5].HeaderText = "DNI";
            grid.Columns[6].HeaderText = "Fecha Nac.";

            grid.Columns[1].Width = 200;
            grid.Columns[2].Width = 200;
            grid.Columns[3].Width = 220;
            grid.Columns[4].Width = 50;
            grid.Columns[4].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            grid.Columns[5].Width = 75;
            grid.Columns[6].Width = 100;
            grid.Columns[6].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;

            grid.Columns[7].Visible = false;
            grid.AlternatingRowsDefaultCellStyle.BackColor = Color.AliceBlue;
            grid.ClearSelection();

            //this.Limpiar();
```

```
            //this.AdControles(false);

            //El formulario incia como proceso NUEVO
            this.Limpiar();
            txtID.Text = objPersona.GenerarID();
            this.AdControles(true);
            //fin del proceso nuevo

            cboTipoFiltro.SelectedIndex = 0;
        }

        private void txtDNI_KeyPress(object sender, KeyPressEventArgs e)
        {
            Sistema.soloNumeros(sender, e, 'E');

            if (e.KeyChar == Convert.ToChar(Keys.Enter))
                this.guardarPersona();
        }

        private void btnNuevo_Click(object sender, EventArgs e)
        {
            this.Limpiar();
            txtID.Text = objPersona.GenerarID();
            this.AdControles(true);

            txtNombres.Focus();
        }

        private void btnCancelar_Click(object sender, EventArgs e)
        {
            if (Sistema.preguntar("¿Está seguro de cancelar el proceso?") ==
DialogResult.Yes)
            {
                this.Limpiar();
                this.AdControles(false);
                cboSexo.SelectedIndex = 0;
            }
        }

        private void btnGuardar_Click(object sender, EventArgs e)
        {
            this.guardarPersona();
        }
```

```
    private void guardarPersona()
    {
        if (txtApellidoPat.Text.Trim() == "" || txtApellidoMat.Text.Trim() == "" ||
txtNombres.Text.Trim() == "")
        {
            Sistema.informar("Ingrese la información solicitada para continuar");
            txtApellidoPat.Focus();
        }
        else
        {
            if (txtDNI.Text.Trim().Length != 8)
            {
                Sistema.advertir("Error en el número de DNI, recuerde que debe poseer 8
dígitos");
                txtDNI.Focus();
            }
            else
            {
                if (Sistema.preguntar("¿Está seguro de guardar este registro?") ==
DialogResult.Yes)
                {
                    Nullable<DateTime> FechaNacimiento = null;
                    if (txtFecha.Text.Trim().Length == 10) { FechaNacimiento =
Convert.ToDateTime(txtFecha.Text); }

                    if (objPersona.EnviarPersona(IdPersona, txtApellidoPat.Text,
txtApellidoMat.Text, txtNombres.Text, Convert.ToChar(cboSexo.Text.Substring(0, 1)),
txtDNI.Text, FechaNacimiento))
                    {
                        tPersona = objPersona.ListarPersonas();
                        grid.DataSource = tPersona;
                        grid.ClearSelection();

                        this.Limpiar();
                        this.AdControles(false);

                        btnNuevo.Focus();
                    }
                }
            }
        }

    private void grid_CellEnter(object sender, DataGridViewCellEventArgs e)
```

```
    {
        IdPersona = grid.CurrentRow.Cells[0].Value.ToString();
    }

    private void btnModificar_Click(object sender, EventArgs e)
    {
        if (grid.Rows.Count > 0)
        {
            if (IdPersona.Trim() == "")
            {
                Sistema.informar("Debe seleccionar una persona de la lista para continuar");
            }
            else
            {
                txtID.Text = IdPersona;

                txtApellidoPat.Text = grid.CurrentRow.Cells[1].Value.ToString();
                txtApellidoMat.Text = grid.CurrentRow.Cells[2].Value.ToString();
                txtNombres.Text = grid.CurrentRow.Cells[3].Value.ToString();

                if (grid.CurrentRow.Cells[4].Value.ToString() == "M") { cboSexo.SelectedIndex
= 0; } else { cboSexo.SelectedIndex = 1; }

                txtDNI.Text = grid.CurrentRow.Cells[5].Value.ToString();
                txtFecha.Text = grid.CurrentRow.Cells[6].Value.ToString();

                this.AdControles(true);
                txtApellidoPat.Focus();
                txtApellidoPat.SelectionStart = txtApellidoPat.Text.Trim().Length;
            }
        }
        else
        {
            Sistema.informar("No existen registros para modificar");
        }
    }

    private void txtValorFiltro_TextChanged(object sender, EventArgs e)
    {
        DataView dv = tPersona.DefaultView;
        string consulta = "";

        switch (cboTipoFiltro.SelectedIndex)
        {
```

```
            case 0: consulta = "[ApellidoPat] LIKE '%" + txtValorFiltro.Text + "%'"; break;
            case 1: consulta = "[ApellidoMat] LIKE '%" + txtValorFiltro.Text + "%'"; break;
            case 2: consulta = "[Nombres] LIKE '%" + txtValorFiltro.Text + "%'"; break;
            case 3: consulta = "[DNI] LIKE '%" + txtValorFiltro.Text + "%'"; break;
        }

        dv.RowFilter = consulta;
    }

    private void txtValorFiltro_KeyUp(object sender, KeyEventArgs e)
    {
        if (txtValorFiltro.Text.Trim() == "") grid.ClearSelection();
    }

    private void cboTipoFiltro_SelectedIndexChanged(object sender, EventArgs e)
    {
        txtValorFiltro.Clear();
        txtValorFiltro.Focus();
    }

    private void txtApellidoPat_Enter(object sender, EventArgs e)
    {
        txtApellidoPat.BackColor = Color.FromArgb(255, 255, 192);
    }

    private void txtApellidoMat_Enter(object sender, EventArgs e)
    {
        txtApellidoMat.BackColor = Color.FromArgb(255, 255, 192);
    }

    private void txtNombres_Enter(object sender, EventArgs e)
    {
        txtNombres.BackColor = Color.FromArgb(255, 255, 192);
    }

    private void txtDNI_Enter(object sender, EventArgs e)
    {
        txtDNI.BackColor = Color.FromArgb(255, 255, 192);
    }

    private void txtFecha_Enter(object sender, EventArgs e)
    {
        txtFecha.BackColor = Color.FromArgb(255, 255, 192);
    }
```

```
        private void txtApellidoPat_Leave(object sender, EventArgs e)
        {
            txtApellidoPat.BackColor = Color.White;
        }

        private void txtApellidoMat_Leave(object sender, EventArgs e)
        {
            txtApellidoMat.BackColor = Color.White;
        }

        private void txtNombres_Leave(object sender, EventArgs e)
        {
            txtNombres.BackColor = Color.White;
        }

        private void txtDNI_Leave(object sender, EventArgs e)
        {
            txtDNI.BackColor = Color.White;
}

        private void txtFecha_Leave(object sender, EventArgs e)
        {
            txtFecha.BackColor = Color.White;
        }

        private void btnExportar_Click(object sender, EventArgs e)
        {

        }

        private void txtFecha_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (e.KeyChar == Convert.ToChar(Keys.Enter))
                this.guardarPersona();
        }

        private void txtNombres_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Mantenimientos
{
    public partial class frmTrabajador : Form
    {
        Trabajador objTrab = new Trabajador();
        string idPersona = "";

        public frmTrabajador()
        {
            InitializeComponent();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void btnCancelar_Click(object sender, EventArgs e)
        {
            idPersona = "";
            txtDniPersona.Clear();
            txtDatosCliente.Clear();

            adControles(false);
        }

        private void btnNuevo_Click(object sender, EventArgs e)
        {
            adControles(true);

            txtID.Text = objTrab.generarID();
            txtDniPersona.Focus();
        }
```

```
    private void frmTrabajador_Load(object sender, EventArgs e)
    {
        grid.DataSource = objTrab.listarTrabajadores();
        grid.Columns[0].Visible = false;
        grid.Columns[1].Width = 400;
        grid.Columns[2].Width = 110;
        grid.ClearSelection();

        adControles(false);
    }

    private void adControles(bool sw)
    {
        gInfo.Enabled = sw;
        btnNuevo.Enabled = !sw;
        btnGuardar.Enabled = sw;
        btnCancelar.Enabled = sw;
        btnExportar.Enabled = sw;
        grid.Enabled = sw;
    }

    private void btnBuscarCliente_Click(object sender, EventArgs e)
    {
        frmPersona f = new frmPersona();
        f._Sub = true;

        if (f.ShowDialog() == DialogResult.OK)
        {
            if (f.grid.CurrentRow != null)
            {
                idPersona = f.grid.CurrentRow.Cells[0].Value.ToString();
                txtDatosCliente.Text = f.grid.CurrentRow.Cells[1].Value.ToString() + " " +
f.grid.CurrentRow.Cells[2].Value.ToString() + " " +
f.grid.CurrentRow.Cells[3].Value.ToString();
                txtDniPersona.Text = f.grid.CurrentRow.Cells[5].Value.ToString();
            }
            else
            {
                Sistema.informar("No seleccionó ningún elemento");
            }
        }
    }
```

```
        private void btnGuardar_Click(object sender, EventArgs e)
        {
            if (idPersona == string.Empty)
            {
                Sistema.advertir("Debe especificar una persona como trabajador");
            }
            else
            {
                if (Sistema.preguntar("¿Está seguro de guardar este trabajador?") ==
DialogResult.Yes)
                {
                    if (objTrab.registrarTrabajador(idPersona))
                    {
                        idPersona = "";
                        txtDniPersona.Clear();
                        txtDatosCliente.Clear();

                        adControles(false);

                        grid.DataSource = objTrab.listarTrabajadores();
                        grid.ClearSelection();
                    }
                    else
                    {
                        Sistema.advertir("No se ha podido completar el proceso, asegúrese que la
persona seleccionada ya posea una cuenta de trabajador");
                    }
                }
            }
        }

        private void txtDatosCliente_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Mantenimientos
{
   public partial class frmMarca : Form
   {
      Marca objMarca = new Marca();
      string IdMarca = "";

      public frmMarca()
      {
         InitializeComponent();
      }

      private void btnCerrar_Click(object sender, EventArgs e)
      {
         this.Close();
      }

      private void frmMarca_Load(object sender, EventArgs e)
      {
         grid.DataSource = objMarca.ListarMarca(1);
         grid.Columns[0].Visible = false;
         grid.Columns[2].Visible = false;
         grid.Columns[1].Width = 285;
         grid.Columns[1].HeaderText = "Nombre de la marca";
         grid.AlternatingRowsDefaultCellStyle.BackColor = Color.AliceBlue;

         grid.ClearSelection();
         IdMarca = "";
      }

      private void btnGuardar_Click(object sender, EventArgs e)
      {
         this.Guardar();
      }

      private void Guardar()
      {
         if (txtNombre.Text.Trim() == string.Empty)
         {
```

```
            Sistema.advertir("Debe completar los datos para continuar");
        }
        else
        {
            if (btnGuardar.Text.Trim() == "Guardar") { IdMarca = ""; }

            if (objMarca.EnviarMarca(IdMarca, txtNombre.Text.Trim()))
            {
                grid.DataSource = objMarca.ListarMarca(1);

                txtNombre.Clear();
                btnGuardar.Text = "Guardar";
                btnEliminar.Text = "Eliminar";
                grid.Enabled = true;
                grid.ClearSelection();
                IdMarca = "";
            }
            else
            {
                Sistema.error("No se ha podido completar el proceso, verifique que el valor
ingresado no haya sido ingresado con anterioridad.");
            }
        }

        txtNombre.Focus();
    }

    private void btnEliminar_Click(object sender, EventArgs e)
    {
        if (btnEliminar.Text.Trim() == "Eliminar")
        {
            if (grid.Rows.Count == 0)
            {
                Sistema.informar("No existen registros para continuar");
            }
            else
            {
                if (IdMarca.Trim() == string.Empty)
                {
                    Sistema.informar("Debe seleccionar un elemento");
                }
                else
                {
                    if (objMarca.Eliminar(IdMarca))
```

```
                {
                    grid.DataSource = objMarca.ListarMarca(1);
                }
            }
        }
    }
    else
    {
        txtNombre.Clear();
        btnGuardar.Text = "Guardar";
        btnEliminar.Text = "Eliminar";
        grid.Enabled = true;
    }

    txtNombre.Focus();
    grid.ClearSelection();
    IdMarca = "";
}

private void optActivos_CheckedChanged(object sender, EventArgs e)
{
    btnEliminar.Enabled = true;
    btnActivar.Visible = false;

    grid.DataSource = objMarca.ListarMarca(1);
    grid.ClearSelection();
    IdMarca = "";
}

private void optInactivos_CheckedChanged(object sender, EventArgs e)
{
    btnEliminar.Enabled = false;
    btnActivar.Visible = true;

    grid.DataSource = objMarca.ListarMarca(0);
    grid.ClearSelection();
    IdMarca = "";
}

private void optTodos_CheckedChanged(object sender, EventArgs e)
{
    btnEliminar.Enabled = false;
    btnActivar.Visible = false;
```

```
    grid.DataSource = objMarca.ListarMarca(null);
    grid.ClearSelection();
    IdMarca = "";
}

private void btnActivar_Click(object sender, EventArgs e)
{
    if (grid.Rows.Count == 0)
    {
        Sistema.informar("No existen registros para continuar");
    }
    else
    {
        if (IdMarca.Trim() == string.Empty)
        {
            Sistema.informar("Debe seleccionar un elemento");
        }
        else
        {
            if (objMarca.Activar(IdMarca))
            {
                grid.DataSource = objMarca.ListarMarca(0);
                grid.ClearSelection();
                IdMarca = "";
            }
        }
    }
}

private void grid_CellDoubleClick(object sender, DataGridViewCellEventArgs e)
{
    txtNombre.Text = grid.CurrentRow.Cells[1].Value.ToString();

    btnGuardar.Text = "Modificar";
    btnEliminar.Text = "Cancelar";

    txtNombre.Focus();
    txtNombre.SelectionStart = txtNombre.Text.Trim().Length;

    grid.Enabled = false;
}

private void grid_CellEnter(object sender, DataGridViewCellEventArgs e)
{
```

```
            IdMarca = grid.CurrentRow.Cells[0].Value.ToString();
        }

        private void txtNombre_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (e.KeyChar == Convert.ToChar(Keys.Enter))
                this.Guardar();
        }

        private void txtNombre_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Mantenimientos
{
    public partial class frmMagnitud : Form
    {
        Magnitud objMagnitud = new Magnitud();
        string IdMagnitud = "";

        public frmMagnitud()
        {
            InitializeComponent();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            this.Close();
        }
```

```
    private void frmMagnitud_Load(object sender, EventArgs e)
    {
        grid.DataSource = objMagnitud.ListarMagnitudes(1);
        grid.Columns[0].Visible = false;
        grid.Columns[2].Visible = false;
        grid.Columns[1].Width = 285;
        grid.Columns[1].HeaderText = "Nombre de la marca";
        grid.AlternatingRowsDefaultCellStyle.BackColor = Color.AliceBlue;

        grid.ClearSelection();
        IdMagnitud = "";
    }

    private void Guardar()
    {
        if (txtNombre.Text.Trim() == string.Empty)
        {
            Sistema.advertir("Debe completar los datos para continuar");
        }
        else
        {
            if (btnGuardar.Text.Trim() == "Guardar") { IdMagnitud = ""; }

            if (objMagnitud.EnviarMarca(IdMagnitud,txtNombre.Text.Trim()))
            {
                grid.DataSource = objMagnitud.ListarMagnitudes(1);

                txtNombre.Clear();
                btnGuardar.Text = "Guardar";
                btnEliminar.Text = "Eliminar";
                grid.Enabled = true;
                grid.ClearSelection();
                IdMagnitud = "";
            }
            else
            {
                Sistema.error("No se ha podido completar el proceso, verifique que el valor
ingresado no haya sido ingresado con anterioridad.");
            }
        }

        txtNombre.Focus();
    }
```

```
private void btnEliminar_Click(object sender, EventArgs e)
{
    if (btnEliminar.Text.Trim() == "Eliminar")
    {
        if (grid.Rows.Count == 0)
        {
            Sistema.informar("No existen registros para continuar");
        }
        else
        {
            if (IdMagnitud.Trim() == string.Empty)
            {
                Sistema.informar("Debe seleccionar un elemento");
            }
            else
            {
                if (objMagnitud.Eliminar(IdMagnitud))
                {
                    grid.DataSource = objMagnitud.ListarMagnitudes(1);
                }
            }
        }
    }
    else
    {
        txtNombre.Clear();
        btnGuardar.Text = "Guardar";
        btnEliminar.Text = "Eliminar";
        grid.Enabled = true;
    }

    txtNombre.Focus();
    grid.ClearSelection();
    IdMagnitud = "";
}

private void optActivos_CheckedChanged(object sender, EventArgs e)
{
    btnEliminar.Enabled = true;
    btnActivar.Visible = false;

    grid.DataSource = objMagnitud.ListarMagnitudes(1);
    grid.ClearSelection();
    IdMagnitud = "";
```

```
}

private void optInactivos_CheckedChanged(object sender, EventArgs e)
{
    btnEliminar.Enabled = false;
    btnActivar.Visible = true;

    grid.DataSource = objMagnitud.ListarMagnitudes(0);
    grid.ClearSelection();
    IdMagnitud = "";
}

private void optTodos_CheckedChanged(object sender, EventArgs e)
{
    btnEliminar.Enabled = false;
    btnActivar.Visible = false;

    grid.DataSource = objMagnitud.ListarMagnitudes(null);
    grid.ClearSelection();
    IdMagnitud = "";
}

private void btnActivar_Click(object sender, EventArgs e)
{
    if (grid.Rows.Count == 0)
    {
        Sistema.informar("No existen registros para continuar");
    }
    else
    {
        if (IdMagnitud.Trim() == string.Empty)
        {
            Sistema.informar("Debe seleccionar un elemento");
        }
        else
        {
            if (objMagnitud.Activar(IdMagnitud))
            {
                grid.DataSource = objMagnitud.ListarMagnitudes(0);
                grid.ClearSelection();
                IdMagnitud = "";
            }
        }
    }
```

```
        }

        private void grid_CellDoubleClick(object sender, DataGridViewCellEventArgs e)
        {
            txtNombre.Text = grid.CurrentRow.Cells[1].Value.ToString();

            btnGuardar.Text = "Modificar";
            btnEliminar.Text = "Cancelar";

            txtNombre.Focus();
            txtNombre.SelectionStart = txtNombre.Text.Trim().Length;

            grid.Enabled = false;
        }

        private void grid_CellEnter(object sender, DataGridViewCellEventArgs e)
        {
            IdMagnitud = grid.CurrentRow.Cells[0].Value.ToString();
        }

        private void btnGuardar_Click(object sender, EventArgs e)
        {
            this.Guardar();
        }

        private void txtNombre_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (e.KeyChar == Convert.ToChar(Keys.Enter))
                this.Guardar();
        }

        private void txtNombre_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Procesos
{
   public partial class frmRegistraUsuario : Form
   {

      Usuario objUsuario = new Usuario();
      string IdUsuario = "", IdPersona = "";
      byte na = 1;

      public frmRegistraUsuario()
      {
         InitializeComponent();
      }

      private void btnCerrar_Click(object sender, EventArgs e)
      {
         this.Close();
      }

      private void AdControles(bool sw)
      {
         gDatos.Enabled = sw;
         btnNuevo.Enabled = !sw;
         btnGuardar.Enabled = sw;
         btnModificar.Enabled = !sw;
         btnCancelar.Enabled = sw;
         btnEliminar.Enabled = !sw;
         grid.Enabled = !sw;
      }

      private void Limpiar()
      {
         txtID.Clear();
         txtPersona.Clear();
         txtNombreUsuario.Clear();
         txtClave1.Clear();
         txtClave2.Clear();
         optAdministrador.Checked = true;
         IdUsuario = "";
```

```
            IdPersona = "";
        }

        private void frmRegistraUsuario_Load(object sender, EventArgs e)
        {
            this.Limpiar();
            this.AdControles(false);

            grid.DataSource = objUsuario.ListarUsuarios();
            grid.Columns[0].Visible = false;
            grid.Columns[1].Width = 310;
            grid.Columns[2].HeaderText = "Usuario";
            grid.Columns[3].Visible = false;
            grid.Columns[4].HeaderText = "Administrador";
            grid.ClearSelection();

            IdUsuario = "";
            IdPersona = "";
        }

        private void btnNuevo_Click(object sender, EventArgs e)
        {
            this.AdControles(true);
            this.Limpiar();

            txtID.Text = objUsuario.GenerarID();
            grid.ClearSelection();
        }

        private void btnCancelar_Click(object sender, EventArgs e)
        {
            if (Sistema.preguntar("¿Está seguro de cancelar el proceso?") ==
DialogResult.Yes)
            {
                this.Limpiar();
                this.AdControles(false);
                grid.ClearSelection();
            }
        }

        private void btnBuscar_Click(object sender, EventArgs e)
        {
            Mantenimientos.frmPersona f = new Mantenimientos.frmPersona();
            f._Sub = true;
```

```
            f.ShowDialog();

            if (f.grid.CurrentRow != null)
            {
                IdPersona = f.grid.CurrentRow.Cells[0].Value.ToString();
                txtPersona.Text = f.grid.CurrentRow.Cells[1].Value.ToString() + " " +
f.grid.CurrentRow.Cells[2].Value.ToString() + " " +
f.grid.CurrentRow.Cells[3].Value.ToString();
                txtNombreUsuario.Focus();
            }
            else
            {
                IdPersona = "";
                txtPersona.Clear();
            }
        }

        private void btnGuardar_Click(object sender, EventArgs e)
        {
            if (IdPersona == string.Empty)
            {
                Sistema.informar("Debe especificar una persona");
            }
            else
            {
                if (txtNombreUsuario.Text.Trim() == string.Empty)
                {
                    Sistema.informar("Debe escribir el nombre de usuario");
                    txtNombreUsuario.Focus();
                }
                else
                {
                    if (txtClave1.Text.Trim() == "" || (txtClave1.Text.Trim() !=
txtClave2.Text.Trim()))
                    {
                        Sistema.informar("Las contraseñas no deben estar vacías y ser iguales");
                        txtClave1.Focus();
                    }
                    else
                    {
                        if (Sistema.preguntar("¿Está seguro de registrar este nuevo usuario?") ==
DialogResult.Yes)
                        {
```

```
                    if (objUsuario.RegistrarUsuario(IdPersona,
txtNombreUsuario.Text.Trim(), txtClave1.Text.Trim(), na))
                    {
                        grid.DataSource = objUsuario.ListarUsuarios();
                        grid.ClearSelection();
                        this.Limpiar();
                        this.AdControles(false);
                    }
                    else
                    {
                        Sistema.error("No se ha podido completar el proceso, verifique los
datos ingresados.\nEl nombre de usuario no debe repetirse");
                    }
                }
            }
        }
    }

    private void optAdministrador_CheckedChanged(object sender, EventArgs e)
    {
        na = 1;
    }

    private void optInvitado_CheckedChanged(object sender, EventArgs e)
    {
        na = 0;
    }

    private void btnModificar_Click(object sender, EventArgs e)
    {
        string nivelAcceso = "USUARIO", reves = "ADMINISTRADOR";
        byte nAcceso = 0;

        if (IdUsuario != "")
        {
            if (Convert.ToBoolean(grid.CurrentRow.Cells[4].Value))
            {
                nivelAcceso = "ADMINISTRADOR";
                reves = "USUARIO";
                nAcceso = 1;
            }
```

```
            if (Sistema.preguntar("\t\tAVISO IMPORTANTE\t\t\t\n\nEl usuario: " +
grid.CurrentRow.Cells[1].Value.ToString() + " posee una cuenta de nivel " + nivelAcceso +
". ¿Está seguro de cambiar el nivel de\nacceso a " + reves + "?") == DialogResult.Yes)
            {
                if (objUsuario.CambiarNivelAcceso(IdUsuario, 0))
                {
                    grid.DataSource = objUsuario.ListarUsuarios();
                }
            }
        }
        else
        {
            Sistema.informar("Debe seleccionar un elemento para continuar");
        }
    }

    private void txtNombreUsuario_Enter(object sender, EventArgs e)
    {
        txtNombreUsuario.BackColor = Color.FromArgb(192, 255, 192);
    }

    private void txtClave1_Enter(object sender, EventArgs e)
    {
        txtClave1.BackColor = Color.FromArgb(192, 255, 192);
    }

    private void txtClave2_Enter(object sender, EventArgs e)
    {
        txtClave2.BackColor = Color.FromArgb(192, 255, 192);
    }

    private void txtNombreUsuario_Leave(object sender, EventArgs e)
    {
        txtNombreUsuario.BackColor = Color.White;
    }

    private void txtClave1_Leave(object sender, EventArgs e)
    {
        txtClave1.BackColor = Color.White;
    }

    private void txtClave2_Leave(object sender, EventArgs e)
    {
        txtClave2.BackColor = Color.White;
```

```
        }

        private void grid_CellEnter(object sender, DataGridViewCellEventArgs e)
        {
            IdUsuario = grid.CurrentRow.Cells[1].Value.ToString();
        }

        private void txtPersona_TextChanged(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Procesos
{
    public partial class frmRegistroSerie : Form
    {
        Compra objCompra = new Compra();
        int filas = 0, totalFilas;
        string codigoActivo, detalleCompra;

        public frmRegistroSerie()
        {
            InitializeComponent();
        }

        private void frmRegistroSerie_Load(object sender, EventArgs e)
        {
            gridCodigos.ClearSelection();

            codigoActivo = txtCodigoBarra.Text.Trim();
            txtCodigoBarra.Font = new Font("barcode font", 72);
```

```
            //Estos datos son importantes para la autogeneración de filas
            totalFilas = gridCodigos.Rows.Count;
            detalleCompra = gridCodigos.Rows[0].Cells[1].Value.ToString();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void txtCodigoBarra_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (e.KeyChar == Convert.ToChar(Keys.Enter))
            {
                if (filas < gridCodigos.Rows.Count)
                {
                    //No podemos agregar un código repetido
                    int j = 0;
                    bool repetido = false;

                    while (j < gridCodigos.Rows.Count && !repetido)
                    {
                        if (txtCodigoBarra.Text.Trim() ==
gridCodigos.Rows[j].Cells[2].Value.ToString().Trim())
                        {
                            repetido = true;
                        }
                        j++;
                    }

                    if (!repetido)
                    {
                        gridCodigos.Rows[filas].Cells[2].Value = txtCodigoBarra.Text.Trim();
                        txtCodigoBarra.Clear();
                        txtCodigoBarra.Focus();
                        filas++;
                    }
                    else
                    {
                        txtCodigoBarra.Clear();
                        Sistema.advertir("Este código ya ha sido registrado en la lista");
                        txtCodigoBarra.Focus();
                    }
```

```
        }
    }
}

private void lblLimpiar_Click(object sender, EventArgs e)
{
    txtCodigoBarra.Clear();
    txtCodigoBarra.Focus();
}

private void lblLimpiar_MouseEnter(object sender, EventArgs e)
{
    lblLimpiar.Font = new Font(lblLimpiar.Font,FontStyle.Underline);
}

private void lblLimpiar_MouseLeave(object sender, EventArgs e)
{
    lblLimpiar.Font = new Font(lblLimpiar.Font, FontStyle.Regular);
}

private void btnReiniciar_Click(object sender, EventArgs e)
{
    filas = 0;

    for (int i = 0; i < gridCodigos.Rows.Count; i++)
    {
        gridCodigos.Rows[i].Cells[2].Value = "";
    }
    txtCodigoBarra.Clear();
    txtCodigoBarra.Focus();
}

private void btnGuardar_Click(object sender, EventArgs e)
{
    int contador = 0;

    for (int i = 0; i < gridCodigos.Rows.Count; i++)
    {
        if (gridCodigos.Rows[i].Cells[2].Value.ToString() != string.Empty)
        {
            contador++;
        }
    }
```

```
            if (contador == gridCodigos.Rows.Count)
            {
                for (int j = 0; j < gridCodigos.Rows.Count; j++)
                {
                    objCompra.RegistrarSerie(gridCodigos.Rows[j].Cells[1].Value.ToString(),
gridCodigos.Rows[j].Cells[2].Value.ToString());
                }

                Sistema.informar("Se han registrado los código de barra/ número de serie
correctamente");
                this.Close();
            }
            else
            {
                Sistema.informar("Faltan especificar algunos códigos");
            }
        }

        private void chkCodigoUnico_CheckedChanged(object sender, EventArgs e)
        {
            //En cualquier caso el GRID se reinicia
            filas = 0;
            gridCodigos.Rows.Clear();

            if (chkCodigoUnico.Checked)
            {
                gridCodigos.Rows.Add("1",detalleCompra,"");
            }
            else
            {
                //Reestucturar todo de nuevo
                for (int i = 0; i < totalFilas; i++)
                {
                    gridCodigos.Rows.Add((gridCodigos.Rows.Count + 1).ToString(),
detalleCompra, "");
                }
            }

            gridCodigos.ClearSelection();
            txtCodigoBarra.Focus();
        }

        private void txtCodigoBarra_TextChanged(object sender, EventArgs e)
        {
```

```
        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace DESIGNER.Reportes
{
    public partial class frmAdminReportes : Form
    {
        public frmAdminReportes()
        {
            InitializeComponent();
        }

        private void btnReporte1_Click(object sender, EventArgs e)
        {
            frmReporteVenta f = new frmReporteVenta();
            f.ShowDialog();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void btnReporte2_Click(object sender, EventArgs e)
        {
            frmReporteTrabajador f = new frmReporteTrabajador();
            f.ShowDialog();
        }

        private void btnReporte3_Click(object sender, EventArgs e)
        {
            Reportes.frmReporteVentaDetalladado f = new frmReporteVentaDetalladado();
            f.ShowDialog();
        }
```

```
        private void btnReporte4_Click(object sender, EventArgs e)
        {
            frmReporteExistencias f = new frmReporteExistencias();
            f.ShowDialog();
        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace DESIGNER.Reportes
{
    public partial class frmReporte : Form
    {
        public frmReporte()
        {
            InitializeComponent();
        }

        private void visor_Load(object sender, EventArgs e)
        {

        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;
```

```
namespace DESIGNER
{
    public partial class frmLogin : Form
    {
        Usuario objUsuario = new Usuario();
        DataTable tUsuario = new DataTable();

        public frmLogin()
        {
            InitializeComponent();
        }

        private void btnCerrar_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void Acceder()
        {
            string usuario = txtUsuario.Text.Trim();
            string clave = txtClave.Text.Trim();

            if (usuario == string.Empty)
            {
                error.SetError(txtUsuario, "Escriba el nombre de usuario");
                txtUsuario.Focus();
            }
            else
            {
                if (clave == string.Empty)
                {
                    error.SetError(txtClave, "Escriba la contraseña de usuario");
                    txtClave.Focus();
                }
                else
                {
                    tUsuario = objUsuario.Login(usuario, clave);

                    if (tUsuario.Rows.Count == 0)
                    {
                        Sistema.error("Los datos ingresados son incorrectos");
                    }
                    else
                    {
```

```
                //Datos correctos
                if (Convert.ToBoolean(tUsuario.Rows[0][4].ToString()))
                {
                    Sistema._Administrador = true;
                }

                Sistema._IdUsuario = tUsuario.Rows[0][0].ToString();
                Sistema._NombreUsuario = tUsuario.Rows[0][2].ToString();
                Sistema._Clave = tUsuario.Rows[0][3].ToString();

                frmPrincipal f = new frmPrincipal();
                f.Show();
                this.Hide();
            }
        }
    }
}

private void btnIngresar_Click(object sender, EventArgs e)
{
    this.Acceder();
}

private void txtUsuario_KeyPress(object sender, KeyPressEventArgs e)
{
    if (e.KeyChar == Convert.ToChar(Keys.Enter)) { txtClave.Focus(); }
}

private void txtClave_KeyPress(object sender, KeyPressEventArgs e)
{
    if (e.KeyChar == Convert.ToChar(Keys.Enter)) { this.Acceder(); }
}

private void btnEntrar_Click(object sender, EventArgs e)
{
    this.Acceder();
}

private void frmLogin_Load(object sender, EventArgs e)
{

}
}
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

namespace DESIGNER
{
   static class Program
   {
      /// <summary>
      /// Punto de entrada principal para la aplicación.
      /// </summary>
      [STAThread]
      static void Main()
      {
         Application.EnableVisualStyles();
         Application.SetCompatibleTextRenderingDefault(false);
         Application.Run(new frmLogin());
      }
   }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Reportes
{
   public partial class frmReporteTrabajador : Form
   {
      DataTable dtVenta = new DataTable();
      Venta objVenta = new Venta();

      //Instancia para emitir reportes
      Reportes.rptReporteVendedorDia reporte = new rptReporteVendedorDia();
      Reportes.rptReporteVendedorIntervalo reporteInt = new
rptReporteVendedorIntervalo();
```

```
    char tipo = 'D';

    public frmReporteTrabajador()
    {
        InitializeComponent();
    }

    private void btnBuscar_Click(object sender, EventArgs e)
    {
        if (!chkFin.Checked)
        {
            try
            {
                dtVenta = objVenta.ventaDia(Convert.ToDateTime(txtInicio.Text.Trim()));

                //Se cargan los datos
                reporte.SetDataSource(dtVenta);
                reporte.Refresh();
                reporte.SetParameterValue("local", Sistema._NombreLocal);

                if (dtVenta.Rows.Count > 0)
                {
                    tipo = 'D';
                    reporte.SetParameterValue("subtitulo", "Ventas del día: " + txtInicio.Text);
                    visor.ReportSource = reporte;
                }
                else
                {
                    reporte.SetParameterValue("subtitulo", "No existen registros para el día " +
txtInicio.Text);
                    Sistema.informar("No existen registros de venta para la fecha: " +
txtInicio.Text);
                    txtInicio.Focus();
                }

                visor.Refresh();
            }
            catch
            {
                Sistema.error("Debe ingresar la fecha utilizando el formato: dd/mm/aaaa");
                txtInicio.Focus();
            }
        }
        else
```

```
        {
            //se busca en un intervalo de fechas
            try
            {

                dtVenta =
objVenta.ventaIntervalos(Convert.ToDateTime(txtInicio.Text.Trim()),
Convert.ToDateTime(txtTermino.Text.Trim()));

                //Se cargan los datos
                reporteInt.SetDataSource(dtVenta);
                reporteInt.Refresh();
                reporteInt.SetParameterValue("local", Sistema._NombreLocal);

                if (dtVenta.Rows.Count > 0)
                {
                    tipo = 'I';
                    reporteInt.SetParameterValue("subtitulo", "Ventas desde el día: " +
txtInicio.Text + " hasta " + txtTermino.Text);
                    visor.ReportSource = reporteInt;
                }
                else
                {
                    reporteInt.SetParameterValue("subtitulo", "No existen registros");
                    Sistema.informar("No existen registros de venta para el intervalo
especificado");
                    txtInicio.Focus();
                }

                visor.Refresh();
            }
            catch
            {
                Sistema.error("Debe ingresar la fecha utilizando el formato: dd/mm/aaaa");
                txtInicio.Focus();
            }
        }
    }

    private void chkFin_CheckedChanged(object sender, EventArgs e)
    {
        txtTermino.Enabled = chkFin.Checked;

        if (!chkFin.Checked) { txtTermino.Clear(); }
```

```
    }

    private void btnReiniciar_Click(object sender, EventArgs e)
    {
        txtInicio.Clear();
        chkFin.Checked = false;
        txtTermino.Clear();
        dtVenta = null;
        visor.ReportSource = null;
        visor.Refresh();

        txtInicio.Focus();
    }

    private void btnPDF_Click(object sender, EventArgs e)
    {
        if (dtVenta.Rows.Count == 0)
        {
            Sistema.advertir("No existen registros para exportar");
            txtInicio.Focus();
        }
        else
        {
            if (tipo == 'D')
            {
                SaveFileDialog sd = new SaveFileDialog();
                sd.Title = "Exportar datos como PDF - Ventas";
                sd.FileName = "Record del día " + txtInicio.Text.Trim().Replace('/', '-');
                sd.Filter = "Archivo en formato PDF (*.PDF)|*.pdf";

                if (sd.ShowDialog() == DialogResult.OK && sd.FileName.Trim() !=
string.Empty)
                {

reporte.ExportToDisk(CrystalDecisions.Shared.ExportFormatType.PortableDocFormat,
sd.FileName);
                    Sistema.informar("Se ha terminado de crear el archivo correctamente");
                }
            }
            else
            {
                //Por intervalo
                SaveFileDialog sd = new SaveFileDialog();
                sd.Title = "Exportar datos como PDF - Ventas";
```

```
                sd.FileName = "Record del " + txtInicio.Text.Trim().Replace('/', '-') + " al " +
txtTermino.Text.Trim().Replace('/', '-');
                sd.Filter = "Archivo en formato PDF (*.PDF)|*.pdf";

                if (sd.ShowDialog() == DialogResult.OK && sd.FileName.Trim() !=
string.Empty)
                {

reporteInt.ExportToDisk(CrystalDecisions.Shared.ExportFormatType.PortableDocFormat,
sd.FileName);
                    Sistema.informar("Se ha terminado de crear el archivo correctamente");
                }
            }
        }
    }

    private void btnXLS_Click(object sender, EventArgs e)
    {
        if (dtVenta.Rows.Count == 0)
        {
            Sistema.advertir("No existen registros para exportar");
            txtInicio.Focus();
        }
        else
        {
            if (tipo == 'D')
            {
                SaveFileDialog sd = new SaveFileDialog();
                sd.Title = "Exportar datos como XLS - Ventas";
                sd.FileName = "Record del día " + txtInicio.Text.Trim().Replace('/', '-');
                sd.Filter = "Microsoft Office Excel (*.XLS)|*.xls";

                if (sd.ShowDialog() == DialogResult.OK && sd.FileName.Trim() !=
string.Empty)
                {
                    reporte.ExportToDisk(CrystalDecisions.Shared.ExportFormatType.Excel,
sd.FileName);
                    Sistema.informar("Se ha terminado de crear el archivo correctamente");
                }
            }
            else
            {
                //Por intervalo
                SaveFileDialog sd = new SaveFileDialog();
```

```
                sd.Title = "Exportar datos como XLS - Ventas";
                sd.FileName = "Record del " + txtInicio.Text.Trim().Replace('/', '-') + " al " +
txtTermino.Text.Trim().Replace('/', '-');
                sd.Filter = "Microsoft Office Excel (*.XLS)|*.xls";

                if (sd.ShowDialog() == DialogResult.OK && sd.FileName.Trim() !=
string.Empty)
                {
                    reporteInt.ExportToDisk(CrystalDecisions.Shared.ExportFormatType.Excel,
sd.FileName);
                    Sistema.informar("Se ha terminado de crear el archivo correctamente");
                }
            }
        }
    }

    private void panel1_Paint(object sender, PaintEventArgs e)
    {

    }
  }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Reportes
{
  public partial class frmReporteVenta : Form
  {
    DataTable dtVenta = new DataTable();
    Venta objVenta = new Venta();

    //Instancia para emitir reportes
    Reportes.rptReporteVentaDiario reporte = new rptReporteVentaDiario();
    Reportes.rptReporteVentaIntervalo reporteInt = new rptReporteVentaIntervalo();
```

```
    char tipo = 'D';

    public frmReporteVenta()
    {
        InitializeComponent();
    }

    private void frmReporteVenta_Load(object sender, EventArgs e)
    {
        txtTermino.Enabled = false;
    }

    private void chkFin_CheckedChanged(object sender, EventArgs e)
    {
        txtTermino.Enabled = chkFin.Checked;

        if (!chkFin.Checked) { txtTermino.Clear(); }
    }

    private void btnBuscar_Click(object sender, EventArgs e)
    {
        if (!chkFin.Checked)
        {
            try
            {
                dtVenta = objVenta.ventaDia(Convert.ToDateTime(txtInicio.Text.Trim()));

                //Se cargan los datos
                reporte.SetDataSource(dtVenta);
                reporte.Refresh();
                reporte.SetParameterValue("local", Sistema._NombreLocal);

                if (dtVenta.Rows.Count > 0)
                {
                    tipo = 'D';
                    reporte.SetParameterValue("subtitulo", "Ventas del día: " + txtInicio.Text);
                    visor.ReportSource = reporte;
                }
                else
                {
                    reporte.SetParameterValue("subtitulo", "No existen registros para el día " +
txtInicio.Text);
                    Sistema.informar("No existen registros de venta para la fecha: " +
txtInicio.Text);
```

```
                txtInicio.Focus();
            }

            visor.Refresh();
        }
        catch
        {
            Sistema.error("Debe ingresar la fecha utilizando el formato: dd/mm/aaaa");
            txtInicio.Focus();
        }
    }
    else
    {
        //se busca en un intervalo de fechas
        try
        {

            dtVenta =
objVenta.ventaIntervalos(Convert.ToDateTime(txtInicio.Text.Trim()),
Convert.ToDateTime(txtTermino.Text.Trim()));

            //Se cargan los datos
            reporteInt.SetDataSource(dtVenta);
            reporteInt.Refresh();
            reporteInt.SetParameterValue("local", Sistema._NombreLocal);

            if (dtVenta.Rows.Count > 0)
            {
                tipo = 'I';
                reporteInt.SetParameterValue("subtitulo", "Ventas desde el día: " +
txtInicio.Text + " hasta " + txtTermino.Text);
                visor.ReportSource = reporteInt;
            }
            else
            {
                reporteInt.SetParameterValue("subtitulo", "No existen registros");
                Sistema.informar("No existen registros de venta para el intervalo
especificado");
                txtInicio.Focus();
            }

            visor.Refresh();
        }
        catch
```

```
            {
                Sistema.error("Debe ingresar la fecha utilizando el formato: dd/mm/aaaa");
                txtInicio.Focus();
            }
        }
    private void btnReiniciar_Click(object sender, EventArgs e)
    {
        txtInicio.Clear();
        chkFin.Checked = false;
        txtTermino.Clear();
        dtVenta = null;
        visor.ReportSource = null;
        visor.Refresh();

        txtInicio.Focus();
    }

    private void btnPDF_Click(object sender, EventArgs e)
    {
        if (dtVenta.Rows.Count == 0)
        {
            Sistema.advertir("No existen registros para exportar");
            txtInicio.Focus();
        }
        else
        {
            if (tipo == 'D')
            {
                SaveFileDialog sd = new SaveFileDialog();
                sd.Title = "Exportar datos como PDF - Ventas";
                sd.FileName = "Ventas del día " + txtInicio.Text.Trim().Replace('/', '-');
                sd.Filter = "Archivo en formato PDF (*.PDF)|*.pdf";

                if (sd.ShowDialog() == DialogResult.OK && sd.FileName.Trim() !=
string.Empty)
                {

reporte.ExportToDisk(CrystalDecisions.Shared.ExportFormatType.PortableDocFormat,
sd.FileName);
                    Sistema.informar("Se ha terminado de crear el archivo correctamente");
                }
            }
            else
            {
```

```
                //Por intervalo
                SaveFileDialog sd = new SaveFileDialog();
                sd.Title = "Exportar datos como PDF - Ventas";
                sd.FileName = "Ventas del " + txtInicio.Text.Trim().Replace('/', '-') + " al " +
txtTermino.Text.Trim().Replace('/', '-');
                sd.Filter = "Archivo en formato PDF (*.PDF)|*.pdf";

                if (sd.ShowDialog() == DialogResult.OK && sd.FileName.Trim() !=
string.Empty)
                {

reporteInt.ExportToDisk(CrystalDecisions.Shared.ExportFormatType.PortableDocFormat,
sd.FileName);
                    Sistema.informar("Se ha terminado de crear el archivo correctamente");
                }
            }
        }
    }

    private void btnXLS_Click(object sender, EventArgs e)
    {
        if (dtVenta.Rows.Count == 0)
        {
            Sistema.advertir("No existen registros para exportar");
            txtInicio.Focus();
        }
        else
        {
            if (tipo == 'D')
            {
                SaveFileDialog sd = new SaveFileDialog();
                sd.Title = "Exportar datos como XLS - Ventas";
                sd.FileName = "Ventas del día " + txtInicio.Text.Trim().Replace('/', '-');
                sd.Filter = "Microsoft Office Excel (*.XLS)|*.xls";

                if (sd.ShowDialog() == DialogResult.OK && sd.FileName.Trim() !=
string.Empty)
                {
                    reporte.ExportToDisk(CrystalDecisions.Shared.ExportFormatType.Excel,
sd.FileName);
                    Sistema.informar("Se ha terminado de crear el archivo correctamente");
                }
            }
            else
```

```
            {
                //Por intervalo
                SaveFileDialog sd = new SaveFileDialog();
                sd.Title = "Exportar datos como XLS - Ventas";
                sd.FileName = "Ventas del " + txtInicio.Text.Trim().Replace('/', '-') + " al " +
txtTermino.Text.Trim().Replace('/', '-');
                sd.Filter = "Microsoft Office Excel (*.XLS)|*.xls";

                if (sd.ShowDialog() == DialogResult.OK && sd.FileName.Trim() !=
string.Empty)
                {
                    reporteInt.ExportToDisk(CrystalDecisions.Shared.ExportFormatType.Excel,
sd.FileName);
                    Sistema.informar("Se ha terminado de crear el archivo correctamente");
                }
            }
        }
    }

    private void panel1_Paint(object sender, PaintEventArgs e)
    {

    }
  }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Data.SqlClient;
using BOL;

namespace DESIGNER.Reportes
{
  public partial class frmReporteVentaDetalladado : Form
  {
    Venta objVenta = new Venta();
    DataTable tabla = new DataTable();
```

```
//Variables para la búsqueda
DateTime fecha1;
Nullable<DateTime> fecha2 = null;

public frmReporteVentaDetalladado()
{
   InitializeComponent();
}

private void chkFin_CheckedChanged(object sender, EventArgs e)
{
   if (chkFin.Checked)
   {
      txtTermino.Clear();
      txtTermino.Enabled = true;
   }
   else
   {
      txtTermino.Enabled = false;
   }
}

private void btnBuscar_Click(object sender, EventArgs e)
{
   try
   {
      if (chkFin.Checked)
      {
         if (txtTermino.Text.Trim().Length == 10)
         {
            fecha2 = Convert.ToDateTime(txtTermino.Text.Trim());
         }
         else
         {
            Sistema.error("Ingrese la fecha correctamente: dd/mm/aaaa");
            return;
         }
      }
      else
      {
         fecha2 = null;
      }
```

```
            Reportes.rptReporteDiarioDetallado reporte = new
Reportes.rptReporteDiarioDetallado();
            tabla = objVenta.VerReporteVentaDetallado(Convert.ToDateTime(txtInicio.Text),
fecha2);

            //Se cargan los datos
            reporte.SetDataSource(tabla);
            reporte.Refresh();
            //reporte.SetParameterValue("local", Sistema._NombreLocal);

            if (tabla.Rows.Count > 0)
            {
               if (chkFin.Checked)
               {
                  reporte.SetParameterValue("subtitulo", "Ventas del " + txtInicio.Text + " al "
+ txtTermino.Text);
               }
               else
            {
                  reporte.SetParameterValue("subtitulo", "Ventas del día: " + txtInicio.Text);
               }

               visor.ReportSource = reporte;
            }
            else
            {
               reporte.SetParameterValue("subtitulo", "No existen registros para el día " +
txtInicio.Text);
               Sistema.informar("No existen registros de venta para la fecha: " +
txtInicio.Text);
               txtInicio.Focus();
            }

            visor.Refresh();
         }
         catch
         {
            Sistema.error("Debe ingresar la fecha utilizando el formato: dd/mm/aaaa");
            txtInicio.Focus();
         }
      }

      private void label4_Click(object sender, EventArgs e)
            }
```

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

- ✓ Con la implementación del Sistema Computarizado para la Administración del Departamento Comercial, se han obtenido resultados satisfactorios para la Librería y Distribuidora Sánchez S.R.L, con lo que se cumple con el objetivo general ya que ha contribuido de manera eficiente en la automatización de la información y de los procesos de la institución.

- ✓ La participación del cliente ha sido un aspecto clave en el éxito de la implementación del sistema. Las reuniones que se realizaron ayudaron a la definición y validación de los requerimientos del sistema.

- ✓ El desarrollo de los prototipos del sistema, agilizó la implementación del sistema ayudando al entendimiento del cliente, para las modificaciones requeridas tanto de interfaz como de funcionalidad.

- ✓ El Sistema para el departamento Comercial fue desarrollado con el lenguaje de programación C# y como gestor de base de datos con Microsoft SQL Server Standard Edition, los cuales permitieron desarrollar un sistema robusto y seguro.

- ✓ Mediante la realización de las encuestas se concluye que el Sistema Computarizado para el área Comercial, minimiza el tiempo de atención a los clientes, minimiza el tiempo de dar repuesta a cada uno de las actividades de la empresa, minimiza el tiempo de búsqueda de los productos, genera una comunicación más eficiente, confiable y segura en el manejo de la información en el área comercial.

- ✓ El desarrollo de los reportes estadísticos permiten visualizar información la cual ayuda a una toma de decisiones adecuada.

- ✓ El servidor NTP cumple un papel importante en el correcto funcionamiento del sistema, ya que ayuda a tener la hora correcta, sin poder ser modifcada para intereses propios de algún usuario.

RECOMENDACIONES

- ✓ El Sistema Computarizado para el área Comercial nos permite ver la necesidad de automatizar los demás procesos de la institución ajenos al área Comercial. Por lo que se recomienda crear una cartera de proyectos el cual le permita identificar, priorizar, administrar y controlar para lograr los objetivos de la empresa.

- ✓ Realizar una evaluación periódica de la interacción entre el usuario y el sistema, con el objetivo de realizar mejoras en las funcionalidades del sistema.

- ✓ Realizar el proceso de mantenimiento del sistema, siguiendo la metodología RUP, la cual se detalla en el presente documento.

- ✓ Usar como referencia el manual de procedimientos del sistema para que le den buen uso a la misma.

FUENTES DE INFORMACIÓN

FUENTES DE INFORMACIÓN

- [McConnell, Steve 1996] Desarrollo y Gestión de Proyectos Informáticos

- O Brien Games 2001 sistemas de información gerencial – Manejo de tecnologías de información

- AMO, Fernando Alonso; LOÏC MARTINEZ, Normand; SEGOVIA PÉREZ, Francisco Javier 2005. Introducción a la ingeniería del software.

- BARRANCO DE AREBA, 2001 Jesús. Metodología de análisis estructurado de sistemas.

- CAMPDERRICH FALGUERAS, 2003 Benet. Ingeniería del software.

- SÁNCHEZ GARRETA, 2003 José Salvador. Ingeniería de proyectos informáticos: actividades y procedimientos. Castellón.

LINKOGRAFIA

- Senn, 1992 James...Analista de sistemas y el paradigma

  http://www.monografias.com/trabajos15/analista-sistem/analista-sistem.shtml

- Estudio de ingeniería de métodos, Librería y Papelería LILÍ C.A

  http://www.programa-de-librerias.com/software-para-gestionar-librerias-y-papelerias-verial.html

- Geslib. El software de gestión para su librería

  http://editorial.trevenque.es/soluciones/software-librerias-geslib/

- Estudio de ingeniería de métodos, Librería y papelería latina

  http://www.monografias.com/trabajos91/estudio-ingenieria-metodos-libreria-y-papeleria-latina/estudio-ingenieria-metodos-libreria-y-papeleria-latina.shtml

- Desarrollo de plan comercial de la librería dimeiggs s.a. y estudio de posible plan de expansión.

  http://tesis.uchile.cl/tesis/uchile/2007/dayoub_g/sources/dayoub_g.pdf

ANEXOS

❖ ENTREVISTA

1. ¿Cuánto tiempo de antigüedad tienes en el cargo?
2. ¿Qué procesos administrativos se controlan en esta área?
3. ¿Cuáles de procesos considera usted los más importantes?
4. ¿Cuál de los procesos que menciono en la pregunta anterior cree que puede ser mejorado mediante la implementación de un sistema?
5. Describa detalladamente (paso a paso) la secuencia de los procesos elegidos
6. En la actualidad que complicaciones y errores se generan en el proceso
7. ¿Cuál cree usted que son las causas que generan dichas fallas?
8. ¿Cómo sugiere que se ejecuten las áreas fallidas?
9. ¿Qué tareas del proceso no deben ser modificados?