BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

"Sistema Administrativo de una Bisutería"

Tesis que Presenta:
Erick Cruz Campos

Para Obtener el Título de:
Licenciado en Ciencias de la Computación

Director de la Tesis:

M.C. Meliza Contreras González

Puebla, Pue. Noviembre, 2008

INDICE

INTRODUCCION

Los negocios de bisutería en los últimos años han cobrado vital importancia, esto es debido a que resulta más económico elaborar una pieza que adquirirla a un precio demasiado costoso.

Sin embargo estos negocios aún manejan sus procesos administrativos de forma manual, el departamento de compras realiza los pedidos a los principales proveedores del material de bisutería, éste se incorpora al inventario considerando los precios de proveedor y los precios al público. Una vez realizado esto en el departamento de manufactura se elaboran los precios sobre los modelos de bisutería con el nuevo material adquirido.

Así los clientes adquieren la materia prima o solicitan la elaboración de productos, por lo que se emite una cotización y si al cliente le satisface el precio, se genera el pedido. La forma de pago la realiza por depósitos o cheques o por abonos sobre el saldo que lleva en la cartera.

Ante esta necesidad, surge este proyecto con el objetivo de automatizar el flujo de información de la empresa, como primera etapa del proyecto se realizará el análisis de los requerimientos de la empresa, para posteriormente diseñar el sistema y la base de datos que sustente el flujo de información, como tercera etapa se realizará la implementación del sistema con una interfaz adecuada y la seguridad que implica para finalmente describir las pruebas pertinentes del funcionamiento y robustez del sistema.

La empresa actualmente invierte demasiado tiempo en los procesos de realización de inventario, solicitud de pedidos y compras, la elaboración de notas y facturas, la administración de abonos se realiza de forma manual y todos los documentos ocupan un espacio necesario para el inventario, por esta razón el sistema propuesto administrará y automatizará todos los procesos involucrados

OBJETIVO GENERAL:

Desarrollar un sistema de control local o red de procesos de un negocio de bisutería

OBJETIVOS ESPECÍFICOS:

- Administrar la información de los clientes, los proveedores, las solicitudes de compras, pedidos, reposición de stock de material y productos elaborados.
- Visualizar de forma amigable el contenido del inventario de productos y materiales y los precios y cotizaciones.
- Personalizar las vistas de facturas, remisiones y cotizaciones adecuadas a las necesidades del cliente
- Generar el historial de los pedidos, deudas a saldar, abonos.
- Impresión de facturas, notas, cotizaciones y reposición de stock

CAPITULO 1. MARCO TEORICO.

En este capitulo se darán definiciones de conceptos esenciales, metodologías y descripción de las herramientas utilizadas para la elaboración de la documentación así como para programación del sistema.
Para UML

1.1 UML

Lenguaje Unificado de Modelado (**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 de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, 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 aplica en una gran variedad de formas para dar soporte a una metodología de desarrollo de software (Proceso Unificado Racional o RUP), pero no especifica en sí mismo qué metodología o proceso usar.

UML no se compara con la programación estructurada, pues UML significa (Lengua de Modelación Unificada), 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 orientación a objetos es un complemento perfecto de UML.

UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas [stephen2000].

1.1.1 Vistas de UML.

En la construcción de software usando UML, existen cinco vistas para visualizar, especificar, construir y documentar la arquitectura del software. UML permite representar cada vista mediante un conjunto de diagramas, las vistas son las siguientes:

- Vista de casos de uso: Muestra la funcionalidad del sistema desde el punto de vista de un actor externo que interactúa con él. Esta vista es útil a clientes, diseñadores y desarrolladores.

- Vista de diseño: Muestra la funcionalidad del diseño dentro del sistema en términos de la estructura estática y comportamiento dinámico del sistema. Esta vista es útil a diseñadores y desarrolladores. Se definen propiedades tales como: persistencia, concurrencia, interfaces y estructuras internas a las clases.

- Vista de procesos: Muestra la concurrencia del sistema, comunicación y sincronización. Útil a desarrolladores e integradores.

- Vista de implementación: Muestra la organización de los componentes de código. Útil a desarrolladores.

- Vista de implantación (también conocida como vista de despliegue): Muestra la implantación del sistema en la arquitectura física. Útil a desarrolladores, integradores y verificadores [stephen2000].

1.1.2 Definición de modelo.

Un sistema (tanto en el mundo real como en el mundo del software) suele ser extremadamente intrincado, por ello es necesario dividir el sistema en partes o fragmentos si queremos entender y administrar su complejidad. Estas partes se representan como modelos que describen y abstraen sus aspectos esenciales.

Un modelo captura una vista de un sistema del mundo real. Es una abstracción de dicho sistema considerando un cierto propósito. Así, el modelo describe completamente aquellos aspectos del sistema que son relevantes al propósito del modelo y a un apropiado nivel de detalle.

Los modelos se componen de otros modelos, de diagramas y documentos que describen detalles del sistema. UML especifica varios diagramas. Si queremos caracterizar los modelos, se especifica la información estática o dinámica del sistema. Un modelo estático describe las propiedades estructurales del sistema; en cambio, un modelo dinámico describe las propiedades de comportamiento de un sistema [stephen200].

Es importante mencionar que UML es un lenguaje para construir modelos; no guía al desarrollador en la forma de realizar el análisis y diseño orientado a objetos ni indica cuál proceso de desarrollo adoptar.

Para modelar un sistema es suficiente utilizar una parte de UML, "el 80 por ciento de la mayoría de los problemas pueden modelarse usando alrededor del 20 por ciento de UML".

1.1.3 Diagramas UML.

UML es un lenguaje notacional. Parte importante de esta notación son los diagramas que nos permiten modelar un sistema. Un diagrama es una representación gráfica de una colección de elementos de modelado, la mayoría de las veces mostrados como grafo conexo de vértices (cosas) y arcos (relaciones). Los buenos diagramas hacen el sistema que se está desarrollando más comprensible y cercano a los objetivos. En UML se definen nueve diagramas, los cuales se mezclan si es necesario en cada vista [stephen2000]:

Los ***Diagramas Estructurales*** enfatizan en los elementos que existirán en el sistema modelado:

- Diagrama de clases
- Diagrama de componentes
- Diagrama de objetos
- Diagrama de estructura compuesta (UML 2.0)
- Diagrama de despliegue
- Diagrama de paquetes

Los ***Diagramas de Comportamiento*** enfatizan en lo que sucederá en el sistema modelado:

- Diagrama de actividades
- Diagrama de casos de uso
- Diagrama de estados

Los ***Diagramas de Interacción*** son un subtipo de diagramas de comportamiento, que enfatiza sobre el flujo de control y de datos entre los elementos del sistema modelado:

- Diagrama de secuencia
- Diagrama de colaboración
- Diagrama de tiempos (UML 2.0)
- Diagrama de vista de interacción (UML 2.0)

1.1.3.1 Diagramas estructurales.

Los cuatro diagramas estructurales de UML existen para visualizar, especificar, construir y documentar los aspectos estáticos del sistema. Están organizados sobre grupos de objetos que se encontrarán cuando se esté modelando un sistema [stephen2000].

1. Diagrama de clases: Un diagrama de este tipo muestra un conjunto de clases, interfaces, y sus relaciones.

2. Diagrama de objetos: Muestra un conjunto de objetos y sus relaciones. A diferencia de los diagramas anteriores, estos diagramas se enfocan en la perspectiva de casos de uso, y prototipos.

3. Diagrama de componentes: Muestra el conjunto de componentes y sus relaciones y se utilizan para ilustrar la vista de la implementación estática de un sistema.

4. Diagrama de implantación: Muestra un conjunto de nodos y sus relaciones, se usan para ilustrar la vista de implantación estática de un sistema.

1.1.3.2 Diagramas de comportamiento.

Los cinco diagramas de comportamiento de UML son usados para visualizar, especificar, construir y documentar los aspectos dinámicos de un sistema. Se consideran los aspectos dinámicos como las representaciones de las partes cambiantes del sistema [stephen2000].

1. Diagrama de casos de uso: Muestra el conjunto de casos de uso y actores (incluyendo sus relaciones). Estos diagramas se utilizan para ilustrar la vista del caso de uso del sistema.

2. Diagrama de secuencia: Es un diagrama de interacción que enfatiza el orden en tiempo de los mensajes.

3. Diagrama de colaboración: Es un diagrama de interacción que enfatiza la organización estructural de los objetos que envían y reciben mensajes. El diagrama de colaboración muestra un conjunto de objetos, las ligas entre ellos y los mensajes enviados y recibidos por dichos objetos.

4. Diagrama de estado: Muestra una máquina de estado, consistente en estados, transiciones, eventos y actividades. Estos diagramas enfatizan el comportamiento ordenado por eventos de un objeto.

5. Diagrama de actividad: Muestra el flujo de una actividad a otra dentro del sistema. Ha sido diseñado para mostrar una visión simplificada de lo que ocurre durante una operación o suceso.

1.1.4 UML y su Relación con los Procesos de Desarrollo de Software.

Un proceso de desarrollo de software es un método de organizar las actividades relacionadas con la creación, presentación y mantenimiento de los sistemas de software [pressman2000].

El lenguaje UML no define un proceso oficial de desarrollo, en realidad UML se combina con un proceso de desarrollo para obtener un producto final. Craig Larman da dos razones importantes que explican esto.

1. Aumentar la posibilidad de una aceptación generalizada de la notación estándar del modelado, sin la obligación de adoptar un proceso oficial.

2. La esencia de un proceso apropiado admite mucha variación y depende de las habilidades del personal, de la razón investigación-desarrollo, de la naturaleza del problema y de las herramientas.

1.1.4.1 Proceso Unificado de Desarrollo de Software.

Uno de los procesos más ocupados y recomendados para trabajar con UML es el proceso unificado de desarrollo de software (The Unified Software Development Process). Este proceso fue elaborado por los creadores del UML (Jacobson, Booch y Rumbaugh). Sus características principales son [pressman2000]:

- Es dirigido por los casos de uso; acciones realizadas (interacción) entre los usuarios y el sistema.

- Se centra en el diseño de una arquitectura central, la cual guía el proceso de construcción de software.
- Es un proceso que utiliza un desarrollo iterativo e incremental:
    - Las iteraciones son controladas sobre los diferentes pasos del proceso.
    - Es incremental porque en cada iteración el software se va ampliando y mejorando.

1.1.4.2 Desarrollo Iterativo e Incremental.

Un ciclo de vida iterativo se basa en el agrandamiento y perfeccionamiento secuencial de un sistema a través de múltiples ciclos de desarrollo, análisis, diseño, implementación y pruebas. El sistema crece al incorporar nuevas funciones en cada ciclo de desarrollo. "En cada ciclo se aborda un conjunto relativamente pequeño de requerimientos, pasando por el análisis, el diseño, la construcción y las pruebas. El sistema va creciendo con cada ciclo que concluye".

1.2 Ingeniería de Software

La Ingeniería del Software es una disciplina o área de la informática o ciencias de la computación, que ofrece métodos y técnicas para desarrollar y mantener software de calidad que resuelven problemas de todo tipo. Hoy día es cada vez más frecuente la consideración de la Ingeniería del Software como una nueva área de la ingeniería, y el Ingeniero del Software comienza a ser una profesión implantada en el mundo laboral internacional, con derechos, deberes y responsabilidades que cumplir, junto a una, y reconocida consideración social en el mundo empresarial y, por suerte, para esas personas con brillante futuro [pressman2000].

La Ingeniería del Software trata áreas muy diversas de la informática y de las ciencias de la computación, tales como construcción de compiladores, sistemas operativos o desarrollos en Intranet/Internet, abordando las fases del ciclo de vida del desarrollo de cualquier tipo de sistemas de Información y aplicables a una infinidad de áreas tales como: negocios, investigación científica, medicina, producción, logística, banca, control de tráfico, meteorología, la red de redes Internet, redes Intranet y Extranet, etc.

1.2.1 Definición del Termino "Ingeniería del Software".

El término Ingeniería se define en el DRAE (Diccionario de la Real Académica Española) como: Conjunto de conocimientos y técnicas que permiten aplicar el saber científico a la utilización de la materia y de las fuentes de energía [stephen2000].

Dentro de las definiciones más importantes se consideran las siguientes:

- Ingeniería del Software es el estudio de los principios y metodologías para desarrollo y mantenimiento de sistemas de software.
- Ingeniería del Software es la aplicación práctica del conocimiento científico en el diseño y construcción de programas de computadora y la documentación asociada requerida para desarrollar, operar (funcionar) y mantenerlos.
- Ingeniería del software trata del establecimiento de los principios y métodos de la ingeniería a fin de obtener software de modo rentable que sea fiable y trabaje en máquinas reales.
- La aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo operación (funcionamiento) y mantenimiento del software: es decir, el estudio de enfoques con la aplicación de ingeniería al software. [stephen2000]

1.2.2 Modelos de Ingeniería de Software.

La ingeniería de software proporciona varios modelos o paradigmas de desarrollo en los cuales se apoya para la realización de software, de los cuales se destacan a éstos por ser los más utilizados y los más completos: [benet2000]

a) Modelo en cascada o Clásico (modelo tradicional).
b) Modelo en espiral (modelo evolutivo).
c) Modelo de prototipos.
d) Desarrollo por etapas.
e) Desarrollo iterativo y creciente o Iterativo Incremental.
f) RAD (Rapid Application Development).

1.2.2.1 Modelo en cascada o Clásico.

Es el enfoque metodológico que ordena rigurosamente las etapas del **ciclo de vida del software**, de forma tal que el inicio de cada etapa espera a la finalización de la inmediatamente anterior [benet2000].

Un ejemplo de una metodología de desarrollo en cascada es:

1. Análisis de requisitos.
2. Diseño del Sistema.
3. Diseño del Programa.
4. Codificación.
5. Pruebas.
6. Implantación.
7. Mantenimiento.

De esta forma, cualquier error de diseño detectado en la etapa de prueba conduce necesariamente al rediseño y nueva programación del código afectado, aumentando los costes del desarrollo. La palabra *cascada* sugiere, mediante la metáfora de la fuerza de la gravedad, el esfuerzo necesario para introducir un cambio en las fases más avanzadas de un proyecto.
Si bien ha sido ampliamente criticado desde el ámbito académico y la industria, sigue siendo el paradigma más seguido al día de hoy.

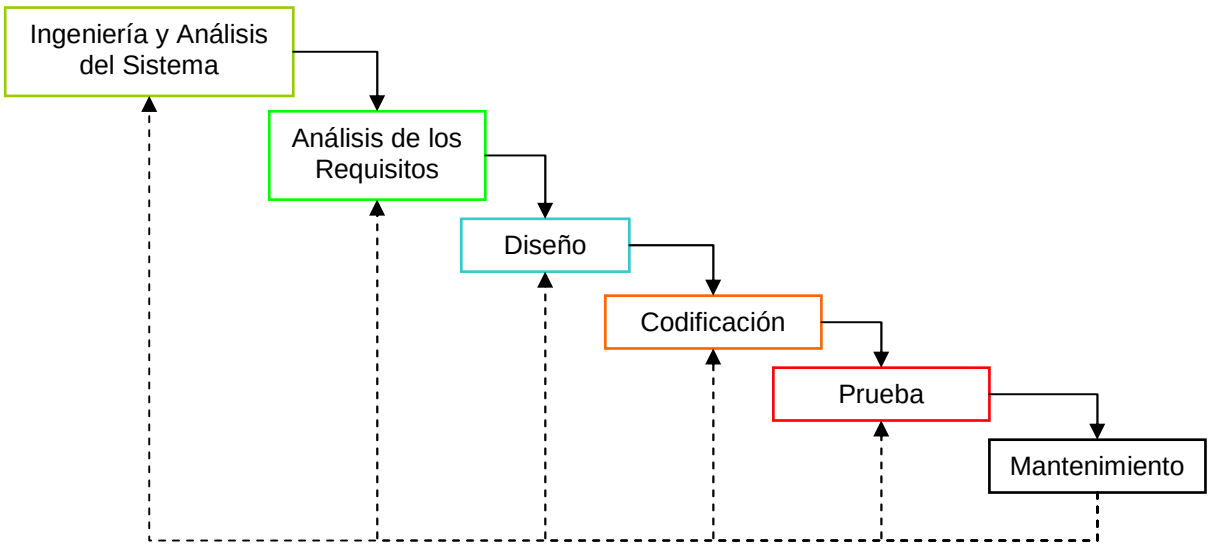

Figura 1.1 Modelo de Cascada

Análisis de requisitos
Se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. De esta fase surge una memoria llamada SRD (Documento de Especificación de Requisitos), que contiene la especificación completa de lo que realizará el sistema sin entrar en detalles internos.

Es importante señalar que en esta etapa se consensua todo lo que se requiere del sistema y será aquello lo que seguirá en las siguientes etapas, evitando requerir nuevos resultados a mitad del proceso de elaboración del software.

Diseño del Sistema
Se descompone y organiza el sistema en elementos que se elaboran por separado, aprovechando las ventajas del desarrollo en equipo. Como resultado surge el SDD (Documento de Diseño del Software), que contiene la descripción de la estructura relacional global del sistema y la especificación de lo que realizará cada una de sus partes, así como la manera en que se combinan unas con otras.

Diseño del Programa
Es la fase en donde se realizan los algoritmos necesarios para el cumplimiento de los requerimientos del usuario así como también los análisis necesarios para saber que herramientas utilizar en la etapa de Codificación.

Codificación
Es la fase de programación o implementación propiamente dicha. Aquí se implementa el código fuente, haciendo uso de prototipos así como pruebas y ensayos para corregir errores.

Dependiendo del lenguaje de programación y su versión se crean las librerías y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido [benet2000].

Pruebas
Los elementos, ya programados, se ensamblan para componer el sistema y se comprueba que funciona correctamente antes de ser puesto en explotación.

Implantación
El software obtenido se pone en producción. Se implantan los niveles software y hardware que componen el proyecto. La implantación es la fase con más duración y con más cambios en el ciclo de elaboración de un proyecto. Es una de las fases finales del proyecto

Durante la explotación del sistema, en el software tal vez sucedan cambios, bien para corregir errores o bien para introducir mejoras. Todo ello se recoge en los Documentos de Cambios.

1.2.2.2 Modelo en Espiral.

Es un modelo de **ciclo de vida** desarrollado por Barry Boehm en 1985, utilizado generalmente en la Ingeniería de software. Las actividades de este modelo son una espiral, cada bucle es una actividad. Las actividades no están fijadas a prioridad, sino que las siguientes se eligen en función del análisis de riesgo, comenzando por el bucle interior.
Para cada actividad habrá cuatro tareas [benet2000]:

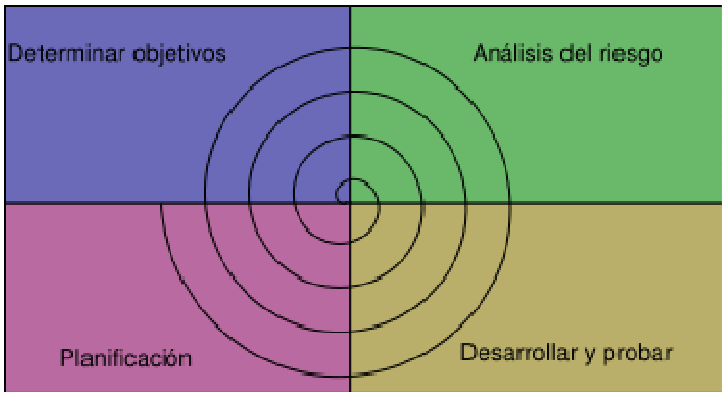

Figura1.2 Modelo en Espiral

Determinar o fijar objetivos

- Fijar también los productos definidos a obtener: requerimientos, especificación, manual de usuario.
- Fijar las restricciones. Identificación de riesgos del proyecto y estrategias alternativas para evitarlos.
- Hay una cosa que solo se hace una vez: planificación inicial o previa.

Análisis del riesgo

Se estudian todos los riesgos potenciales y se seleccionan una o varias alternativas propuestas para reducir o eliminar los riesgos.

Desarrollar, verificar y validar (probar)

- Tareas de la actividad propia y de prueba.
- Análisis de alternativas e identificación resolución de riesgos.
- Dependiendo del resultado de la evaluación de los riesgos, se elige un modelo para el desarrollo, considerando cualquiera de los otros existentes, como formal, evolutivo, cascada, etc. Así si, por ejemplo si los riesgos en la interfaz de usuario son dominantes, un modelo de desarrollo apropiado seria la construcción de prototipos evolutivos. Si lo riesgos de protección son la principal consideración, un desarrollo basado en transformaciones formales seria el más apropiado.

Planificar

Revisar todo lo hecho, evaluándolo, y con ello decidimos si se continua con las fases siguientes y planificar la próxima actividad.

1.2.2.3 Modelo de Prototipos.

En Ingeniería de software el desarrollo con prototipación, también llamado modelo de prototipos o modelo de desarrollo evolutivo, se inicia con la definición de los objetivos globales para el software, luego se identifican los requisitos conocidos y las áreas del esquema en donde es necesaria más definición. Entonces se plantea con rapidez una iteración de construcción de prototipos y se presenta el modelado (en forma de un diseño rápido) [pressman2000].

El diseño rápido se centra en una representación de aquellos aspectos del software que serán visibles para el cliente o el usuario final (por ejemplo, la configuración de la interfaz con el usuario y el formato de los despliegues de salida). El diseño rápido conduce a la construcción de un prototipo, el cual es evaluado por el cliente o el usuario para una retroalimentación; gracias a ésta se refinan los requisitos del software que se desarrollará.

La iteración ocurre cuando el prototipo se ajusta para satisfacer las necesidades del cliente. Esto permite que al mismo tiempo el desarrollador entienda mejor lo que se realizará y el cliente vea resultados a corto plazo.

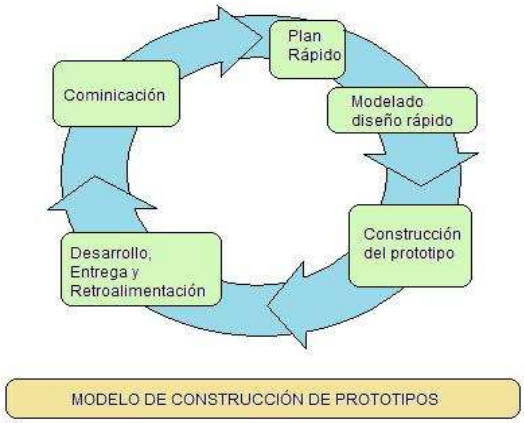

Figura 1.3 Modelo de Prototipos

1.2.2.4 Desarrollo por Etapas.

El modelo de desarrollo de software por etapas es similar al Modelo de prototipos ya que se muestra al cliente el software en diferentes estados sucesivos de desarrollo, se diferencia en que las especificaciones no son conocidas en detalle al inicio del proyecto y por tanto se van desarrollando simultáneamente con las diferentes versiones del código [stephen2002].

Se distinguen las siguientes fases:

- Especificación conceptual
- Análisis de requerimientos
- Diseño inicial
- Diseño detallado, codificación, depuración y liberación

Estas diferentes fases se van repitiendo en cada etapa del diseño.

1.2.2.5 Desarrollo Iterativo y Creciente.

Es un proceso de desarrollo de software, creado en respuesta a las debilidades del modelo tradicional de cascada. Para apoyar el desarrollo de proyectos por medio de este modelo se han desarrollado entornos de trabajo, de los cuales los dos más famosos son el Rational Unified Process y el Dynamic Systems Development Method. El desarrollo incremental e iterativo es también una parte esencial de un tipo de programación conocido como programación extrema y los demás entornos de trabajo de desarrollo rápido de software.

Ciclo de vida

La idea principal detrás del mejoramiento iterativo es desarrollar un sistema de programas de manera incremental, permitiéndole al desarrollador sacar ventaja de lo que se ha aprendido a lo largo del desarrollo anterior, incrementando, versiones entregables del sistema. El aprendizaje viene de dos vertientes: el desarrollo del sistema y su uso (mientras sea posible).

Los pasos claves en el proceso son comenzar con una implementación simple de los requerimientos del sistema e iterativamente mejorar la secuencia evolutiva de versiones hasta que el sistema completo esté implementado. En cada iteración, se realizan cambios en el diseño y se agregan nuevas funcionalidades y capacidades al sistema [stephen2002].

El proceso en sí mismo consiste de:

- Etapa de inicialización
- Etapa de iteración
- Lista de control de proyecto

Etapa de inicialización

Se crea una versión del sistema. La meta de esta etapa es crear un producto con el que el usuario interactúe, y por ende retroalimentar el proceso. Debe ofrecer una muestra de los aspectos claves del problema y proveer una solución lo suficientemente simple para ser comprendida e implementada fácilmente. Para guiar el proceso de iteración, una lista de control de proyecto se crea, y esta lista contiene un historial de todas las tareas que necesitan ser realizadas. Incluye cosas como nueva funcionalidades para ser implementadas, y áreas de rediseño de la solución ya existente. Esta lista de control se revisa periódica y constantemente como resultado de la fase de análisis.

Etapa de iteración

Esta etapa involucra el rediseño e implementación de una tarea de la lista de control de proyecto, y el análisis de la versión más reciente del sistema. La meta del diseño es ser simple, directa y modular, para soportar el rediseño de la etapa o como una tarea añadida a la lista de control de proyecto [stephen2002].

El código representará la mayor fuente de documentación del sistema. El análisis de una iteración se basa en la retroalimentación del usuario y en el análisis de las funcionalidades disponibles del programa. Involucra el análisis de la estructura, modularidad, usabilidad, confiabilidad, eficiencia y eficacia (alcanzar las metas). La lista de control del proyecto se modifica bajo la luz de los resultados del análisis [stephen2002].

Las guías primarias que conducen a la implementación y el análisis incluyen:

- Cualquier dificultad en el diseño, codificación y prueba de una modificación apuntará a la necesidad de rediseñar o recodificar.
- Las modificaciones se ajustarán fácilmente a los módulos fáciles de encontrar y a los aislados. Si no es así, entonces se requiere algún grado de rediseño.
- Las modificaciones a las tablas serán especialmente fáciles de realizar. Si dicha modificación no ocurre rápidamente, se aplicará algo de rediseño.
- Las modificaciones serán más fáciles de hacer conforme avanzan las iteraciones. Si no es así, hay un problema primordial usualmente encontrado en un diseño débil o en la proliferación excesiva de parches al sistema.
- Los parches normalmente permanecerán solo por una o dos iteraciones. Se hacen necesarios para evitar el rediseño durante una fase de implementación.

- La implementación existente será analizada frecuentemente para determinar que tan bien se ajusta a las metas del proyecto.
- Las facilidades para analizar el programa serán utilizadas cada vez para ayudar en el análisis de implementaciones parciales.
- La opinión del usuario será solicitada y analizada para indicar deficiencias en la implementación referida por él.

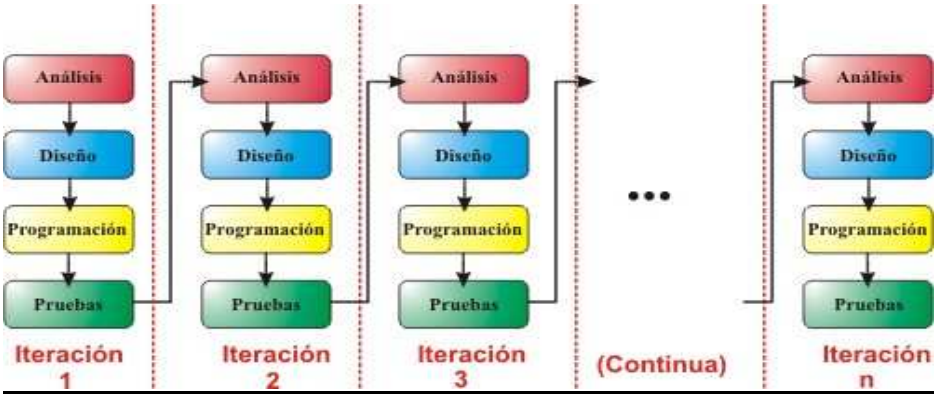

Figura 1.4 Modelo Iterativo y Creciente

1.2.2.6 Desarrollo Rápido de Aplicaciones (RAD).

Es un proceso de desarrollo de software, en inglés: Software Development Process, desarrollado inicialmente por James Martin en 1980. El método comprende el desarrollo interactivo, la construcción de prototipos y el uso de utilidades CASE (Computer Aided Software Engineering). Tradicionalmente, el desarrollo rápido de aplicaciones tiende a englobar también la usabilidad, utilidad y la rapidez de ejecución.

Hoy en día se suele utilizar para referirnos al desarrollo rápido de GUI's tal como Glade, o IDEs de desarrollo completas como Delphi, Foxpro o Anjuta. Uno de los programas más usados para generar aplicaciones rápidamente es Visual Basic [benet2000].

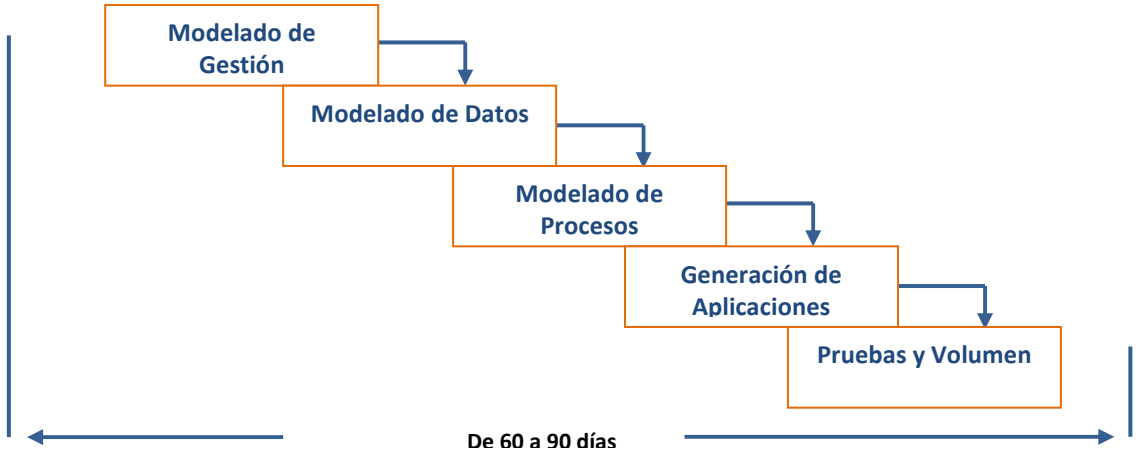

Figura1.5 Modelo RAD

1.3 Bases de Datos

Una base de datos es un “almacén” que permite guardar grandes cantidades de información de forma organizada para que luego se encuentre y utilice fácilmente.

El término de bases de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California, USA. Una base de datos se define como un conjunto de información relacionada que se encuentra agrupada ó estructurada.

Desde el punto de vista informático, la base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulen ese conjunto de datos.

Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más **columnas** y **filas**. Las columnas guardan una sección de la información sobre cada elemento que se requieran guardar en la tabla, cada fila de la tabla conforma un registro [korth2002].

1.3.1 Definición de base de datos.

Se define una base de datos como una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular.

1.3.2 Características.

Entre las principales características de los sistemas de base de datos se mencionan:

- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Consultas complejas optimizadas.
- Seguridad de acceso y auditoria.
- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

1.3.3 Sistema de Gestión de Base de Datos (SGBD).

Los Sistemas de Gestión de Base de Datos (DataBase Management System) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta [korth2002].

1.3.4 Ventajas de las bases de datos.

Control sobre la redundancia de datos:
Los sistemas de archivos almacenan varias copias de los mismos datos en archivos distintos. Esto hace que se desperdicie espacio de almacenamiento, además de provocar la falta de consistencia de datos [elmasri2000].

En los sistemas de bases de datos todos estos archivos están integrados, por lo que no se almacenan varias copias de los mismos datos. Sin embargo, en una base de datos no se elimina la redundancia completamente, ya que en ocasiones es necesaria para modelar las relaciones entre los datos.

Consistencia de datos:
Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias. Si un dato está almacenado una sola vez, cualquier actualización se realizará sólo una vez, y está disponible para todos los usuarios inmediatamente. Si un dato está duplicado y el sistema conoce esta redundancia, el propio sistema se encarga de garantizar que todas las copias se mantengan consistentes.

Compartición de datos:
En los sistemas de archivos, los archivos pertenecen a las personas o a los departamentos que los utilizan. Pero en los sistemas de bases de datos, la base de datos pertenece a la empresa y será compartida por todos los usuarios que estén autorizados.

Mantenimiento de estándares:
Gracias a la integración es más fácil respetar los estándares necesarios, tanto los establecidos a nivel de la empresa como los nacionales e internacionales. Estos estándares se establecen sobre el formato de los datos para facilitar su intercambio, incluyendo estándares de documentación, procedimientos de actualización y también reglas de acceso [elmasri2002].

Mejora en la integridad de datos:
La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se violaran. Estas restricciones se aplicarán tanto a los datos, como a sus relaciones, y es el SGBD quien se encarga de mantenerlas.

Mejora en la seguridad:
La seguridad de la base de datos es la protección de la base de datos frente a usuarios no autorizados. Sin medidas de seguridad, la integración de datos en los sistemas de bases de datos hace que éstos sean más vulnerables que en los sistemas de archivos.

Mejora en la accesibilidad a los datos:
La mayoría de los SGBD proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos, sin que sea necesario que un programador escriba una aplicación que realice tal tarea [elmasri2002].

Mejora en la productividad:
El SGBD proporciona varias de las funciones estándar que el programador escribe en un sistema de archivos. A nivel básico, el SGBD proporciona todas las rutinas de manejo de archivos típicas de los programas de aplicación.

El hecho de disponer de estas funciones permite al programador centrarse mejor en la función específica requerida por los usuarios, sin preocuparse de los detalles de implementación de bajo nivel.

Mejora en el mantenimiento:
En los sistemas de archivos, las descripciones de los datos se encuentran inmersas en los programas de aplicación que los manejan. Esto hace que los programas sean dependientes de los datos, de modo que un cambio en su estructura, o un cambio en el modo en que se almacena en disco, requiere cambios importantes en los programas cuyos datos se ven afectados [elmasri2002].

Sin embargo, los SGBD separan las descripciones de los datos de las aplicaciones. Esto es lo que se conoce como independencia de datos, gracias a la cual se simplifica el mantenimiento de las aplicaciones que acceden a la base de datos.

Aumento de la concurrencia:
En algunos sistemas de archivos, si hay varios usuarios que pueden acceder simultáneamente a un mismo archivo, es posible que el acceso interfiera entre ellos de modo que se pierda información o se pierda la integridad. La mayoría de los SGBD gestionan el acceso concurrente a la base de datos y garantizan que no ocurran problemas de este tipo.

Mejora en los servicios de copias de seguridad:
Muchos sistemas de archivos dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema o en las aplicaciones. Los usuarios realizan copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos [elmasri2002].

En este caso, todo el trabajo realizado sobre los datos desde que se hizo la última copia de seguridad se pierde y se volverá a realizar. Sin embargo, los SGBD actuales funcionan de modo que se minimiza la cantidad de trabajo perdido cuando se produce un fallo.

1.3.5 Desventajas de las Bases de Datos.

Complejidad:
Los SGBD son conjuntos de programas que son complejos con una gran funcionalidad. Es preciso comprender muy bien esta funcionalidad para realizar un buen uso de ellos [pons2002].

Costo del equipamiento adicional:
Tanto el SGBD, como la propia base de datos, provocan que sea necesario adquirir más espacio de almacenamiento. Además, para alcanzar las prestaciones deseadas, es posible que sea necesario adquirir una máquina más grande o una máquina que se dedique solamente al SGBD. Todo esto hará que la implantación de un sistema de bases de datos sea más cara.

Vulnerable a los fallos:
El hecho de que todo esté centralizado en el SGBD hace que el sistema sea más vulnerable ante los fallos que se produzcan. Es por ello que se almacenan copias de seguridad (Backup).

Tipos de Campos:
Cada Sistema de Base de Datos posee tipos de campos que son similares o diferentes [pons2002].

Entre los más comunes se encuentran:

- **Numérico:** entre los diferentes tipos de campos numéricos están los enteros "sin decimales" y reales "decimales".
- **Booleanos:** poseen dos estados: Verdadero "Si" y Falso "No".
- **Memos:** son campos alfanuméricos de longitud ilimitada. Presentan el inconveniente de no ser indexados.
- **Fechas:** almacenan fechas facilitando posteriormente su explotación. De esta forma posibilita ordenar los registros por fechas o calcular los días entre una fecha y otra.
- **Alfanuméricos:** contienen cifras y letras. Presentan una longitud limitada (255 caracteres).
- **Autoincrementables:** son campos numéricos enteros que incrementan en una unidad su valor para cada registro incorporado. Su utilidad es servir de identificador ya que resultan exclusivos de un registro.

1.3.6 Tipos de manejadores de bases de datos.

Entre los diferentes tipos de base de datos, se consideran los siguientes:

- **MySql:** Es un manejador de base de datos con licencia GPL basada en un servidor. Se caracteriza por su rapidez. No es recomendable usar para grandes volúmenes de datos.
- **PostgreSql y Oracle:** Son sistemas de base de datos poderosos. Administra muy bien grandes cantidades de datos, y son utilizadas en intranets y sistemas de gran calibre.

- **Access:** Es un manejador de base de datos desarrollada por Microsoft. Esta base de datos, debe es creada bajo el programa access, el cual crea un archivo .mdb.
- **Microsoft SQL Server:** Es un manejador de base de datos más potente que Access desarrollada por Microsoft. Se utiliza para manejar grandes volúmenes de información [korth2002].

1.4 Modelo Entidad – Relación

Este modelo se obtiene en tiempo de diseño de la base de datos. Fue propuesto por Peter Chen en 1976 y desde entonces se viene utilizando de una forma muy global. Se caracteriza por utilizar una serie de símbolos y reglas para representar los datos y sus relaciones.

Con este modelo conseguimos representar de manera gráfica la estructura lógica de una base de datos. Los principales elementos del modelo entidad-relación son las entidades con sus atributos y las relaciones entre entidades [stephen2002].

1.4.1 Elementos del Modelo Entidad-Relación.

Entidad
Se trata de un objeto del que se obtiene información de interés. Gráficamente se representan mediante un rectángulo. Un ejemplo sería la entidad banco, donde se obtendrían los datos relativos a ese banco, como el nombre, el número de sucursal, la dirección, etc.

Las entidades son fuertes o débiles. Las fuertes son las que no dependen de otras entidades para existir, mientras que las entidades débiles siempre dependen de otra entidad de lo contrario no tienen sentido por ellas mismas.

Relación
Se define una relación como una asociación de dos o más entidades. A cada relación se le asigna un nombre para distinguirla de las demás y saber su función dentro del modelo entidad-relación. Otra característica es el grado de relación, siendo las de grado 1 relaciones que solo vinculan una entidad consigo misma. Las de grado 2 son relaciones que asocian dos entidades distintas, y las de grado n que se tratan de relaciones que unen más de dos entidades [stephen2002].

Las relaciones se representan gráficamente con rombos, dentro de ellas se coloca el nombre de la relación.

Otra característica es el tipo de correspondencia entre dos relaciones;

- 1:1 Uno a uno, a cada ocurrencia de una entidad le corresponde como máximo una ocurrencia de la otra entidad relacionada.
- 1:N Uno a Muchos, a cada ocurrencia de la entidad A le corresponde varias de la entidad B.
- N:M Muchos a muchos, cada ocurrencia de una entidad contiene varias de la otra entidad relacionada y viceversa.

La cardinalidad define el número máximo y mínimo de ocurrencias de cada tipo de entidad. Se representa con los valores (máximo, mínimo).

Atributo
Se define como cada una de las propiedades de una entidad o relación. Cada atributo posee un nombre y todos los posibles valores que alcanzara. Dentro de una entidad existe un atributo principal que identifica a la entidad y su valor es único. Un ejemplo de atributo principal seria el curp dentro de la entidad persona.

Observe el siguiente esquema del modelo entidad-relación [stephen2002].

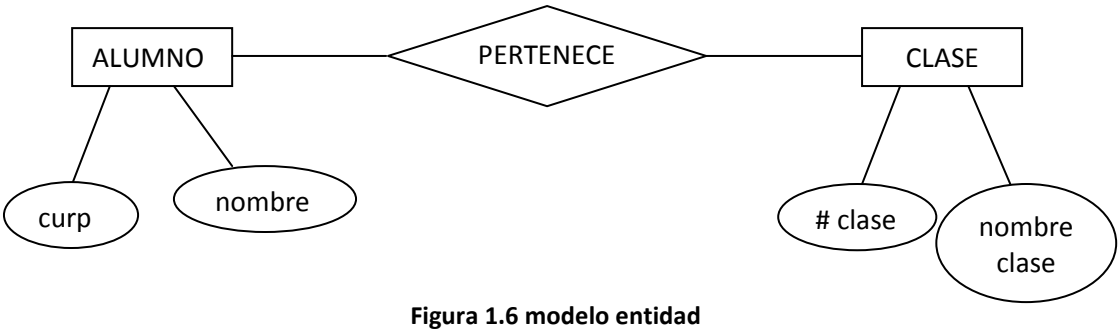

Figura 1.6 modelo entidad

1.5 Modelo Relacional

Las bases de datos relacionales son el modelo de bases de datos actualmente más difundido. Los motivos de este éxito son fundamentalmente dos [pons2000].

1. Ofrecen sistemas simples y eficaces para representar y manipular los datos
2. Se basan en un modelo, el relacional, con sólidas bases teóricas

El modelo relacional fue propuesto originariamente por E.F. Codd en un ya famoso artículo de 1970. Gracias a su coherencia y facilidad de uso, el modelo se ha convertido en los años 80 en el más usado para la producción de DBMS (Sistema Gestor de Bases de Datos).

La estructura fundamental del modelo relacional es precisamente esa, "relación", es decir una tabla bidimensional constituida por líneas (tuplas) y columnas (atributos). Las relaciones representan las entidades que se consideran interesantes en la base de datos. Cada instancia de la entidad encontrará sitio en una tupla de la relación, mientras que los atributos de la relación representarán las propiedades de la entidad. Por ejemplo, si en la base de datos se representan personas, se definirá una relación llamada "Personas", cuyos atributos describen las características de las personas (tabla siguiente). Cada tupla de la relación "Personas" representará una persona concreta [pons2000].

Persona				
Nombre	Apellido	Nacimiento	Sexo	Estado Civil
Juan	Loza	15/06/1971	H	Soltero
Isabel	Gálvez	23/12/1969	M	Casada
Micaela	Ruiz	02/10/1985	M	Soltera

Tabla1.5.1 Ejemplo de una Tabla.

En realidad, siendo rigurosos, una relación es sólo la definición de la estructura de la tabla, es decir su nombre y la lista de los atributos que la componen. Cuando se genera con las tuplas, se habla de "instancia de relación". Por eso, la tabla anterior representa una instancia de la relación persona. Una representación de la definición de esa relación sería la siguiente:

Personas (nombre, apellido, fecha_nacimiento, sexo, estado_civil)

A continuación, se indicarán ambas (relación e instancia de relación) con el término "relación".

Las tuplas en una relación son una colección no ordenada de elementos diferentes. Para distinguir una tupla de otra, se recurre al concepto de "llave primaria", es decir un conjunto de atributos que permiten identificar unívocamente una tupla en una relación [pons2000].

Naturalmente, en una relación existirán más combinaciones de atributos que permitan identificar unívocamente una tupla ("llaves candidatas"), pero entre éstas se elegirá una sola para utilizar como llave primaria. Los atributos de la llave primaria nunca asumirán el valor nulo (que significa un valor no determinado), en tanto que ya no permitirían identificar una tupla concreta en una relación.

Esta propiedad de las relaciones y de sus llaves primarias está bajo el nombre de integridad de las entidades (entity integrity). A menudo, para obtener una llave primaria "económica", es decir compuesta de pocos atributos fácilmente manipulables, se introducen uno o más atributos ficticios, con códigos identificativos unívocos para cada tupla de la relación [pons2000].

Cada atributo de una relación se caracteriza por un nombre y por un dominio. El dominio indica qué valores son asumidos por una columna de la relación. A menudo un dominio se define a través de la declaración de un tipo para el atributo (por ejemplo diciendo que es una cadena de diez caracteres), pero también es posible definir dominios más complejos y precisos. Por ejemplo, para el atributo "sexo" de la relación "Personas" se define un dominio por el cual los únicos valores válidos son 'M' y 'F'; o bien por el atributo "fecha_nacimiento" se plantea un dominio por el que se consideren válidas sólo las fechas de nacimiento después del primero de enero de 1960, si la base de datos no está previsto que se contemplen personas con fecha de nacimiento anterior a esa. El motor de datos se ocupará de controlar que en los atributos de las relaciones se incluyan sólo los valores permitidos por sus dominios.

Característica fundamental de los dominios de una base de datos relacional es que sean "atómicos", es decir que los valores contenidos en las columnas no se separen en valores de dominios más simples. Más formalmente se dice que no es posible considerar atributos multivalor (multivalued). Por ejemplo, si una característica de las personas en la base de datos fuese la de tener uno o más hijos, no sería posible escribir la relación Personas de la siguiente manera:

Personas (nombre, apellido, fecha_nacimiento, sexo, estado_civil, hijos)

En efecto, el atributo hijos es un atributo no-atómico, bien porque una persona tiene más de un hijo o porque cada hijo tendrá diferentes características que lo describen. Para representar estas entidades en una base de datos relacional hay que definir dos relaciones:

Personas (*número_persona, nombre, apellido, fecha_nacimiento, sexo, estado_civil)
Hijos (*número_persona, *nombre_apellido, edad, sexo)

En las relaciones precedentes, los asteriscos (*) indican los atributos que componen sus llaves primarias. Nótese la introducción en la relación Personas del atributo número_persona, a través del cual se asigna a cada persona un identificativo numérico unívoco que se usa como llave primaria [pons2000].

Estas relaciones contienen sólo atributos atómicos. Si una persona tiene más de un hijo, éstos se representarán en tuplas diferentes de la relación Hijos. Las diferentes características de los hijos las representan los atributos de la relación Hijos. La unión entre las dos relaciones está constituida por los atributos número_persona que aparecen en ambas relaciones y que permiten que se asigne cada tupla de la relación hijos a una tupla concreta de la relación Personas [pons2000].

Más formalmente se dice que el atributo número_persona de la relación Hijos es una llave externa (foreign key) hacia la relación Personas. Una llave externa es una combinación de atributos de una relación que son, a su vez, una llave primaria para otra relación. Una característica fundamental de los valores presentes en una llave externa es que, a no ser que no sean null, tienen que corresponder a valores existentes en la llave primaria de la relación a la que se refieren. En el ejemplo, esto significa que no existirá en la relación Hijos una tupla con un valor del atributo número_persona sin que también en la relación Personas exista una tupla con el mismo valor para su llave primaria. Esta propiedad recibe el nombre de integridad referencial (referential integrity).
Una de las grandes ventajas del modelo relacional es que define también un álgebra, llamada "álgebra relacional". Todas las manipulaciones posibles sobre las relaciones se obtienen gracias a la combinación de tan sólo cinco operadores: RESTRICT, PROJECT, TIMES, UNION y MINUS.

También se han definido tres operadores adicionales que de todos modos se obtienen aplicando los cinco fundamentales: JOIN, INTERSECT y DIVIDE. Los operadores relacionales reciben como argumento una relación o un conjunto de relaciones y restituyen una única relación como resultado.

Veamos brevemente estos ocho operadores [pons2000]:

RESTRICT: restituye una relación que contiene un subconjunto de las tuplas de la relación a la que se aplica. Los atributos se quedan como estaban.

PROJECT: restituye una relación con un subconjunto de los atributos de la relación a la que viene aplicado. Las tuplas de la relación resultado se componen de las tuplas de la relación original, de manera que siguen siendo un conjunto en sentido matemático.

TIME: se aplica a dos relaciones y efectúa el producto cartesiano de las tuplas. Cada tupla de la primera relación está concatenada con cada tupla de la segunda.

JOIN: se concatenan las tuplas de dos relaciones de acuerdo con el valor de un conjunto de sus atributos.

UNION: aplicando este operador a dos relaciones compatibles, se obtiene una que contiene las tuplas de ambas relaciones. Dos relaciones son compatibles si tienen el mismo número de atributos y los atributos correspondientes en las dos relaciones tienen el mismo dominio.

MINUS: aplicado a dos relaciones compatibles restituye una tercera que contiene las tuplas que se encuentran sólo en la primera relación pero no en la segunda.

INTERSECT: aplicado a dos relaciones compatibles restituye una relación que contiene las tuplas que existen en ambas.

DIVIDE: aplicado a dos relaciones que tengan atributos comunes, restituye una tercera que contiene todas las tuplas de la primera relación que hacen que correspondan con todos los valores de la segunda relación.

Las bases de datos relacionales efectúan todas las operaciones en las tablas usando el álgebra relacional, aunque normalmente no le permiten al usuario emplearla. El usuario interacciona con la base de datos a través de una interfaz diferente el lenguaje SQL, un lenguaje declarativo que permite escribir conjuntos de datos. Las instrucciones SQL vienen descompuestas por el motor de datos en una serie de operaciones relacionales [pons2000].

1.6 MySQL

Es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB desarrolla MySQL como software libre en un esquema de licenciamiento dual. En enero de 2008, **MySQL AB** fué adquirida por Sun Microsystems, y por tanto MySQL [perez2008].

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero las empresas que requieran incorporarlo en productos privativos compran a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario de los proyectos como Apache, donde el software es desarrollado por una comunidad pública y el copyright del código está en poder del autor individual, MySQL es propiedad y está patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

SQL (*Lenguaje de Consulta Estructurado*) fue comercializado por primera vez en 1981 por IBM, el cual fue presentado a ANSI y desde ese entonces ha sido considerado como un estándar para las bases de datos relacionales.

Desde 1986, el estándar SQL ha aparecido en diferentes versiones como por ejemplo: SQL:92, SQL:99, SQL:2003. MySQL es una idea originaria de la empresa opensource MySQL AB establecida inicialmente en Suecia en 1995 y cuyos fundadores son David Axmark, Allan Larsson, y Michael "Monty" Widenius. El objetivo que persigue esta empresa consiste en que **MySQL** cumpla el estándar SQL, pero sin sacrificar velocidad, fiabilidad o usabilidad [perez2008].

Michael Widenius en la década de los 90 trató de usar *mSQL* (Mini SQL) para conectar las tablas usando rutinas de bajo nivel ISAM (Método de Acceso Secuencial Indexado), sin embargo, mSQL no era rápido y flexible para sus necesidades. Esto lo conllevó a crear una API SQL denominada **MySQL** para bases de datos muy similar a la de *mSQL* pero más portable.

Existen varias APIs que permiten, a aplicaciones escritas en diversos lenguajes de programación, acceder a las bases de datos MySQL, incluyendo C, C++, C#, Pascal, Delphi (via dbExpress), Eiffel, Smalltalk, Java (con una implementación nativa del driver de Java), Lisp, Perl, PHP, Python, Ruby,Gambas, REALbasic (Mac), FreeBASIC, y Tcl; cada uno de estos utiliza una API específica. También existe un interfaz ODBC, llamado MyODBC que permite a cualquier lenguaje de programación que soporte ODBC comunicarse con las bases de datos MySQL.

MySQL es muy utilizado en aplicaciones web como MediaWiki, Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL [perez2008].

MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero provoca problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

Especificaciones:

1.6.1 Plataformas.

MySQL funciona sobre múltiples plataformas, incluyendo [sanchez2004]:

- AIX
- BSD
- FreeBSD
- HP-UX
- GNU/Linux
- Mac OS X
- NetBSD
- Novell Netware
- OpenBSD
- OS/2 Warp
- QNX
- SGI IRIX
- Solaris
- SunOS
- SCO OpenServer
- SCO UnixWare
- Tru64
- Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, Windows Vista y otras versiones de Windows [sanchez2004].

1.6.2 Características de la versión 5.0.22.

- Un amplio subconjunto de ANSI SQL 99, y varias extensiones.
- Soporte a multiplataforma

- Procedimientos almacenados
- Triggers
- Cursores
- Vistas actualizables
- Soporte a VARCHAR
- INFORMATION_SCHEMA
- Modo Strict
- Soporte X/Open XA de transacciones distribuidas; transacción en dos fases como parte de esto, utilizando el motor InnoDB de Oracle [sanchez2004].
- Motores de almacenamiento independientes (MyISAM para lecturas rápidas, InnoDB para transacciones e integridad referencial)
- Transacciones con los motores de almacenamiento InnoDB, BDB Y Cluster; puntos de recuperación(savepoints) con InnoDB
- Soporte para SSL
- Query caching
- Sub-SELECTs (o SELECTs anidados)
- Réplica con un maestro por esclavo, varios esclavos por maestro, sin soporte automático para multiples maestros por esclavo.
- indexing y buscando campos de texto completos usando el motor de almacenamiento MyISAM
- Embedded database library
- Soporte completo para Unicode
- Conforme a las reglas ACID usando los motores InnoDB, BDB y Cluster
- Shared-nothing clustering through MySQL Cluster [sanchez2004].

1.6.3 Características Adicionales:

- Usa GNU Automake, Autoconf, y Libtool para portabilidad
- Uso de multihilos mediante hilos del kernel.
- Usa tablas en disco b-tree para búsquedas rápidas con compresión de índice
- Tablas hash en memoria temporales.
- El código MySQL se prueba con Purify (un detector de memoria perdida comercial) así como con Valgrind, una herramienta GPL

- Completo soporte para operadores y funciones en cláusulas select y where.
- Completo soporte para cláusulas group by y order by, soporte de funciones de agrupación.
- Seguridad: ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está cifrado al conectarse a un servidor.
- Soporta gran cantidad de datos. MySQL Server tiene bases de datos de hasta 50 millones de registros [sanchez2004].
- Se permiten hasta 64 índices por tabla (32 antes de MySQL 4.1.2). Cada índice consiste desde 1 hasta 16 columnas o partes de columnas. El máximo ancho de límite son 1000 bytes (500 antes de MySQL 4.1.2).
- Los clientes se conectan al servidor MySQL usando sockets TCP/IP en cualquier plataforma. En sistemas Windows se conectan usando named pipes y en sistemas Unix usando archivos socket Unix.
- En MySQL 5.0, los clientes y servidores Windows se conectan usando memoria compartida.
- MySQL contiene su propio paquete de pruebas de rendimiento proporcionado con el código fuente de la distribución de MySQL [sanchez2004].

1.6.4 Características (versión 4.0).

Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, atrajo a los desarrolladores de páginas web con contenido dinámico, justamente por su simplicidad.

Poco a poco los elementos de los que carecía MySQL están siendo incorporados tanto por desarrollos internos, como por desarrolladores de software libre. Entre las características disponibles en las últimas versiones se destaca [perez2008]:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
- Disponibilidad en gran cantidad de plataformas y sistemas.
- Diferentes opciones de almacenamiento según si se desea velocidad en las operaciones o el mayor número de operaciones disponibles.
- Transacciones y claves foráneas.
- Conectividad segura.

- Replicación.
- Búsqueda e indexación de campos de texto.

MySQL es un sistema de administración de bases de datos. Para agregar, acceder y procesar datos guardados en una computadora, usted necesita un administrador como MySQL Server. Dado que las computadoras manejan grandes cantidades de información, los administradores de bases de datos juegan un papel central en aplicaciones independientes o como parte de otras aplicaciones [perez2008].

Las siguientes características son implementadas únicamente por MySQL:

- Múltiples motores de almacenamiento (MyISAM, Merge, InnoDB, BDB, Memory/heap, MySQL Cluster, Federated, Archive, CSV, Blackhole y Example en 5.x), permitiendo al usuario escoger la que sea más adecuada para cada tabla de la base de datos.
- Agrupación de transacciones, reuniendo múltiples transacciones de varias conexiones para incrementar el número de transacciones por segundo.

1.7 NetBeans

Se refiere a una plataforma para el desarrollo de aplicaciones de escritorio usando Java y a un entorno de desarrollo integrado (IDE) usando la Plataforma NetBeans.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados *módulos*. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos son extendidas agregándole nuevos módulos. Debido a que los módulos son desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans tiene la facultad de ser extendidas fácilmente por otros desarrolladores de software [wikipedia2008].

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos.

1.7.1 Historia.

NetBeans comenzó como un proyecto estudiantil en Republica Checa (originalmente llamado Xelfi), en 1996 bajo la tutoría de la Facultad de Matemáticas y Física en la Universidad de Charles en Praga. La meta era escribir un entorno de desarrollo integrado (IDE) para Java parecida a la de Delphi. Xelfi fue el primer entorno de desarrollo integrado escrito en Java, con su primer pre-release en 1997 [wikipedia2008].

Xelfi fue un proyecto divertido para trabajar, ya que las IDEs escritas en Java eran un territorio desconocido en esa época. El proyecto atrajo suficiente interés, por lo que los estudiantes, después de graduarse, decidieron que se convertiría en un proyecto comercial. Prestando espacios web de amigos y familiares, formaron una compañía alrededor de esto. Casi todos ellos siguen trabajando en NetBeans.

Tiempo después, ellos fueron contactados por Roman Stanek, un empresario que ya había estado relacionado con varias iniciativas en la Republica Checa. Él estaba buscando una buena idea en que invertir, y encontró en Xelfi una buena oportunidad. Ellos se reunieron, y el negocio surgió [wikipedia2008].

El plan original era desarrollar unos componentes JavaBeans para redes. Jarda Tulach, quien diseñó la arquitectura básica de la IDE, surgió con la idea de llamarlo NetBeans, con el fin de describir lo que ellos harían. Cuando las especificaciones de los Enterprise JavaBeans salieron, ellos decidieron trabajar con este estándar, ya que no tenía sentido competir con él, sin embargo el nombre de NetBeans se quedó.

En la primavera de 1999, Netbeans DeveloperX2 fue lanzado, soportando Swing. Las mejoras de rendimiento que llegaron con el JDK 1.3, lanzado en otoño de 1999, hicieron a NetBeans una alternativa realmente viable para el desarrollo de herramientas.

En el verano de 1999, el equipo trabajó duro para rediseñar a DeveloperX2 en un NetBeans más modular, lo que lo convirtió en la base de NetBeans hoy en día.

Algo más pasó en el verano de 1999. Sun Microsystems quería una mejor herramienta de desarrollo de Java, y comenzó a estar interesado en NetBeans. En otoño de 1999, con la nueva generación de NetBeans en Beta, el acuerdo fue realizado.

Sun adquirió otra compañía de herramientas al mismo tiempo, Forté, y decidió renombrar NetBeans a Forté for Java. El nombre de NetBeans desapareció de vista por un tiempo.

meses después, se tomó la decisión de hacer a NetBeans open source. Mientras que Sun había contribuido considerablemente con líneas de código en varios proyectos de código abierto a través de los años, NetBeans se convirtió en el primer proyecto de código abierto patrocinado por ellos. En Junio del 2000 NetBeans.org fue lanzado [wikipedia2008].

1.7.2 NetBeans Hoy.

Un proyecto de código abierto no es nada más ni nada menos que un proceso. Toma tiempo encontrar el equilibrio. El primer año, fue crucial como inicio. Los dos años siguientes, se orientó hacia código abierto. Como muestra de lo abierto que era, en los primeros dos años había más debate que implementación.

Con NetBeans 3.5 se mejoró enormemente en desempeño, y con la llegada de NetBeans 3.6, se re implementó el sistema de ventanas y la hoja de propiedades, y se limpió enormemente la interfaz. NetBeans 4.0 fue un gran cambio en cuanto a la forma de funcionar del IDE, con nuevos sistemas de proyectos, con el cambio no solo de la experiencia de usuario, sino del reemplazo de muchas piezas de la infraestructura que tuvo NetBeans anteriormente.

NetBeans IDE 5.0 introdujo un soporte mucho mejor para el desarrollo de nuevos módulos, el nuevo constructor intuitivo de interfaces Matisse, un nuevo y rediseñado soporte de CVS, soporte a Sun ApplicationServer 8.2, Weblogic9 y JBoss 4 [wikipedia2008].

1.7.3 La Plataforma NetBeans.

Durante el desarrollo del NetBeans IDE ocurrió una cosa interesante. La gente empezó a construir aplicaciones usando el NetBeans core runtime con sus propios plug-ins, de hecho, esto se convirtió en un mercado bastante grande.

La Plataforma NetBeans es una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes.

Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones [wikipedia2008].

La plataforma ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma están:

- Administración de las interfaces de usuario (ej. menús y barras de herramientas)
- Administración de las configuraciones del usuario
- Administración del almacenamiento (guardando y cargando cualquier tipo de dato)
- Administración de ventanas
- Framework basado en asistentes (diálogos paso a paso)

1.7.4 NetBeans IDE.

El IDE NetBeans es un IDE - una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero sirve para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso.

El NetBeans IDE es un IDE de código abierto escrito completamente en Java usando la plataforma NetBeans. El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Entre sus características se encuentra un sistema de proyectos basado en Ant, control de versiones y refactoring [wikipedia2008].

La versión actual es NetBeans IDE 5.5.1, la cual fue lanzada en Mayo de 2007. NetBeans IDE 5.5 extiende las características existentes del Java EE (*Java Enterprise Edition*). Adicionalmente, el NetBeans Enterprise Pack soporta el desarrollo de Aplicaciones empresariales con Java EE 5, incluyendo herramientas de desarrollo visuales de SOA (*Arquitectura Orientada a Servicios*), herramientas de esquemas XML (*Extensible Markup Language*), orientación a web servicies y modelado UML.

El NetBeans C/C++ Pack soporta proyectos de C/C++.

Modularidad. Todas las funciones del IDE son provistas por módulos. Cada módulo provee una función bien definida, tales como el soporte de Java, edición, o soporte para el sistema de control de versiones. NetBeans contiene todos los módulos necesarios para el desarrollo de aplicaciones Java en una sola descarga, permitiéndole al usuario comenzar a trabajar inmediatamente [wikipedia2008].

Sun Studio, Sun Java Studio Enterprise, y Sun Java Studio Creator de Sun Microsystems han sido todos basados en el IDE NetBeans.

Desde Julio de 2006, NetBeans IDE es licenciado bajo la Common Development and Distribution License (CDDL), una licencia basada en la Mozilla Public License (MPL).

NetBeans Enterprise Pack
Provee Soporte para la creación de aplicaciones Orientadas a Servicios (SOA), incluyendo herramientas de esquemas XML, un editor WSDL (*Web Services Description Language*), y un editor BPEL (*Lenguaje de Ejecución de Procesos de Negocio*) para web services.

Pack de Movilidad de NetBeans
El Pack de Movilidad de NetBeans es una herramienta para desarrollar aplicaciones que se ejecutan en teléfonos móviles

El Pack de Movilidad puede ser usado para escribir, probar, y depurar aplicaciones para la plataforma Java ME, tecnología existente en dispositivos móviles [wikipedia2008].

Profiler de NetBeans
El **Profiler de NetBeans** es una herramienta para la optimización de aplicaciones Java: Ayuda a encontrar cuellos de botella en la memoria y a optimizar la velocidad de las aplicaciones.

El C/C++ Native Development Module
Soporta proyectos de C/C++.

El Visual Web Pack
Permite rápida y visualmente, construir aplicaciones web estándar, incluyendo Soporte para AJAX (*Asynchronous JavaScript And XML*) y componentes JSF.

Ruby
Posee facilidades para el desarrollo de aplicaciones en ruby y ruby on rails, con mejoras sobresalientes en las nuevas versiones del IDE.

1.8 Modelo Cliente – Servidor

Esta arquitectura consiste básicamente en que un programa -el cliente- realiza peticiones a otro programa -el servidor- que le da respuesta. Aunque esta idea se aplica a programas que se ejecutan sobre una sola computadora es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras [ceballos2006].

En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un sólo programa.

Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo, los servidores del correo, etc. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma.

Una disposición muy común son los sistemas multicapa en los que el servidor se descompone en diferentes programas que son ejecutados por diferentes computadoras aumentando así el grado de distribución del sistema.

La arquitectura cliente-servidor sustituye a la arquitectura monolítica en la que no hay distribución, tanto a nivel físico como a nivel lógico [ceballos2006].

1.8.1 Características de un cliente.

En la arquitectura C/S el **remitente de una solicitud** es conocido como cliente. Sus características son:

- Es quien inicia solicitudes o peticiones, tienen por tanto un papel activo en la comunicación (dispositivo **maestro** o **amo**).
- Espera y recibe las respuestas del servidor.
- Por lo general, pueden conectase a varios servidores a la vez.
- Normalmente interactúa directamente con los usuarios finales mediante una interfaz gráfica de usuario.

1.8.2 Características de un servidor.

En los sistemas C/S el receptor de la solicitud enviada por cliente se conoce como servidor. Sus características son [ceballos2006]:

- Al iniciarse esperan a que lleguen las solicitudes de los clientes, desempeñan entonces un papel pasivo en la comunicación (dispositivo esclavo).
- Tras la recepción de una solicitud, la procesan y luego envían la respuesta al cliente.

- Por lo general, aceptan conexiones desde un gran número de clientes (en ciertos casos el número máximo de peticiones puede estar limitado).
- No es frecuente que interactúen directamente con los usuarios finales.

1.8.3 Comparación de la Arquitectura Cliente-Servidor con otras Arquitecturas de Red.

Comparación con las redes de pares

Las redes de pares, también conocidas como redes par-a-par o peer-to-peer (abreviado con las siglas P2P) son otro tipo de arquitectura de red; en ellas cada nodo o elemento del sistema puede actuar al mismo tiempo como cliente y como servidor; cada nodo tiene, por tanto, las responsabilidades y estados de ambos elementos.

Comparación con la arquitectura Cliente-Cola-Cliente

Si bien la clásica arquitectura C/S requiere uno de los puntos terminales de comunicación para actuar como un servidor, la arquitectura Cliente-Cola-Cliente habilita a todos los nodos para actuar como clientes simples, mientras que el servidor actúa como una cola que va capturando las peticiones de los clientes (un proceso que debe pasar sus peticiones a otro, lo hace a través de una cola, por ejemplo, una consulta a una base de datos, entonces, el segundo proceso conecta con la base de datos, elabora la petición, la coloca en la base de datos, etc.).

Arquitectura multi-capas

La arquitectura cliente/servidor genérica tiene dos tipos de nodos en la red: clientes y servidores. Consecuentemente, estas arquitecturas genéricas se refieren a veces como arquitecturas de dos niveles o dos capas.

Algunas redes disponen de tres tipos de nodos:

- Clientes que interactúan con los usuarios finales.
- Servidores de aplicación que procesan los datos para los clientes.
- Servidores de la base de datos que almacenan los datos para los servidores de aplicación.

Esta configuración se llama una arquitectura de la tres-capas.

- Ventajas de las arquitecturas n-capas:
  - La ventaja fundamental de una arquitectura n-capas comparado con una arquitectura de dos niveles (o una tres-capas con una de dos niveles) es que separa hacia fuera el proceso, eso ocurre para mejorar el balance la carga en los diversos servidores; es más escalable.

- Desventajas de las arquitecturas de la n-capas:
  - Coloca más carga en la red, debido a una mayor cantidad de tráfico de la red.
  - Es mucho más difícil programar y probar el software que en arquitectura de dos niveles porque tienen que comunicarse más dispositivos para terminar la transacción de un usuario [ceballos2006].

1.9 Microsoft Visio

Microsoft Visio es un software de dibujo vectorial para Microsoft Windows. Visio comenzó a formar parte de los productos de Microsoft cuando fue adquirida la compañía Visio en el año 2000 [microsoft 2008].

Las herramientas que lo componen permiten realizar diagramas de oficinas, diagramas de bases de datos, diagramas de flujo de programas, UML, y más, que permiten iniciar al usuario en los lenguajes de programación.

El navegador Internet Explorer incluye un visor de diagramas Visio, cuya extensión es vsd, llamado Visio Viewer.

Aunque originalmente apuntaba a ser una aplicación para dibujo técnico para el campo de Ingeniería y Arquitectura; con añadidos para desarrollar diagramas de negocios, su adquisición por Microsoft implicó drásticos cambios de directrices de tal forma que a partir de la versión de Visio para Microsoft Office 2003 el desarrollo de diagramas para negocios pasó a ser de un añadido a ser el núcleo central de negocio, minimizando las funciones para desarrollo de planos de Ingeniería y Arquitectura que se habían mantenido como principales hasta antes de la compra [microsoft 2008].

CAPITULO 2. ANALISIS Y ESPECIFICACION DE REQUERIMIENTOS.

2.1 Introducción.

Para que un proyecto de desarrollo de software tenga éxito es crucial realizar una comprensión total de los requerimientos del software a diseñar.

En la etapa del análisis y la especificación de requerimientos, tanto el cliente, como el desarrollador juegan un rol fundamental, debido a que el primero se encarga de describir las necesidades que le apremian, mientras que el segundo es el encargado de dar solución a dichas necesidades. Debido a que la especificación es complicada de detallar, desde el comienzo del desarrollo de los sistemas se ha tratado de realizar una adecuada identificación de los requisitos del sistema derivados de las necesidades de los usuarios.

Por todo esto, dentro de la Ingeniería existe una rama que se dedica a la captura de requerimientos, la cual es la Ingeniería de requerimientos cuyo propósito general es desarrollar técnicas para que este proceso fundamental se realice en forma eficiente y segura.

La ingeniería de requerimientos se divide en cuatro etapas, las cuales son: estudio de factibilidad, obtención y análisis de requerimientos, especificación de requerimientos y validación de requerimientos. Estas etapas se observan en la figura 2.1:

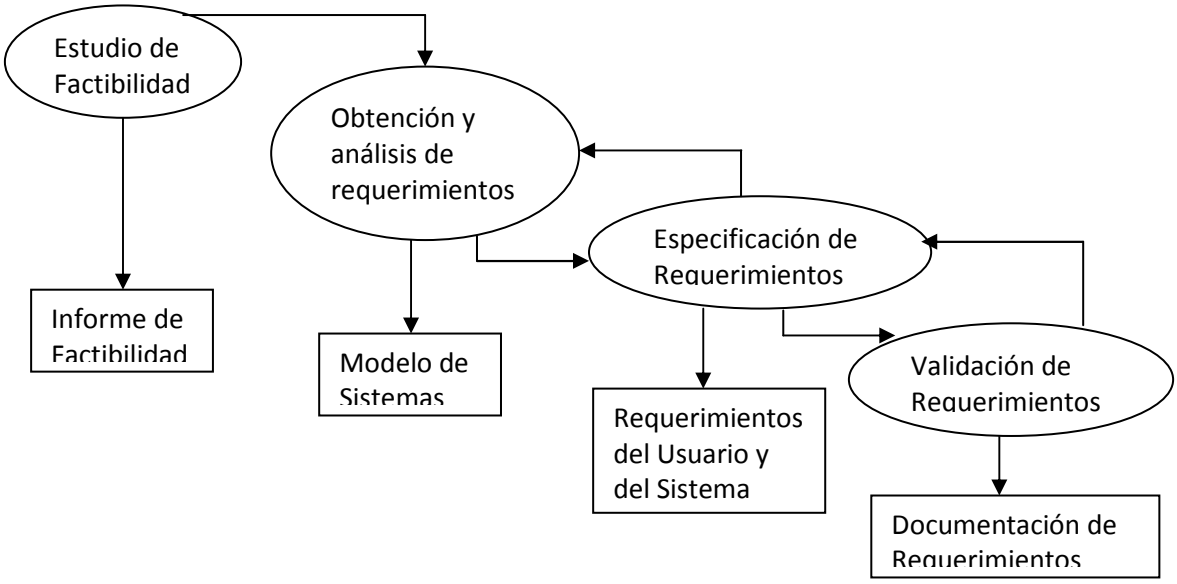

Figura 2.1 Etapas de la Ingenieria de Requerimientos

2.1.1 Etapas del Análisis y Especificación de Requerimientos

El proceso de Ingeniería de requerimientos posee tres etapas: estudio de factibilidad, obtención de requerimientos y validación de requerimientos. Cada una de las siguientes etapas se especifica a continuación.

a. Estudio de factibilidad

El resultado de esta etapa es producir un informe de factibilidad como se ilustra en la figura 2.1 que consiste, tanto en realizar una recolección y evaluación de la información, como redactar el informe del estudio de la factibilidad.

b. Obtención y análisis de requerimientos

El objetivo de esta etapa es determinar: el dominio de la aplicación, desempeño del sistema, las restricciones que el sistema debe poseer, entre otras cosas. En esta etapa toman principal importancia los *stakeholders*, los cuales son aquellas personas con alguna influencia ya sea directa o indirecta en los requerimientos del sistema, es decir, pueden ser los usuarios finales, ingenieros desarrolladores, ingenieros de mantenimiento, etc.

c. Validación de Requerimientos

En esta etapa se establecen los requerimientos finales ó completos que definirán el sistema que el cliente desea.

2.2 Estudio de factibilidad

En la red existen diversas páginas de venta de bisutería las cuales ofrecen sus servicios, en estas páginas de cada negocio sus servicios están delimitados en diversas funciones pues algunas solo ofrecen vistas de sus materiales o productos solo mostrando imágenes y descripciones de estos sin permitir el manejo de información como capturar nuevos clientes, realizar compras o visualizar un catalogo más especifico como en:
www.meximanos.com, www.bimoda.com, www.felixbisuteria.com, www.bisutalia.com, www.bijoux-guenita.com/bisuteria,
www.elgaleon.com.mx/?gclid=CL_siNypspECFSTMiQodCVRyNg.
Otras páginas de internet ofrecen un registro de clientes y vista de productos pero no realizan ventas en ese momento y sólo se visualiza la información delos productos. Así se observa que cada página solo muestra funciones o porciones de las mismas que delimitan el alcance de lo que es un sistema completo o el progreso del mismo negocio.

Otro aspecto muy importante es que los establecimientos de bisuterías no cuentan con un sistema que controle su negocio de forma interna por cuestión de costos lo que hace que tarden tiempo en levantar pedidos o facturar compras en el aspecto del llenado a mano.

Así, este sistema ofrece la facilidad de contar con una cartera de clientes y proveedores para llevar un control sobre los mismos, además de considerar un sistema de impresiones lo cual les ahorrará tiempo en el llenado ya que este es automático, al realizar una venta de sus productos esta será más rápida y eficaz lo cual hará que se levanten más pedidos en menos tiempo. Tendrán una vista digital de sus catálogos para que el cliente tenga una mejor elección sobre que productos comprar.

Todo esto será realizado en diversas máquinas ya que el programa esta disponible incluso para situarse en red local y así ser manipulado por diversas personas a la vez.

Con este sistema se pretende que el negocio de la bisutería mantenga un control más estricto en sus inventarios considerando evitar las pérdidas de material y productos, control en pagos de sus clientes y los pagos a proveedores, realizar ventas de forma correcta y eficaz así como en menos tiempo, también un llenado automático de facturas y remisiones. Además de que el programa contará con los siguientes beneficios hacia los usuarios finales:

- Un manejo amigable con las interfaces del sistema, un mayor control en sus abonos a sus proveedores y de sus clientes.
- Personalización en las impresiones de sus reportes de facturas, remisiones, cotizaciones, inventarios, etc.
- Manejo de seguridad en el sistema mediante usuario-password para evitar la alteración de la información por personas ajenas al sistema.
- El sistema ofrece una conexión cliente-servidor para que este pueda ser utilizado por varios usuarios con sus debidas restricciones sobre el sistema.
- Calidad en el software.
- El proyecto sustenta una metodología efectiva de la ingeniería de software en sistemas administrativos complejos.

2.3 Obtención y Análisis de Requerimientos

Durante la investigación en diversos negocios de bisuterías se recolectó la siguiente información que se considera necesaria para llevar un correcto control de la misma:

1.- Cartera de Clientes:

Aquí se llevará un control absoluto de todos los clientes así como la manipulación de la información del mismo como son: Alta de Clientes, Buscar Clientes, Modificar Clientes y Eliminar Clientes.

2.- Cartera de Proveedores:

Aquí se llevará un control absoluto de todos los proveedores así como la manipulación de la información del mismo como son: Alta de Proveedores, Buscar Proveedores, Modificar Proveedores y Eliminar Proveedores.

3.- Inventario:

En este apartado se llevará el control de productos y de los materiales necesarios para elaborar cada producto.

4.- Catálogos:

Se visualizará un catálogo de materiales y uno de productos en los cuales se mostrará la imagen del producto o material así como su descripción.

5.- Ventas o Pedidos:

En este apartado se realizarán las ventas de productos o materiales según la petición del cliente, para cada venta se remisionará o facturará según la preferencia del cliente.

6.- Compras:

En este apartado se realizarán las compras a sus proveedores tanto de materiales como de productos para llevar un control interno de lo que se compra.

7- Stock:

Aquí se levará el control de las existencias en bodega de los productos y de los materiales, así como la cantidad que se necesita para cubrir su mínimo de stock.

8.- impresiones:

En este punto el usuario realizará las impresiones o reimpresiones dependiendo el caso de sus documentos como los son: Reportes de Stock para productos y materiales, facturas o remisiones de sus ventas, reporte de precios de productos o materiales, reportes de existencias de productos o materiales.

9.- Nuevos:

Se ingresarán los nombres de las nuevas categorías en las que se dividen los productos y los materiales por ejemplo: collares, pulseras, anillos, etc.

10.- Administración:

Solo el usuario que sea administrador accederá a éste apartado, en él se darán de alta los nuevos usuarios así como su eliminación, además la realización de los respaldos de su información.

A continuación se detallará la información necesaria para cada atributo que se describió anteriormente.

1.- Cartera de Cliente:

- Clave de Cliente.
- Nombre Cliente:
  - Apellido paterno.
  - Apellido Materno
  - Nombre(s).
- RFC.
- Dirección:
  - Calle.
  - Número.
  - Colonia.
  - CP.
- Teléfono Local.
- Teléfono Móvil o celular.
- Correo Electrónico.

2.- Cartera de Proveedores:

- Clave de Proveedor.
- Razón Social.
- RFC.
- Dirección:
  - Calle.
  - Número.
  - Colonia.
  - CP.
  - Estado.
  - País.
- Teléfono Local.
- Teléfono Móvil.
- Correo Electrónico.
- Página Web.
- Contacto:
  - Nombre.
  - Puesto.

3.- Inventarios:

- Materiales:
  - Clave del Material.
  - Categoría.
  - Nombre del Proveedor.
  - Descripción del Material.
  - Precio Proveedor.
  - Precio Menudeo.
  - Precio Mayoreo.
  - Existencia Máxima.
  - Existencia Mínima.
  - Existencia en Bodega.
  - Imagen.
- Productos:
  - Clave del Producto.
  - Descripción del Producto.
  - Tipo_producto (manufacturado, comprado).
  - Nombre del Proveedor
  - Precio Proveedor.
  - Precio Menudeo.
  - Precio Mayoreo.
  - Categoría.
  - Existencia Máxima.
  - Existencia Mínima.
  - Existencia en Bodega.
  - Materiales Necesarios para la Fabricación:
    - Material.
    - Cantidad de Material a Utilizar.
  - Imagen.

4.- Catálogos:

- Productos:
  - Imagen.
  - Descripción.
  - Precio.
- Materiales:
  - Imagen.
  - Descripción.
  - Precio.

5.- Ventas o Pedidos:

- Folio del Pedido.
- Clave del Cliente.
- Fecha de la venta.
- Tipo de Venta (factura o Remisión).
- Producto o Material Vendido:
    - Tipo (Producto o Material).
    - Categoría.
    - Nombre del Material.
    - Cantidad.
    - Precio.
    - Precio Unitario.
    - Precio Total.
- Subtotal.
- IVA.
- Total
- Cantidad con Letra.

6.- Compras:

- Folio de la Compra.
- Clave del Proveedor.
- Fecha de la Compra.
- Tipo de Compra (factura o Remisión).
- Producto o Material Vendido:
    - Tipo (Producto o Material).
    - Categoría.
    - Nombre del Material o Producto.
    - Cantidad.
    - Precio Unitario.
    - Precio Total.
- Subtotal.
- IVA.
- Total
- Cantidad con Letra.

7.- Stock:

- Productos:
  - Fecha de Consulta.
  - Solicitante del Reporte.
  - Cantidad en Bodega.
  - Cantidad a Comprar.
  - Clave del Producto.
  - Nombre del Producto.
- Materiales:
  - Fecha de Consulta.
  - Solicitante del Reporte.
  - Cantidad en Bodega.
  - Cantidad a Comprar.
  - Clave del Producto.
  - Nombre del Producto.

8.- Impresiones:

- Factura de Ventas en formatos de la Bisutería.
- Remisión de Ventas con diseño de Formato.
- Reportes de Ventas por Intervalos de Fecha.
- Reportes General de Ventas.
- Reportes de Compras por Intervalos de Fecha.
- Reporte General de Compras.
- Reportes de Stock:
  - Materiales.
  - Productos.
- Reporte de fechas de precios para materiales y productos:
  - Por intervalo de fecha:
    - Mejor precio de proveedor.
    - Peor precio de proveedor.
    - Mejor precio de menudeo.
    - Peor precio de menudeo.
    - Mejor precio de mayoreo.
    - Peor precio de mayoreo.
  - General:
    - Todas las fechas para precios de proveedor existentes.
    - Todas las fechas para precios de menudeo existentes.
    - Todas las fechas para precios de mayoreo existentes.

- Reporte de fechas de existencias para materiales y productos:
  - Por intervalo de fecha:
    - Muestra todas las existencias mínimas en el intervalo de fecha ingresado.
    - Muestra todas las existencias máximas en el intervalo de fecha ingresado.
    - Muestra todas las existencias en bodega en el intervalo de fecha ingresado.
  - General:
    - Todas las existencias mínimas.
    - Todas las existencias máximas.
    - Todas las existencias en bodega.

9.- Categorías:

- Tipo_categoria (si es descripción de un material o de un producto).
- Categoría (descripción de la categoría).

10.- Administración:

- Nuevo Usuario:
  - Nombre de Usuario.
  - Contraseña.
  - Confirmación de la Contraseña.
  - Tipo de usuario.
- Ver Usuarios:
  - Tipo de Usuario.
  - Nombre de Usuario.
  - Contraseña.
- Respaldar Información.

CAPITULO 3. DIAGRAMAS

En este capitulo se representará de forma gráfica el software mediante diagramas de entidad relación, casos de uso y el diccionario de datos, describiendo cada diagrama para su mayor comprensión, estos diagramas nos mostraran de forma gráfica el funcionamiento que tendrá el sistema.

3.1 Diagrama Entidad Relación.

Identificación de entidades y atributos.

CLIENTES	PROVEEDORES	MATERIALES	PRODUCTOS	VENTA_CLIENTES
clave_cliente	clave_prov	clave_mat	clave_prod	id_venta_c
ap	razon_social	descripcion	descripcion	fecha
am	rfc	maximo	maximo	status
nombre	puesto	minimo	minimo	tipo_venta(factura,remision)
rfc	contacto	bodega	bodega	
calle	tel_local	imagen	imagen	
numero	tel_movil	precio_prov	precio_prov	
colonia	calle	precio_mayoreo	precio_mayoreo	
cp	numero	precio_menudeo	precio_menudeo	
tel_local	colonia	categoria	categoria	
tel_movil	cp		tipo_prod	
mail	edo			
	pais			
	mail			
	web			

COMPRA_PROVEEDORES	DESC_VENTA_CLI	DESC_COMPRA_PROV	PRECIOS_PRODUCTO
id_compra_p	id_venta_c	id_compra_p	tipo_precio
	cantidad	cantidad	fecha
fecha	tipo	tipo	precio
status	precio_u	precio_u	
tipo_compra(factura,remision)			

PRECIOS_MATERIAL	EXISTENCIAS_PRODUCTO	EXISTENCIAS_MATERIAL	CATALOGO CATEGORIAS
tipo_precio	tipo_existencia	tipo_existencia	tipo_categoria
fecha	Fecha	Fecha	categoria
precio	cantidad	cantidad	

A continuación en la figura 3.1.1 se muestra el diagrama entidad relación del sistema para administración de bisuterías

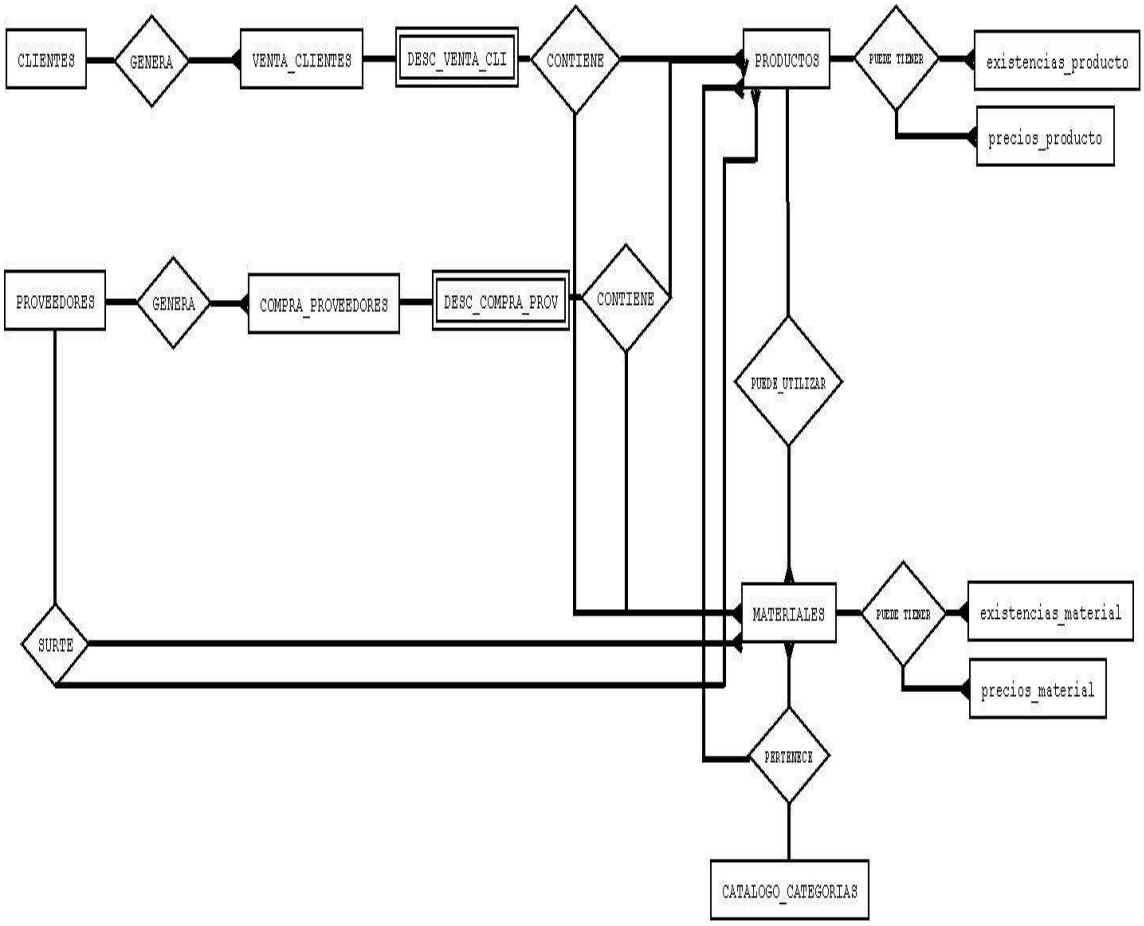

Figura 3.1.1

3.2 Diagrama de Casos de Uso

Identificacion de Actores.
A continuacion se identifican y describen los usuarios finales, tambien llamados actores, los cuales interactuarán con el sistema.

- VENDEDOR: Es la persona encargada de la venta de los productos y materiales, asi como de la atención de la bisuteria, este actor realizará las siguientes funciones:

- Altas y Búsquedas de Clientes.
- Realizar, Buscar e Imprimir Ventas.
- Visualizar el catálogo de Productos/ Materiales.
- Alta, Búsqueda, Actualización y Baja de Productos/Materiales.

- ADMINISTRADOR: Es la persona encargada del correcto funcionamiento de la bisuteria en todos los aspectos como lo son: empleados, atención a cliente y el correcto uso del sistema. Este actor tendrá acceso a todas las funciones del sistema:

- Altas, Baja, Actualización y Búsqueda de Clientes/Proveedores.
- Altas, Baja, Actualización y Búsqueda de Productos/Materiaes.
- Visualizar Catálogos de Productos/Materiales.
- Realizar, Modificar y Cancelar una Venta.
- Realizar, Modificar y Cancelar una Compra.
- Impresión de reportes y Ventas.

Identificacion de Casos de Uso.

- **Login:** Permite validar al usuario para que acceda al sistema asi como los permisos con los que cuenta.
- **Ingresar un Nuevo Cliente:** Permite ingresar los datos necesarios del cliente.
- **Consultar un Cliente por Clave.** Permite realizar la búsqueda de un cliente en específico ingresando su clave de control para mostrar sus datos personales.
  - **Modificar Datos del Cliente:** Permite modificar los datos personales del cliente seleccionado.
  - **Eliminar al Cliente:** Permite eliminar al cliente de la base de datos.
- **Consultar un Cliente por Nombre.** Permite realizar la búsqueda de un cliente en específico ingresando su nombre completo para mostrar sus datos personales.
  - **Modificar Datos del Cliente:** Permite modificar los datos personales del cliente seleccionado.
  - **Eliminar al Cliente:** Permite eliminar al cliente de la base de datos.

- **Ingresar un Nuevo Proveedor:** Permite ingresar los datos necesarios del proveedor.

- **Consultar un Proveedor por Clave.** Permite realizar la búsqueda de un proveedor en específico ingresando su clave de control para mostrar sus datos.
  - **Modificar Datos del Proveedor:** Permite modificar los datos personales del proveedor seleccionado.
  - **Eliminar al Proveedor:** Permite eliminar al proveedor de la base de datos.

- **Consultar un Proveedor por Razon Social.** Permite realizar la búsqueda de un proveedor en específico ingresando su razón social para mostrar sus datos.
  - **Modificar Datos del Proveedor:** Permite modificar los datos personales del proveedor seleccionado.
  - **Eliminar al Proveedor:** Permite eliminar al proveedor de la base de datos.

- **Ingresar un Nuevo Producto:** Permite ingresar los datos para almacenar un nuevo producto en bodega.

- **Consultar un Producto por Clave:** Permite realizar la búsqueda de un producto en específico ingresando su clave de control para mostrar sus datos.
    - **Modificar Datos de un Producto:** Permite modificar los datos del producto seleccionado.
    - **Eliminar un Producto:** Permite eliminar el producto de la base de datos.

- **Consultar un Producto por Nombre:** Permite realizar la búsqueda de un producto en específico ingresando su nombre o descripción para mostrar sus datos.
    - **Modificar Datos de un Producto:** Permite modificar los datos del producto seleccionado.
    - **Eliminar un Producto:** Permite eliminar el producto de la base de datos.

- **Ingresar un Nuevo Material** Permite ingresar los datos para almacenar un nuevo material en bodega.

- **Consultar un Material por Clave:** Permite realizar la búsqueda de un material en específico ingresando su clave de control para mostrar sus datos.
    - **Modificar Datos de un Material:** Permite modificar los datos del material seleccionado.
    - **Eliminar un Material:** Permite eliminar el material de la base de datos.

- **Consultar un Material por Nombre:** Permite realizar la búsqueda de un material en específico ingresando su nombre o descripción para mostrar sus datos.
    - **Modificar Datos de un Material:** Permite modificar los datos del material seleccionado.
    - **Eliminar un Material:** Permite eliminar al material de la base de datos.

- **Ver Catálogo de Productos:** Permite visualizar la imagen de los productos almacenados en bodega.

- **Ver Catálogo de Materiales:** Permite visualizar la imagen de los materiales almacenados en bodega.

- **Levantar una Compra:** Permite realizar una compra de materiales y/o productos para almacenarlos en bodega.

- **Buscar una Compra por Folio:** Permite buscar una compra ya almacenada por su folio y presentar los datos.
    - **Modificar Compra:** Permite modificar los datos de la compra.
    - **Cancelar Compra:** Permite Cancelar la compra.

- **Buscar una Compra por Proveedor** Permite buscar todas las compras que se hayan realizado al proveedor y seleccionando el folio, muestra los datos.
  - **Modificar Compra:** Permite modificar los datos de la compra.
  - **Cancelar Compra:** Permite Cancelar la compra.

- **Levantar una Venta** Permite realizar una venta de materiales y/o productos.

- **Imprimir Venta:** Permite imprimir la venta en el formato que seleccione el usuario (Factura, remisión).

- **Reimprimir Venta:** Permite reimprimir la venta en el formato que seleccione el usuario (Factura, remisión).

- **Buscar una Venta por Folio:** Permite buscar una venta ya almacenada por su folio y presentar los datos.
  - **Modificar Venta:** Permite modificar los datos de la venta.
  - **Cancelar Venta:** Permite Cancelar la venta.

- **Buscar una Venta por Cliente:** Permite buscar todas las ventas que tiene un cliente y seleccionando el folio muestra los datos.
  - **Modificar Venta:** Permite modificar los datos de la venta.
  - **Cancelar Venta:** Permite Cancelar la venta.

- **Generar Reportes.**
  - **Generar Reporte de Ingresos:** Permite generar el reporte de ingresos en un determinado intervalo de fecha.
  - **Generar Reporte de Egresos:** Permite generar el reporte de egresos en un determinado intervalo de fecha.
  - **Generar Reporte de Stock:** Permite generar un reporte de las existencias en bodega.
  - **Generar Reporte de Balance Genaral:** Permite generar un reporte en el cual nos indica el balance en un determinado rango de fecha.

- **Imprimir Reporte:** Permite imprimir el reporte seleccionado.

- **Nuevo Usuario:** Permite ingresar un nuevo usuario al sistema.

- **Ver Usuarios:** Permite visualizar todos los usuarios existentes.
  - **Eliminar Usuario:** Permite eliminar al usuario de la base de datos.

A continuación se muestran los diagramas de flujo del sistema para administración de bisuterías:

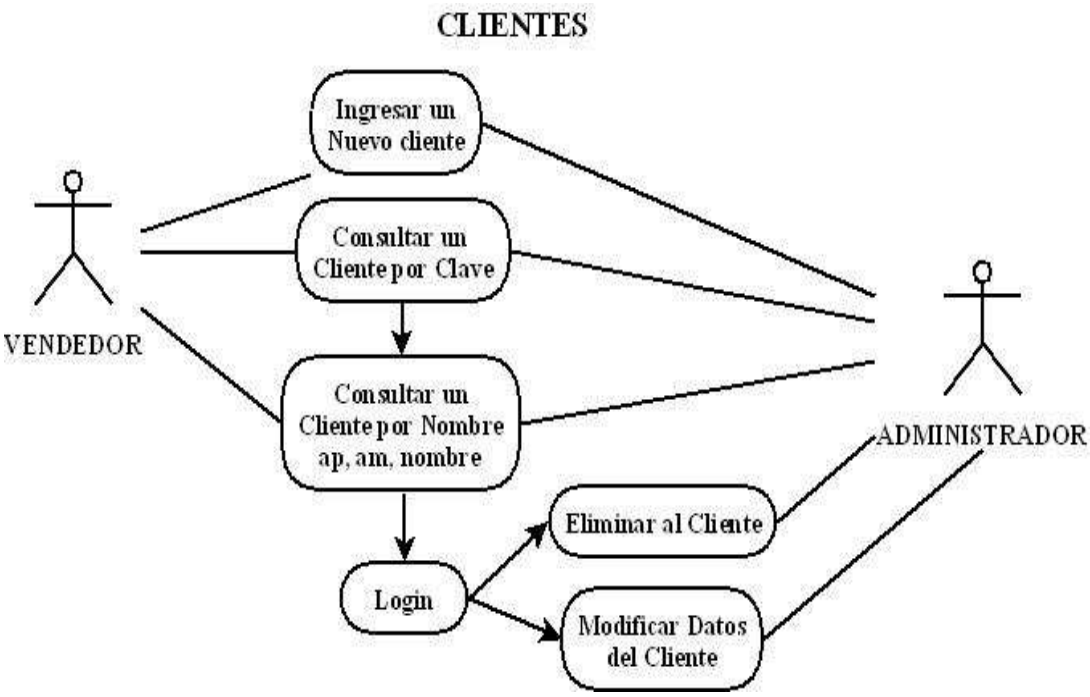

Figura 3.2.1

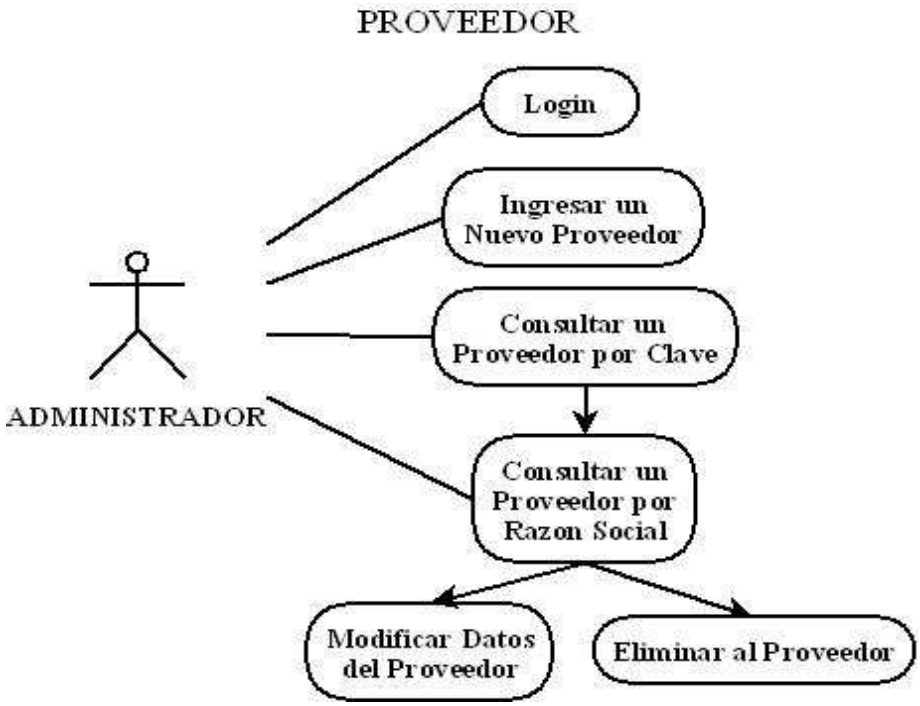

Figura 3.2.2

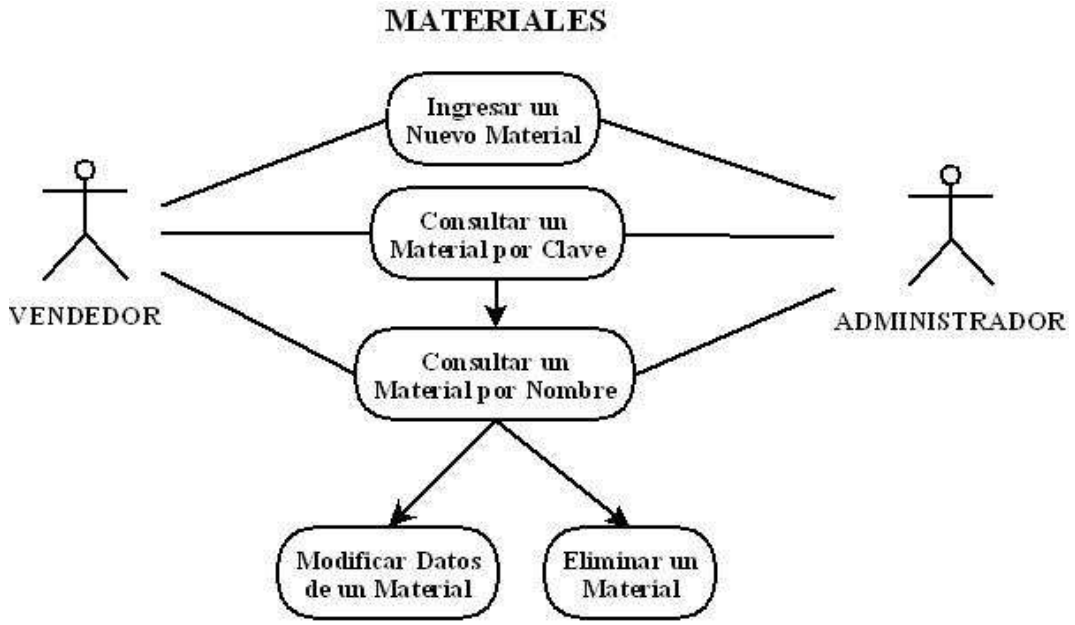

Figura 3.2.3

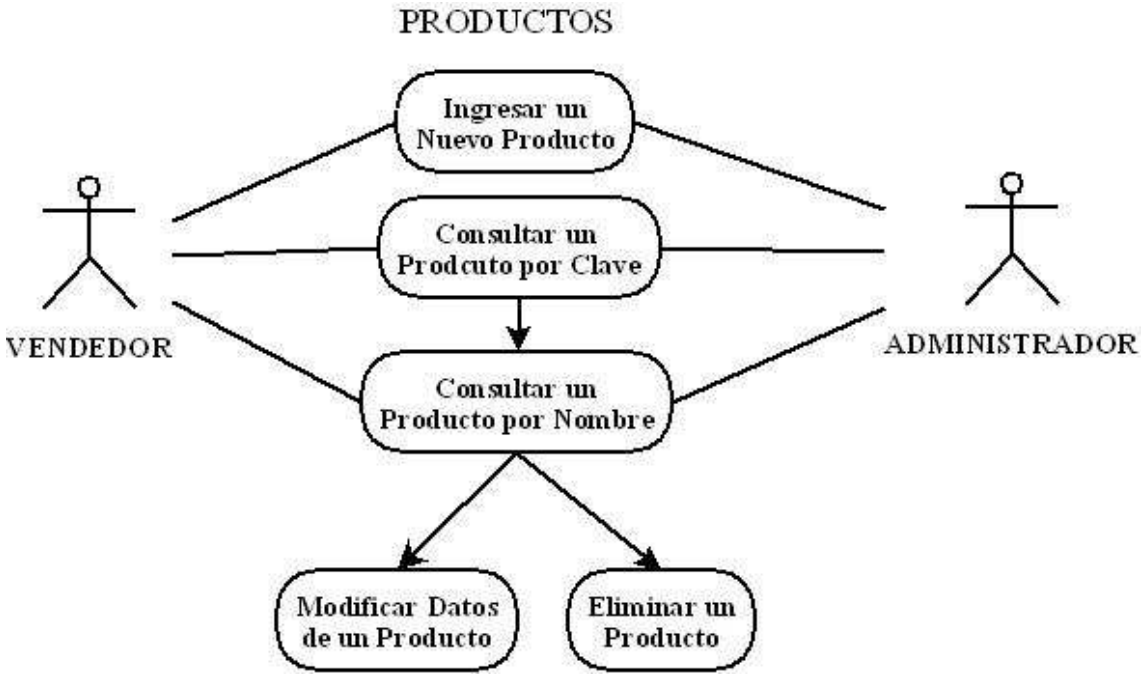

Figura 3.2.4

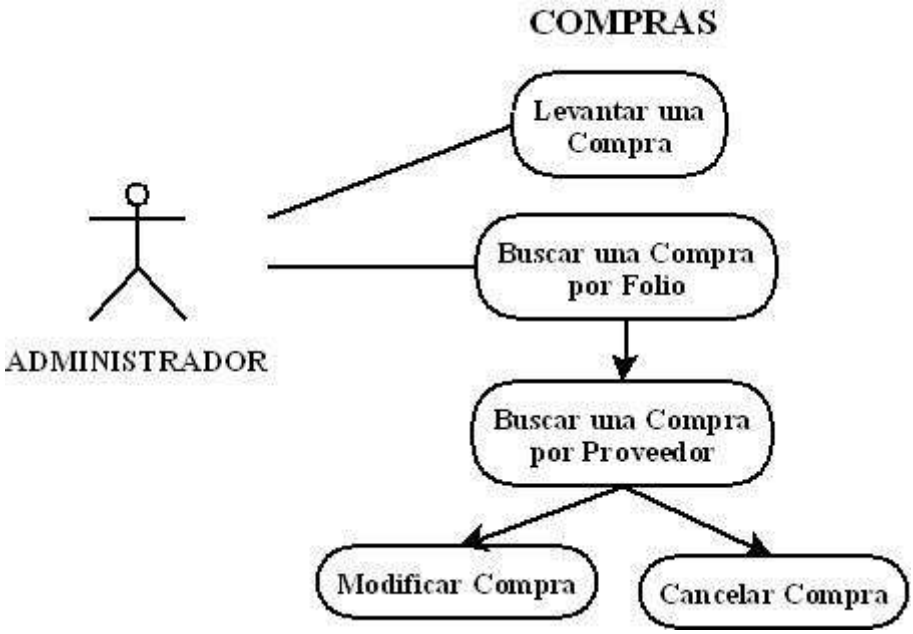

Figura 3.2.5

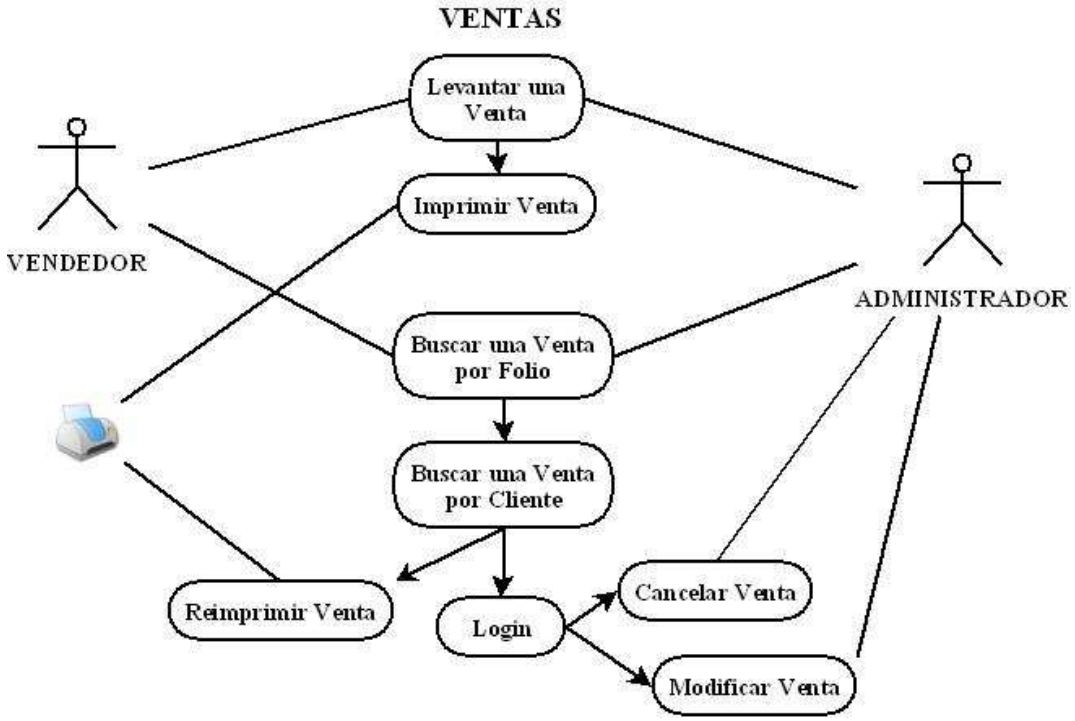

Figura 3.2.6

Figura 3.2.7

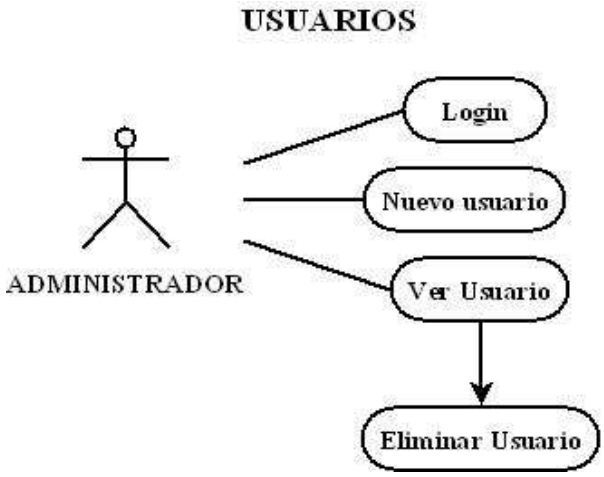

Figura 3.2.8

Figura 3.2.9

Acontinuación se describe de forma detallada algunos de los casos de uso más relevantes en el sistema:

Nombre:	Ingresar un Nuevo Cliente (Figura 2.4.2.1)
Autor:	Erick Cruz Campos
Fecha:	10/11/2008
Descripción: Permite al usuario ingresar los datos de un nuevo cliente al sistema.	
Actores: Usuario del sistema (Administrador o Usuario Limitado).	
Precondiciones: Ninguna.	
Flujo Normal: ➢ El usuario elije la opción “Nuevo Cliente” en el menú o en el botón de acceso rápido. ➢ El Usuario Ingresa los datos del cliente. ➢ Presiona el botón “Guardar”. ➢ El sistema comprueba la validez de los datos ingresados y los almacena.	
Flujo Alternativo: ➢ El sistema comprueba la validez de los datos, si éstos no son los correctos avisa al actor de los errores cometidos para que los pueda modificar.	

Nombre:	Ingresar un Nuevo Producto (Figura 2.4.2.4)
Autor:	Erick Cruz Campos
Fecha:	10/11/2008
Descripción: Permite al usuario ingresar los datos de un nuevo producto elaborado y los materiales que se utilizan para elaborarlo.	
Actores: Usuario del sistema (Administrador o Usuario Limitado).	
Precondiciones: Elegir el tipo de producto a ingresar.	
Flujo Normal: ➢ El usuario ingresa los datos del producto y los materiales que lo conforman. ➢ Presiona el botón “Enviar”. ➢ El sistema comprueba la validez de los datos ingresados y los almacena.	
Flujo Alternativo: ➢ El sistema comprueba la validez de los datos, si éstos no son los correctos avisa al actor de los errores cometidos para que los pueda modificar.	

Nombre:	Levantar una Venta (Figura 2.4.2.6)
Autor:	Erick Cruz Campos
Fecha:	10/11/2008

Descripción:
Permite al usuario ingresar una venta al sistema.

Actores:
Usuario del sistema (Administrador o Usuario Limitado).

Precondiciones:
Tener al cliente registrado en el sistema.

Flujo Normal:

- El usuario elije al cliente al cual se levantara}á la venta.
- El usuario Ingresa los datos necesarios para la venta.
- Presiona el botón “Guardar”.
- El sistema comprueba la validez de los datos ingresados y los almacena.
- Imprime la remisión o factura.

Flujo Alternativo:

- El sistema comprueba la validez de los datos, si éstos no son los correctos avisa al actor de los errores cometidos para que los pueda modificar.

Nombre:	Nuevo Usuario (Figura 2.4.2.8)
Autor:	Erick Cruz Campos
Fecha:	10/11/2008

Descripción:
Permite al Administrador ingresar nuevos usuarios para tener acceso al sistema.

Actores:
Usuario del sistema(Administrador).

Precondiciones:
El usuario del sistema debe ser administrador.

Flujo Normal:

- El usuario elije la opción “Nuevo Usuario” en el menú.
- Ingresa los datos del nuevo usuario.
- Ingresa los permisos que el nuevo usuario tendrá sobre el sistema.
- Presiona el botón “Guardar”.
- El sistema comprueba la validez de los datos ingresados y los almacena.

Flujo Alternativo:

- El sistema comprueba la validez de los datos, si éstos no son los correctos avisa al actor de los errores cometidos para que los pueda modificar.

3.3 Diccionario de Datos.

Clientes.

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
clave_cliente	varchar(10)	No		✓	
ap	varchar(30)	No			✓
am	varchar(30)	No			✓
nombre	varchar(60)	No			✓
rfc	varchar(13)	Sí	*NULL*		
calle	varchar(25)	Sí	*NULL*		
numero	int(5)	Sí	*0*		
colonia	varchar(30)	Sí	*NULL*		
cp	int (5)	Sí	*0*		
tel_local	int(12)	Sí	*0*		
tel_movil	int(12)	Sí	*0*		
mail	varchar(50)	Sí	*NULL*		

Proveedores.

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
clave_prov	varchar(10)	No		✓	
razon_social	varchar(60)	No			✓
rfc	varchar(13)	No			
puesto	varchar(20)	Sí	*NULL*		
contacto	varchar(60)	Sí	*NULL*		
tel_local	int(12)	Sí	*0*		
tel_movil	int(12)	Sí	*0*		
calle	varchar(25)	Sí	*NULL*		
numero	int(5)	Sí	*0*		
colonia	varchar(50)	Sí	*NULL*		
cp	int(5)	Sí	*0*		
edo	varchar(60)	Sí	*NULL*		
pais	varchar(60)	Sí	*NULL*		
mail	varchar(50)	Sí	*NULL*		
web	varchar(50)	Sí	*NULL*		

Materiales.

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
clave_mat	varchar(10)	No		✓	
clave_prov	varchar(10)	No			✓
descripcion	varchar(50)	No			✓
maximo	int(4)	Sí	0		
minimo	int(4)	Sí	0		
bodega	int(4)	Sí	0		
imagen	longblob	Sí	*NULL*		
precio_prov	float(8,2)	Sí	0.00		
precio_mayoreo	float(8,2)	Sí	0.00		
precio_menudeo	float(8,2)	Sí	0.00		
categoria	varchar(30)	No			✓

Productos

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
clave_prod	varchar(10)	No		✓	
clave_prov	varchar(10)	No			✓
descripcion	varchar(50)	No			✓
maximo	int(4)	Sí	0		
minimo	int(4)	Sí	0		
bodega	int(4)	Sí	0		
imagen	longblob	Sí	*NULL*		
precio_prov	float(8,2)	Sí	0.00		
precio_mayoreo	float(8,2)	Sí	0.00		
precio_menudeo	float(8,2)	Sí	0.00		
categoria	varchar(30)	No			✓
tipo_prod	varchar(15)	No			✓

Matariales Utilizados

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
clave_mat	varchar(10)	No			✓
clave_prod	varchar(10)	No			✓
cantidad	int(3)	Sí	0		

Existencias de Materiales

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
clave_mat	varchar(10)	No			✓
tipo_existencia	varchar(6)	No			✓
fecha	date	No			✓
cantidad	int(4)	No			✓

Existencias de Productos

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
clave_prod	varchar(10)	No			✓
tipo_existencia	varchar(6)	No			✓
fecha	date	No			✓
cantidad	int(4)	No			✓

Precios de Materiales

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
clave_mat	varchar(10)	No			✓
tipo_precio	varchar(10)	No			✓
fecha	date	No			✓
precio	float(8,2)	No			✓

Precios de Productos

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
clave_prod	varchar(10)	No			✓
tipo_precio	varchar(10)	No			✓
fecha	date	No			✓
precio	float(8,2)	No			✓

Catalogo de Categorias

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
tipo_categoria	varchar(10)	No			
categoria	varchar(10)	No			✓

Factura para Cliente

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
id_factura_c	int(10)	No		✓	
clave_cliente	varchar(10)	No			✓
fecha	date	No			✓
status	varchar(9)	Sí	VIGENTE		

Descripcion de la Factura del cliente

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
id_factura_c	int(10)	No			✓
clave_mat	varchar(10)	Sí	*NULL*		✓
clave_prod	varchar(10)	Sí	*NULL*		✓
cantidad	int(10)	No			
tipo	varchar(8)	No			
precio_u	float(8,2)	No			

Factura de Proveedor

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
id_factura_p	int(10)	No		✓	
clave_prov	varchar(10)	No			✓
fecha	date	No			✓
status	varchar(9)	Sí	VIGENTE		

Descripción de la Factura de Proveedor

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
id_factura_p	int(10)	No			✓
clave_mat	varchar(10)	Sí	*NULL*		✓
clave_prod	varchar(10)	Sí	*NULL*		✓
cantidad	int(10)	No			
tipo	varchar(8)	No			
precio_u	float(8,2)	No			

Remisión para Cliente

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
id_remision_c	int(10)	No		✓	
clave_cliente	varchar(10)	No			✓
fecha	date	No			✓
status	varchar(9)	Sí	VIGENTE		

Descripcion de la Remisión del cliente

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
id_remision_c	int(10)	No			✓
clave_mat	varchar(10)	Sí	*NULL*		✓
clave_prod	varchar(10)	Sí	*NULL*		✓
cantidad	int(10)	No			
tipo	varchar(8)	No			
precio_u	float(8,2)	No			

Remisión de Proveedor

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
id_remision_p	int(10)	No		✓	
clave_prov	varchar(10)	No			✓
fecha	date	No			✓
status	varchar(9)	Sí	VIGENTE		

Remisión de la Factura de Proveedor

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
id_remision_p	int(10)	No			✓
clave_mat	varchar(10)	Sí	*NULL*		✓
clave_prod	varchar(10)	Sí	*NULL*		✓
cantidad	int(10)	No			
Tipo	varchar(8)	No			
precio_u	float(8,2)	No			

Empresa

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
id_empresa	int(3)	No		✓	
nombre_emp	varchar(200)	Sí	*MI EMPRESA*		✓
calle	varchar(50)	Sí	*CONOCIDA*		
num	int(5)	Sí	*0*		
colonia	varchar(100)	Sí	*CENTRO*		
cp	int(5)	Sí	*72000*		
ciudad	varchar(150)	Sí	*PUEBLA, PUEBLA*		
telefono1	int(12)	Sí	*0*		
telefono2	int(12)	Sí	*0*		

Usuarios

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
Login	varchar(10)	No		✓	
Pass	varchar(10)	No			✓
tipo_usuario	varchar(15)	No			

Permisos

Campo	Tipo	Nulo	Predeterminado	Llave Primaria	Índice
Login	varchar(10)	No			✓
permiso	varchar(20)	No			✓

CAPITULO 4. IMPLEMENTACIÓN Y PRUEBAS.

Se implementan en el SGBD de MySQL los diseños realizados en el cápitulo anterior, se hace uso de la herramienta PHPMyAdmin, la cual nos permite gestionar todas las funciones del MySQL con la ventaja de que todo esto se realiza en una interfaz gráfica. Con esta herramienta crearemos la base de datos, la cuál se llamará ***bisuteria,*** asi como las tablas que contendrá la base de datos.

Para generar las interfaces gráficas del sistema, la interacción con MySQL y la manipulación de la información contenida en la base de datos se hace uso del NETBEANS 6.1.

4.1 Equipos para la implementación y ejecución del sistema.

4.1.1 Equipo utilizado para Implementación del Sistema para Control de Bisutería.

- PC Centrino a 2.2 GHz.
- Disco Duro de 80 GB.
- 768 MB en Memoria RAM.
- Quemador de DVD.
- Monitor de 19”.
- Sistema Operativo Windows XP Service Pack2.

4.1.2 Equipo para el servidor con ejecución del Sistema para Control de Bisuteria.

- PC Pentium4.
- Disco Duro de 40 GB a 80 GB.
- 512 MB en Memoria RAM.
- Monitor de 17” o mas Según la Preferencia del Usuario.
- Sistema Operativo Windows XP Service Pack2.
- WampServer 5.
- jdk-6u3-windows-i586-p.
- Explorer 6 o Mayor.

4.1.3 Equipo Requerido para la Terminal con Ejecucion del Sistema para Control de Bisuteria.

- PC Pentium4 o Mayor.
- Disco Duro de 40 GB a 80 GB.
- 512 MB en Memoria RAM.
- Monitor de 17” o mas Según la Preferencia del Usuario.
- Sistema Operativo Windows XP Service Pack2.
- jdk-6u3-windows-i586-p.

A continuación se muestran imágenes de la implementación de la base de datos con la herramienta WampServer.

4.2 Definición de la Base de Datos.

La siguiente imagen muetra la definición de la base de datos: ***bisuteria*** asi como algunas de las tablas que la conforman.

Tabla	Acción	Registros	Tipo	Cotejamiento	Tamaño
categorias		4	InnoDB	latin1_swedish_ci	32.0 KB
cliente		3	InnoDB	latin1_spanish_ci	64.0 KB
desc_fac_cli		2	InnoDB	latin1_swedish_ci	64.0 KB
desc_fac_pro		2	InnoDB	latin1_swedish_ci	64.0 KB
desc_remi_cli		2	InnoDB	latin1_swedish_ci	64.0 KB
desc_remi_pro		2	InnoDB	latin1_swedish_ci	64.0 KB
empresa		1	InnoDB	latin1_swedish_ci	32.0 KB
existencias_mat		16	InnoDB	latin1_swedish_ci	80.0 KB
existencias_prod		13	InnoDB	latin1_swedish_ci	80.0 KB
factura_cliente		2	InnoDB	latin1_swedish_ci	32.0 KB
factura_proveedor		1	InnoDB	latin1_swedish_ci	32.0 KB
inicio		1	InnoDB	latin1_swedish_ci	16.0 KB
material		2	InnoDB	latin1_spanish_ci	80.0 KB
materiales_utilizados		1	InnoDB	latin1_swedish_ci	48.0 KB
permisos		36	InnoDB	latin1_spanish_ci	48.0 KB
precios_mat		13	InnoDB	latin1_swedish_ci	80.0 KB
precios_prod		8	InnoDB	latin1_swedish_ci	80.0 KB
producto		2	InnoDB	latin1_swedish_ci	80.0 KB
proveedor		2	InnoDB	latin1_spanish_ci	32.0 KB
remision_cliente		1	InnoDB	latin1_swedish_ci	32.0 KB
remision_proveedor		1	InnoDB	latin1_swedish_ci	32.0 KB
usuarios		6	InnoDB	latin1_swedish_ci	16.0 KB

Figura 4.2.1

Definición de la Tabla cliente.

Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
clave_cliente	varchar(10)	latin1_spanish_ci		No			
ap	varchar(30)	latin1_spanish_ci		No			
am	varchar(30)	latin1_spanish_ci		No			
nombre	varchar(60)	latin1_spanish_ci		No			
rfc	varchar(13)	latin1_spanish_ci		Sí	*NULL*		
calle	varchar(40)	latin1_spanish_ci		Sí	*NULL*		
numero	int(5)			Sí	0		
colonia	varchar(50)	latin1_spanish_ci		Sí	*NULL*		
cp	int(5)			Sí	0		
tel_local	int(12)			Sí	0		
tel_movil	int(12)			Sí	0		
mail	varchar(50)	latin1_spanish_ci		Sí	*NULL*		

Figura 4.2.2

Definición de la Tabla proveedor.

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	clave_prov	varchar(10)	latin1_spanish_ci		No			
☐	razon_social	varchar(60)	latin1_spanish_ci		No			
☐	rfc	varchar(13)	latin1_spanish_ci		No			
☐	puesto	varchar(20)	latin1_spanish_ci		Sí	NULL		
☐	contacto	varchar(60)	latin1_spanish_ci		Sí	NULL		
☐	tel_local	int(12)			Sí	0		
☐	tel_movil	int(12)			Sí	0		
☐	calle	varchar(40)	latin1_spanish_ci		Sí	NULL		
☐	numero	int(5)			Sí	0		
☐	colonia	varchar(50)	latin1_spanish_ci		Sí	NULL		
☐	cp	int(5)			Sí	0		
☐	edo	varchar(60)	latin1_spanish_ci		Sí	NULL		
☐	pais	varchar(60)	latin1_spanish_ci		Sí	NULL		
☐	mail	varchar(50)	latin1_spanish_ci		Sí	NULL		
☐	web	varchar(50)	latin1_spanish_ci		Sí	NULL		

Figura 4.2.3

Definición de la Tabla material.

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	clave_mat	varchar(10)	latin1_spanish_ci		No			
☐	clave_prov	varchar(10)	latin1_spanish_ci		No			
☐	descripcion	varchar(50)	latin1_spanish_ci		No			
☐	maximo	int(4)			Sí	0		
☐	minimo	int(4)			Sí	0		
☐	bodega	int(4)			Sí	0		
☐	imagen	longblob		BINARY	Sí	NULL		
☐	precio_prov	float(8,2)			Sí	0.00		
☐	precio_mayoreo	float(8,2)			Sí	0.00		
☐	precio_menudeo	float(8,2)			Sí	0.00		
☐	categoria	varchar(30)	latin1_spanish_ci		No			

Figura 4.2.4

Definición de la Tabla factura_cliente.

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	id_factura_c	int(10)			No			
☐	clave_cliente	varchar(10)	latin1_spanish_ci		No			
☐	fecha	date			No			
☐	status	varchar(9)	latin1_spanish_ci		Sí	VIGENTE		

Figura 4.2.5

Definición de la tabla producto.

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	clave_prod	varchar(10)	latin1_spanish_ci		No			
☐	clave_prov	varchar(10)	latin1_spanish_ci		No			
☐	descripcion	varchar(50)	latin1_spanish_ci		No			
☐	maximo	int(4)			Sí	0		
☐	minimo	int(4)			Sí	0		
☐	bodega	int(4)			Sí	0		
☐	imagen	longblob		BINARY	Sí	*NULL*		
☐	precio_prov	float(8,2)			Sí	0.00		
☐	precio_mayoreo	float(8,2)			Sí	0.00		
☐	precio_menudeo	float(8,2)			Sí	0.00		
☐	categoria	varchar(30)	latin1_spanish_ci		No			
☐	tipo_prod	varchar(15)	latin1_spanish_ci		No			

Figura 4.2.6

Definición de la tabla desc_fac_cli.

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	id_factura_c	int(10)			No			
☐	clave_mat	varchar(10)	latin1_spanish_ci		Sí	*NULL*		
☐	clave_prod	varchar(10)	latin1_spanish_ci		Sí	*NULL*		
☐	cantidad	int(10)			No			
☐	tipo	varchar(8)	latin1_spanish_ci		No			
☐	precio_u	float(8,2)			No			

Figura 4.2.7

Definición de la tabla factura_proveedor.

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	id_factura_p	int(10)			No			
☐	clave_prov	varchar(10)	latin1_spanish_ci		No			
☐	fecha	date			No			
☐	status	varchar(9)	latin1_spanish_ci		Sí	VIGENTE		

Figura 4.2.8

Definición de la Tabla desc_fac_pro.

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	id_factura_p	int(10)			No			
☐	clave_mat	varchar(10)	latin1_spanish_ci		Sí	*NULL*		
☐	clave_prod	varchar(10)	latin1_spanish_ci		Sí	*NULL*		
☐	cantidad	int(10)			No			
☐	tipo	varchar(8)	latin1_spanish_ci		No			
☐	precio_u	float(8,2)			No			

Figura 4.2.9

4.3 Desarrollo de Interfaces.

A continuación se describirá funcionamiento de algunas ventanas mas importantes del sistema para su mayor comprensión.

Para accesar al sistema el usuario ingresa su nombre de usuario y contraseña para ser validado por el sistema. Esta validación determina según los datos ingresados por el usuario si este es administrador o un usuario limitado para así darle los derechos correspondientes sobre el sistema, ya que este es correctamente validado aparecerá la pantalla principal (Fig. 4.3.1).

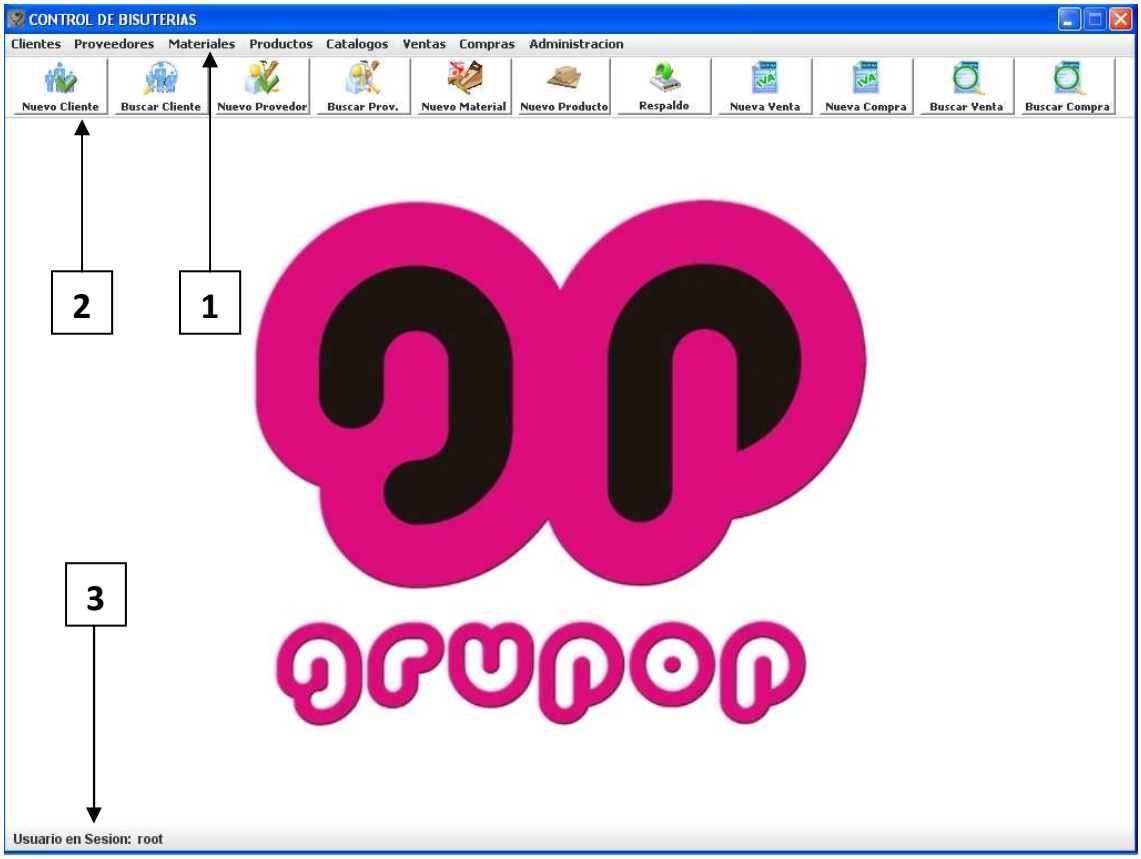

Figura 4.3.1

1.- Barra de Menús.
En cada menú y submenú contenido en esta barra se encuentran las funciones que el sistema realizará, como son:

- Clientes.- El usuario podrá ingresar la información necesaria de un cliente para almacenarla en la base de datos, también podrá realizar una búsqueda del mismo para que la información sea visualizada, eliminada o modificada de la base de datos.
- Proveedores.- El usuario podrá ingresar la información necesaria de un proveedor para almacenarla en la base de datos, también podrá realizar una búsqueda del mismo para que la información sea visualizada, eliminada o modificada de la base de datos.

- Materiales.- Se podrá ingresar la información necesaria para almacenar materiales en la base de datos, también podrá realizar una búsqueda del material para que esta sea visualizada, eliminada o modificada de la base de datos.
- Productos.- Se podrá ingresar la información necesaria para almacenar productos, asi como los materiales que lo componen, también podrá realizar una búsqueda del producto para que esta sea visualizada, eliminada o modificada de la base de datos.
- Catálogos.- Se visualizaran las imágenes, precios (mayoreo y menudeo) y descripción del producto o material elegido.
- Ventas.- Se cargarán ventas a los clientes registrados (factura o remisión) para su almacenamiento en la base de datos e impresión de las mismas, también podrán ser buscadas para visualizarlas o ser canceladas.
- Compras.- Se cargarán compras a los proveedores registrados (factura o remisión) para su almacenamiento en la base de datos e impresión de las mismas, también podrán ser buscadas para visualizarlas o ser canceladas.
- Administración.- Aquí se tiene las opciones de ingresar nuevas categorías para los productos o materiales, generar reportes, modificar datos de su empresa, cambiar el color de la apariencia del sistema y realizar respaldos de la información contenida en la base de datos.

2.- Barra de Accesos Directos.
Estos botones accesan directamente a la función que describe cada uno, en esta barra están contenidas las funciones más comunes y utilizadas por el usuario como lo son: nuevo cliente, buscar cliente, nuevo proveedor, buscar proveedor, nuevo material, nuevo producto, respaldo de información, nueva venta, nueva compra, buscar venta y buscar compra.
3.- Barra de Sesión.
Esta Barra nos muestra que usuario se encuentra en sesion.

Menú Clientes.
En este aparatado se realizan las operaciones de inserción, búsqueda, modificación y eliminación de los datos de un cliente en particular. En todos los procedimientos anteriores el sistema pide una confirmación del usuario para realizar cada acción.

a) Nuevo Cliente (Fig. 4.3.2)

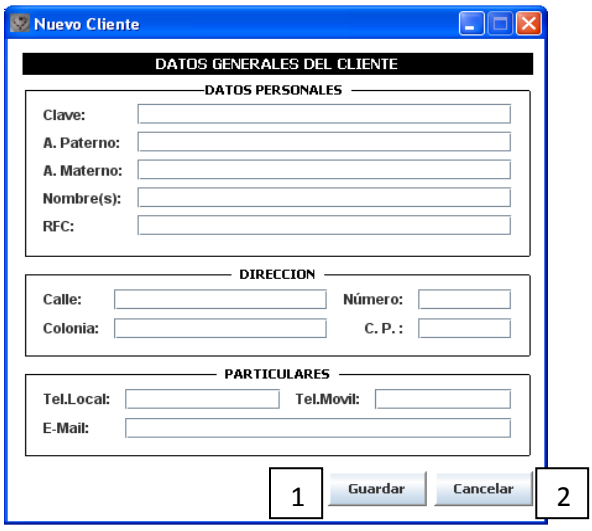

Figura 4.3.2

1.- Guarda en la base de datos la informacion del cliente previamente ingresada en los cuadros de texto por el usuario.
2.- Sale del formulario de Nuevo Cliente y nos manda a la pantalla principal.

b) Buscar Cliente (Fig. 4.3.3)
En este apartado se muestran los 2 tipos de consulta que se pueden ejecutar en el módulo, cada una de estas consultas muestran la información del cliente de forma detallada.

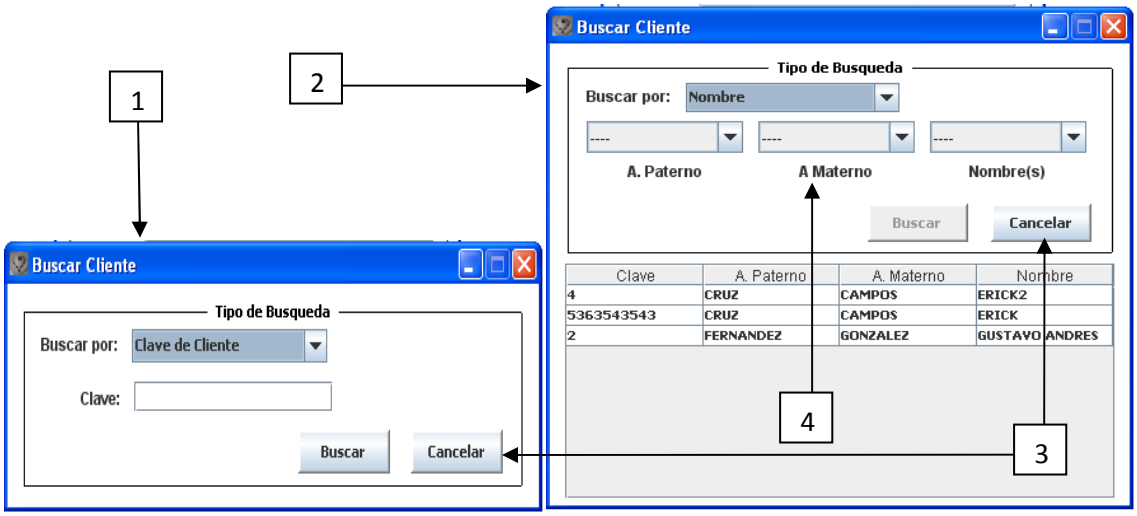

Figura 4.3.3

La busqueda del cliente puede ser realizada de dos formas: clave del cliente (1) o por nombre (2) si la eleccion del usuario es por clave solo vasta ingresar en el cuadro de texto la clave para que ña informacion sea visualizada, si el cliente elije la busqueda por nombre nos apareceran 3 lista las cuales almacenaran los apellidos paternos, maternos y nombre de los clientes existentes(4), el usuario elijira el apellido paterno , meterno o nombre del cliente y el la tabla inferior (3) se iran dezplegando los clientes que conciden con la eleccion de las listas dezplegables, al dar click con el mouse sobre algun cliente se visualizara la información del cliente seleccionado.

c) Modificar Cliente (Fig. 4.3.4)
Despues de realizar la búsqueda del cliente (Ver detalles Fig. 4.3.3), nos manda al formulario donde se visualizan los datos generales del cliente el cual nos dará la opción de modificar y guardar los datos actualizados del cliente con previa confirmación del usuario.

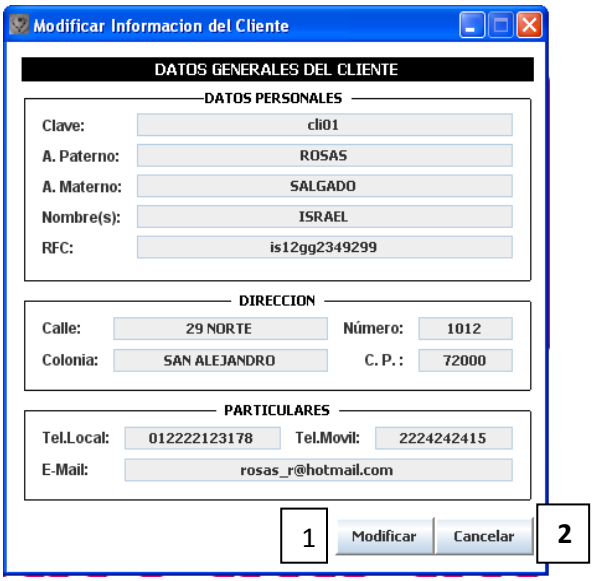

Figura 4.3.4

1.- Modifica en la base de datos la informacion del cliente previamente ingresada en los cuadros de texto por el usuario.
2.- Sale del formulario de modificación del cliente y nos manda a la pantalla principal.

d) Eliminar Cliente (Fig. 4.3.5)
Despues de realizar la búsqueda del cliente (Ver detalles Fig. 4.3.3), nos manda al formulario donde se visualizan los datos generales del cliente el cual nos dará la opción de eliminar los datos del cliente con previa confirmación del usuario.

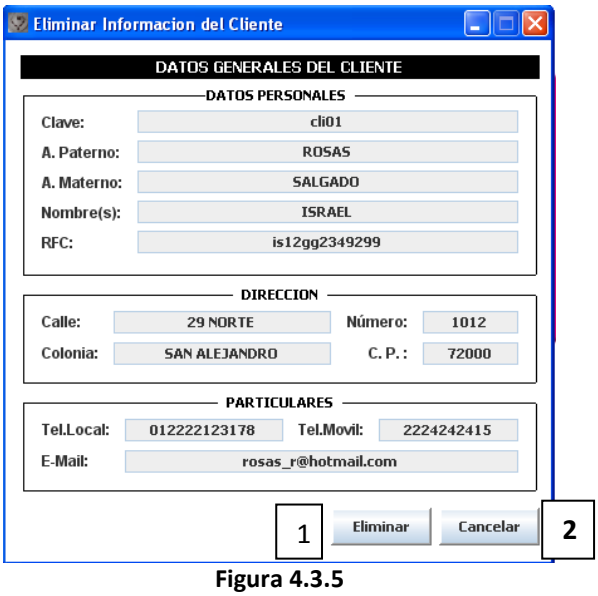

Figura 4.3.5

1.- Elimina la informacion del cliente de la base de datos.
2.- Sale del formulario de eliminación del cliente y nos manda a la pantalla principal.

Menú Productos.
En este apartado se realizan las operaciones de inserción, búsqueda, modificación y eliminación de los datos de un producto en particular. En todos los procedimientos anteriores el sistema pide una confirmación del usuario para realizar cada acción.

a) Nuevo producto (Fig.4.3.6)

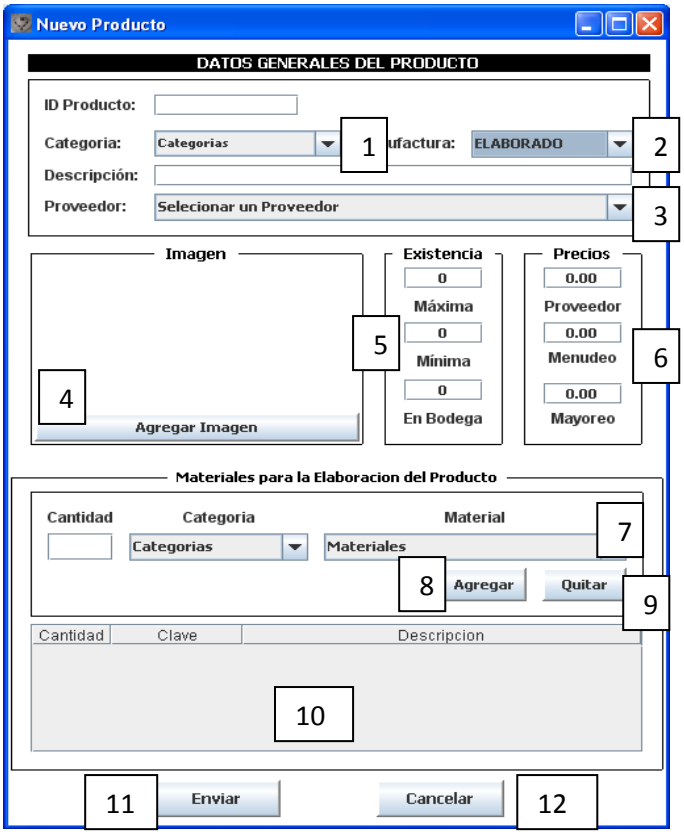

Figura 4.3.6

1.- Despliega las categorías que pertenecen a los productos, previamente almacenadas.
2.- Muestra los tipos de producto (ELABORADO Y COMPRADO), si el producto es elaborado se tienen que insertar los materiales que se necesitan para elaborarlo, si el producto es comprado sólo se llena la información particular del producto.
3.- Despliega una lista la cual contiene los proveedores registrados en la base de datos, aquí el usuario elegirá el proveedor que le distribuye ese producto.
4.- El usuario ingresara la imagen del producto, al presionar el botón (Agregar Imagen) aparecerá un cuadro de dialogo para que el usuario escoja la imagen de su producto.
5.- El usuario ingresará la cantidad de producto para cada existencia (minima, maxima y bodega).

6.- El usuario ingresará el precio del producto para cada tipo de precio (mayoreo, menudeo y proveedor).
7.- En esta área el usuario eligira que materia(es) asi como la cantidad que se necesita para la elaboracion del producto, si el usuario elige uqe el producto es comprado esta area no sera visible.
8.- Agrega a la tabla los datos del material que lo compone.
9.- Elimina de la tabla el material seleccionado.
10.- Visualiza la cantidad, descripción y clave de los materiales utilizados para elaborar el producto.
11.- Guarda el producto en la base de datos con previa confirmación del usuario.
12. Cierra el formulario de nuevo Producto.

b) Buscar Producto (Fig. 4.3.7)
El Usuario puede consultar la información de un producto en especifico seleccionando el tipo de búsqueda y el producto a consultar y al dar click sobre el mandará la información completa del producto seleccionado.

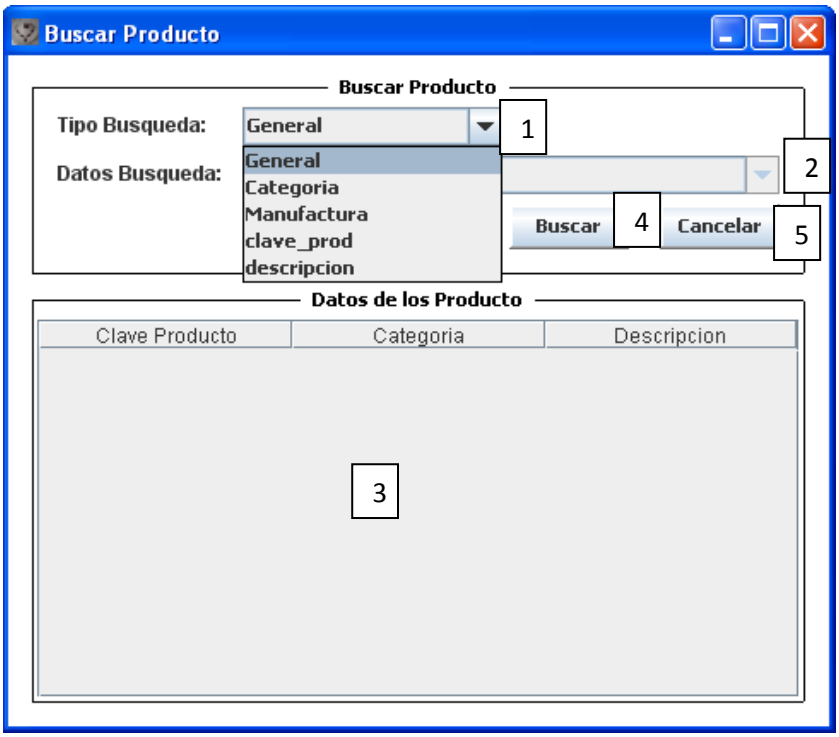

Figura 4.3.7

1.- El usuario podra elegir el tipo de consulta que desea realizar, en las que se encuentran: busqueda general, por categoria, por manufactura, por clave del producto o por descripcion del producto, si la busqueda es general se visualisan los datos de todos los productos existentes en la base de datos.
2.- Esta lista contiene la información del producto previamente solicitada en el tipo de búsqueda (1) , p/e: si es elegida la búsqueda por clave en esta lista se desplegarán todas las claves de los prodcutos existentes.

3.- Se mostrarán los datos del producto seleccionado (clave, categoria y descripcion).
4.- Envia a la tabla los productos correspondientes a la búsqueda seleccionada.
5.- Cierra el formulario de búsqueda.

c) Modificar Producto (Fig. 4.3.8)
Permite al usuario modificar la información del producto seleccionado a excepción de la clave y existencia en bodega en caso de un producto comprado ya que ésta sólo será modificada desde las compras a proveedor.

Figura 4.3.8

1.- Guarda la información actualizada del producto con previa confirmación.
2.- Cierra el formulario.

e) Ingresar Productos Elaborados (Fig.4.3. 9)
Permite al usuario ingresar productos elaborados a su existencia en bodega ya que éstos no se ingresan mediante una compra porque sólo necesitan el material para elaborarlos.

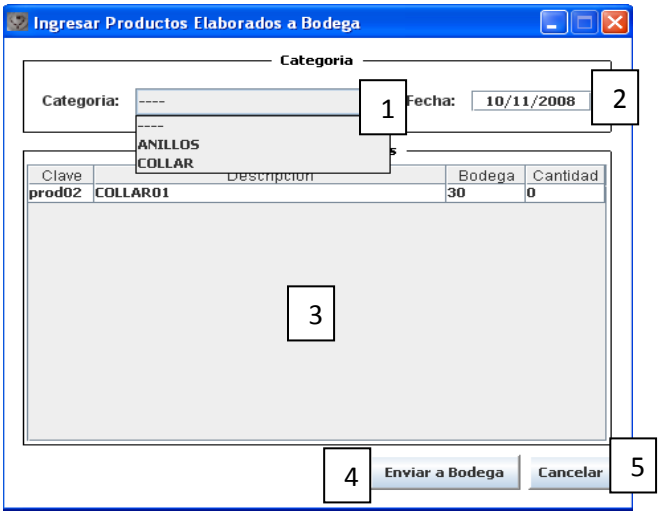

Figura 4.3.9

1.- Despliega las categorías que pertenecen a los productos, previamente almacenadas.
2.- Fecha de inserción de productos elaborados, se deja abierta para que el usuario inserte la fecha que el desee.
3.- Muestra los productos elaborados pertenecientes a la categoría seleccionada, mostrando la clave , descripción, bodega y cantidad la cual es editable para ingresar más productos a bodega.
4.- Actualiza la existencia en bodega de cada producto donde se ingresó una cantidad diferente de cero.
5.- Cierra el formulario.

Ventas.

a) Nueva Venta (Fig. 4.3.10)
El usuario podrá cargar su venta al cliente, teniendo 2 opciones factura o remisión, si elige la factura esta se imprimirá en el formato de factura de su negocio y si la opción es remisión esta se imprimirá en un formato ya predeterminado en el sistema, eligiendo cualquiera de las opciones ésta se guardará dentro de la base de datos para futuras consultas y reimpresiones.

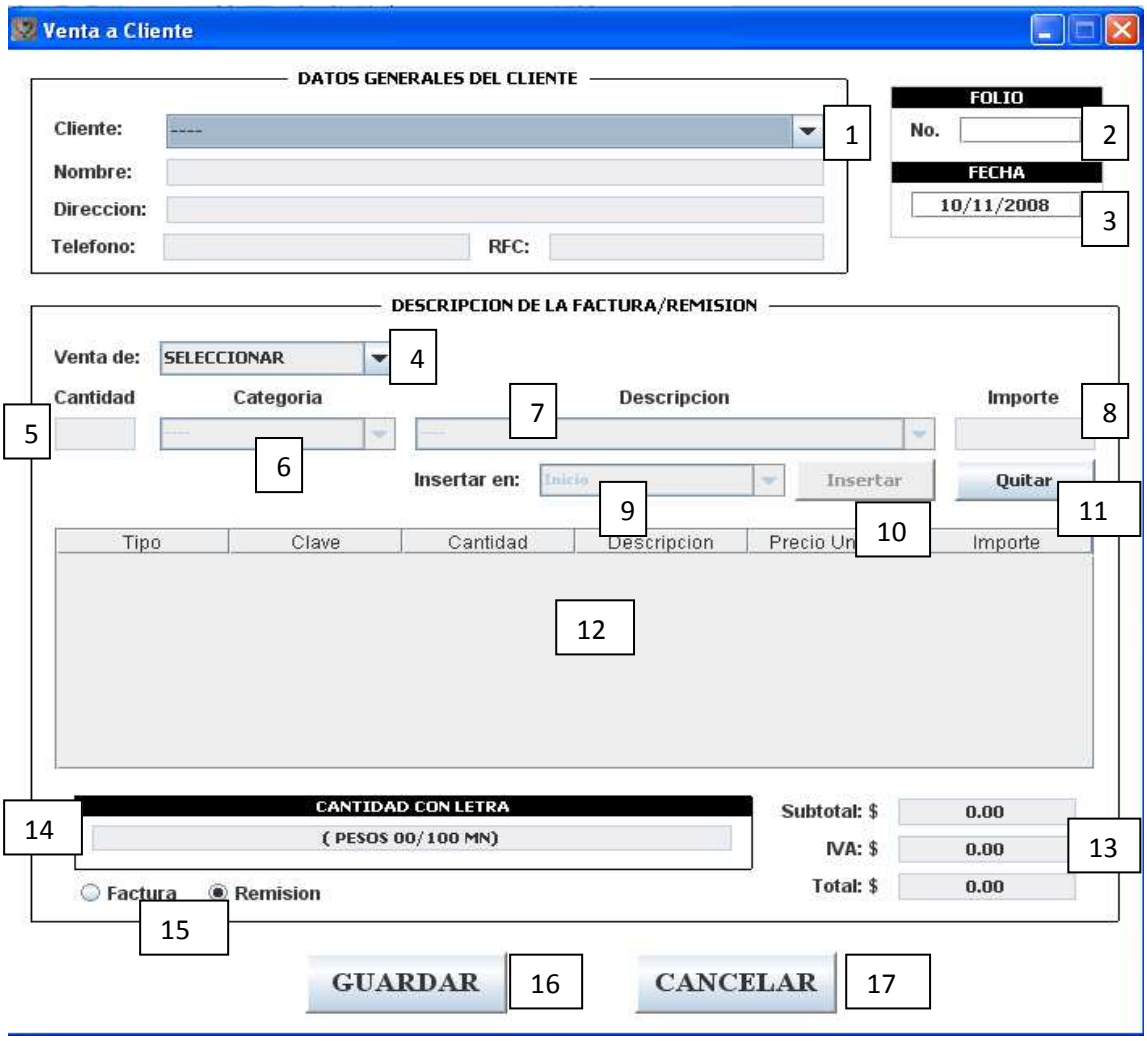

Figura 4.3.10

1.- Despliega una lista con los clientes existentes en la base de datos, del cual será seleccionado uno para que se le cargue la venta.
2.- Se inserta el número de folio de la factura o remisión.
3.- Fecha en que se realiza la venta, por default aparece la fecha del sistema pero esta es editable para ingresar la fecha que el usuario desee.
4.- El usuario seleccionará si el articulo a ingresar a la venta es un MATERIAL o PRODUCTO.
5.- Cantidad a vender del articulo.
6.- Selección de categorías de material o producto.
7.- El usuario selecciona que material o producto va a ingresar a la tabla mediante su descripción.
8.- Al elegir el material o producto el importe se inserta automáticamente, si la cantidad es mayor a 5 muestra el precio de mayoreo de lo contrario muestra el precio de menudeo.
9.- Permite al usuario elegir en que posición de la tabla desea insertar el artículo (inicio, final o después de algún otro artículo).

10.- Inserta el artículo en la tabla.
11.- Elimina el artículo seleccionado de la tabla.
12.- Visualiza todos los artículos que serán vendidos (tipo, clave, cantidad, descripción, precio unitario, importe).
13.- Visualiza el subtotal, iva y total de la venta.
14.- Muestra la cantidad con letra del total de la venta.
15.- Tipo de venta (factura o remisión).
16.- Guarda la venta y muestra el cuadro de diálogo para la impresión.
17.- Cierra el formulario.

b) Buscar Venta (Fig. 4.3.11)

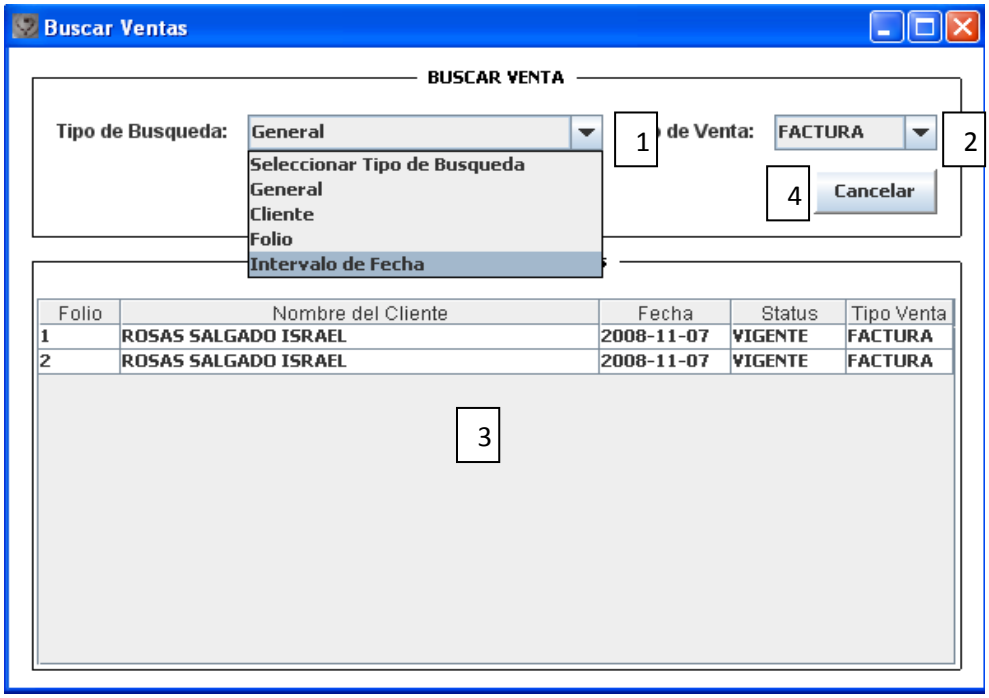

Figura 4.3.11

1.- Despliega los tipos de búsqueda para consultar una venta:

- Si elige General: Lista todas las ventas existentes en la base de datos dependiendo el tipo de veta seleccionada (2).
- Si elige Cliente: Lista las ventas que tiene el cliente seleccionado.
- Folio: Muestra la venta con el folio seleccionado.
- Intervalo de fecha: Lista todas las ventas que se encuentran dentro de ese rango de fecha.

2.- Lista los tipos de venta a consultar (FACTURA o REMISIÓN).
3.- Visualiza las ventas contenidas según el tipo de búsqueda (folio, nombre del cliente, fecha, status, tipo de venta) al dar click sobre algún artículo de la tabla muestra la venta completa.
4.- Cierra el formulario.

c) Cancelar Venta (Fig. 4.3.12)
Permite al usuario cancelar una venta o una remisión.

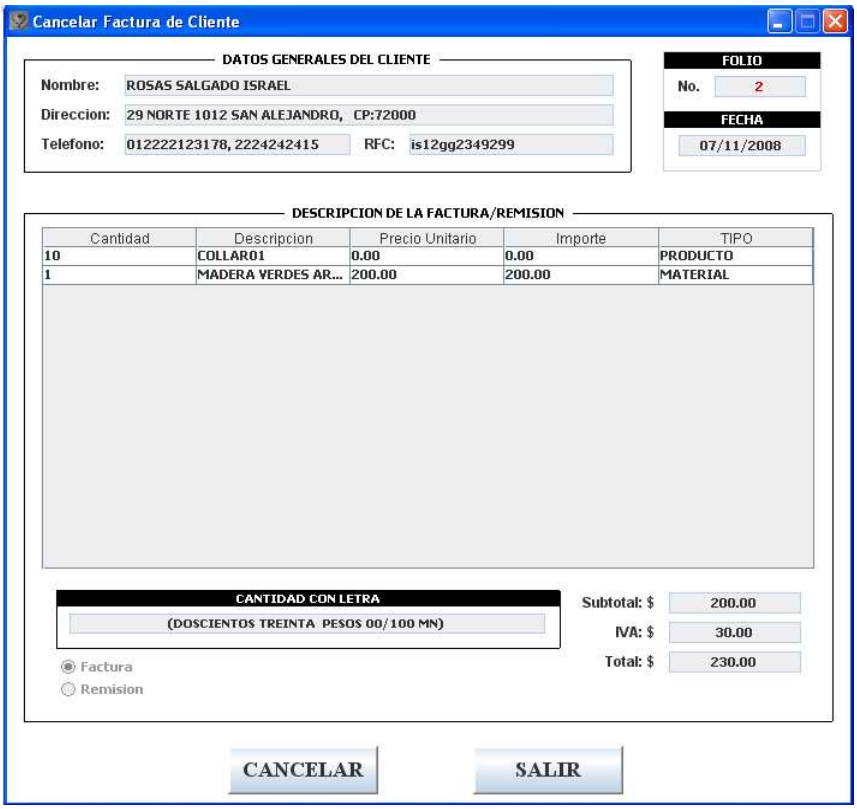

Figura 4.3.12

Administración.

Usuarios (Fig.4.3.13)
Permite al usuario ingresar nuevos usuarios a su sistema así como elegir que permisos tendrán sobre el mismo.

Figura 4.3.13

1.- El usuario ingresa los datos necesarios para identificar al usuario y porteriormente para accesar al sistema (usuario y clave).
2.- Enlista los tipos de usuario que puede haber en el sistema (si es administrador podra tener acceso a todas la operaciones sobre el sistema: ingresar, busca, modificar y eliminar informacion, si el tipo es limitado solo podra ingresar y buscar informacion).
3.- El usuario elige a que funciones tendra acceso el nuevo usuario.

Reportes.
El usuario podrá generar reportes para materiales y productos acerca de:

- El más vendido y el menos vendido (Fig.4.3. 14).

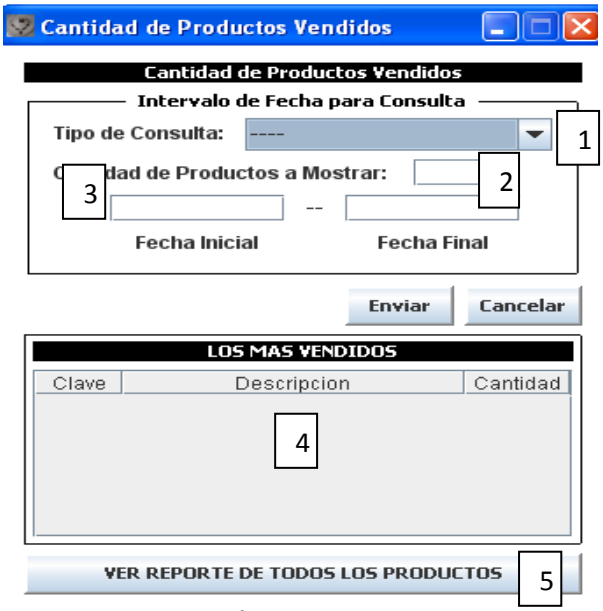

Figura 4.3.14

1.- El usuario elegirá que tipo de reporte desea que se muestre(EL MAS VENDIDO, EL MENOS VENDIDO).
2.- Cantidad de productos o materiales a mostrar.
3.- Intervalo de fecha a Consultar.
4.- Visualiza el o los productos según la cantidad ingresada, el tipo de consulta y el intervalo de fecha ingresado.
5.- Muestra el reporte con la cantidad vendida de los productos según el tipo de consulta seleccionado.

- Mejor y Peor Precio(Fig. 4.3.15).

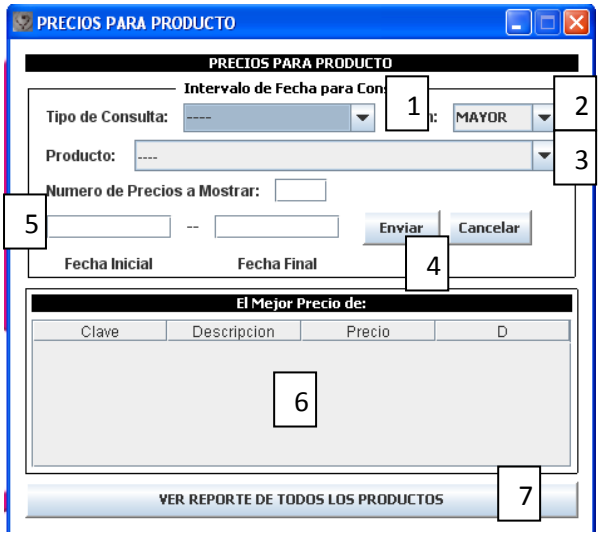

Figura 4.3.15

1.- Lista los tipos de precios que contienen los materiales o productos (menudeo, mayoreo, proveedor).
2.- Orden en que se presentarán los datos (mayor, menor).
3.- Lista los productos existentes en la bases de datos para saber cuál fue su mayor o menor precio.
4.- Cantidad de precios a mostrar.
5.- Intervalo de fecha en el que se tienen que encontrar los precios.
6.- Visualiza los precios según el tipo de consulta y el orden seleccionado.
7.- Muestra el reporte con todos los precios de cada producto en el intervalo de fecha, tipo de consulta y orden seleccionado.

- Ingresos, Egresos y Balance General (Fig. 4.3.16).

Figura 4.3.16

1.- Visualiza el reporte con los ingresos, egresos o balance general que fueron generados el día de hoy.
2.- Visualiza el reporte con los ingresos, egresos o balance general que fueron generados el día de ayer.

3.- Visualiza el reporte con los ingresos, egresos o balance general que fueron generados del de hoy un mes hacia atrás.
4.- Visualiza el reporte con los ingresos, egresos o balance general que fueron generados dentro del intervalo de fecha.

4.4 Pruebas del Sistema.

En este proceso se trata de encontrar errores que son el resultado de procedimientos no previstos y de verificar el correcto cumplimiento de los requerimientos iniciales. En este caso mostraremos pruebas de funcionamiento de algunas ventanas del sistema.

1.- Nuevo Cliente.
Vayamos a la prueba de inserción de un cliente, tenemos el caso donde el usuario ingresa una clave ya almacenada previamente con otro cliente, el sistema lanza este mensaje de erro diciéndole al usuario que esta clave ya existe figura 4.4.1, si el cliente no ingresa ninguna clave el sistema manda el mensaje diciendo al cliente que ingrese una clave figura 4.4.2.

Figura 4.4.1

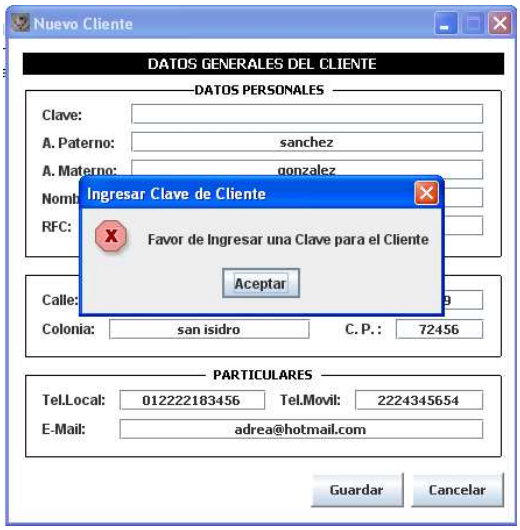

Figura 4.4.2

Otro caso es cuando el usuario olvida ingresar el nombre del cliente (apellido paterno, apellido materno, nombre) y el rfc lanza el siguiente mensaje para pedirle al cliente que los ingrese ya que estos datos son necesarios cuando el usuario levanta una venta figura 4.4.3

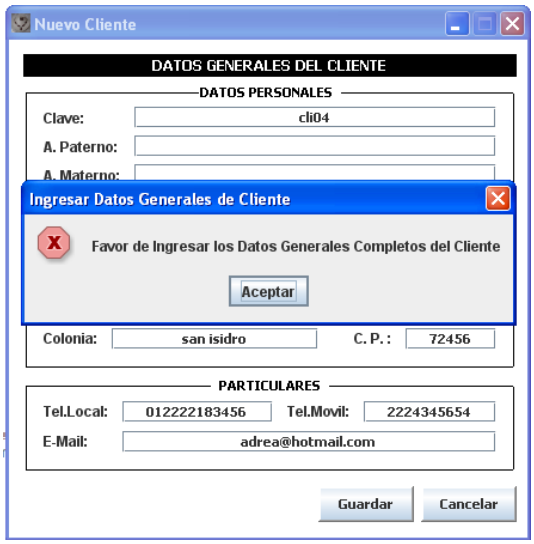

Figura 4.4.3

2.- Nuevo Material.

Otro caso mas evaluado en las pruebas del sistema es cuando el usuario ingresa al sistema un nuevo material, se evaluan el caso cuando el cliente olvida ingresar la clave del material, el sistema envia un mensaje pidiendo al usuario que ingrese la clave como se muestra en la figura 4.4.4, lo mismo sucede cuando el usuario olvida seleccionar una categoria, un proveedor e ingresar la descripcion del material.

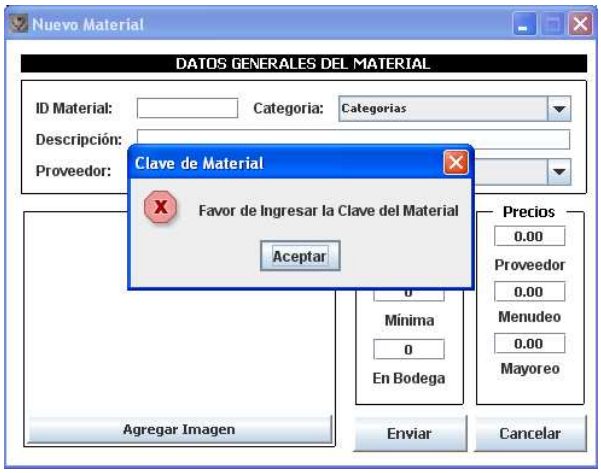

Figura 4.4.4

Un caso mas en la insercion de nuevos materiales es cuando el usuario ingresa una clave que ya ha sido insertado previamente en otro material, cuando esto sucede el sistema envia el siguiente mensaje pidiendo al usuario que ingrese otra clave como se muestra en la figura 4.4.5, cuando el usuario ingresa una descripcion del material existente en la base de datos el sistema lanza un mensaje al usuario para que inserte otra descripcion que no sea repetida como lo muestra la figura 4.4.6.

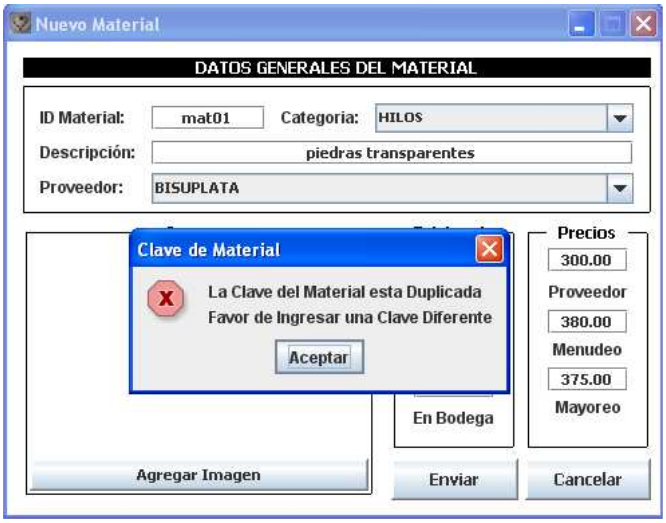

Figura 4.4.5

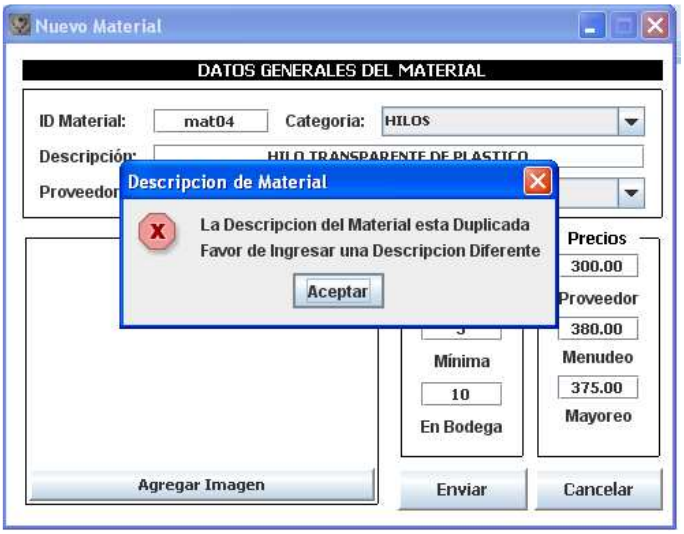

Figura 4.4.6

3.- Nueva Venta.

Un caso mas enlas prubas del sistema es la generacion de una venta, cuando el usuario presiona el boton guardar y la factura no tiene un cliente seleccionado el sistema lanza un mesaje avisando que debe seleccionar un cliente como se muestra en la figura 4.4.7, lo mismo cuando el usuario olvida ingresar el folio de la venta figura 4.4.8 y cuando no ingresa una fecha figura 4.4.8.

Figura 4.4.7

Figura 4.4.8

Figura 4.4.9

Otro caso es cuando el usuario del sistema ingresa un numero de folio duplicado ya sea de una remision o de una factura, cuando esto sucede el sitema envia un mensaje informando al usuario la duplicidad del folio como se muestra en la figura 4.4.10 y 4.4.11.

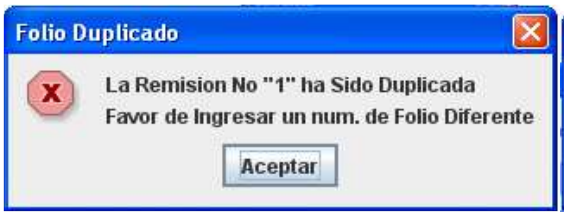

Figura 4.4.10

Figura 4.4.11

CAPITULO 5. CONCLUSIONES.

- Podemos concluir que gracias a las metodologías aplicadas en el desarrollo de este sistema se genera una aplicación confiable, segura y funcional para los usuarios finales, asi como brindarles una interfaz agradable para su uso y principalmente realizar un sistema que sea utilizado al 100% .

- Con este sistema se pretende que el negocio de la bisutería mantenga un control más estricto en sus inventarios, realizar ventas de forma correcta y eficaz así como invertir menos tiempo en el desarrollo de éstas.

- Este sistema ofrecerá una mejor atención al cliente en cuestión de tiempo y trato personal cliente-vendedor ya que habrá mayor interacción entre ellos, otro aspecto es que el cliente perderá menos tiempo al levantar su pedido.

- El sistema permitirá obtener más ganancias y además evitar pérdidas de mercancía, gastarán menos papelería, venderán más y en menos tiempo, y se llevará un mayor control en sus pagos tanto a sus proveedores como en los pagos de sus clientes.

- El sistema puede ser modificado para funcionar en red local y éste pueda ser manipulado por varios equipos a la vez haciendo que el negocio sea todavía más productivo en el aspecto tiempo.

BIBLIOGRAFIA

1.- [stephen2000].

- Stephen R. Schach "Ingeniería de Software Clasica y Orientada a Objetos", McGraw Hill, 2006.

2.- [benet 2000].

- Benet Campderrich Falgueras (Ed. Uoc)
  ISBN: 8484297934. ISBN-13: 9788484297932

3.- [pressman 2002].

- Pressman. S. Roger "Ingeniería de Software. Un enfoque práctico", McGraw Hill, 2002.

4.- [korth2002].

- F. Korth Henry , Silberschatz Abraham, Sudarshan S. "Fundamentos de Bases de Datos", McGraw Hill, 2002.

5.- [elmasri 2000].

- Elmasri Y. Navathe. "Fundamentos de Sistemas de Bases de Datos", Adisson Wesley, 2000.

6.- [pons2000].

- Olga Pons Capote; Nicolás Marín Ruiz; Juan Miguel Medina Rodríguez; Silvia Acid Carrillo, "Introducción a las Bases de Datos. El Modelo Relacional" ,2000.

7.- [perez 2008].

- Pérez César. "MySQL para Windows y Linux", Alfaomega Ra-MA, 2008.

8.- [ceballos2006].

- JAVA 2: LENGUAJE Y APLICACIONES, CEBALLOS, F.J. (Editorial Ra-ma).

9.- [sanchez2004]

- Sanchez Jorge, "MySQL Guia Rapida", 2004.

10.- [wikipedia2008].

- WIKIPEDIA, *NetBeans*,2008, http://es.wikipedia.org/wiki/NetBeans_IDE, [Consulta: 20 de Agosto de 2008]

11.- [microsoft2008].

- MICROSOFT, *Visio 2003: Información de Producto*, ©2008 Microsoft Corporation. Todos los derechos reservados, http://www.microsoft.com/latam/office/visio/prodinfo/default.mspx,[Consulta: 23 de Agosto de 2008]