PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
SEDE AMBATO

ESCUELA DE INGENIERIA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TITULO DE INGENIERIA DE SISTEMAS

"SISTEMA PARA EL DISEÑO DE MUEBLES MODULARES Y LA OBTENCIÓN DE PRESUPUESTO DEL MISMO"

AUTOR:
GIOCONDA VANESSA LARA PERALTA

DIRECTOR DE LA DISERTACIÓN
ING. WIGBERTO SANCHEZ

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
SEDE AMBATO
BIBLIOTECA

AMBATO, 2006

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
SEDE AMBATO
SECRETARIA
ESCUELA DE INGENIERIA
DE SISTEMAS
12-I-07
17130

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
SEDE AMBATO
BIBLIOTECA

DEDICATORIA

Cada una de las letras de este proyecto
va dedicado a las personas más importantes
de mi vida que son mis padres, mi hija, mi esposo,
mis hermanos y amigos.

AGRADECIMIENTO

A Dios por darme sabiduría y fortaleza, a mis padres, y en especial a mi madre, mi hermano Xavier, mi esposo y a mi hija por el apoyo brindado. A mi director de disertación Ing. Wigberto Sánchez por los conocimientos brindados.

INTRODUCCIÓN

Dentro de un mundo globalizado, la tecnología avanza a un ritmo acelerado, la misma que ofrece servicios, métodos, herramientas y equipos modernos para facilitar el desarrollo de actividades personales y empresariales. En el campo informático existen muchas herramientas con las cuales se automatizan actividades y procesos manuales, teniendo importancia dentro de la organización de las empresas, en donde ocupan un lugar privilegiado, además de que han aportado enormemente al progreso, mejoramiento y eficiencia de las actividades que en esta se efectúen.

Actualmente en el mercado se encuentran las más variadas herramientas para desarrollar sistemas informáticos manejables y confiables, capaces de realizar cualquier trabajo, migrar a cualquier plataforma utilizada, adaptarse a cualquier tipo de hardware, compartir información con otros sistemas, etc., por lo que se facilita el uso de la información existente, la misma que de acuerdo a las necesidades del usuario será identificada, procesada y evaluada en su presentación final. En tal virtud se ha investigado ampliamente la forma como se realiza manualmente los procesos, fuentes de información, almacenes de datos, procesos intermedios, captura y evaluación de la información existente, visualización de resultados.

CAPITULO I

1. PROYECTO DE INVESTIGACIÓN

1.1. PLANTEAMIENTO DEL PROBLEMA

1.1.1. PROBLEMA

La carencia de un sistema informático que proporcione al cliente en forma rápida y precisa la obtención de un presupuesto y adicionalmente el diseño de un mueble modular.

1.1.2. PROBLEMATIZACIÓN

- Limitaciones técnicas que no permitan que el cliente tenga opciones a cambios del diseño de su mueble.
- Problema a los empleados en el momento de la elaboración del diseño del mueble modular para el cliente, sin poder tener un costo de dicho mueble en el instante mismo que se lo esta diseñando.
- Dificultad de obtener en forma rápida, exacta y oportuna el presupuesto de un mueble modular a partir del diseño solicitado por el cliente.

1.2. DELIMITACIÓN

El estudio y la elaboración del sistema se realizarán en la empresa CRUMAD de la ciudad de Ambato. La solución del problema estará en desarrollar un sistema informático que trabaje bajo la plataforma Windows 2000, con un front end realizado en Power Builder, herramienta de cuarta generación; Sysbase 11.5 como base de datos y Autocad 2000 como herramienta de integración y visualización del diseño de un mueble modular.

El Sistema se orienta a la elaboración de presupuestos y diseño únicamente para muebles modulares de centros comerciales y estaciones de trabajo.

1.3. JUSTIFICACIÓN E IMPORTANCIA

En tal virtud se ha encontrado que dentro de la empresa CRUMAD de la ciudad de Ambato, se viene desarrollando un proceso manual, el mismo que tiene la necesidad de automatizarse, para que la atención se efectúe más rápidamente, obteniéndose de esta manera resultados oportunos, confiables y veraces.

Este proyecto es importante ya que facilitará al dueño del negocio la presentación de sus proyectos, proformas y presupuestos, esto aumentará la calidad de servicio al cliente.

El impacto es de tipo científico – técnico ya que se generará un documento válido para consultas de docentes y estudiantes.

Por lo tanto el tema de la tesis es: *"Sistema para el diseño de muebles modulares y la obtención del presupuesto del mismo"*.

1.4. OBJETIVOS

1.4.1. OBJETIVO GENERAL

- Desarrollar un sistema automatizado que facilite la obtención de presupuestos a partir del diseño de un mueble modular.

1.4.2. OBJETIVOS ESPECÍFICOS

- Elaborar un módulo donde se permita al cliente observar el mueble modular terminado el costo del mismo.
- Adecuar el sistema para que permita que el cliente pueda seleccionar a su gusto y preferencia cualquier tipo de cambio que necesite realizar a su mueble y por consiguiente la variación de su presupuesto.
- Implantar un sistema informático que facilite él cálculo de presupuestos de un mueble modular y que colabore en la obtención rápida y precisa de la información.

1.5. HIPÓTESIS

El sistema automatizado simplificará el trabajo del usuario del sistema además dándole al cliente la facilidad de obtener su presupuesto y diseño del mueble modular que necesite en forma inmediata y precisa.

1.6. ASPECTOS METODOLÓGICOS

1.6.1. FUNDAMENTOS TEÓRICOS

El proyecto se desarrollará a partir de tres modelos: empirista, racionalista y pragmatista.

Empirista ya que el proyecto se desarrollará en función de la práctica y el conocimiento adquirido a lo largo de todo el proyecto.

Racionalista ya que todo lo que se piensa se llevará a la práctica en el desarrollo del proyecto.

Pragmatista porque el resultado final será el presupuesto generado por el sistema partiendo del diseño del mueble modular.

1.6.2. MÉTODOS DE INVESTIGACIÓN

El proceso que se utilizará en este proyecto será el método científico para establecer la base de la investigación.

De igual manera, la investigación será experimental ya que se realizará estudios y pruebas en la utilización del software.

El nivel de investigación:

- Realizar un estudio de los modelos de muebles modulares así como sus costos por metro cuadrado o lineal dependiendo del caso.
- Descripción de los componentes software y hardware.
- Definir las tareas a realizar.

CAPITULO II

2. MARCO TEÓRICO

2.1 SISTEMA

2.1.1 CONCEPTO DE SISTEMA

El concepto de sistema en general está sustentado sobre el hecho de que ningún sistema puede existir aislado completamente y siempre tendrá factores externos que lo rodean y pueden afectarlo, por lo tanto podemos referir a Muir citado en Puleo (1985) que dijo: "Cuando tratamos de tomar algo, siempre lo encontramos unido a algo más en el Universo".

Puleo define sistema como " un conjunto de entidades caracterizadas por ciertos atributos, que tienen relaciones entre sí y están localizadas en un cierto ambiente, de acuerdo con un cierto objetivo".

- *Una Entidad* es lo que constituye la esencia de algo y por lo tanto es un concepto básico. Las entidades pueden tener una existencia concreta, si sus atributos pueden percibirse por los sentidos y por lo tanto son medibles y una existencia abstracta si sus atributos están relacionados con cualidades inherentes o propiedades de un concepto.

- *Los Atributos* determinan las propiedades de una entidad al distinguirlas por la característica de estar presentes en una forma cuantitativa o cualitativa.

  Los atributos cuantitativos tienen dos percepciones: La dimensión y la magnitud. La dimensión es una percepción que no cambia y que identifica al atributo, para lo cual se utilizan sistemas de medida basado en unidades o patrones, tales como el CGS, MKS, etc.; ejemplos de dimensión son Kg., tamaño, sexo, color, etc. La magnitud es la percepción que varía y que determina la intensidad del atributo en un instante dado de tiempo, para lo cual

se utilizan escalas de medida, tales como: la nominal, la ordinal, la de intervalo y la de razón, ejemplos de magnitud son: 30 Kg., 20 empleados, etc.

- *Las Relaciones* determinan la asociación natural entre dos o más entidades o entre sus atributos. Estas relaciones pueden ser estructurales, si tratan con la organización, configuración, estado o propiedades de elementos, partes o constituyentes de una entidad y son funcionales, si tratan con la acción propia o natural mediante la cual se le puede asignar a una entidad una actividad en base a un cierto objetivo o propósito, de acuerdo con sus aspectos formales (normas y procedimientos) y modales (criterios y evaluaciones).

- *El Ambiente* es el conjunto de todas aquellas entidades, que al determinarse un cambio en sus atributos o relaciones pueden modificar el sistema.

- *El Objetivo* es aquella actividad proyectada o planeada que se ha seleccionado antes de su ejecución y está basada tanto en apreciaciones subjetivas como en razonamientos técnicos de acuerdo con las características que posee el sistema.

2.1.2 SISTEMA INFORMÁTICO

Un sistema informático es aquel conjunto de elementos que tiene como misión tratar alguna información de forma automática. Dicho de otra manera, para hacer uso de la informática necesitaremos un sistema informático.
En él, distinguiremos tres elementos

- Hardware: Parte física. Los componentes físicos del ordenador, cpu, memoria, tarjetas...

- Software: Parte lógica. Aplicaciones y programas que hacen posible el tratamiento de información.

- Personal: Nosotros, los informáticos o usuarios, también formamos

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
SEDE AMBATO
BIBLIOTECA

parte de ese sistema.

Todo tratamiento, siempre cumple el mismo esquema. Entrada - Proceso - Salida. En la entrada el ordenador recibe los datos (introducidos por teclado, extraídos del disco duro, de un disquette...), en el proceso actúa con ellos y en la salida ofrece los resultados de esa actuación. Por ejemplo, supongamos que deseamos calcular la suma 3 + 4. En primer lugar metemos los datos en el programa destinado a realizar esas tareas (por ejemplo, la típica calculadora de Windows), el ordenador lo calcula (proceso) y devuelve el resultado (salida).

2.2 BASES DE DATOS

Posiblemente uno de los elementos informáticos de mayor desarrollo desde el comienzo de la informática. Una base de datos es el conjunto de información organizada sistemáticamente. El diseño de una base de datos se puede dividir en tres etapas: análisis conceptual (que sería la descripción de toda la información que se quiere incluir), el diseño lógico y el diseño físico.

2.2.1 MODELOS DE BASE DE DATOS

Para organizar esa información, de forma que sea de fácil acceso, se establecen modelos de bases de datos. Estos modelos son principalmente cuatro:

1. El modelo jerárquico, en el cual los enlaces entre las entidades son jerárquicos.
2. El modelo en red, en el cual la base de datos se describen como un conjunto de entidades unidas entre sí por enlaces.
3. El modelo relacional, en el que la base de datos se describe como un conjunto de tablas.
4. El modelo de bases de datos deductivas, es un sistema de bases de datos que tiene la capacidad de definir reglas con las cuales deduce o infiere información adicional a partir de hechos almacenados.

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
SEDE AMBATO
BIBLIOTECA

Los dos primeros modelos se utilizan solamente en el contexto de grandes sistemas. El modelo más usado es el relacional, por ser el de concepción y utilización más sencilla. Está basado en la agrupación de los datos en forma de tablas formadas por filas y columnas. Las filas de una tabla se las llama registros y a las columnas, campos. La intersección entre la fila y la columna es el dato. Las tablas tienen que contener uno o varios campos claves. El campo clave debe ser único, o sea su contenido no debe repetirse en dos registros distintos y por tanto se puede decir que es el campo que identifica unívocamente al registro.

2.2.2 GESTOR DE BASE DE DATOS.

La herramienta más potente de las bases de datos es la creación de diferentes bases de datos sobre un problema concreto, relacionadas entre sí, de forma que en vez de crear macro bases de datos de difícil gestión, lo que se irá creando son diferentes bases enlazadas entre sí por campos de las relaciones y que permiten gestionarlas de una manera rápida y eficaz.

Por ejemplo, si se tiene una base de datos con clientes, en las que figuran todos los datos de dichos clientes, se tiene dos opciones cuando se gestionen los pedidos que hacen dichos clientes: la primera, en esa misma base de datos donde figuran sus datos, ampliando el número de campos para dar cabida a los pedidos, con lo que se generará un registro por cada pedido, en el que estarán repetidos los datos del cliente y sólo variará lo que haya pedido en cada momento, o una segunda opción, que es crear una base de datos donde sólo figuren los pedidos y que esté enlazada con los datos del cliente por un campo clave que puede ser, por ejemplo, el número de cliente. A esta segunda opción se la llama crear bases de datos relacionales.

Existen multitud de gestores de bases de datos, pero los más famosos son: Paradox. dBASE, Oracle. Access. Approach. FileMaker y Clipper.

2.2.3 BASES DE DATOS RELACIONALES

Es un conjunto exhaustivo de datos estructurados, fiables y homogéneos, organizados independientemente de su utilización y de su implementación en máquina, accesibles en tiempo real, compartibles por usuarios concurrentes que tienen necesidades de información diferentes y no predecibles en el tiempo.

Cuenta con propiedades en los datos de manera que estos son:

- Estructurados independientemente de las aplicaciones y del soporte de almacenamiento que los contiene.
- Presentan la menor redundancia posible y son compartidos por varios usuarios y/o aplicaciones.
- Están bajo un control centralizado.

Los sistemas relacionales operan conceptualmente sobre archivos o **Tablas** de datos y no sobre los datos individuales contenidos en el archivo, las tablas permiten representar la información de forma mas compacta y son posibles de acceder a la información contenida en dos o más tablas simultáneamente.

Entre las características de las Tablas se puede citar las siguientes:

- Una tabla está formada por Filas y Columnas.
- Las **Filas** son equivalentes a los **Registros** de un archivo clásico (contienen los valores de los objetos o entidades descritas).
- Las **Columnas** son equivalentes a los **Campos** (que representan los atributos de los objetos o entidades descritas). El conjunto de valores que puede tomar un campo se llama ***dominio***.

2.3 HERRAMIENTAS DE BASE DE DATOS.

2.3.1. SYBASE ASE.

Sybase Adaptive Server Enterprise es uno de los manejadores relaciónales más conocido y poderoso del mercado actual. Su capacidad de almacenamiento, ambiente multiusuario y excelente desempeño lo han convertido en uno de los DBMS más exitosos de la historia de la computación, ganándose una enorme reputación en menos de 10 años, gracias a sus innovaciones, confiabilidad y facilidad de uso.

Sybase ofrece soluciones de software corporativos que permiten el acceso a la información y a las aplicaciones e-Business desde cualquier lugar y en cualquier momento. Hoy en día, cualquier negocio necesita estar disponible de forma global 24 horas al día.

Sin embargo, el uso de distintas tecnologías provoca que el desafío actual se encuentre en una adecuada integración de los sistemas para que se haga realidad la disponibilidad total de sitios unificados y accesibles desde cualquier lugar, en cualquier momento y para cualquier persona.

Con su plataforma e-Business, Sybase garantiza la integración de los sistemas y el lanzamiento a la Web de forma rápida y con bajo coste. La tecnología de Sybase es sobre todo flexible, lo que permite integrar cualquier aplicación y plataforma de negocio ya existente con las nuevas necesidades de la empresa.

Sybase Enterprise Application Studio es el completo conjunto de herramientas para construir y desplegar aplicaciones cliente / servidor, distribuidas y Web. Su misión: mover aplicaciones de negocio a Internet - rápidamente, inteligentemente y de manera segura, tan sólo utilizando Enterprise Application Studio de Sybase.

Todo lo que se necesita para construir aplicaciones cliente / servidor, distribuidas y Web, en una sola caja. Enterprise Application Studio integra la tecnología de Desarrollo Rápido de Aplicaciones (RAD - Rapid Application Development) con:

Sybase PowerBuilder, el 4GL líder de la industria. • *Sybase PowerJ*, el robusto ambiente de desarrollo Java. • *Sybase Enterprise Application Server*, es el poderoso servidor de aplicaciones de Sybase de peso ligero en relación con productos competitivos, ofrece integración de componentes tales como: C/C++, *PowerBuilder, CORBA, Java (J2EE) y ActiveX (Windows).* Esta enfocado a solucionar los problemas del negocio yendo mas allá de la estructura de la programación de aplicaciones. Brinda una Arquitectura "Multi-Proceso" e incorpora un Web Server para páginas estáticas.

Soporta protocolos distribuidos como IIOP, 110P/SSL, HTTP / HTTPS; conectividad con bases de datos vía JDBC, ODBC, nativo ORACLE 7.x, 8.x, Sybase y EJB 1.1, 2.0. Soporte de Seguridad: SSL 3.0, X.509 v3, RSA, Entrust y tunneling. Características avanzadas como: JDMK, in-Memory Fail over, Procedimientos Almacenados, Clustering, Fail over of Stateless Components, Two Phase Commit (Unix), Al CICS components y Fail over of State full.

2.4 HERRAMIENTAS DE ANÁLISIS Y DISEÑO DEL SISTEMA.

2.4.1. POWER DESIGNER

Power Designer es un ambiente completamente integrado de Ingeniería de Software para el análisis y diseño de aplicaciones empresariales, con capacidades para el modelamiento de negocios, datos y objetos, sus diferentes alternativas de licenciamiento contemplan:

Modelamiento de Procesos de Negocio: Power Designer brinda poder a los usuarios no técnicos para diseñar y modelar procesos de negocio en términos reales del negocio, usando un modelo simple, fácil de usar, altamente gráfico, y no técnico. Incluye soporte a la generación e ingeniería reversa de código XML

2.4.1.1. MODELAMIENTO DE DATOS:

Power Designer diseña y genera el esquema de la base de datos a través de un verdadero modelamiento de bases de datos relaciónales de dos niveles (conceptual y físico), basado en métodos probados. Power Designer también soporta técnicas específicas de modelamiento para Datawarehouse.

2.4.1.2. MODELAMIENTO DE OBJETOS:

Power Designer completa el análisis y el diseño usando técnicas UML estándar. Genera todos los diagramas contemplados en UML. A partir de un diagrama de clase, Power Designer automáticamente genera y realiza ingeniería reversa de ambientes populares como Java™ (incluyendo EJB 2.0), XML, Servicios Web, C++, PowerBuilder®, Visual Basic® y más, a través de un generador personalizable.

2.4.1.3. REPOSITORIO EMPRESARIAL:

La versión Enterprise de Power Designer agrega el valor de un repositorio de clase empresarial. El repositorio permite fácilmente visualizar y compartir modelos y otra información entre todos los miembros del equipo de desarrollo. El repositorio es altamente escalable y soporta seguridad basada en roles, control de versiones, búsqueda y generación de reportes.

2.5 LENGUAJES DE PROGRAMACIÓN.

Es un conjunto de sintaxis y reglas semánticas que definen los programas del computador. Es una técnica estándar de comunicación para entregarle instrucciones al computador. Un lenguaje le da la capacidad al programador de especificarle al computador, qué tipo de datos actúan y que acciones tomar bajo una variada gama de circunstancias, utilizando un lenguaje relativamente próximo al lenguaje humano.

Un programa escrito en un lenguaje de programación necesita pasar por un proceso de compilación, es decir, ser traducido al lenguaje de máquina para que pueda ser ejecutado por el ordenador.

2.5.1. TIPOS DE LENGUAJES

Los lenguajes de programación pueden ser clasificados de acuerdo a diversos criterios:

- Forma de ejecución
  - Lenguajes interpretados
  - Lenguajes compilados
- Paradigma de programación
  - Lenguajes declarativos
  - Lenguajes imperativos
    - Programación orientada a objetos
    - Programación estructurada
  - Lenguajes funcionales
  - Lenguajes esotéricos
  - Lenguajes visuales
- Nivel de abstracción
  - Lenguajes de bajo nivel
  - Lenguajes de alto nivel
- Generación
  - Primera generación
  - Segunda generación
  - Tercera generación
  - Cuarta generación
  - Quinta generación

2.5.1.1. LENGUAJES DE CUARTA GENERACIÓN.

4GL (Fourth Generation Language), los lenguajes de cuarta generación suponen una evolución de los de tercera generación. En estos lenguajes de programación

avanzados, el programador no incorpora el procedimiento a seguir, ya que el propio lenguaje es capaz de indicar al ordenador cómo debe ejecutar el programa,

Los lenguajes de cuarta generación son más fáciles de usar que los 3GL: suelen incluir interfaces gráficos y capacidades de gestión avanzadas, pero consumen muchos más recursos del ordenador que la generación de lenguajes anterior.

Estos tipos de lenguajes cuentan con las siguientes características:

- Lenguajes de dominio especifico para sistemas de negocios basados en un sistema de manejo de bases de datos.
- Normalmente incluye un lenguaje de queries de bases de datos, un generador de reportes y una hoja de cálculo.
- Disponible en las herramientas CASE.
- Efectivo en costos para sistemas pequeños y medianos

2.6 POWER BUILDER 9.0

Es un entorno de programación que está compuesto de diferentes herramientas para el desarrollo rápido que se manejan con el ratón, y un lenguaje de programación. Está pensada especialmente para el entorno cliente servidor, a través de odbc o de drivers nativos para las bases de datos más importantes del mercado.

El lenguaje de programación se llama Power Script, y posee una implementación de la programación orientada a objetos muy completa. Con PowerBuilder se puede crear objetos genéricos, crear herencias de éstos y generar objetos más especializados, implementar la herencia múltiple a través de los objetos de servicio, y finalmente crear aplicaciones reutilizando el código al máximo. PowerBuilder 9.0 es el primer entorno de desarrollo visual que proporciona la capacidad que se necesita para construir y desplegar aplicaciones web y distribuidas, con la misma facilidad de los desarrolladores cliente / servidor. La última versión ofrece la tecnología clave del producto, así como mejoras en las herramientas de desarrollo de aplicaciones. Mejorando el entorno tradicional de desarrollo cliente / servidor para incluir desarrollo de aplicaciones basadas en la Web, esta versión ayuda a los

desarrolladores a integrar PowerBuilder con servidores Web o de aplicaciones. Esto significa que los usuarios de sitios de comercio electrónico y e-Business realizados con PowerBuilder podrán acceder a la información de forma más sencilla y rápida.

PowerBuilder 9.0 incluye nuevas características para desarrolladores en el entorno cliente / servidor y para aquellas organizaciones que se encuentren en procesos de migración hacia entornos Web empresariales distribuidos.

Entre las nuevas funcionalidades de PowerBuilder se destacan:

- Soporte Double Byte Character Ser Soporte extendido para bases de datos.
- Mejora de las capacidades AutoScript. Excepciones de AutoScript cuando existen errores de sistema.
- Librerías altamente ejecutables personalizables. Desarrollo Web en modo compatible 4GL
- Gestión de archivos HTML, imágenes y otros activos del sitio Web. Desarrollo de páginas eficientes basado en arquitectura de eventos Integración mejorada con EAServer.

2.6.1 APLICACIÓN POWERBUILDER.

La interfase de usuario de una aplicación PowerBuilder, consiste en menús y ventanas que interactúan con el usuario. Las aplicaciones pueden incluir todos los controles estándar como botones, casilleros de selección (Checkboxes), listas dinámicas, cajas de selección, las que hacen que la aplicación sea fácil de desarrollar y usar.

2.6.2 LENGUAJE POWERSCRIPT

Los scripts consisten en comandos, funciones, declaraciones, estas mejoran los procesos y optimizan la respuesta en un evento. La ejecución de un evento puede además desencadenar una serie más de eventos, esto quiere decir que mientras se corre un grupo de sentencias, este puede llamar a que se ejecuten otras de otro evento diferente o complementario al proceso que se está ejecutando.

PowerBuider además utiliza una programación orientada a objetos, cualquier menú o ventana creada en PowerBuilder contiene a si misma un módulo llamado objeto. La construcción básica de bloques de programación en PowerBuilder son los objetos que el usuario crea y cada uno de estos contienen unas características particulares y conductas propias entre las que podemos citar: propiedades, eventos y funciones.

Además estas son apropiadas para utilizar las técnicas de programación llamados como:

- Encapsulamiento.- Archivo que contiene instrucciones PostScript encapsuladas.
- Herencia.- Características heredadas de otro objeto.
- Polimorfismo.- Cualidad de lo que tiene o puede tener distintas formas

2.7 ARQUITECTURA CLIENTE-SERVIDOR ("Modelo estándar para la ejecución de aplicaciones de redes y para sistemas operativos distribuidos")

Es una interacción de dos procesos, que se ejecutan en forma simultánea. Este modelo es una comunicación basada en una serie de preguntas y respuestas, que asegura que si dos aplicaciones intentan comunicarse, una comienza la ejecución y espera indefinidamente que la otra responda y luego continua con el proceso.

La mayoría de las comunicaciones extremo a extremo en las redes, están basadas en el modelo Cliente-Servidor.

Cliente: aplicación que inicia la comunicación, es dirigida por el usuario.

Servidor: es quien responde a los requerimientos de los clientes, son procesos que se están ejecutando indefinidamente.

Los procesos clientes son más sencillos que los procesos de los servidores, los primeros **no** requieren de privilegios de sistemas para operar, en cambio los procesos servidores sí.

Los usuarios cuando quieren acceder a un servicio de red, ejecutan un software cliente. El diseño de los servidores debe ser muy cuidadoso, debe incluir código para el manejo de:

Autenticación: verificar la identidad del cliente.
Seguridad de datos: para que estos puedan ser accedidos en forma apropiada.
Privacidad: garantizar que la información privada de un usuario, no sea accedida por alguien no autorizado.
Protección: asegurar que las aplicaciones no monopolicen los recursos del sistema.
Autorización: verificar si el cliente tiene acceso al servicio proporcionado por el servidor.

La tecnología denominada Cliente-Servidor es utilizada por todas las aplicaciones de Internet / intranet:

- Un *servidor* es un Computador remoto - en algún lugar de la red - que proporciona información según petición.
- Un *cliente* funciona en su Computador local, se comunica con el servidor remoto, y pide a éste información.
- El servidor envía la información solicitada.

Un único servidor típicamente sirve a una multitud de clientes, ahorrando a cada uno de ellos el problema de tener la información instalada y almacenada localmente.
Los sistemas Cliente-Servidor pueden ser de muchos tipos, dependiendo de las aplicaciones que el servidor pone a disposición de los clientes. Entre otros, existen:

- Servidores de Impresión, mediante el cual los usuarios comparten impresoras.
- Servidores de Archivos, con el cual los clientes comparten discos duros.

- Servidores de Bases de Datos, donde existe una única base de datos.
- Servidores de Lotus Notes, que permite el trabajo simultáneo de distintos clientes con los mismos datos, documentos o modelos.
- *Servidores Web* que también utilizan la tecnología Cliente-Servidor, aunque añaden aspectos **nuevos y propios a la misma.**

De acuerdo con la forma de prestar el servicio, se pueden considerar 2 tipos de servidores:

Servidores Interactivos: El servidor recoge la petición de servicio y además se encarga el mismo de atenderla.
Servidores Concurrentes: El servidor recoge cada una de las peticiones de servicio y crea otros procesos para que se encarguen de atenderlas.

2.7.1. VENTAJAS DE LA ARQUITECTURA CLIENTE-SERVIDOR

- El servidor no necesita tanta potencia de procesamiento, parte del proceso se reparte con los clientes.
- Se reduce el tráfico de red considerablemente. Idealmente, el cliente se conecta al servidor cuando es estrictamente necesario, obtiene los datos que necesita y cierra la conexión dejando la red libre.

2.7.2. HARDWARE Y SOFTWARE CLIENTE-SERVIDOR.

La arquitectura cliente-servidor permite al usuario en una máquina, llamado el cliente, requerir algún tipo de servicio de una máquina a la que está unido, llamado el servidor, mediante una red como una LAN (Red de Área Local) o una WAN (Red de Área Mundial).

Estos servicios pueden ser peticiones de datos de una base de datos, de información contenida en archivos o los archivos en sí mismos o peticiones de imprimir datos en una impresora asociada.

Aunque clientes y servidores suelen verse como máquinas separadas, pueden, de hecho, ser dos áreas separadas en la misma máquina. Por tanto, una única máquina Windows2000 puede ser al mismo tiempo cliente y servidor. Además una máquina cliente unida a un servidor puede ser a su vez servidor de otro cliente y el servidor puede ser un cliente de otro servidor en la red.

También es posible tener el *cliente* corriendo en un sistema operativo y el *servidor* en otro distinto.

Hay varios tipos comunes de máquinas clientes en entornos cliente-servidor. Uno de los clientes más populares es una computadora personal basada en Intel que ejecuta aplicaciones de DOS en un entorno Windows. Otro cliente popular es una Terminal X; de hecho, el sistema XWindows es un modelo cliente-servidor clásico. Hay también clientes Unix que ejecutan sistemas operativos como UnixWare. Un servidor que pide cosas a otro servidor es un cliente de la máquina a la que está pidiendo. Sin considerar el tipo de cliente que se esté usando en una red cliente-servidor, realizando al menos una de las funciones básicas descritas aquí como funciones del cliente.

2.8 TCP/IP

En el mundo de TCP/IP las comunicaciones entre Computadores se rigen básicamente por lo que se llama *modelo cliente-servidor,* este es un modelo que funciona extraordinariamente bien y es responsable de la flexibilidad en las comunicaciones.

2.8.1. TRANSPORTE Y APLICACIÓN.

Transporte y aplicación en el modelo cliente-servidor Usan UDP o TCP como mecanismo de transporte. Recordar que UDP no es fiable ni ofrece control de flujo, por lo que en este caso la aplicación debe proporcionar sus propias rutinas de recuperación de errores y de control de flujo. Suele ser más fácil desarrollar aplicaciones sobre **TCP,** un protocolo fiable, orientado a conexión. La mayoría de los protocolos de aplicación utilizan TCP, pero algunas aplicaciones se construyen

sobre UDP para proporcionar un mejor rendimiento reduciendo la carga del sistema que genera el protocolo.

Todos los protocolos de alto nivel tienen algunas características en común. Pueden ser aplicaciones escritas por el usuario o aplicaciones estandarizadas y distribuidas con un producto TCP/IP. De hecho, la pila TCP/IP incluye protocolos de aplicación tales como:

- ***TELNET*** para el acceso interactivo de una terminal a un host remoto.

- ***FTP*** **("File Transfer Protocol")** para transferencias de alta velocidad de un disco a otro.

- ***SMTP*** **("simple mail transfer protocol") como** sistema de correo de Internet.

Estas son las aplicaciones implementadas más ampliamente, pero existen muchas otras. Cada imp TCP/IP particular incluye un conjunto más o menos restringido de protocolos de aplicación.

TCP es un protocolo orientado a conexión. No hay relaciones maestro-esclavo. Las aplicaciones, sin embargo, utilizan un modelo cliente-servidor en las comunicaciones.

Un *servidor es* una aplicación que ofrece un servicio a usuarios de Internet; un *cliente* es el que pide ese servicio. Una aplicación consta de una parte de servidor y una de cliente, que se pueden ejecutar en el mismo o en diferentes sistemas.

Los usuarios invocan la parte cliente de la aplicación, que construye una *solicitud* para ese servicio y se la envía al servidor de la aplicación que usa TCP/IP como transporte.

El servidor es un programa que recibe una solicitud, realiza el servicio requerido y devuelve los resultados en forma de una respuesta. Generalmente un servidor puede tratar múltiples peticiones (múltiples clientes) al mismo tiempo.

Algunos servidores esperan las solicitudes en *puertos conocidos* de modo que sus clientes saben a que zócalo IP deben dirigir sus peticiones. El cliente emplea un puerto arbitrario para comunicarse. Los clientes que se quieren comunicar con un servidor que no usa un puerto conocido tienen otro mecanismo para saber a qué puerto dirigirse. Este mecanismo podría usar un servicio de registro como Portmap, que utiliza un puerto conocido.

El primer problema con el que se debe lidiar al querer comunicar dos procesos es la sincronización *(rendez-vous).* Suponer por ejemplo que existen dos programas que desean comunicarse en forma síncrona a través de un pipe.

La mejor forma de lograr la sincronización es que uno de los dos programas espere hasta que llegue algún indicio de que existe otro programa que se quiere comunicar con él. En el modelo cliente-servidor, el servidor siempre está esperando una conexión (inicia la conexión en forma pasiva) y el cliente inicia la conexión en forma activa. En general, un servidor debe ser capaz de atender simultáneamente a varios clientes, por esto son en general más complejos.

2.9 REDES LAN

Una LAN o Red de Área Local permite compartir información entre PC's sobre una rápida y confiable conexión entre los equipos que conforman la red. La mayoría de las LAN están basadas estructuras cableadas. Fig. 1

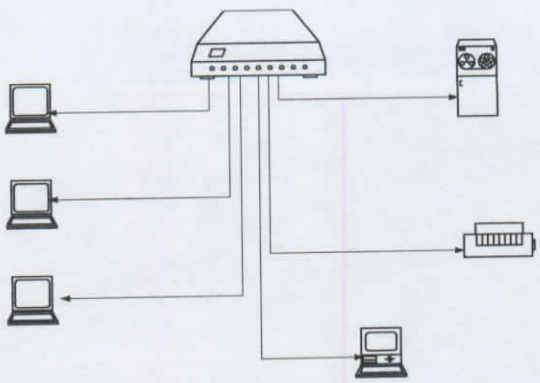


Fig 2.1

Ethernet es la tecnología más popular y más usada en redes locales. Compartir el medio significa que solo un dispositivo por vez puede enviar datos en la red, los otros solo pueden escuchar y decidir cuando deben recibir datos. Sin embargo todos los dispositivos conectados a la red comparten los 10/100 Mbps de ancho de banda

2.9.1. BENEFICIOS DE UNA RED DE DATOS

Datos o archivos compartidos

- Se puede intercambiar archivos desde una PC a otra.
- Se puede guardar información en otra PC sin el uso de medios como los disquetes o cualquier otro medio, solo se deberá configurar el software.

Compartir impresoras u otros dispositivos.

- Si se posee una impresora y varias PCs, la impresora puede ser compartida por todos los usuarios. Esto significa que se puede imprimir desde cualquier PC conectada a red en la impresora.

2.10 SISTEMAS OPERATIVOS

2.10.1. MICROSOFT WINDOWS 2000 PROFESSIONAL.

Windows 2000 proporciona un rendimiento más rápido, mayor fiabilidad, seguridad mejorada y un escritorio más manejable. Gracias a su amplia compatibilidad con aplicaciones y hardware, Windows 2000 es la mejor plataforma para ejecutar el software y hardware más reciente.

Windows 2000 amplia de forma significativa las capacidades existentes de Windows NT, lo que proporciona una respuesta del equipo mejor y más estable. El nuevo Sistema de archivos de cifrado asegura que la información es completamente privada y segura, incluso para usuarios que comparten un equipo. Ahora se puede crear fácilmente una red privada virtual para conectar equipos a través de Internet de forma segura. Windows 2000 admite las siguientes tecnologías de seguridad

escalables: Protocolo Kerberos V5, tarjeta inteligente, cifrado de clave pública y seguridad de Protocolo Internet (IPSec, *Internet Protocol security).* Sólo se necesita conectarse **una** vez para utilizar todos los recursos de red de Windows 2000 Server.

Con la última tecnología de instalación, puede instalar, personalizar y quitar programas de manera más fiable. El Instalador de Windows ayuda a evitar conflictos en los archivos del sistema, repara instalaciones incompletas, actualiza los componentes antiguos y quita todas las trazas de un programa, incluidas las entradas del Registro.

Para una asignación del espacio de disco más eficaz, Windows 2000 admite el sistema de archivos 32 bits de tabla de asignación de archivos (FAT, *File Allocation Table).* El sistema de archivos NTFS incluye muchas mejoras de rendimiento y nuevas características como cuotas de disco por usuario y cifrado de archivos.

La información viaja más rápido con el rendimiento mejorado de la pila TCP/IP, con lo que se reduce el tráfico en la red. Los archivos construidos con Lenguaje de marcado de hipertexto dinámico (DHTML, *Dynamic Hipar-texto Markup Language)* se cargan más rápidamente. Puede acomodar las últimas tecnologías de transmisión, incluidos los módems de cable, mediante la compatibilidad con la tecnología de modo de transferencia asincrónico.

En una *red cliente-servidor,* los clientes (equipos de escritorio normales) están conectados a un servidor (un equipo más capaz) que almacena y distribuye información, dispositivos y programas de software a los clientes. Las redes cliente-servidor son propias de las grandes organizaciones.

El administrador de la red establece permisos en una red basada en servidor para proporcionar acceso a los equipos, impresoras, información, etc. El servidor autentifica a los usuarios cuando éstos utilizan la red y comprueba los niveles de acceso que tiene cada uno.

2.11 DISEÑO DE MUEBLES MODULARES

2.11.1 TEORÍA DE LA FABRICACIÓN DE MUEBLES

Materiales y tecnología. "Hasta los años 60, los adelantos mayores de muebles de este siglo dependieron todos, en una forma u otra, sobre avances tecnológicos" escribe Sparke. Ella se refiere no solamente a las máquinas nuevas, que ella enumera meticulosamente para cada período subsecuente, pero también a los materiales novedosos de construcción. No contando la madera recta unida tradicional, Sparke enumera los materiales siguientes que han motivado a diseñadores para inventar las formas radicalmente novedosas para muebles:

- Madera doblada como típicamente en los productos de la fábrica Thonet en Austria (a la derecha).

- Tubos de acero. La primera base teórica para su uso en muebles fue dada probablemente por Le Corbusier, que deseó incluir muebles en la "máquina para vivir". Otra meta era economizar en material y espacio. Las máquinas para el nuevo material vinieron en el principio de industrias automovilísticas y de la construcción naval y avión. A partir de 1928 la fábrica Thonet modificó sus máquinas de doblar madera para manejar el acero también, y comenzó producir los diseños de acero de Marcel Breuer.

- Contrachapado. El primer uso de él estaba en las espaldas de guardarropas y en fondos de cajones, pero se amplió gradualmente a otras partes planas de muebles y animó a diseñadores a utilizar formas planas más frecuentemente. Un uso novelo de él comenzó con la invención de contrachapado doblado, por Alvar Aalto y otros.

- Tablero de partículas. Su rigidez permite prescindir del marco y también satisface bien al hacer se mismo (do-it-yourself). Favorece diseño caja-formado, e igual como contrachapado se puede manejar con las máquinas normales de la carpintería.

- Plástico, como contraste, aboga diseñar con formas curvadas tridimensionales, especialmente cuando se utiliza la fibra de vidrio o la tecnología ABS. Estos materiales y la tecnología vienen inicialmente desde la construcción de barcos. El PVC inflable presenta otro método de construir una silla plástica. Confía en la presión de aire de guardar la forma y por lo tanto requiere formas semejantes a cojines. La espuma del poliuretano se puede utilizar semejantemente para definir la forma de un asiento suave sin marco rígido.

- Papel (quizás reforzado con tela, plástico etc.) se ha utilizado de vez en cuando en muebles "pop", quizás para profesar una forma de vida poste-moderna, una movilidad social, una libertad intelectual y una protesta contra sociedad tradicional y sus estilos de período. Las formas posibles al diseñar con el papel son restrictas, siendo sobre todo variantes de cilindros y conos.

2.11.2 TEORÍA DEL DISEÑO DE MUEBLES

Los muebles pertenecen a esos productos que funcionan como una "segunda piel" de la gente en el mismo sentido que ropas, coches y hogares. Los muebles se asocian cerca con su usuario o dueño, y al seleccionar sus muebles el usuario puede definir el retrato que otras personas tendrán de él o de ella.

Qué gente desea decir sobre sí mismos, depende sobre todo de la estructura social local. En una comunidad rural tradicional un campesino quizás desea ser visto simplemente como digno de sus padres. Por consiguiente, él tiene toda razón de pegarse a los muebles vernáculos ancestrales.

Por otra parte, las sociedades urbanas occidentales tienen una herencia fuerte de la jerarquía social. Muchas personas querrían ascender a una clase social más alta o, si una subida inmediata se pareciera imposible, ellos quieren mostrar por lo menos que ellos son dignos y preparados a tal subida. Los muebles, así como ropas, se convienen bien para indicar la posición social y también la capacidad de una persona para una subida eventual del estado. Ésa es la razón por la que los estilos de

los muebles de la aristocracia europea han sido imitados de largo por las clases más bajas coexistentes.

La imitación de clases más altas explica cómo los estilos de muebles propagan a través de la sociedad, y también explica porqué los nuevos estilos vienen en existencia repetidas veces. La razón es que la aristocracia reinante no puede permitirse adherir de largo a ningún solo estilo. Si hacen así, los imitadores tarde o temprano alcancen una perfección igual del estilo. La clase alta debe desarrollar continuamente su estilo de vanguardia en un nivel más refinado. De esta manera condenan a los imitadores por siempre para tener un estilo levemente pasado, por lo tanto inferior.

El mecanismo socio psicológica antedicho, que Simmel primero expone en las modas de ropa, y Bourdieu (1984 a) entonces listamente analizó en el libro Distinción, explica mucho del desarrollo de estilos nuevos de muebles, por lo menos en Europa hasta bien en el 19 siglo. Los funcionamientos del mecanismo también se reflejan en el hecho de que los estilos del período franceses todavía llevan los nombres de los monarcas reinantes: "estilo Louis XV" etc.

Aún hoy, cuándo aristocracia ha perdido su posición como la clase social más envidiable, mucha gente todavía trae el modelo para su habitus y muebles de uno o de otro grupo aparentemente más distinguido en la sociedad. Puede ser que sea hoy la nueva clase de tecnócratas o de "meritocratas", o los ídolos de la música o deporte, los hogares de los cuales se publican en las revistas ilustradas.

La gran diferencia a generaciones precedentes es que hoy personas, especialmente los más jóvenes, son libres escoger sus modelos para el habitus. Ellos ahora pueden reflexionar y escoger en qué grupo de gente desean ser contados y qué estilo o genere de muebles es apropiado - él no necesita ser artículos del período aristocráticos. Están incluso libres volver nostálgico al estilo vernáculo de sus antepasados.

Además, el símbolo tradicional del estado o de la asociación de un grupo son hoy solamente unos de muchos mensajes alternativos posibles. Otras alternativas son

actitudes, humores y sentimientos que se pueden propagar hoy con la selección conveniente de muebles, de ropas y de otras pertenencias personales. Las personas imaginativas o amantes de diversiones pueden crear para sí mismos o encontrar hoy en las tiendas los productos ingeniosos que, colocados en contextos convenientes, pueden transmitir varios mensajes, o vagos y subconscientes, o abiertos e inequívocos. Algunos productos incluso se han dado un nombre que indica lo que simboliza, por ejemplo la silla "Hormiga" por Arne Jacobsen, a la derecha.

La gran diversidad de los mensajes que el público desea hoy entregar, hace casi imposible componer una teoría semiótica de muebles. La verdad es que hay un paradigma semiótica general (él se parafrasea en la página de Mensaje de un producto) pero muy poco de sus resultados hasta este momento se puede aplicar en el diseño de muebles. El método normal cuando un diseñador desea comunicar un mensaje en su trabajo es simplemente hacerlo por el método de tanteos. El público entonces decidirá tomarlo o dejarlo.

Los estándares son otra forma típica de presentación de la teoría del diseño. Los productos industriales suelen ser bastante similares entre ellos, y su diseño los problemas se suelen repetir con sólo pequeñas variaciones. No sería sensato que cada diseñador "comenzase en la casilla 1" y gastase mucho tiempo en inventar sus propias soluciones para problemas viejos y bien conocidos.

En lugar de ello, suele ser aconsejable usar soluciones generalmente aplicables a los problemas típicos de diseño. Su propósito es ahorrar tiempo, no sólo en el diseño sino también en la fase de la producción y a veces también en el uso, servicio y reparación del producto. Por otro lado, suele ser posible ahorrar materias primas. Todo esto lleva a ahorros en los costes, y habitualmente el único inconveniente es una reducción de la elección de productos por el consumidor. Los objetivos económicos de arriba son típicos en los estándares creados por la industria privada. Entre los tipos habituales de estándares se incluyen:

- estándares de métodos de producción,
- estándares de materiales, y
- estándares de productos y sus detalles

- esto representa el producto acabado o al detalle, o
- esto especifica los requisitos que el producto tiene que o debiera cumplir.

- métodos para probar todos los requisitos antedichos.

Los estándares son generalmente voluntarios, pero pueden ser declarados como obligatorios por una agencia gubernamental, la gerencia de la compañía u otro agente que tiene interés y el poder de hacer así.] En todo caso, la preparación de un estándar es tarea de los investigadores que suelen trabajar en "institutos de normalización" específicos. Éstos son financiados por una u otra de las partes implicadas en el diseño, como:

- Empresas privadas, que así se aseguran que todos los productos de la compañía se adhieren a principios uniformes de apariencia o son adecuados para las líneas de producción de la empresa,
- Un grupo o una asociación de empresas industriales puede elegir cooperar en el desarrollo de una serie de estándares.
- Hay institutos internacionales que coordinan o refunden estándares nacionales. Ejemplos notables son el ANSI, en los EE.UU., y la organización internacional ISO.

2.12 CALCULO DE PRESUPUESTOS

2.12.1 PRESUPUESTO

Un presupuesto es un plan integrador y coordinador que expresa en términos financieros con respecto a las operaciones y recursos que forman parte de una empresa para un periodo determinado, con el fin de lograr los objetivos fijados por la alta gerencia.

Los principales elementos del presupuesto son:

- *Es un plan,* esto significa que el presupuesto expresa lo que la administración tratará de realizar.

- *Integrador.* Indica que toma en cuenta todas las áreas y actividades de la empresa. Dirigido a cada una de las áreas de forma que contribuya al logro del objetivo global. Es indiscutible que el plan o presupuesto de un departamento de la empresa no es funcional si no se identifica con el objetivo total de la organización, a este proceso se le conoce como presupuesto maestro, formado por las diferentes áreas que lo integran.
- *Coordinador.* Significa que los planes para varios de los departamentos de la empresa deben ser preparados conjuntamente y en armonía.
- *En términos monetarios:* significa que debe ser expresado en unidades monetarias.
- *Operaciones:* uno de los objetivos primordiales del presupuesto es el de la determinación de los ingresos que se pretenden obtener, así como los gastos que se van a producir. Esta información debe elaborarse en la forma más detallada posible.
- *Recursos:* No es suficiente con conocer los ingresos y gastos del futuro, la empresa debe planear los recursos necesarios para realizar sus planes de operación, lo cual se logra, con la planeación financiera que incluya:

    - Presupuesto de Efectivo.
    - Presupuesto de adiciones de activos.

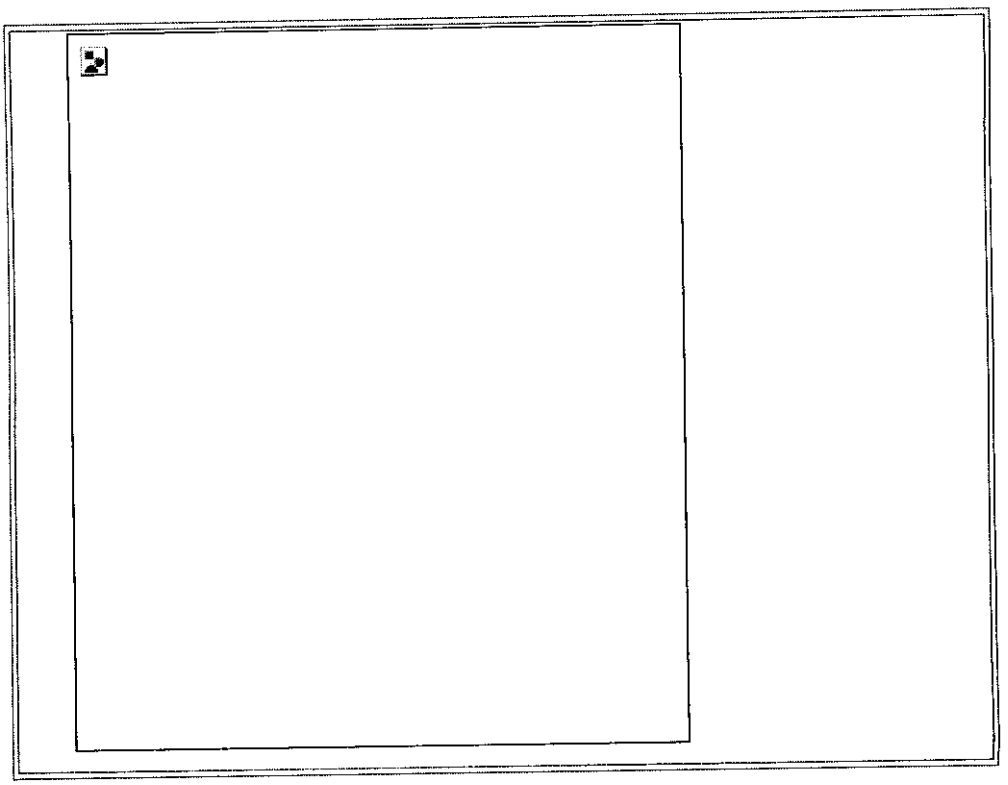
  Dentro de un periodo futuro determinado.

2.12.2 PRESUPUESTO MAESTRO

Está integrado básicamente por dos áreas que son:

- El presupuesto de operación.
- El presupuesto financiero.

Diagrama de la secuencia del presupuesto Maestro.



2.12.3 PRESUPUESTO DE VENTAS

La organización deberá determinar el comportamiento de su demanda, es decir, conocer qué se espera que haga el mercado, al concluir con esto, podrá elaborar un presupuesto propio de producción.

Normalmente se realiza este procedimiento en la mayoría de las empresas, ya que cuentan con una capacidad ociosa, es decir que la demanda es menor que la capacidad instalada para producir.

Existen casos en que las empresas elaboran el presupuesto de producción como primer paso.

También están las empresas del sector público que presupuestan primeramente sus gastos o necesidades y con base en ello determinan los ingresos que habrán de recaudar por vía impositiva (impuestos, derechos, etc).

Para desarrollar el presupuesto de ventas es recomendable la siguiente secuencia:

1. Determinar claramente el objetivo que desea lograr la empresa con respecto al nivel de ventas en un periodo determinado, así como las estrategias que se desarrollarán para lograrlo.

2. Realizar un estudio del futuro de la demanda, apoyado en ciertos métodos que garanticen la objetividad de los datos, como análisis de regresión y correlación, análisis de la industria, análisis de la economía, etc.

3. Basándose en los datos deseados para el futuro que generó el pronóstico y en el juicio profesional de los ejecutivos de ventas, elaborar el presupuesto de éstas tratando de dividirlo por zonas, divisiones, líneas, etcétera, de tal forma que se facilite su ejecución.

Una vez aceptado el presupuesto de ventas, debe comunicarse a todas las áreas de la organización para que se planifique el presupuesto de insumos.

2.12.4 PRESUPUESTO DE PRODUCCIÓN

Una vez que ya se determinó el presupuesto de ventas, se debe elaborar el plan de producción. Éste es importante ya que de él depende todo el plan de requerimientos con respecto a los diferentes insumos o recursos que se utilizarán en el proceso productivo.
Para determinar la cantidad que se debe producir de cada una de las líneas que se vende la organización, hay que considerar las siguientes variables:

- Ventas presupuestadas de cada línea.
- Inventarios finales deseados para cada tipo de línea.
- Inventarios iniciales con que se cuente para cada línea.

2.12.4.1 PRESUPUESTO DE PRODUCCIÓN POR LÍNEA

Ventas presupuestadas + Inventario final deseado de Arts. Terminados -Inventario inicial de Arts. Terminados. = Presupuesto de producción por línea

La fórmula anterior supone que los inventarios en proceso tienen cambios poco significativos. De no ser así, se deberían considerar dentro del análisis para determinar la producción de cada línea.

Hasta ahora se ha planteado la necesidad de conocer los niveles de inventarios al principio y al final del periodo productivo, sin embargo, dentro de dicho periodo hay que determinar cuál es la política deseada por cada empresa con respecto a la producción.

Las políticas más comunes son:

- Producción estable e inventario variable.
- Producción variable e inventario estable.
- Combinación de las dos anteriores.

2.12.5 PRESUPUESTO DE NECESIDADES DE MATERIA PRIMA Y DE COMPRAS.

Una vez concluido el presupuesto de producción, se puede diagnosticar las necesidades de los diferentes insumos.

Bajo condiciones normales, cuando no se espera escasez de materia prima, la cantidad debe estar en función del estándar que se haya determinado para cada tipo de ellas por producto, así como la cantidad presupuestada para producir en cada línea, indicando a la vez el tiempo en que se requerirá.

Materia prima"A" requerida= Producción presupuestada de una línea * estándar materia prima "A"

El presupuesto de requerimientos de materia prima se debe expresar en unidades monetarias una vez que el departamento de compras defina el precio al que se va a adquirir, lo cual constituye el costo del material presupuestado.

En este presupuesto se incluye únicamente el material directo, ya que los materiales indirectos (lubricantes, accesorios, etc) se incorporan en el presupuesto de gastos indirectos de fabricación.

2.12.6 PRESUPUESTO DE MANO DE OBRA DIRECTA.

Este presupuesto trata de diagnosticar claramente las necesidades de recursos humanos y cómo actuar, de acuerdo con dicho diagnóstico para satisfacer los requerimientos de la producción planeada.

Debe permitir la determinación del estándar en horas de mano de obra para cada tipo de línea que produce la empresa, así como la calidad de mano de obra que se requiere, con lo cual se puede detectar si se necesitan más recursos humanos o sí los actuales son suficientes.

Una vez calculado el número de obreros requeridos, se debe determinar qué costará esa cantidad de recursos humanos, o sea, traducir el presupuesto de mano de obra directa, expresada en horas estándar o en número de personas y calidad, a unidades monetarias, es decir, calcular el presupuesto del costo de mano de obra.

2.12.7 PRESUPUESTO DE GASTOS INDIRECTOS DE FABRICACIÓN.

El presupuesto debe elaborarse con anticipación de todos los centros de responsabilidad del área productiva que efectúan cualquier gasto productivo indirecto. Es importante que, al elaborar dicho presupuesto, se detecta perfectamente el comportamiento de cada una de las partidas de gastos indirectos, de manera que los gastos de fabricación variables se presupuesten en función del volumen de producción previamente determinado y los gastos de fabricación fijos se planeen dentro de un tramo determinado de capacidad independiente al volumen de producción presupuestado.

Cuando se ha elaborado el presupuesto de gastos de fabricación, debe calcularse la tasa de aplicación tanto en su parte variable como en su parte fija y elegir una base

que sea adecuada para la estructura del presupuesto de gastos de fabricación indirectos.

Se puede sintetizar lo anterior por medio de la fórmula:

$Y = a + bx$

Donde:

a = los gastos de fabricación fijos.

b = gastos variables por unidad a producir.

x = volumen de actividad.

En el contexto del desarrollo del plan maestro es muy importante utilizar el denominado presupuesto flexible, que consiste en presupuestar según diferentes niveles de actividad tanto los ingresos como los gastos, de acuerdo con el comportamiento que manifiesten ambos en función de una actividad determinada.

Presupuesto de gastos de operación

Este presupuesto tiene por objeto planear los gastos en que incurrirán las funciones de distribución y administración de la empresa para llevar a cabo las actividades propias de su naturaleza.

Al igual que los gastos indirectos de producción, los gastos de operación deben ser separados en todas las partidas de gastos variables y fijos para aplicar el presupuesto flexible a estas áreas, utilizando costeo con base en actividades.

El volumen según el cual cambiarán las partidas variables no será el de producción sino el adecuado a su función generadora de costos.

2.12.8 ESTADO DE RESULTADOS PRESUPUESTADO.

El presupuesto maestro está constituido por dos presupuestos el de operación y el financiero. El primero de ellos se refiere propiamente a las actividades de producir,

vender y administrar la organización, que son las actividades típicas a través de las cuáles una empresa realiza su misión de ofrecer productos o servicios a la sociedad. Dichas actividades dan origen a los presupuestos de ventas, de producción, de compras, de requerimientos de materia prima, de mano de obra, de gastos indirectos y los gastos de operación, costos de ventas.

Estos a su vez requieren ser resumidos en un reporte que permita a la administración, conocer hacia dónde se dirigirán los esfuerzos en torno a la operación de la compañía, lo cual se logra a través del Estado de Resultados presupuestado.

2.12.9 PRESUPUESTO FINANCIERO.

El plan maestro debe culminar con la elaboración de los estados financieros presupuestados, que son reflejo del lugar en donde la administración quiere colocar la empresa, así como cada una de las áreas, de acuerdo con los objetivos que se fijaron para lograr la situación global.

A parte de los estados financieros presupuestados anuales, pueden elaborarse reportes financieros mensuales o trimestrales o cuando se juzgue conveniente para efectos de retroalimentación, lo que permite tomar las acciones correctivas que se juzguen oportunas en cada situación.

El presupuesto financiero aunado a ciertos datos del estado de resultados presupuestado expresa el estado de situación financiera presupuestado y el estado de flujo de efectivo presupuestado.

El estado de resultados, el estado de situación financiera y el estado de flujo de efectivo presupuestado indican la situación proyectada. Con estos informes queda concluida la elaboración del plan anual o plan maestro de una empresa.

CAPITULO III

3. SISTEMA PARA EL DISEÑO DE MUEBLES MODULARES Y LA OBTENCIÓN DE PRESUPUESTO DEL MISMO.

3.1 INTRODUCCIÓN

El advenimiento de nuevas herramientas para el diseño de muebles modulares y la realización de presupuestos de los mismos, han permitido a las empresas fabricantes de muebles modulares un cambio substancial en la sociedad de consumo, inundándonos con un mundo de posibilidades de mejorar sus productos.

El desarrollo de un sistema que sea capaz de obtener el presupuesto de un mueble modular y al mismo tiempo visualizar el diseño del mismo facilitara el trabajo de la empresa para la cual se desarrolla este sistema. En nuestro país y en particular en la ciudad de Ambato en base a un sondeo se puede decir que no es muy común el uso de estos sistemas ya que el proceso lo realizan en forma manual.

Un área interesante para aplicar este sistema y es la empresa CRUMAD, es por esta razón el interés de la presente investigación en el desarrollo de un sistema para el diseño de muebles modulares y la obtención de presupuesto del mismo "SIDIMMOP".

En éste capítulo se explica la información técnica utilizada en el Sistema antes mencionado. Se describe la metodología de desarrollo de software aplicada, así

como la estructuración de la información, las herramientas utilizadas y las pruebas de software realizadas.

3.2 ANÁLISIS DEL SISTEMA

En el análisis del sistema, se detalla el proceso que conlleva a la realización del Sistema para el Diseño de Muebles Modulares y la obtención de presupuesto del mismo.

3.2.1 OBJETIVOS

Determinar los requerimientos, delimitaciones, procesos y funciones del sistema SIDIMMOP.

3.2.2 ALCANCE

El sistema SIDIMMOP es una herramienta de Software desarrollado como un recurso de ayuda para la visualización del diseño de un mueble modular y la obtención del presupuesto del mismo para la empresa CRUMAD de la ciudad de Ambato.

SIDIMMOP ha sido diseñado en una arquitectura cliente-servidor la cual facilita la administración del software para varios usuarios.

El sistema en su estructura, tiene mantenimiento de información, procesos, reportes y control de permisos de acceso por menú.

En el sistema se han determinado los siguientes procesos: presupuesto, proforma y facturación.

En la proforma se establecerá el RUC, cliente, fecha, detalle, observación, producto, cantidad, valor, IVA y total, además se podrá visualizar los muebles modulares que están siendo presupuestados.

Los beneficios que brindará SIDIMMOP, son varios tales como:

- Control clientes y proveedores.
- Control de los materiales o materia prima con sus precios.
- Control de las clases de muebles.
- Control de Bancos.
- Control de las autorizaciones de las facturas que otorga el SRI.
- Control de Usuarios con sus respectivos permisos.

De todo lo detallado anteriormente el objetivo de SIDIMMOP es de facilitar los procesos de obtención de proformas y facturación de la empresa CRUMAD y alcanzar de esta manera una reducción de tiempo. Brindar a la empresa CRUMAD una automatización correcta de su trabajo y permitir el uso de Herramientas informáticas, que están a disposición hoy en día.

3.2.3 VISIÓN GENERAL

La implementación del sistema permitirá que la información de la empresa CRUMAD sea transparente, confiable, segura y organizada; adicionalmente permitirá mayor eficiencia en el proceso de obtención de proformas.

3.3 DESCRIPCIÓN GENERAL

3.2.4 PERSPECTIVA

SIDIMMOP, es un producto de software diseñado para plataformas Windows en una arquitectura cliente – servidor. Desarrollado en una poderosa herramienta de cuarta generación, como es Power Builder 9.0, la cual se acopla al sistema operativo, permitiendo el acceso y uso a todos los periféricos con los que cuenta el computador. Se utilizará como motor de Base de Datos Sybase Central, en su versión 11.5, el cual posee una buena integración con la herramienta de desarrollo antes mencionada.

3.2.5 FUNCIONES

- **Entrada**

Dentro de las Funciones de Entrada de SIDIMMOP, se requerirá en la configuración inicial el ingreso de datos sobre el Usuario Administrador, el cual

será el encargado de controlar todo el sistema así como su correcto funcionamiento.

Con SIDIMMOP se crean funciones de entrada que permitan, el ingreso, modificación, y eliminación, así como su correcto registro de provincias, ciudades, clientes, proveedores, materiales, bancos, clases de muebles y usuarios.

- **Procesamiento**

Con la información en el sistema, se realizarán los siguientes procesos:

- Validación de Usuario, por clave y por accesos a menú a procesos.
- Validación de datos, para información requerida sobre el acceso de usuario y control.
- Verificación de la existencia de información necesaria, requerida para la realización de los procesos, de manera adecuada
- Control de diferentes estados de los procesos, que permitan conservar la integridad de la información y permitir una secuencia lógica de los procesos a seguir.
- Control de ingreso de datos correctos y completos de los diferentes procesos.

- **Salidas**

SIDIMMOP, permitirá la salida de la información procesada por pantalla e impresora, por medio de diferentes formatos, informes y reportes que se generan de acuerdo a las necesidades y requerimientos de la Empresa.

3.2.6 CARACTERÍSTICAS DE USUARIO

El sistema va dirigido al propietario de la empresa CRUMAD, con el conocimiento de los procesos que se realizan en dicha empresa. La interfaz se ha diseñado de una manera que facilita su uso y entendimiento, así como la interacción entre el usuario y la información.

Dentro de SIDIMMOP, se permitirán dos tipos de Usuarios: Administrador y no Administrador.

El usuario **Administrador**, tendrá acceso a todo el sistema, sin restricción alguna, pudiendo realizar cualquier actividad y siendo el responsable del correcto funcionamiento del Sistema, así como la gestión de los permisos de acceso a los usuario no Administradores.

Un usuario **no Administrador**, estará sujeto a los permisos que se le asigne, determinando una función específica, y de esta manera controlando el manejo de la información. En cada opción de menú, se tendrá la posibilidad de permitir cuatro acciones, como se explica a continuación.

Opción Menú	Activa	Crear	Editar	Imprimir	Borrar
Tipo de Opción	Si / No	Si / No	Si / No	Si / No	Si / No

En cada opción, se le asignará, que acciones se pueden realizar a cada usuario, permitiendo de esta manera una administración del manejo de información.

3.2.7 RESTRICCIONES

Para el correcto funcionamiento del software en una arquitectura cliente – servidor se requiere habilitar el servicio de acceso a la base de datos.

El acceso a la información se lo realizará por medio de permisos por menú, el cual será asignado a cada uno de los usuarios según el nivel de privilegios que el administrador le otorgue. Todos los usuarios administradores tienen un acceso total a los datos y procesos, sin ningún tipo de restricción, como se detalló anteriormente, debiendo el Usuario administrador, el responsable de controlar todos los accesos al Sistema, y a cuales procesos un usuario podría acceder así como las acciones a realizar.

3.2.8 SUPOSICIONES Y DEPENDENCIAS

Los requerimientos necesarios que se asumen para el correcto funcionamiento del sistema, es un sistema operativo Windows NT 4.0 o Windows 2000.

Para el funcionamiento del Software será necesario las librerías de ejecución de Power Builder 9.0, y los controladores del motor de Base de Datos, los cuales estarán incluidos en el instalador, el cual los direccionará y registrará de manera automática.

3.2.9 REQUISITOS ESPECÍFICOS

3.2.9.1 REQUISITOS DE LA INTERFAZ EXTERNA

Se presentará al usuario una interfaz de manera general, en la que se podrá ingresar, modificar, eliminar y guardar la información, brindándole al usuario, un entorno de trabajo sencillo y fácil de utilizar, para que la información sea comprendida y manejada de la mejor manera.

Se estructurara una interfaz como la siguiente:

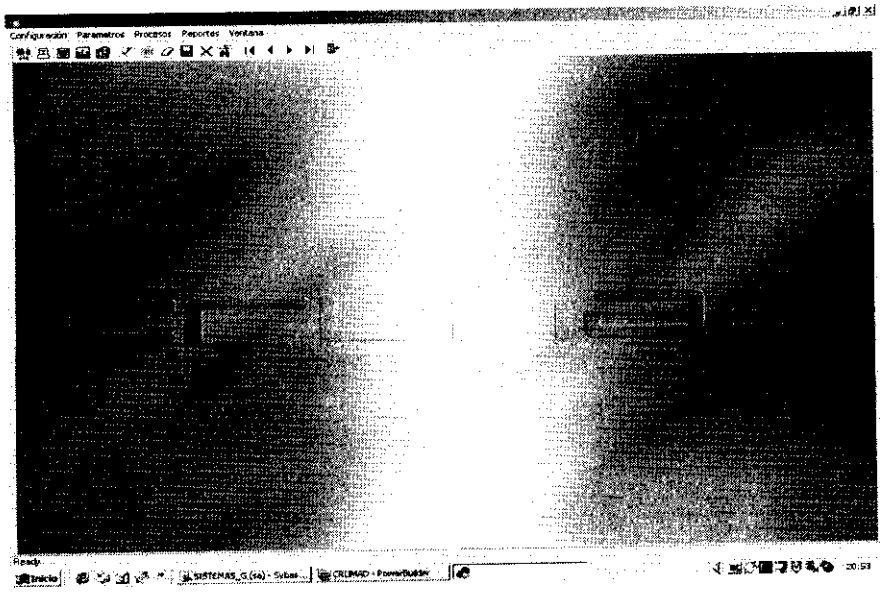


Figura 2 Diseño de Interfaz Externo

Esta interfaz está conformada por un menú principal, que permite el acceso a todos los procesos, así como a cada acción (menú), de cada uno de los procesos.

Se tendrá una barra de Herramienta de la aplicación, la cual facilitará, la configuración de impresoras y otros elementos globales. Cada ventana de procesos, tendrá su barra de herramientas, la cual le facilitara el acceso a cada acción (botón) de cada uno de los procesos, pudiéndose realizar los mismo, por medio del menú.

Se tendrá un área de manejo de Información, en la cual se presentará toda la información relativa al proceso en ejecución, en la que se permitirá, el ingreso de un nuevo registro, la modificación, la eliminación de los mismos, así como la visualización de los reportes.

Dentro del sistema, se han determinado los siguientes procesos:

Parámetros

- Provincias
- Ciudades
- Clientes
- Proveedores
- Bancos
- Materiales
- Clases de muebles
- Autorización de SRI

- Usuarios

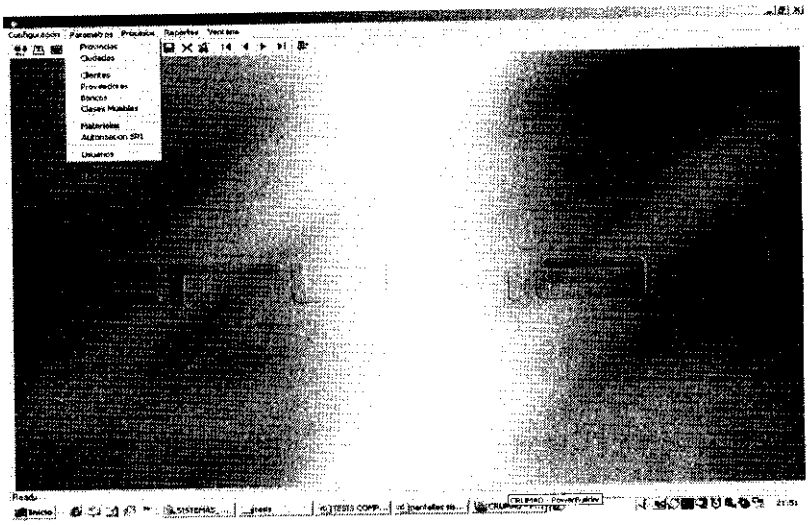


Fig. 3 Menú Parámetros

Procesos:

- Presupuestos
- Proforma
- Facturación

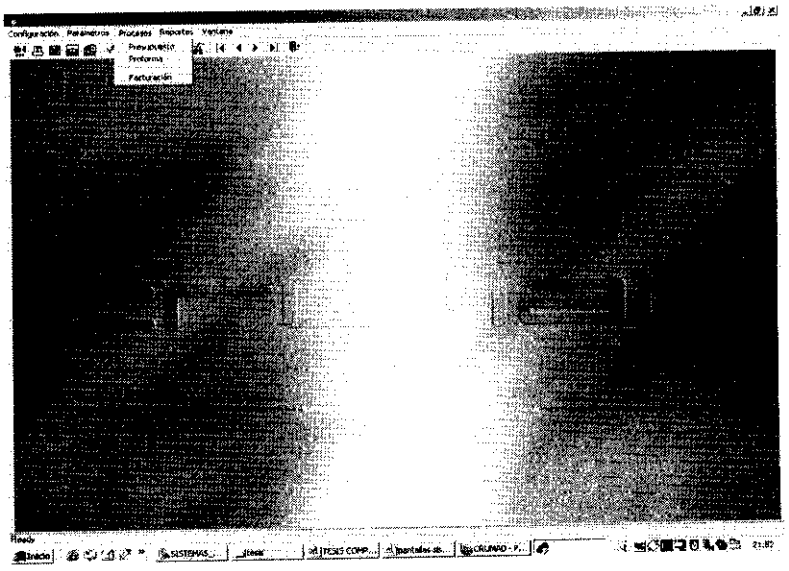


Fig. 4 Menú Procesos

Cada uno de ellos, tendrá la interfaz anteriormente diseñada, cambiándose, en el área de Manejo de Información, los datos que son presentados.

3.2.9.2 REQUISITOS FUNCIONALES

Para el correcto funcionamiento del sistema será importante realizar el ingreso de información de acuerdo a un orden lógico, con esto se garantizará que el procesamiento y salida de la información sea coherente. El sistema proveerá los controles necesarios para evitar un incorrecto ingreso de la información, sin embargo el usuario debe guardar consistencia con el orden en que los datos son manejados, para su correcto funcionamiento. Es importante, un adecuado ingreso de información, especialmente en los procesos, cuyos datos son necesarios para la generación de Reportes.

3.2.10 OBTENCIÓN DE LOS REQUERIMIENTOS DEL USUARIO.

3.2.10.1 IDENTIFICACIÓN DE LAS NECESIDADES

La obtención de requerimientos del Usuario, se ha elaborado con el Gerente de la empresa CRUMAD quien va a ser el que maneje el sistema.

Se han manifestado por parte del propietario de la empresa los aspectos más relevantes de su gestión a:

Administración de las siguientes variables:

- Clientes
- Proveedores

- Materiales
- Clases de muebles

La creación y administración de éstas variables agilitará el proceso manual que se lo viene realizando y ayudará a evitar posibles errores en el ingreso de la información, para lo cual debe existir un nivel jerárquico y orden establecido.

La formación de un Presupuesto de un mueble estará dada por los materiales y la clase de muebles, a los materiales se le asignarán la cantidad utilizada y el costo de los mismos.

Una vez realizado el presupuesto se procederá a la elaboración de una proforma, esta una vez aprobada por el cliente se procede a la facturación previa verificación en el proceso de autorizaciones si la factura a utilizarse esta vigente o no, de esta manera la facturación estará dada por la proforma.

Otros requerimientos que el sistema debe tener son los siguientes:

- Parametrizar el nombre de la empresa y responsables para el manejo del Sistema.
- Llevar un historial de clientes, proveedores y proformas.
- Controlar de una manera correcta los materiales que puedan ser tomadas para los presupuestos.
- Proteger la información para evitar su alteración.
- Permitir la obtención de reportes de una manera ágil.

- Auditar el ingreso, modificación y eliminación de información, para determinar responsables y llevar un control adecuado del sistema.
- Facilitar la generación de las facturas.

3.3 DISEÑO DEL SISTEMA.

3.3.1 INTRODUCCIÓN

De acuerdo al análisis expuesto y la obtención de los requerimientos, se ha estructurado un diseño, lo más acorde a las necesidades y realidad de la empresa CRUMAD, el que se describe a continuación.

3.3.2 DESCRIPCIÓN DE CADA UNO DE LOS PROCESOS

- **Configuración**

En este proceso se permitirá configurar la impresora, imprimir, exportar e importar datos.

- **Parámetros**

En este proceso se asignaran los datos referentes a las provincias, ciudades, clientes, proveedores, Bancos, la clase de muebles, los materiales, la autorización del SRI para las facturas y los Usuarios todos estos datos

servirán para utilizarse en los procesos de presupuestos, proformas y facturación.

Parámetro Provincias

Este proceso contendrá los datos de las provincias del país que servirán para llevar el control de la provincia de proveniencia tanto de clientes como de proveedores.

Parámetro Ciudades

Este proceso contendrá los datos de las ciudades del país que servirán para llevar el control de la ciudad de proveniencia tanto de clientes como de proveedores.

Parámetro Clientes

Este proceso contendrá los datos de todos los clientes de la empresa tanto los que soliciten una proforma como los que realizan la compra.

Parámetro Proveedores

Este proceso contendrá los datos de todos los proveedores de la empresa tanto proveedores permanentes como ocasionales.

Parámetro Clase de Muebles

Este proceso contendrá los tipos de muebles que se fabrican en la empresa CRUMAD.

Parámetro Bancos

Este proceso contendrá el listado de Bancos que servirán para la utilización como formas de pago en la facturación.

Parámetro Materiales

Este proceso contendrá el listado de los tipos materiales con los cuales se fabrican los muebles modulares.

Parámetro Autorización SRI

Con este proceso se llevara un control de las autorizaciones otorgadas por el SRI para las facturas que se emiten dentro del proceso de facturación, se controlara la fecha de caducidad y abra un historial de las autorizaciones otorgadas anteriormente.

Parámetro Usuarios

Con este proceso se llevara un control de los usuarios con los diversos

permisos otorgados para el manejo del sistema.

- **Procesos**

Proceso Presupuesto

En este proceso se permitirá la inserción, modificación, actualización de datos, también se permitirá la eliminación de datos siempre y cuando no exista dependencia de información; es decir, estos procesos manejarán la parametrización, para que los procesos consiguientes funcionen de una manera correcta.

Se obtendrá el presupuesto de determinado mueble modular en base a los costos de material, dimensiones y otros datos que se obtienen de acuerdo a los parámetros configurados.

También dentro de este proceso se podrá visualizar el mueble modular terminado que se ha presupuestado.

Proceso Proforma

En este proceso se permitirá la inserción, modificación, actualización de datos, también se permitirá la eliminación de datos siempre y cuando no exista dependencia de información; es decir, estos procesos manejarán la parametrización, para que los procesos consiguientes funcionen de una manera correcta.

Se obtendrá la proforma de varios muebles modulares que fueron presupuestados en el proceso anterior.

También dentro de este proceso se podrá visualizar los muebles modulares terminados que se ha presupuestado.

Proceso Facturación

En este proceso se permitirá la inserción, modificación, actualización de datos, también se permitirá la eliminación de datos siempre y cuando no exista dependencia de información; es decir, estos procesos manejarán la parametrización, para que los procesos consiguientes funcionen de una manera correcta.

Se obtendrá la factura de determinado mueble modular en base a la proforma.

Dentro de este proceso se podrá también visualizar los muebles modulares que se están facturando.

- **Reportes**

Reporte Clientes

En este proceso se obtendrá el reporte del historial de clientes que han solicitado proformas y de los que ha realizado compras, con datos como el RUC, nombres, dirección, teléfono entre otros.

Reporte Proveedores

En este proceso se obtendrá el reporte del historial de proveedores que han proporcionado ala empresa de sus materiales utilizados en la fabricación de los muebles modulares, con datos como el RUC, nombres, dirección, teléfono entre otros.

Reporte Proformas

En este proceso se obtendrá En este proceso se obtendrá el reporte del historial de las proformas realizadas, se podrá obtener reportes mensuales y anuales.

Reporte Facturación

En este proceso se obtendrá En este proceso se obtendrá el reporte del historial de las facturas realizadas, se podrá obtener reportes mensuales y anuales.

- **Ventana**

Este proceso permite organizar la vista de la pantalla nos da las opciones de vista de Ventana vertical, Ventana horizontal y Ajuste a pantalla.

3.3.3 DIAGRAMA DE FLUJO DE DATOS (DFD'S)

SIDIMMOP

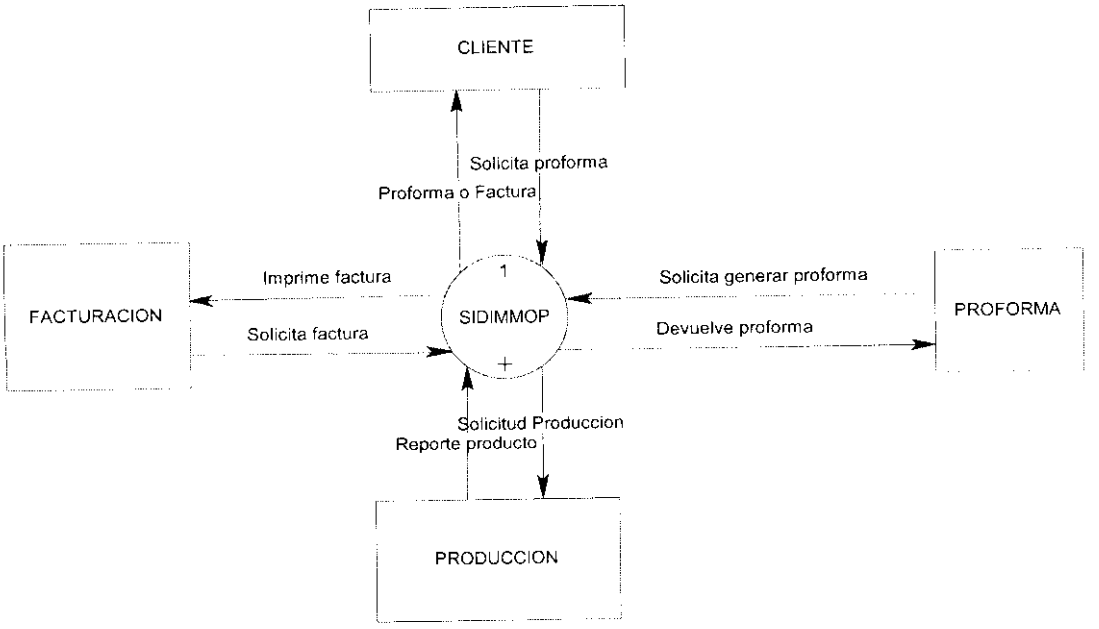


Figura 5: DFD Nivel 0 del Sistema SIDIMMOP

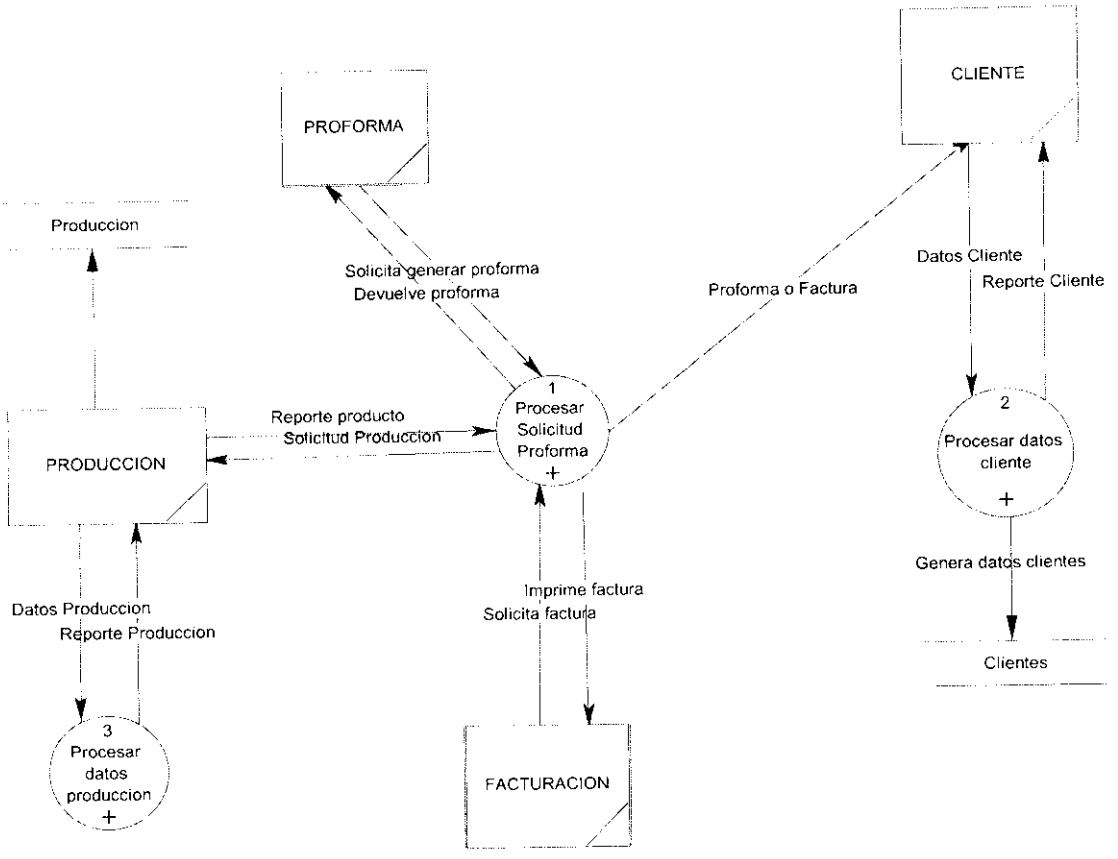


Figura 6: DFD Nivel 1 del Sistema SIDIMMOP

En los diagramas siguientes se muestran los DFD de los procesos más importantes que describen el funcionamiento del Sistema.

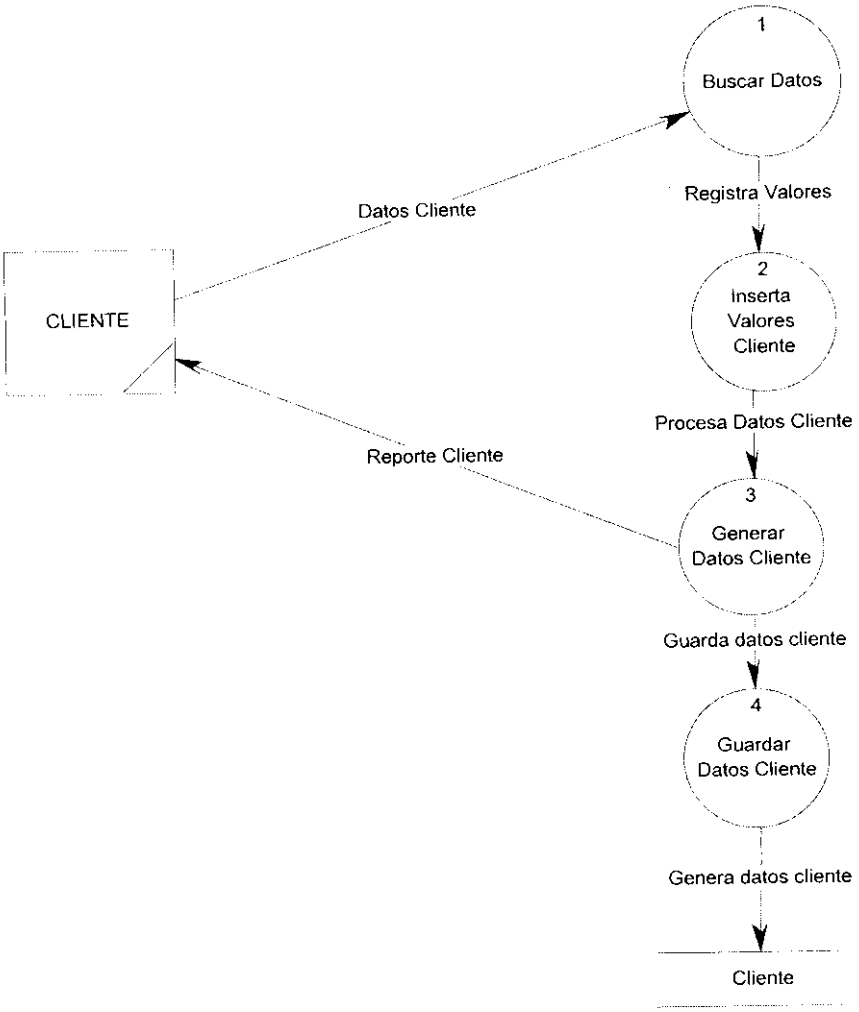


Figura 7: DFD Nivel 1 de creación de CLIENTES

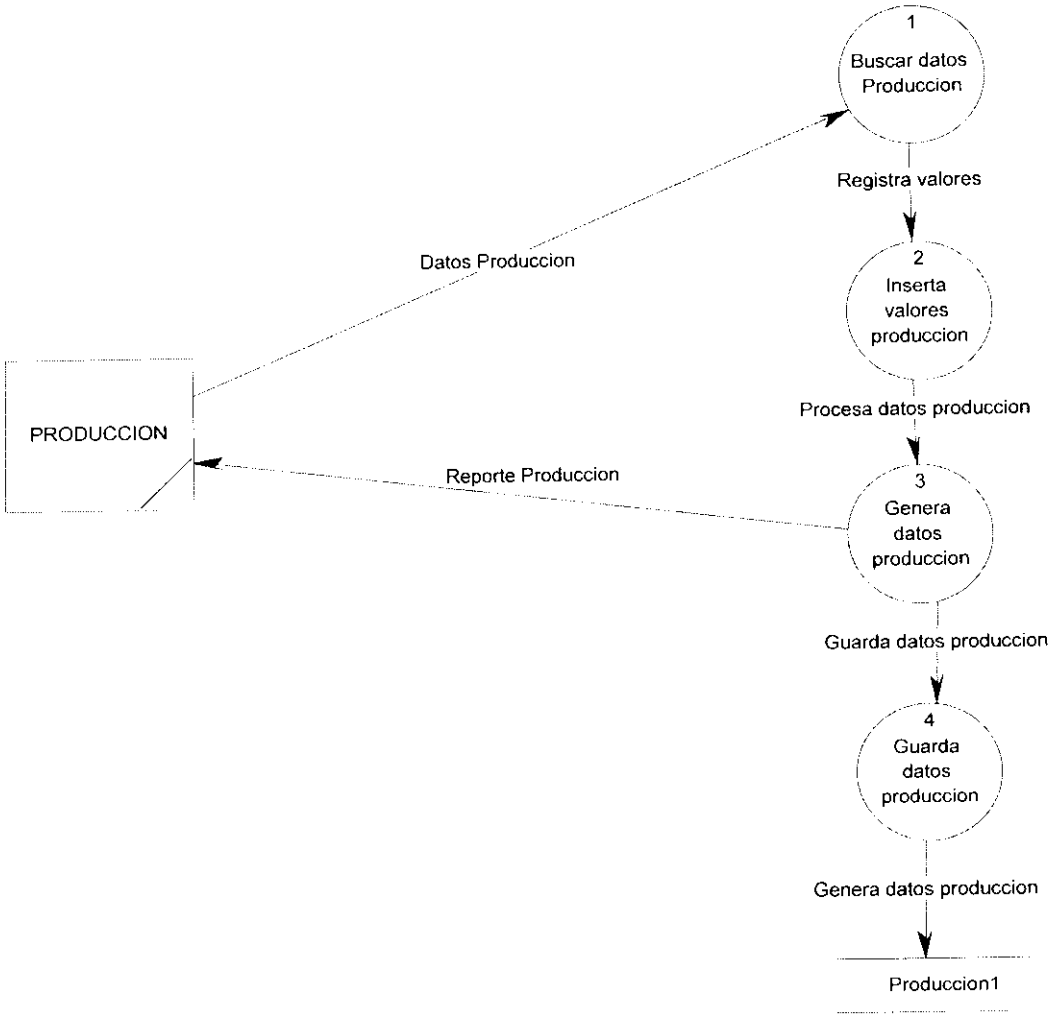


Figura 8: DFD Nivel 1 de creación de MATERIALES

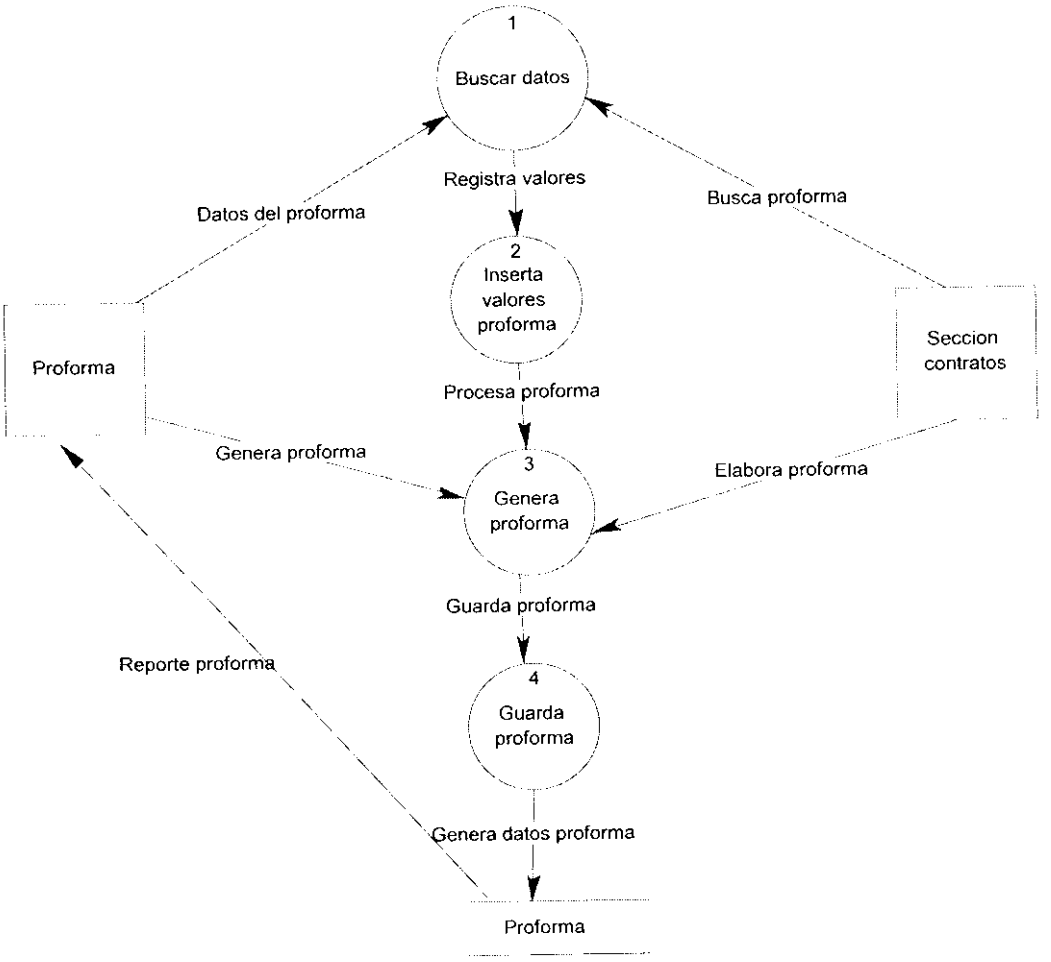


Figura 9: DFD Nivel 1 de creación de PROFORMAS

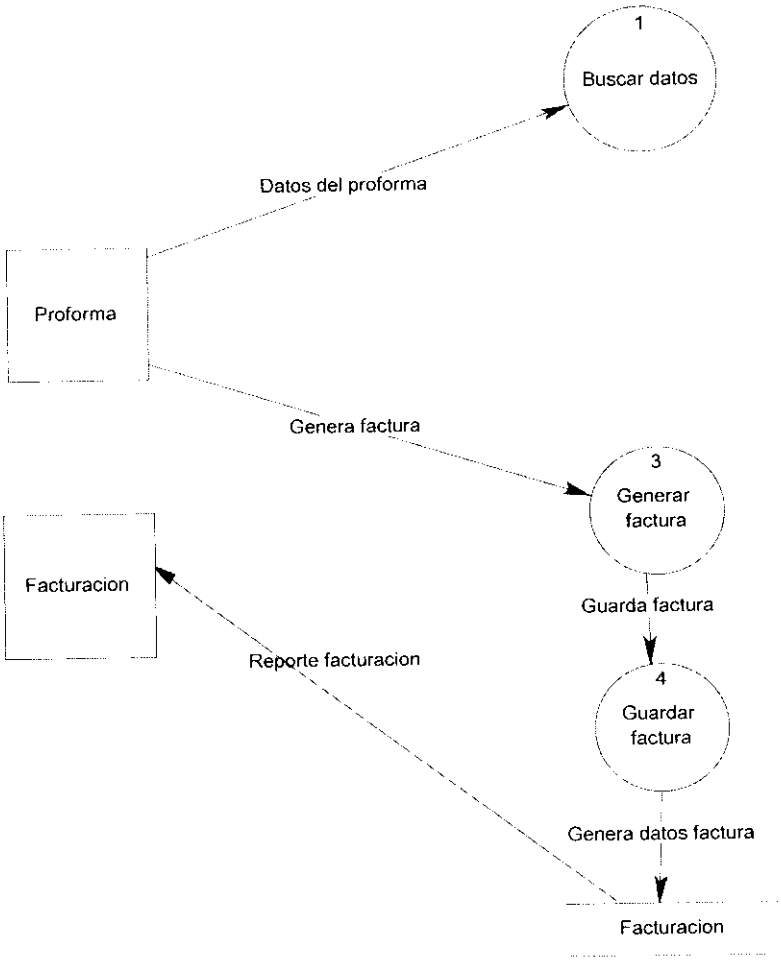


Figura 10: DFD Nivel 1 de creación de FACTURACIÓN

3.3.4 DISEÑO DE LA BASE DE DATOS

En ésta sección se muestra el diseño lógico y físico de la base de datos que estructura toda la información del Sistema SIDIMMOP

3.3.4.1 DISEÑO LÓGICO (CDM)

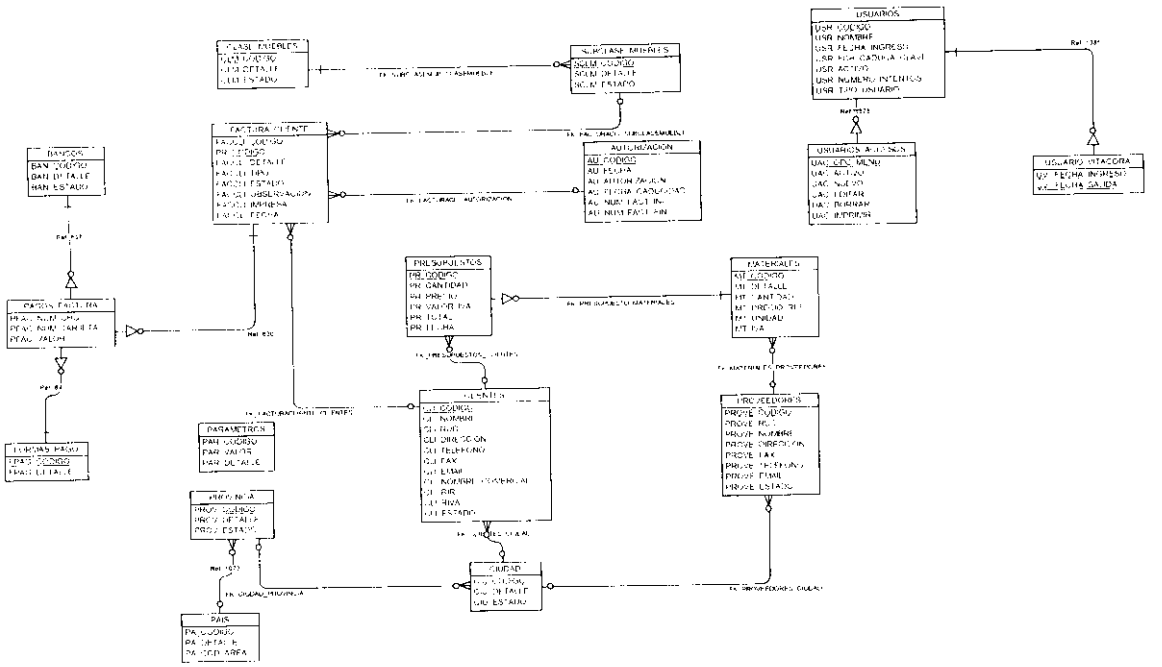


Fig. 11 Diseño Lógico de la Base de Datos

3.3.4.2 DISEÑO FÍSICO (PDM)

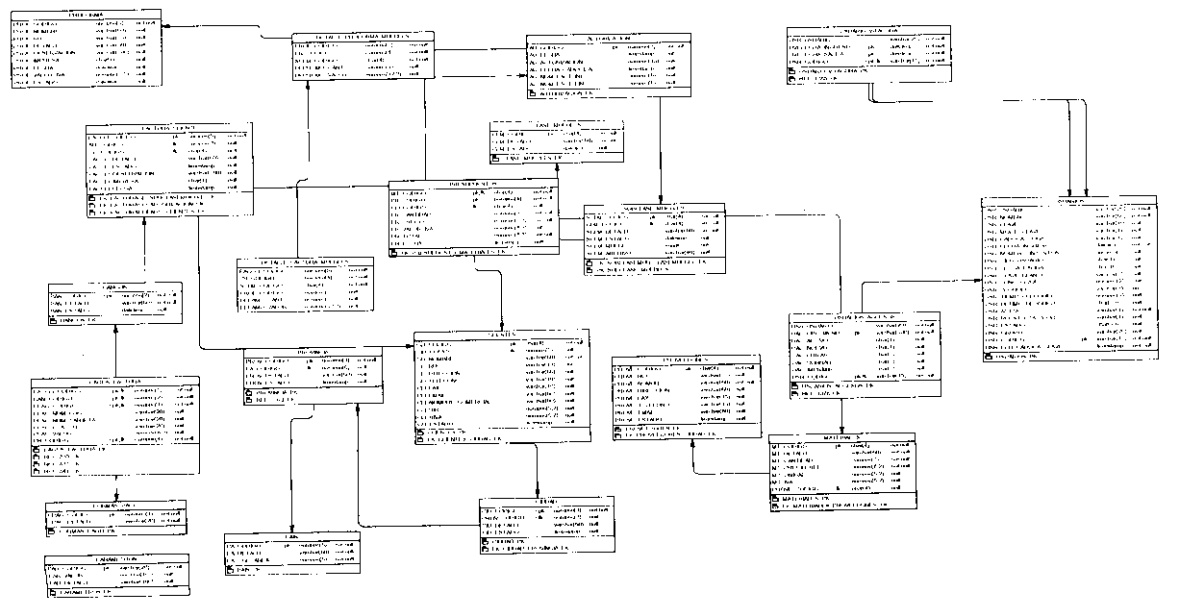


Fig. 12 Diseño Físico de la Base de Datos

3.3.5 DIAGRAMA DE OBJETOS DE POWER BUILDER USADOS EN LA APLICACIÓN

3.4 **DESARROLLO DEL PROYECTO**

3.4.1 DESARROLLO DE BASE DE DATOS

3.4.1.1 DICCIONARIO DE DATOS

Tables

Table List

Name	Code	Number
AUTORIZACIÓN	AUTORIZACIÓN	0
BANCOS	BANCOS	0
CIUDAD	CIUDAD	0
CLASE_MUEBLES	CLASE_MUEBLES	0
CLIENTES	CLIENTES	0
DETALLE_FACTURA_MUEBLES	DETALLE_FACTURA_MUEBLES	0
DETALLE_PROFORMA_MUEBLES	DETALLE_PROFORMA_MUEBLES	0
FACTURA_CLIENTE	FACTURA_CLIENTE	0

Name	Code	Number
FORMAS_PAGO	FORMAS_PAGO	0
MATERIALES	MATERIALES	0
PAGOS_FACTURA	PAGOS_FACTURA	0
PAÍS	PAÍS	0
PARÁMETROS	PARÁMETROS	0
PRESUPUESTOS	PRESUPUESTOS	0
PROFORMA	PROFORMA	0
PROVEEDORES	PROVEEDORES	0
PROVINCIA	PROVINCIA	0
SUBCLASE_MUEBLES	SUBCLASE_MUEBLES	0
USUARIO_VITACORA	USUARIO_VITACORA	0
USUARIOS	USUARIOS	0
USUARIOS_ACCESOS	USUARIOS_ACCESOS	0

Table AUTORIZACIÓN

Name:	AUTORIZACIÓN
Code:	AUTORIZACIÓN
Label:	
Number:	
PK constraint:	
Source:	Entity AUTORIZACIÓN

Column List

Name	Code	Type	P	M
AU_CODIGO	AU_CODIGO	numeric(3)	Yes	Yes
AU_FECHA	AU_FECHA	timestamp	No	No
AU_AUTORIZACION	AU_AUTORIZACION	numeric(13)	No	No
AU_FECHA_CADUCIDAD	AU_FECHA_CADUCIDAD	timestamp	No	No
AU_NUM_FACT_INI	AU_NUM_FACT_INI	numeric(6	No	No

Name	Code	Type	P	M
AU_NUM_FACT_FIN	AU_NUM_FACT_FIN	) numeric(6)	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
AUTORIZACION_PK	Yes	No	Yes	No	AU_CODIGO	ASC

Reference by List

Referenced by	Primary Key	Foreign Key
FACTURA_CLIENTE	AU_CODIGO	AU_CODIGO

Table BANCOS

Name: BANCOS
Code: BANCOS
Label:
Number:
PK constraint:
Source: Entity BANCOS

Column List

Name	Code	Type	P	M
BAN_CODIGO	BAN_CODIGO	numeric(2)	Yes	Yes
BAN_DETALLE	BAN_DETALLE	varchar(6 0)	No	Yes
BAN_ESTADO	BAN_ESTADO	datetime	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
BANCOS_PK	Yes	No	Yes	No	BAN_CODIGO	ASC

Reference by List

Referenced by	Primary Key	Foreign Key
PAGOS_FACTURA	BAN_CODIGO	BAN_CODIGO
PAGOS_FACTURA	BAN_CODIGO	BAN_CODIGO

Table CIUDAD

Name: CIUDAD
Code: CIUDAD
Label:
Number:
PK constraint:
Source: Entity CIUDAD

Column List

Name	Code	Type	P	M
CIU_CODIGO	CIU_CODIGO	numeric(3)	Yes	Yes
PROV_CODIGO	PROV_CODIGO	numeric(3)	No	No
CIU_DETALLE	CIU_DETALLE	varchar(60)	No	No
CIU_ESTADO	CIU_ESTADO	timestamp	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
CIUDAD_PK	Yes	No	Yes	No	CIU_CODIGO	ASC

Index Code	P	F	U	C	Column Code	Sort
FK_CIUDAD_PROVINCIA_FK	No	Yes	No	No	PROV_CODIGO	ASC

Reference to List

Reference to	Primary Key	Foreign Key
PROVINCIA	PROV_CODIGO	PROV_CODIGO

Reference by List

Referenced by	Primary Key	Foreign Key
CLIENTES	CIU_CODIGO	CIU_CODIGO

Table CLASE_MUEBLES

Name:	CLASE_MUEBLES
Code:	CLASE_MUEBLES
Label:	
Number:	
PK constraint:	
Source:	Entity CLASE_MUEBLES

Column List

Name	Code	Type	P	M
CLM_CODIGO	CLM_CODIGO	char(4)	Yes	Yes
CLM_DETALLE	CLM_DETALLE	varchar(60)	No	Yes
CLM_ESTADO	CLM_ESTADO	datetime	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
CLASE_MUEBLES_PK	Yes	No	Yes	No	CLM_CODIGO	ASC

Reference by List

Referenced by	Primary Key	Foreign Key
SUBCLASE_MUEBLES	CLM_CODIGO	CLM_CODIGO

Table CLIENTES

Name:	CLIENTES
Code:	CLIENTES
Label:	
Number:	
PK constraint:	
Source:	Entity CLIENTES

Column List

Name	Code	Type	P	M
CLI_CODIGO	CLI_CODIGO	char(4)	Yes	Yes
CIU_CODIGO	CIU_CODIGO	numeric(3)	No	No
CLI_NOMBRE	CLI_NOMBRE	varchar(60)	No	Yes
CLI_RUC	CLI_RUC	varchar(13)	No	No
CLI_DIRECCION	CLI_DIRECCION	varchar(60)	No	No
CLI_TELEFONO	CLI_TELEFONO	varchar(10)	No	No
CLI_FAX	CLI_FAX	varchar(10)	No	No
CLI_EMAIL	CLI_EMAIL	varchar(60)	No	No
CLI_NOMBRE_COMERCIAL	CLI_NOMBRE_COMERCIAL	varchar(60)	No	No
CLI_RIR	CLI_RIR	numeric(5	No	No

Name	Code	Type	P	M
CLI_RIVA	CLI_RIVA	numeric(5,2)	No	No
CLI_ESTADO	CLI_ESTADO	timestamp	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
CLIENTES_PK	Yes	No	Yes	No	CLI_CODIGO	ASC
FK_CLIENTES_CIUDAD_FK	No	Yes	No	No	CIU_CODIGO	ASC

Reference to List

Reference to	Primary Key	Foreign Key
CIUDAD	CIU_CODIGO	CIU_CODIGO

Reference by List

Referenced by	Primary Key	Foreign Key
FACTURA_CLIENTE	CLI_CODIGO	CLI_CODIGO
PRESUPUESTOS	CLI_CODIGO	CLI_CODIGO

Table DETALLE_FACTURA_MUEBLES

Name:	DETALLE_FACTURA_MUEBLES
Code:	DETALLE_FACTURA_MUEBLES
Label:	
Number:	
PK constraint:	
Source:	

Column List

Name	Code	Type	P	M
FACCLI_CODIGO	FACCLI_CODIGO	numeric(5)	No	Yes
PR_CODIGO	PR_CODIGO	numeric(4)	No	Yes
SCLM_CODIGO	SCLM_CODIGO	char(4)	No	Yes
PROF_CODIGO	PROF_CODIGO	numeric(5)	No	No
DEFAMU_CANT	DEFAMU_CANT	numeric(1)	No	No
DEFAMU_VALOR	DEFAMU_VALOR	numeric(12,2)	No	No

Reference to List

Reference to	Primary Key	Foreign Key
DETALLE_PROFORMA_MUEBLES	PROF_CODIGO	PROF_CODIGO
	PR_CODIGO	PR_CODIGO
	SCLM_CODIGO	SCLM_CODIGO

Table DETALLE_PROFORMA_MUEBLES

Name: DETALLE_PROFORMA_MUEBLES
Code: DETALLE_PROFORMA_MUEBLES
Label:
Number:
PK constraint:
Source:

Column List

Name	Code	Type	P	M
PROF_CODIGO	PROF_CODIGO	numeric(5	No	Yes

Name	Code	Type	P	M
PR_CODIGO	PR_CODIGO	numeric(4)	No	Yes
SCLM_CODIGO	SCLM_CODIGO	char(4)	No	Yes
DEPROMU_CANT	DEPROMU_CANT	numeric(1)	No	No
DEPROMU_VALOR	DEPROMU_VALOR	numeric(12,2)	No	No

Reference to List

Reference to	Primary Key	Foreign Key
PROFORMA	PROF_CODIGO	PROF_CODIGO
SUBCLASE_MUEBLES	SCLM_CODIGO	SCLM_CODIGO

Reference by List

Referenced by	Primary Key	Foreign Key
DETALLE_FACTURA_MUEBLES	PROF_CODIGO	PROF_CODIGO
	PR_CODIGO	PR_CODIGO
	SCLM_CODIGO	SCLM_CODIGO

Table FACTURA_CLIENTE

Name:	FACTURA_CLIENTE
Code:	FACTURA_CLIENTE
Label:	
Number:	
PK constraint:	
Source:	Entity FACTURA_CLIENTE

Column List

Name	Code	Type	P	M
FACCLI_CODIGO	FACCLI_CODIGO	numeric(5)	Yes	Yes
AU_CODIGO	AU_CODIGO	numeric(3)	No	No
CLI_CODIGO	CLI_CODIGO	char(4)	No	No
FACCLI_DETALLE	FACCLI_DETALLE	varchar(60)	No	No
FACCLI_ESTADO	FACCLI_ESTADO	timestamp	No	No
FACCLI_OBSERVACION	FACCLI_OBSERVACION	varchar(100)	No	No
FACCLI_IMPRESA	FACCLI_IMPRESA	char(1)	No	No
FACCLI_FECHA	FACCLI_FECHA	timestamp	No	No

Check

Domain:
Low Value:
High Value:
Default Value:
Unit:
Format:
Uppercase: No **Lowercase:** No **Can't modify:** No
List of Values: S SI
N NO

Index List

Index Code	P	F	U	C	Column Code	Sort
FK_FACTURACLI_SUBCL	No	Yes	No	No		
ASEMUEBLE_FK						
FK_FACTURACLI_AUTOR	No	Yes	No	No	AU_CODIGO	ASC

Index Code	P	F	U	C	Column Code	Sort
IZACION_FK FK_FACTURACLIENTE_C LIENTES_FK	No	Yes	No	No	CLI_CODIGO	ASC

Reference to List

Reference to	Primary Key	Foreign Key
AUTORIZACION	AU_CODIGO	AU_CODIGO
CLIENTES	CLI_CODIGO	CLI_CODIGO

Reference by List

Referenced by	Primary Key	Foreign Key
PAGOS_FACTURA	FACCLI_CODIGO	FACCLI_CODIGO

Table FORMAS_PAGO

Name: FORMAS_PAGO
Code: FORMAS_PAGO
Label:
Number:
PK constraint:
Source: Entity FORMAS_PAGO

Column List

Name	Code	Type	P	M
FPAG_CODIGO	FPAG_CODIGO	numeric(1)	Yes	Yes
FPAG_DETALLE	FPAG_DETALLE	varchar(20)	No	Yes

Name	Code	Type	P	M
PFAC_NUM_CHQ	PFAC_NUM_CHQ	varchar(20)	No	No
PFAC_NUM_TARJETA	PFAC_NUM_TARJETA	varchar(20)	No	No
PFAC_CTA_CTE	PFAC_CTA_CTE	varchar(20)	No	No
PFAC_VALOR	PFAC_VALOR	numeric(8,2)	No	No
PR_CODIGO	PR_CODIGO	numeric(4)	Yes	Yes

Index List

Index Code	P	F	U	C	Column Code	Sort
PAGOS_FACTURA_PK	No	No	Yes	No	FPAG_CODIGO	ASC
					BAN_CODIGO	ASC
					FACCLI_CODIGO	ASC
					PR_CODIGO	ASC
REF_830_FK	No	No	No	No	FACCLI_CODIGO	ASC
					PR_CODIGO	ASC
REF_837_FK	No	Yes	No	No	BAN_CODIGO	ASC
REF_841_FK	No	Yes	No	No	FPAG_CODIGO	ASC

Reference to List

Reference to	Primary Key	Foreign Key
BANCOS	BAN_CODIGO	BAN_CODIGO
FORMAS_PAGO	FPAG_CODIGO	FPAG_CODIGO
FACTURA_CLIENTE	FACCLI_CODIGO	FACCLI_CODIGO
BANCOS	BAN_CODIGO	BAN_CODIGO
FORMAS_PAGO	FPAG_CODIGO	FPAG_CODIGO

Table PARAMETROS

Name:	PARAMETROS
Code:	PARAMETROS
Label:	
Number:	
PK constraint:	
Source:	Entity PARAMETROS

Column List

Name	Code	Type	P	M
PAR_CODIGO	PAR_CODIGO	varchar(20)	Yes	Yes
PAR_VALOR	PAR_VALOR	numeric(10,2)	No	No
PAR_DETALLE	PAR_DETALLE	varchar(100)	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
PARAMETROS_PK	Yes	No	Yes	No	PAR_CODIGO	ASC

Table PRESUPUESTOS

Name:	PRESUPUESTOS
Code:	PRESUPUESTOS
Label:	
Number:	
PK constraint:	
Source:	Entity PRESUPUESTOS

Column List

Name	Code	Type	P	M
MT_CODIGO	MT_CODIGO	char(4)	Yes	Yes
PR_CODIGO	PR_CODIGO	numeric(4)	Yes	Yes
CLI_CODIGO	CLI_CODIGO	char(4)	No	No
PR_CANTIDAD	PR_CANTIDAD	numeric(3)	No	Yes
PR_PRECIO	PR_PRECIO	numeric(8,2)	No	Yes
PR_VALOR_IVA	PR_VALOR_IVA	numeric(8,2)	No	No
PR_TOTAL	PR_TOTAL	numeric(8,2)	No	Yes
PR_FECHA	PR_FECHA	timestamp	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
FK_PRESUPUESTO_MATERIALES_FK	No	Yes	No	No	MT_CODIGO	ASC
FK_PRESUPUESTOS_CLIENTES_FK	No	Yes	No	No	CLI_CODIGO	ASC

Reference to List

Reference to	Primary Key	Foreign Key
MATERIALES	MT_CODIGO	MT_CODIGO
CLIENTES	CLI_CODIGO	CLI_CODIGO

Table PROFORMA

Name:	PROFORMA
Code:	PROFORMA
Label:	
Number:	
PK constraint:	
Source:	

Column List

Name	Code	Type	P	M
PROF_CODIGO	PROF_CODIGO	numeric(5)	No	Yes
PROF_NOMBRE	PROF_NOMBRE	varchar(60)	No	No
PROF_RUC	PROF_RUC	varchar(13)	No	No
PROF_DETALLE	PROF_DETALLE	varchar(60)	No	No
PROF_OBSERVACION	PROF_OBSERVACION	varchar(100)	No	No
PROF_IMPRESA	PROF_IMPRESA	char(1)	No	No
PROF_FECHA	PROF_FECHA	datetime	No	No
PROF_VALOR_IVA	PROF_VALOR_IVA	numeric(8,2)	No	No
PROF_ESTADO	PROF_ESTADO	datetime	No	No

Check

Domain:					
Low Value:					
High Value:					
Default Value:					
Unit:					
Format:					
Uppercase:	No	**Lowercase:**	No	**Can't modify:**	No
List of Values:	S	SI			
	N	NO			

Reference by List

Referenced by	Primary Key	Foreign Key
DETALLE_PROFORMA_MUEBLES	PROF_CODIGO	PROF_CODIGO

Table PROVEEDORES

Name:	PROVEEDORES
Code:	PROVEEDORES
Label:	
Number:	
PK constraint:	
Source:	Entity PROVEEDORES

Column List

Name	Code	Type	P	M
PROVE_CODIGO	PROVE_CODIGO	char(4)	Yes	Yes
PROVE_RUC	PROVE_RUC	varchar(13)	No	No

Name	Code	Type	P	M
PROVE_NOMBRE	PROVE_NOMBRE	varchar(60)	No	Yes
PROVE_DIRECCION	PROVE_DIRECCION	varchar(60)	No	No
PROVE_FAX	PROVE_FAX	varchar(13)	No	No
PROVE_TELEFONO	PROVE_TELEFONO	varchar(13)	No	No
PROVE_EMAIL	PROVE_EMAIL	varchar(80)	No	No
PROVE_ESTADO	PROVE_ESTADO	timestamp	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
PROVEEDORES_PK	Yes	No	Yes	No	PROVE_CODIGO	ASC
FK_PROVEEDORES_CIUDAD_FK	No	Yes	No	No		

Reference by List

Referenced by	Primary Key	Foreign Key
MATERIALES	PROVE_CODIGO	PROVE_CODIGO

Table PROVINCIA

Name: PROVINCIA
Code: PROVINCIA
Label:
Number:
PK constraint:
Source: Entity PROVINCIA

Column List

Name	Code	Type	P	M
PROV_CODIGO	PROV_CODIGO	numeric(3)	Yes	Yes
PA_CODIGO	PA_CODIGO	numeric(6)	No	No
PROV_DETALLE	PROV_DETALLE	varchar(60)	No	No
PROV_ESTADO	PROV_ESTADO	timestamp	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
PROVINCIA_PK	Yes	No	Yes	No	PROV_CODIGO	ASC
REF_1073_FK	No	Yes	No	No	PA_CODIGO	ASC

Reference to List

Reference to	Primary Key	Foreign Key
PAIS	PA_CODIGO	PA_CODIGO

Reference by List

Referenced by	Primary Key	Foreign Key
CIUDAD	PROV_CODIGO	PROV_CODIGO

Table SUBCLASE_MUEBLES

Name:	SUBCLASE_MUEBLES
Code:	SUBCLASE_MUEBLES
Label:	
Number:	
PK constraint:	PK_SUBCLASE_MUEBLES
Source:	Entity SUBCLASE_MUEBLES

Options

with max_rows_per_page = 0

on default

Column List

Name	Code	Type	P	M
SCLM_CODIGO	SCLM_CODIGO	char(4)	Yes	Yes
CLM_CODIGO	CLM_CODIGO	char(4)	No	Yes
SCLM_DETALLE	SCLM_DETALLE	varchar(60)	No	Yes
SCLM_ESTADO	SCLM_ESTADO	datetime	No	No
SCLM_MUEBLE	SCLM_MUEBLE	image	No	No
SCLM_ARCHIVO	SCLM_ARCHIVO	varchar(40)	No	No

Index List

Index Code	P	F	U	C	Column Code	Sort
FK_SUBCLASEMUB_CLASEMUEBLE_FK	No	Yes	No	No	CLM_CODIGO	ASC
PK_SUBCLASE_MUEBLES	Yes	No	Yes	Yes	SCLM_CODIGO	ASC

Reference to List

Reference to	Primary Key	Foreign Key
CLASE_MUEBLES	CLM_CODIGO	CLM_CODIGO

Reference by List

Referenced by	Primary Key	Foreign Key
DETALLE_PROFORMA_MUEBLES	SCLM_CODIGO	SCLM_CODIGO

Table USUARIO_VITACORA

Name:	USUARIO_VITACORA
Code:	USUARIO_VITACORA
Label:	
Number:	
PK constraint:	
Source:	Entity USUARIO_VITACORA

Column List

Name	Code	Type	P	M
USR_USUARIO	USR_USUARIO	varchar(20)	No	Yes
UVI_FECHA_INGRESO	UVI_FECHA_INGRESO	datetime	Yes	Yes
UVI_FECHA_SALIDA	UVI_FECHA_SALIDA	datetime	Yes	Yes
USR_CODIGO	USR_CODIGO	varchar(15)	Yes	Yes

Index List

Index Code	P	F	U	C	Column Code	Sort
USUARIO_VITACORA_PK	No	No	Yes	No	USR_CODIGO	ASC
					UVI_FECHA_INGRESO	ASC
					UVI_FECHA_SALIDA	ASC
REF_1385_FK	No	Yes	No	No	USR_CODIGO	ASC

Reference to List

Reference to	Primary Key	Foreign Key
USUARIOS	USR_USUARIO	USR_USUARIO
USUARIOS	USR_CODIGO	USR_CODIGO

Table USUARIOS

Name:	USUARIOS
Code:	USUARIOS
Label:	
Number:	
PK constraint:	
Source:	Entity USUARIOS

Column List

Name	Code	Type	P	M
USR_USUARIO	USR_USUARIO	varchar(20)	No	Yes
USR_NOMBRE	USR_NOMBRE	varchar(60)	No	Yes
USR_CLAVE	USR_CLAVE	varchar(10)	No	No
USR_MODIF_CLAVE	USR_MODIF_CLAVE	varchar(1)	No	No
USR_CADUCA_CLAVE	USR_CADUCA_CLAVE	varchar(1)	No	No
USR_FECHA_INGRESO	USR_FECHA_INGRESO	datetime	No	Yes
USR_NUMERO_INTENTOS	USR_NUMERO_INTENTOS	numeric(1)	No	No
USR_TIPO_USUARIO	USR_TIPO_USUARIO	char(1)	No	No
USR_FECHA_EXPIRA	USR_FECHA_EXPIRA	char(10)	No	No
USR_CLAVE_BLANCO	USR_CLAVE_BLANCO	varchar(1)	No	No
USR_LONG_CLAVE	USR_LONG_CLAVE	numeric(2)	No	No
USR_BLOQUEO	USR_BLOQUEO	varchar(1)	No	No
USR_TIEMPO_BLOQUEO	USR_TIEMPO_BLOQUEO	numeric(2)	No	No
USR_ULTIMO_BLOQUEO	USR_ULTIMO_BLOQUEO	char(10)	No	No
USR_ACTIVO	USR_ACTIVO	varchar(1)	No	Yes
USR_REEST_CTA_BLOQ	USR_REEST_CTA_BLOQ	varchar(1)	No	No

Name	Code	Type	P	M
USR_ESTADO	USR_ESTADO	char(10)	No	No
USR_GRUPO	USR_GRUPO	varchar(20)	No	No
USR_CODIGO	USR_CODIGO	varchar(15)	Yes	Yes
USR_FCH_CADUCA_CLAVE	USR_FCH_CADUCA_CLAVE	timestamp	No	Yes

Check

Domain:
Low Value:
High Value:
Default Value: M
Unit:
Format:
Uppercase: Yes **Lowercase:** No **Can't modify:** No
List of Values:

A	ADMINISTRADOR
C	CONSULTA
M	USR_MODIFICA

Index List

Index Code	P	F	U	C	Column Code	Sort
USUARIOS_PK	Yes	No	Yes	No	USR_CODIGO	ASC

Reference by List

Referenced by	Primary Key	Foreign Key
USUARIOS_ACCESOS	USR_USUARIO	USR_USUARIO
USUARIO_VITACORA	USR_USUARIO	USR_USUARIO
USUARIOS_ACCESOS	USR_CODIGO	USR_CODIGO
USUARIO_VITACORA	USR_CODIGO	USR_CODIGO

Table USUARIOS_ACCESOS

Name:	USUARIOS_ACCESOS
Code:	USUARIOS_ACCESOS
Label:	
Number:	
PK constraint:	
Source:	Entity USUARIOS_ACCESOS

Column List

Name	Code	Type	P	M
USR_USUARIO	USR_USUARIO	varchar(20)	No	Yes
UAC_OPC_MENU	UAC_OPC_MENU	varchar(100)	Yes	Yes
UAC_ACTIVO	UAC_ACTIVO	char(1)	No	No
UAC_NUEVO	UAC_NUEVO	char(1)	No	No
UAC_EDITAR	UAC_EDITAR	char(1)	No	No
UAC_BORRAR	UAC_BORRAR	char(1)	No	No
UAC_IMPRIMIR	UAC_IMPRIMIR	char(1)	No	No
USR_CODIGO	USR_CODIGO	varchar(15)	Yes	Yes

Index List

Index Code	P	F	U	C	Column Code	Sort
USUARIOS_ACCESOS_PK	No	No	Yes	No	USR_CODIGO	ASC
					UAC_OPC_MENU	ASC
REF_1378_FK	No	Yes	No	No	USR_CODIGO	ASC

Reference to List

Reference to	Primary Key	Foreign Key
USUARIOS	USR_USUARIO	USR_USUARIO
USUARIOS	USR_CODIGO	USR_CODIGO

Trigger List

Table	Trigger	User-Defined
CIUDAD	tib_ciudad	No
CLIENTES	tib_clientes	No
DETALLE_FACTURA_MUEBLES	tib_detalle_factura_muebles	No
DETALLE_PROFORMA_MUEBLES	tib_detalle_proforma_muebles	No
FACTURA_CLIENTE	tib_factura_cliente	No
MATERIALES	tib_materiales	No
PAGOS_FACTURA	tib_pagos_factura	No
PRESUPUESTOS	tib_presupuestos	No
PROVINCIA	tib_provincia	No
SUBCLASE_MUEBLES	tib_subclase_muebles	No
USUARIO_VITACORA	tib_usuario_vitacora	No
USUARIOS_ACCESOS	tib_usuarios_accesos	No

3.4.2 CODIFICACIÓN DE TRIGGER

Trigger tib_ciudad

Name:	CIUDAD	
Code:	CIUDAD	
Trigger:	tib_ciudad	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_ciudad" for table "CIUDAD"
create trigger tib_ciudad before insert on CIUDAD
referencing new as new_ins for each row
begin
   declare user_defined_exception exception for SQLSTATE '99999';
   declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
```

```
  declare found integer;

  % Parent "PROVINCIA" must exist when inserting a child in "CIUDAD"
  if (new_ins.PROV_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
    where exists (select 1
                  from PROVINCIA
                 where PROV_CODIGO = new_ins.PROV_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
   end
   end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_ciudad) of table CIUDAD';
  message '       Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_ciudad) of  table CIUDAD';
  resignal;
end
/
```

Trigger tib_clientes

Name:	CLIENTES	
Code:	CLIENTES	
Trigger:	tib_clientes	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_clientes" for table "CLIENTES"
create trigger tib_clientes before insert on CLIENTES
referencing new as new_ins for each row
begin
  declare user_defined_exception exception for SQLSTATE '99999';
  declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
  declare found integer;

  % Parent "CIUDAD" must exist when inserting a child in "CLIENTES"
  if (new_ins.CIU_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into  found
     from  dummy
    where  exists (select 1
                from  CIUDAD
               where  CIU_CODIGO = new_ins.CIU_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_clientes) of table CLIENTES';
  message '       Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_clientes) of  table CLIENTES';
  resignal;
```

```
end
/
```

Trigger tib_detalle_factura_muebles

Name:	DETALLE_FACTURA_MUEBLES	
Code:	DETALLE_FACTURA_MUEBLES	
Trigger:	tib_detalle_factura_muebles	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_detalle_factura_muebles" for table
"DETALLE_FACTURA_MUEBLES"
create trigger tib_detalle_factura_muebles before insert on
DETALLE_FACTURA_MUEBLES
referencing new as new_ins for each row
begin
  declare user_defined_exception exception for SQLSTATE '99999';
  declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
  declare found integer;

  % Parent "DETALLE_PROFORMA_MUEBLES" must exist when inserting a child
in "DETALLE_FACTURA_MUEBLES"
  if (new_ins.PROF_CODIGO is not null and
    new_ins.PR_CODIGO is not null and
    new_ins.SCLM_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
     where exists (select 1
                from DETALLE_PROFORMA_MUEBLES
                where PROF_CODIGO = new_ins.PROF_CODIGO
```

```
            and   PR_CODIGO = new_ins.PR_CODIGO
            and   SCLM_CODIGO = new_ins.SCLM_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_detalle_factura_muebles) of table
DETALLE_FACTURA_MUEBLES';
  message '      Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_detalle_factura_muebles) of  table
DETALLE_FACTURA_MUEBLES';
  resignal;
end
/
```

Trigger tib_detalle_proforma_muebles

Name:	DETALLE_PROFORMA_MUEBLES	
Code:	DETALLE_PROFORMA_MUEBLES	
Trigger:	tib_detalle_proforma_muebles	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_detalle_proforma_muebles" for table
"DETALLE_PROFORMA_MUEBLES"
create trigger tib_detalle_proforma_muebles before insert on
DETALLE_PROFORMA_MUEBLES
```

```
referencing new as new_ins for each row
begin
   declare user_defined_exception exception for SQLSTATE '99999';
   declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
   declare found integer;

   % Parent "PROFORMA" must exist when inserting a child in
"DETALLE_PROFORMA_MUEBLES"
   if (new_ins.PROF_CODIGO is not null) then
   begin
     set found = 0;
     select 1
      into found
      from dummy
     where exists (select 1
                 from PROFORMA
                where PROF_CODIGO = new_ins.PROF_CODIGO);
     if found <> 1 then
       signal insert_child_parent_exist_exception
     end if;
   end
   end if;

   % Parent "SUBCLASE_MUEBLES" must exist when inserting a child in
"DETALLE_PROFORMA_MUEBLES"
   if (new_ins.SCLM_CODIGO is not null) then
   begin
     set found = 0;
     select 1
      into found
      from dummy
     where exists (select 1
                 from SUBCLASE_MUEBLES
                where SCLM_CODIGO = new_ins.SCLM_CODIGO);
```

```
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_detalle_proforma_muebles) of table
DETALLE_PROFORMA_MUEBLES';
  message '      Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_detalle_proforma_muebles) of  table
DETALLE_PROFORMA_MUEBLES';
  resignal;
end
/
```

Trigger tib_factura_cliente

Name:	FACTURA_CLIENTE	
Code:	FACTURA_CLIENTE	
Trigger:	tib_factura_cliente	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_factura_cliente" for table "FACTURA_CLIENTE"
create trigger tib_factura_cliente before insert on FACTURA_CLIENTE
referencing new as new_ins for each row
begin
  declare user_defined_exception exception for SQLSTATE '99999';
  declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
  declare found integer;
```

```
  % Parent "AUTORIZACION" must exist when inserting a child in
"FACTURA_CLIENTE"
  if (new_ins.AU_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
    where exists (select 1
                  from AUTORIZACION
                 where AU_CODIGO = new_ins.AU_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;

  % Parent "CLIENTES" must exist when inserting a child in
 "FACTURA_CLIENTE"
  if (new_ins.CLI_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
    where exists (select 1
                  from CLIENTES
                 where CLI_CODIGO = new_ins.CLI_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
 exception
```

```
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_factura_cliente) of table FACTURA_CLIENTE';
  message '      Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_factura_cliente) of  table
FACTURA_CLIENTE';
  resignal;
end
/
```

Trigger tib_materiales

Name:	MATERIALES	
Code:	MATERIALES	
Trigger:	tib_materiales	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_materiales" for table "MATERIALES"
create trigger tib_materiales before insert on MATERIALES
referencing new as new_ins for each row
begin
   declare user_defined_exception exception for SQLSTATE '99999';
   declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
   declare found integer;

   % Parent "PROVEEDORES" must exist when inserting a child in "MATERIALES"
   if (new_ins.PROVE_CODIGO is not null) then
   begin
     set found = 0;
     select 1
      into  found
```

```
    from dummy
    where exists (select 1
                  from PROVEEDORES
                  where PROVE_CODIGO = new_ins.PROVE_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_materiales) of table MATERIALES';
  message '      Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_materiales) of  table MATERIALES';
  resignal;
end
/
```

Trigger tib_pagos_factura

Name:	PAGOS_FACTURA	
Code:	PAGOS_FACTURA	
Trigger:	tib_pagos_factura	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_pagos_factura" for table "PAGOS_FACTURA"
create trigger tib_pagos_factura before insert on PAGOS_FACTURA
referencing new as new_ins for each row
begin
   declare user_defined_exception exception for SQLSTATE '99999';
   declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
```

```
declare found integer;

% Parent "FACTURA_CLIENTE" must exist when inserting a child in
"PAGOS_FACTURA"
if (new_ins.FACCLI_CODIGO is not null) then
begin
  set found = 0;
  select 1
   into found
   from dummy
  where exists (select 1
                from FACTURA_CLIENTE
                where FACCLI_CODIGO = new_ins.FACCLI_CODIGO);
  if found <> 1 then
    signal insert_child_parent_exist_exception
  end if;
end
end if;

% Parent "BANCOS" must exist when inserting a child in "PAGOS_FACTURA"
if (new_ins.BAN_CODIGO is not null) then
begin
  set found = 0;
  select 1
   into found
   from dummy
  where exists (select 1
                from BANCOS
                where BAN_CODIGO = new_ins.BAN_CODIGO);
  if found <> 1 then
    signal insert_child_parent_exist_exception
  end if;
end
end if;
```

```
  % Parent "FORMAS_PAGO" must exist when inserting a child in
"PAGOS_FACTURA"
  if (new_ins.FPAG_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
    where exists (select 1
                  from FORMAS_PAGO
                 where FPAG_CODIGO = new_ins.FPAG_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;

  % Parent "BANCOS" must exist when inserting a child in "PAGOS_FACTURA"
  if (new_ins.BAN_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
    where exists (select 1
                  from BANCOS
                 where BAN_CODIGO = new_ins.BAN_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
```

```
  % Parent "FORMAS_PAGO" must exist when inserting a child in
"PAGOS_FACTURA"
  if (new_ins.FPAG_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into  found
     from  dummy
    where  exists (select 1
                from  FORMAS_PAGO
               where  FPAG_CODIGO = new_ins.FPAG_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
   end
   end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_pagos_factura) of table PAGOS_FACTURA';
  message '       Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_pagos_factura) of  table
PAGOS_FACTURA';
  resignal;
end
/
```

Trigger tib_presupuestos

Name:	PRESUPUESTOS	
Code:	PRESUPUESTOS	
Trigger:	tib_presupuestos	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_presupuestos" for table "PRESUPUESTOS"
create trigger tib_presupuestos before insert on PRESUPUESTOS
referencing new as new_ins for each row
begin
  declare user_defined_exception exception for SQLSTATE '99999';
  declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
  declare found integer;

  % Parent "MATERIALES" must exist when inserting a child in "PRESUPUESTOS"
  if (new_ins.MT_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
    where exists (select 1
                  from MATERIALES
                 where MT_CODIGO = new_ins.MT_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;

  % Parent "CLIENTES" must exist when inserting a child in "PRESUPUESTOS"
  if (new_ins.CLI_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
```

```
    where  exists (select 1
                from  CLIENTES
                where  CLI_CODIGO = new_ins.CLI_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_presupuestos) of table PRESUPUESTOS';
  message '       Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_presupuestos) of  table
PRESUPUESTOS';
  resignal;
end
/
```

Trigger tib_provincia

Name:	PROVINCIA	
Code:	PROVINCIA	
Trigger:	tib_provincia	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_provincia" for table "PROVINCIA"
create trigger tib_provincia before insert on PROVINCIA
referencing new as new_ins for each row
begin
   declare user_defined_exception exception for SQLSTATE '99999';
   declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
```

```
declare found integer;

% Parent "PAIS" must exist when inserting a child in "PROVINCIA"
if (new_ins.PA_CODIGO is not null) then
begin
  set found = 0;
  select 1
   into found
   from dummy
  where exists (select 1
                from PAIS
               where PA_CODIGO = new_ins.PA_CODIGO);
  if found <> 1 then
    signal insert_child_parent_exist_exception
  end if;
 end
 end if;
exception
when insert_child_parent_exist_exception then
 message 'Error: Trigger(tib_provincia) of table PROVINCIA';
 message '      Parent code must exist when inserting a child!';
 signal user_defined_exception;
when others then
 message 'Exception in before insert trigger(tib_provincia) of table PROVINCIA';
 resignal;
end
/
```

Trigger tib_subclase_muebles

Name:	SUBCLASE_MUEBLES	
Code:	SUBCLASE_MUEBLES	
Trigger:	tib_subclase_muebles	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_subclase_muebles" for table "SUBCLASE_MUEBLES"
create trigger tib_subclase_muebles before insert on SUBCLASE_MUEBLES
referencing new as new_ins for each row
begin
  declare user_defined_exception exception for SQLSTATE '99999';
  declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
  declare found integer;
  % Parent "CLASE_MUEBLES" must exist when inserting a child in
"SUBCLASE_MUEBLES"
  if (new_ins.CLM_CODIGO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
    where exists (select 1
                  from CLASE_MUEBLES
                 where CLM_CODIGO = new_ins.CLM_CODIGO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_subclase_muebles) of table SUBCLASE_MUEBLES';
  message '      Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_subclase_muebles) of table
SUBCLASE_MUEBLES';
  resignal;
```

```
end
/
```

Trigger tib_usuario_vitacora

Name:	USUARIO_VITACORA	
Code:	USUARIO_VITACORA	
Trigger:	tib_usuario_vitacora	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
% Before insert trigger "tib_usuario_vitacora" for table "USUARIO_VITACORA"
create trigger tib_usuario_vitacora before insert on USUARIO_VITACORA
referencing new as new_ins for each row
begin
   declare user_defined_exception exception for SQLSTATE '99999';
   declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
   declare found integer;

   % Parent "USUARIOS" must exist when inserting a child in
"USUARIO_VITACORA"
   if (new_ins.USR_CODIGO is not null) then
   begin
     set found = 0;
     select 1
      into found
      from dummy
     where exists (select 1
                 from USUARIOS
                where USR_CODIGO = new_ins.USR_CODIGO);
     if found <> 1 then
       signal insert_child_parent_exist_exception
     end if;
   end
   end if;
```

```
  % Parent "USUARIOS" must exist when inserting a child in
"USUARIO_VITACORA"
  if (new_ins.USR_USUARIO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
    where exists (select 1
                  from USUARIOS
                 where USR_USUARIO = new_ins.USR_USUARIO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_usuario_vitacora) of table USUARIO_VITACORA';
  message '       Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_usuario_vitacora) of  table
USUARIO_VITACORA';
  resignal;
end
/
```

Trigger tib_usuarios_accesos

Name:	USUARIOS_ACCESOS	
Code:	USUARIOS_ACCESOS	
Trigger:	tib_usuarios_accesos	
Type:	BeforeInsertTrigger	Default Trigger

Trigger Code

```
        % Before insert trigger "tib_usuarios_accesos" for table
"USUARIOS_ACCESOS"
create trigger tib_usuarios_accesos before insert on USUARIOS_ACCESOS
referencing new as new_ins for each row
begin
   declare user_defined_exception exception for SQLSTATE '99999';
   declare insert_child_parent_exist_exception exception for SQLSTATE '99991';
   declare found integer;

   % Parent "USUARIOS" must exist when inserting a child in
"USUARIOS_ACCESOS"
   if (new_ins.USR_CODIGO is not null) then
   begin
     set found = 0;
     select 1
      into  found
      from  dummy
     where  exists (select 1
                 from  USUARIOS
                where  USR_CODIGO = new_ins.USR_CODIGO);
     if found <> 1 then
       signal insert_child_parent_exist_exception
     end if;
   end
```

```
  end if;

  % Parent "USUARIOS" must exist when inserting a child in
"USUARIOS_ACCESOS"
  if (new_ins.USR_USUARIO is not null) then
  begin
    set found = 0;
    select 1
     into found
     from dummy
    where exists (select 1
                  from USUARIOS
                  where USR_USUARIO = new_ins.USR_USUARIO);
    if found <> 1 then
      signal insert_child_parent_exist_exception
    end if;
  end
  end if;
exception
when insert_child_parent_exist_exception then
  message 'Error: Trigger(tib_usuarios_accesos) of table USUARIOS_ACCESOS';
  message '       Parent code must exist when inserting a child!';
  signal user_defined_exception;
when others then
  message 'Exception in before insert trigger(tib_usuarios_accesos) of table
USUARIOS_ACCESOS';
  resignal;
end
/
```

3.5 IMPLANTACIÓN DEL SISTEMA

3.5.1 INSTALACIÓN DEL SOFTWARE

La instalación del Software, se la ha realizado, a través de herramientas que permiten comprimir los archivos de aplicación, así como crear todas las variables y realizar todo el proceso de instalación, en el equipo computacional en el cual, se desea implementar, para el correcto funcionamiento de la Aplicación, en cualquier equipo, que este sobre una plataforma Windows.

Por medio del lenguaje de Programación, se ha construido un ejecutable, que posee, todo el código fuente convertido en código máquina. A este ejecutable, se han anexado, las librerías del lenguaje, que son necesarios para su funcionamiento, la base de datos, así como sus controladores del motor de base (en el caso de servidor), y los demás iconos y archivos gráficos, que complementan la aplicación.

En el proceso, de instalación, se deberá especificar, el Nombre y la Empresa, en la cual se instalará, se requerirá de una contraseña, con lo que se pretende, evitar instalaciones indebidas, se deberá, especificar la ruta, así como el tipo de instalación que se desea realizar, además se podrá especificar si se desea crear acceso en el escritorio o cambiar el nombre de cómo se presenta la aplicación en el menú inicio.

Una vez ingresados todos estos parámetros, se procederá a la instalación de la aplicación, en la que de manera automática, se creara entradas en el registro del Sistema Operativo, con lo cual se podrá registrar los controladores de la aplicación y de

la base de datos, permitiendo un correcto funcionamiento de la aplicación en cualquier equipo computacional en la cual resida.

CAPITULO IV

4.1 VALIDACIÓN DE RESULTADOS

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
Sede Ambato

ESCUELA DE INGENIERIA DE SISTEMAS

Ambato, 04 de enero de 2007

Ingeniero
Telmo Viteri
DIRECTOR DE LA ESCUELA DE SISTEMAS DE LA PUCESA
Presente.

De mi consideración:

La presente es portadora de un saludo cordial y a la vez informarle que se ha procedido a la validación del trabajo de disertación de la señorita: Gioconda Vanesa Lara Peralta, titulado: "Sistema para el diseño de muebles modulares y la obtención de presupuesto del mismo". Encontrando que el mencionado trabajo esta concluido a cabalidad, cumpliendo los objetivos trazados y funcionando plenamente.
Se destaca en el trabajo la utilidad y facilidad que presta el mencionado sistema para los usuarios, puesto que es un sistema a medida.

Atentamente,

Ing. Santiago Acurio M.
COORDINADOR ACADÉMICO
ESCUELA DE INGENIERIA DE SISTEMAS

Ambato, 04 de enero de 2007

Ingeniero
Telmo Viteri
DIRECTOR DE LA ESCUELA DE SISTEMAS DE LA PUCESA
Presente.

De mi consideración:

Por medio de la presente me permito informar a usted que he realizado validación del trabajo de disertación de la señorita: Gioconda Vanesa Lara Peralta, titulado: "Sistema para el diseño de muebles modulares y la obtención de presupuesto del mismo".

Puedo informar que el sistema presenta ventajas y facilidades en el uso, así como un orden lógico de procesos

Atentamente,

Ing. Paúl Zurita
DOCENTE
ESCUELA DE INGENIERIA DE SISTEMAS

Ambato 04 de Enero de 2007

Ing. Telmo Viteri

Director de la Escuela de Ingeniería de Sistemas de la PUCESA

De mis consideraciones

Yo Juan Carlos Cruz Lascano Gerente propietario de la Empresa CRUMAD de RUC 1803069598001 cumplo con informarle que el sistema SIDIMMOP, "Sistema para el Diseño de Muebles Modulares y la Obtención del Presupuesto del mismo" realizado para mi empresa a la cual represento por la Señora Gioconda Vanessa Lara Peralta cumple con los objetivos que nos planteamos ya que ha sido de gran ayuda para poder lograr automatizar los procesos que los veníamos haciendo manualmente. Encontrando que el sistema esta concluido y funcionando plenamente.

De esta manera pongo en su conocimiento para los fines pertinentes.

Por la atención que se digne a la presente anticipo mis agradecimientos.

Atentamente;

crumad

Juan Carlos Cruz Lascano

C.I. 180306959-8

GERENTE

CAPITULO V

5.1 CONCLUSIONES

- Con el presente sistema se ha llegado a determinar que cuando se mejoran los procesos con una herramienta nueva y poderosa como es Power Builder 9.0 y una base de datos robusta como es Sybase 11.5, se puede mejorar en gran forma el manejo de la información, manteniendo como objetivo principal la seguridad de datos.

- El sistema simplificará el trabajo del usuario del sistema además dándole al cliente la facilidad de obtener su presupuesto y diseño del mueble modular que necesite en forma inmediata y precisa.

- Es sistema proporcionará a la empresa una ventaja frente a la atención a los clientes, a medida que vaya creciendo la demanda, este soportará el crecimiento que vaya dando al pasar el tiempo.

- Los reportes que se generen darán como resultado un mejor control del movimiento económico de la empresa y permitirá establecer parámetros para determinar si los procedimientos realizados están dando los frutos deseados.

5.2 RECOMENDACIONES

- Se recomienda verificar el funcionamiento de los equipos que forman parte integral del sistema, ya sea este en la parte de Hardware como de Software, con el fin de evitar problemas en el momento de implantar el sistema.

- Se recomienda que en los equipos que se utilizarán el sistema, se instale solo el Software necesario, con el único fin de mejorar el rendimiento del mismo.

- Si al momento de la utilización del sistema ocurre algún tipo de error que se sospeche que sea el Software, comunicarse de inmediato con el personal de

desarrollo, con el único fin de solucionar dicho problema y verificar la integridad de información ingresada.

- Para brindar una mejor atención al cliente, se debería utilizar impresoras de calidad y velocidad, ya que esto permitirá agilizar la atención al cliente.

ÍNDICE GENERAL

INTRODUCCIÓN 1
CAPITULO I 2
1. PROYECTO DE INVESTIGACIÓN 2
1.1. PLANTEAMIENTO DEL PROBLEMA 2
1.1.1. PROBLEMA 2
1.1.2. PROBLEMATIZACIÓN 2
1.2. DELIMITACIÓN 2
1.3. JUSTIFICACIÓN E IMPORTANCIA 3
1.4. OBJETIVOS 3
1.4.1. OBJETIVO GENERAL 3
1.4.2. OBJETIVOS ESPECÍFICOS 3
1.5. HIPÓTESIS 4
1.6. ASPECTOS METODOLÓGICOS 4
1.6.1. FUNDAMENTOS TEÓRICOS 4
1.6.2. MÉTODOS DE INVESTIGACIÓN 4

CAPITULO II 6
2. MARCO TEORICO 6
2.1 SISTEMA 6
2.1.1 CONCEPTO DE SISTEMA 6
2.1.2 SISTEMA INFORMATICO 7
2.2 BASES DE DATOS 8
2.2.1 MODELOS DE BASE DE DATOS 8
2.2.2 GESTOR DE BASE DE DATOS. 9
2.2.3 BASES DE DATOS RELACIONALES 10
2.3 HERRAMIENTAS DE BASE DE DATOS. 11
2.3.1. SYBASE ASE. 11
2.4 HERRAMIENTAS DE ANÁLISIS Y DISEÑO DEL SISTEMA. 12
2.4.1. POWER DESIGNER 12
2.4.1.1. MODELAMIENTO DE DATOS: 13
2.4.1.2. MODELAMIENTO DE OBJETOS: 13
2.4.1.3. REPOSITORIO EMPRESARIAL: 13

2.5 LENGUAJES DE PROGRAMACIÓN. 13
2.5.1. TIPOS DE LENGUAJES 14
2.5.1.1. LENGUAJES DE CUARTA GENERACIÓN. 14
2.6 POWER BUILDER 9.0 15
2.6.1 APLICACIÓN POWERBUILDER. 16
2.6.2 LENGUAJE POWERSCRIPT 16
2.7 ARQUITECTURA CLIENTE-SERVIDOR 17
2.7.1. VENTAJAS DE LA ARQUITECTURA CLIENTE-SERVIDOR 19
2.7.2. HARDWARE Y SOFTWARE CLIENTE-SERVIDOR. 19
2.8 TCP/IP 20
2.8.1. TRANSPORTE Y APLICACIÓN. 20
2.9 REDES LAN 22
2.9.1. BENEFICIOS DE UNA RED DE DATOS 23
2.10 SISTEMAS OPERATIVOS 23
2.10.1. MICROSOFT WINDOWS 2000 PROFESSIONAL. 23
2.11 DISEÑO DE MUEBLES MODULARES 25
2.11.1 TEORÍA DE LA FABRICACIÓN DE MUEBLES 25
2.11.2 TEORÍA DEL DISEÑO DE MUEBLES 26
2.12 CALCULO DE PRESUPUESTOS 29
2.12.1 PRESUPUESTO 29
2.12.2 PRESUPUESTO MAESTRO 30
2.12.3 PRESUPUESTO DE VENTAS 31
2.12.4 PRESUPUESTO DE PRODUCCION 32
2.12.4.1 PRESUPUESTO DE PRODUCCION POR LINEA 32
2.12.5 PRESUPUESTO DE NECESIDADES DE MATERIA PRIMA Y DE COMPRAS. 33
2.12.6 PRESUPUESTO DE MANO DE OBRA DIRECTA. 34
2.12.7 PRESUPUESTO DE GASTOS INDIRECTOS DE FABRICACIÓN. 34
2.12.8 ESTADO DE RESULTADOS PRESUPUESTADO. 35
2.12.9 PRESUPUESTO FINANCIERO. 36
3. SISTEMA PARA EL DISEÑO DE MUEBLES MODULARES Y LA OBTENCIÓN DE PRESUPUESTO DEL MISMO. 37
3.2 ANALISIS DEL SISTEMA 38
3.2.1 OBJETIVOS 38

3.2.2 ALCANCE 38
3.2.3 VISIÓN GENERAL 40
3.2.4 Perspectiva 40
3.2.5 Funciones 40
3.2.6 Características de usuario 42
3.2.7 Restricciones 43
3.2.8 Suposiciones y Dependencias 43
3.2.9 Requisitos Específicos 44
3.2.10 Obtención de los requerimientos del Usuario. 47
3.3.1 Introducción 49
3.3.2 Descripción de cada uno de los procesos 49
3.3.3 Diagrama de Flujo de Datos (DFD's) 55
3.3.4 Diseño de la Base de Datos 60
3.3.5 Diagrama de Objetos de Power Builder usados en la Aplicación 62
3.4.1 Desarrollo de Base de Datos 62
3.4.2 Codificación de TRIGGER 88
3.5.1 Instalación del Software 109

CAPITULO IV 110
4.1 VALIDACIÓN DE RESULTADOS 110

CAPITULO V 111
5.1 CONCLUSIONES 111
5.2 RECOMENDACIONES 111
GLOSARIO
BIBLIOGRAFIA
ANEXOS
ÍNDICE GENERAL 113

ÍNDICE DE ILUSTRACIONES

Fig 2.1 22
Figura 2 Diseño de Interfaz Externo 44
Figura 5: DFD Nivel 0 del Sistema SIDIMMOP 55
Figura 6: DFD Nivel 1 del Sistema SIDIMMOP 56
Figura 7: DFD Nivel 1 de creación de CLIENTES 57
Figura 8: DFD Nivel 1 de creación de MATERIALES 58
Figura 9: DFD Nivel 1 de creación de PROFORMAS 59
Figura 10: DFD Nivel 1 de creación de FACTURACIÓN 60

ÍNDICE DE TABLAS

TABLA AUTORIZACIÓN 63
TABLA BANCOS 64
TABLA CIUDAD 65
TABLA CLASE_MUEBLES 66
TABLA CLIENTES 67
TABLA DETALLE_FACTURA_MUEBLES 68
TABLA DETALLE_PROFORMA_MUEBLES 69
TABLA FACTURA_CLIENTE 70
TABLA FORMAS_PAGO 72
TABLA MATERIALES 73
TABLA PAGOS_FACTURA 74
TABLA PAIS 76
TABLA PARAMETROS 77
TABLA PRESUPUESTOS 77
TABLA PROFORMA 79
TABLA PROVEEDORES 80
TABLA PROVINCIA 81
TABLA SUBCLASE_MUEBLES 82
TABLA USUARIO_VITACORA 84
TABLA USUARIOS 85
TABLA USUARIOS_ACCESOS 87

GLOSARIO

Asíncrona.- Se aplica este término a una transmisión de datos en serie que no se producen de una manera síncrona, se trata de una técnica que permite la máxima sencillez del circuito de interconexión a expensas de una menor velocidad de transmisión.

Atributos.- Es una unidad básica e indivisible de información acerca de una entidad o relación que sirve para identificar y describir a los mismos.

Autenticación.- Verificar la identidad del cliente.

Automación.- Verificar si el cliente tiene acceso al servicio proporcionado por el servidor.

Cliente.- Funciona en su ordenador local, se comunica con el servidor remoto, y pide a éste información.

DBMS.- Sistema de manipulación de base de datos.

Dispositivo Periférico.- Dispositivo o máquina que está en conexión con la unidad central que trabaja bajo el control de esta última. Los periféricos desarrollan la función de proporcionar datos y programas a la unidad central describir datos a la salida y de almacenar ficheros.

Drivers.- (controlador) termino ingles que suele utilizarse para aquella parte del software del sistema, que controla un periférico particular. La CPU para comunicarse con un periférico tiene necesidad tanto de las conexiones físicas como también de un programa de gestión adecuado que se suele denominar "Driver". Los programas "drivers" forman parte integrante del Sistema Operativo.

Encapsulamiento.- Es también una forma de ocultar información, en la que una clase esconde a la vista sus detalles internos de aplicación.

Estática.- Propiedades invariantes en el tiempo Ej. Entidades, atributos, dominios, relaciones, representaciones, restricciones inherentes.

FTP.- (File Trasnfer Protocol), protocolo de transferencia de archivos, permite que los usuarios transfieran archivos entre host.

Herencia.- Permite definir nuevas clases (descendientes) desde las clases existentes (antecedentes). Una clase descendiente heredas todas las propiedades y métodos definidos para su clase antecedente.

Ingeniería Inversa.- Conjunto de tareas destinadas a obtener las especificaciones de un sistema de información, partiendo del propio sistema. Es una actividad típica del mantenimiento de aplicaciones cuando no existen las especificaciones de diseño de la aplicación a mantener.

Interacción.- Se aplica este término a un sistema que permite ejecutar programas en forma de diálogo con el usuario.

Interoperabilidad.- Los equipos de distintos fabricantes pudieran funcionar conjuntamente, el desarrollo sería más eficiente y se fomentaría la competitividad entre proveedores.

LAN.- (Red de Área Local), red Network de computadores ubicada usualmente en una sola oficina o en un edificio que permite a varios usuarios compartir los archivos y otros recursos (como impresoras) y posibilita la interoperabilidad entre diversos sistemas.

MAN.- (Red de Área Metropolitana), red pública de alta velocidad (100 bmps) capaz de transmitir voz y datos a larga distancia.

Migración.- Actualiza el sistema operativo del servidor y transfiere objetos Bendery, derechos y asignaciones administrativas.

ODBC.- Controlador de conectividad abierta de bases de datos, permite la conectividad de la base de datos con las aplicaciones.

Pipe.- Sección de memoria que puede ser usada por un comando para pasar información a un segundo comando en el procesamiento.

Polimorfismo.- Es la cualidad que tiene una función para trabajar de diferentes formas ante una misma llamada.

Privacidad.- Garantizar que la información privada de un usuario, no sea accedida por alguien no autorizado.

Procesos.- Conjunto de operaciones desarrolladas por uno o varios procesadores, para ejecutar un programa.

Protección.- Asegurar que las aplicaciones no monopolicen los recursos del sistema.

RAD.- Rapid Aplication Development tecnología de desarrollo rápido de aplicaciones.

Seguridad de datos: para que estos no puedan ser accedidos inapropiadamente.

Servidor.- Es un ordenador remoto en algún lugar de la red que proporciona información según petición.

Síncrona.- Este término se aplica a cualquier término que funcione al mismo tiempo, por Ej. Dos dispositivos deben hacer algo juntos es necesario que su funcionamiento se realice simultáneamente de forma muy precisa. El reloj (clock) del ordenador sirve para sincronizar entre sí las operaciones de las diversas partes.

SMTP.- (simple mail transfer protocol), Protocolo de transferencia utilizado para implementar sistemas de correo electrónico.

TCP/IP.- Siglas de Transmiter Control Protocol / Internet Protocol, conjunto de protocolos de comunicación. Abarca acceso al medio transporte de paquetes, comunicaciones de sesión, transferencia de archivos de correo electrónico y emulación terminal.

TELNET.- Protocolo de emulación de terminal remoto y permite a los clientes se conecten a host remotos de la red.

UDP.- (User Datagram Protocol), protocolo inalámbrico a nivel de transporte que se usa en el grupo de protocolos TCP/IP generalmente unido a al capa de software IP, debido a que UDP no agrega sobrecarga, como lo hace la conexión orientada TCP/UDP se emplea a menudo con aplicaciones SIMMP(Simple Netware Directory Services IMDS).

WAN.- (Red de Área Amplia), se refiere a una red network de computador que usa líneas telefónicas dedicadas y/o satélites para interconectar redes LAN a través de largas distancias geográficas hasta miles de millones de Km de separación.

BIBLIOGRAFÍA

- DORFMAN, M. y THAYER, R..- Software Engineering.- New Jersey: Ed. IEEE Computer Society Press,1997.- pp. 148-156.
- DOUGLAS, E..- Redes de Computadoras; Internet e Interredes.- Madrid: Ed. Prentice Hall, 1999. - pp. 120-130, 200,212.
- PRESSMAN, R..- Ingeniería del Software; Un enfoque práctico.- Trad por: Carlos Cervigon y Luis Hernández.- 3era Ed.- Madrid: Ed. McGrawHill, 1993.- pp. 251-273.
- SYBASE, Inc.- Connecting to your Database.- New York: Ed. IEEE Computer Society Press, 1996.- 551p.
- UMAR, A..- Object-Oriented Client/Server Internet Environments.- New Jersey: Ed. Prentice Hall PTR,1997.-pp. 6-55,146,159,398-420.

- Sybase, Inc.- Configuring Sybase Adaptive Server 'Enterprise' for Windows NT. Sybase 1997, 2450p.
- PowerBuilder 5.0. Objects and Controls, United States of America, May 1996, 345 p.
- SYBASE , Inc.- Data Window Reference, PowerSoft 1996, 88p.
- SYBASE , Inc.- Objects and Controls, PowerSoft 1996, 497p.
- SYBASE, Inc.- Enterprise System Administrative Guide, PowerSoft 1997, 875p.
- SYBASE, Inc.- Enterprise Transact-SQL User's Guide, PowerSoft 1997. 889p.
- SYBASE, Inc.- Enterprise Reference Manual Data Type and System Table, PowerSoft 1997, 765p.
- SYBASE, Inc.- Commands and Functions, PowerSoft 1997, 875p.

DIRECCIONES DE INTERNET

- Ing. Aurelio Ramírez García. Power Builder 6.0
  http://www.baja.gob.mx/organización/dgi/biblioteca/ci/ci10/art_5.html
- Consisa. Controles Avanzados de Power Builder.
  http://www.consisa.com.sv/cad312.asp
- The Learning Center. Características de Power Builder 5.0.
  http://www.tutorialusa.com/training/power-builder-tutorial.htm
- Cliff Berg. El estado del middleware en aplicaciones Java.
  http://www.javaworld.com/javaworld/jw-03-1999/jw-03-middleware.html
- Núcleo la Hechicera. Orientación por Objetos.
  http://www.dsia.adm.ula.ve/Orientacin%20por%20objetos.htm
- Monografías. Sistemas de Procesamiento de Datos Programación Orientada a Objetos.
  http://www.monografias.com/trabajos/objetos/objetos.shtml
- Cristofol Rovira. La orientación a objetos en el diseño de hipertextos para la enseñanza-aprendizaje.
  http://www.ucm.es/info/multidoc/revista/num8/rovira.html
- Ing. Manuel Espinoza Curiel. Sistema Operativo para Redes C/S.
  http://www.geocities.com/SiliconValley/8195/noscs.html
- Pedro M. Cuenca. Orientación a Objetos.

http://www.anayamultimedia.es/librosinternet/gpjava/Ejemplos/Objetos.htm

- Luis Guerrero. Herencia Simple. Herencia múltiple. Polimorfismo y enlace dinámico.
  http://www.dcc.uchile.cl/~luguerre/cc51h
- Microsoft Press. ALS Microsoft Windows 2000 Server.
  http://www.microsoft.com/w2000.htm
- Sybase y Power Builder
  www.sybase.com
- Ingeniería del Software
  http://www.gdl.uag.mx/66/temas.htm

ANEXOS

MANUAL DE ADMINISTRACION

Este manual detalla la instalación de la Base de Datos y el Sistema de Atención al Cliente SIDIMMOP.

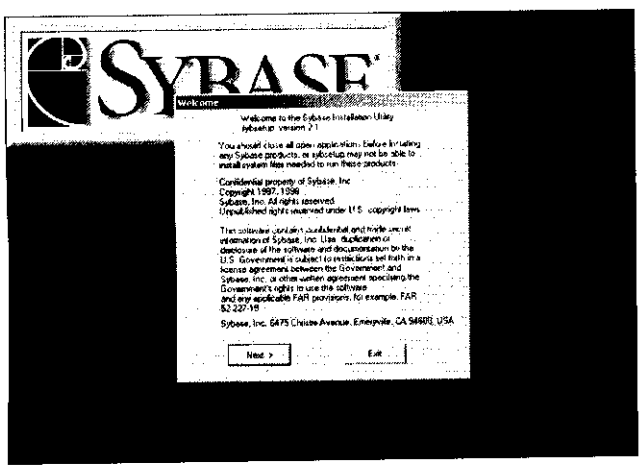
BASE DE DATOS

Requerimientos

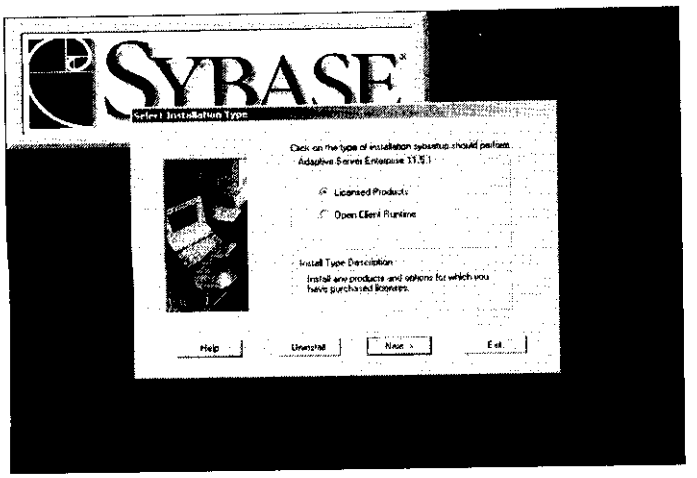
Para la instalación de la Base de Datos SYBASE se requiere un computador Pentium de preferencia PIII o superior, Memoria RAM 64MB (mínimo), 100MB de espacio en Disco (mínimo), sistema operativo Windows NT o superior; en el caso de Windows NT se requiere como base que este instalado el Service Pack 4.0.

El primer paso para el proceso de instalación es el siguiente:

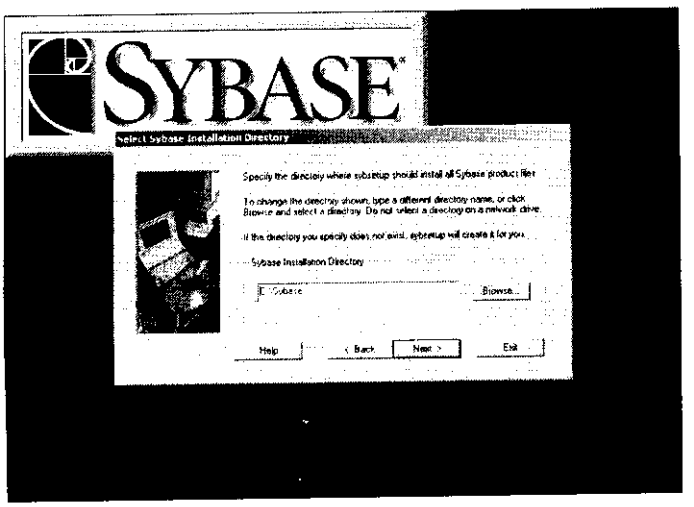
Se coloca el CD de SYBASE, se escoge el programa SETUP desde el explorador de Windows y se lo ejecuta. Aparece la pantalla de bienvenida, seleccionamos NEXT.



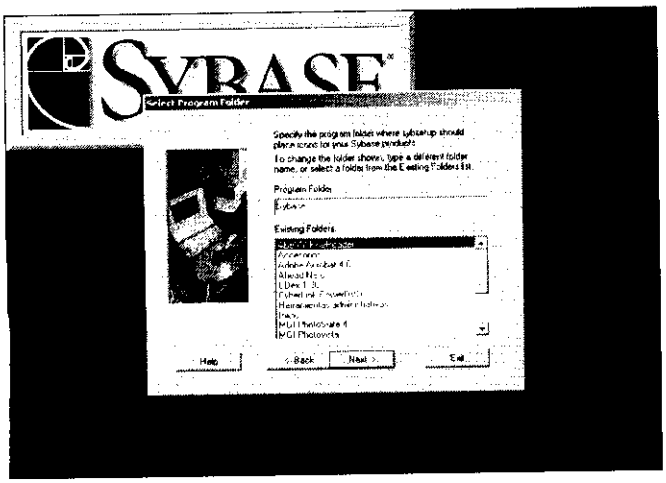
Como siguiente paso seleccionar el tipo de instalación que se desea, Productos licenciados o Instalación del cliente.



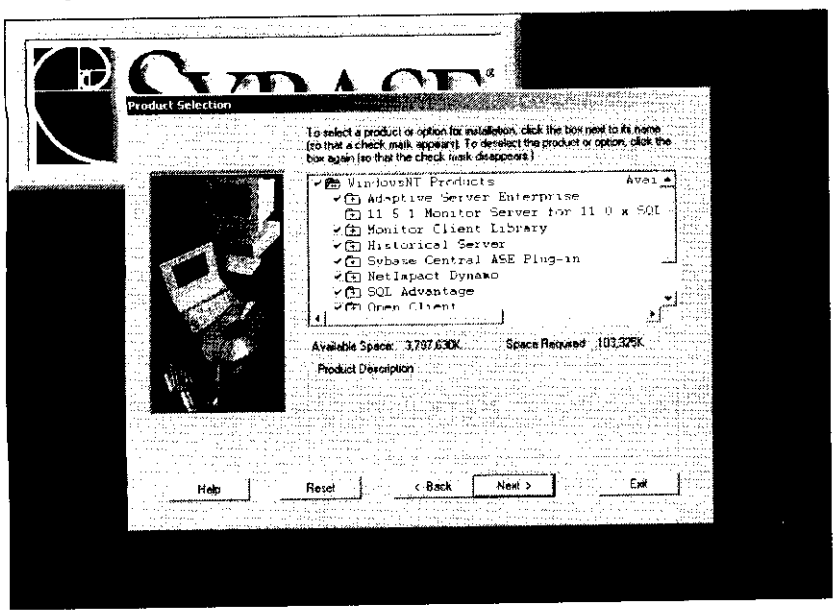
Para instalación del servidor escogemos la primera opción que es Licenced Products y presionamos NEXT; aparece el directorio donde se desea realizar la instalación, por defecto se pone en C: \Sybase, presionamos NEXT.



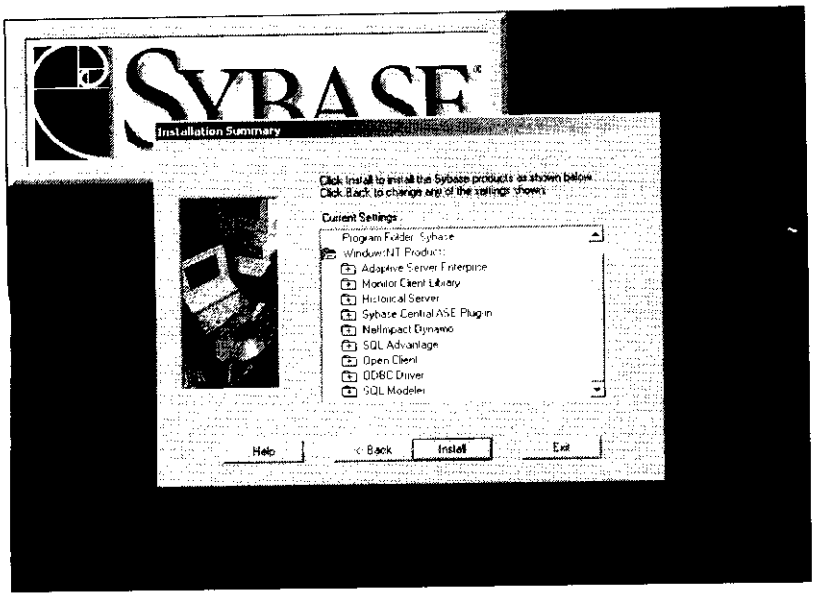
Aparece entonces el nombre de la carpeta donde se colocaran todos los accesos directos a los programas de SYBASE.



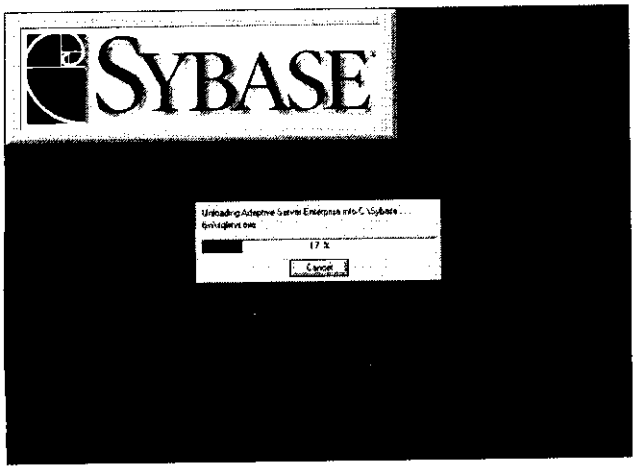
A continuación se detalla todos los programas que se van a instalar, cuando es la primera Instalación el software señala por defecto todos los necesarios para la instalación, por lo que se recomienda continuar con la instalación si alterar ninguna de las opciones; Presionamos NEXT.



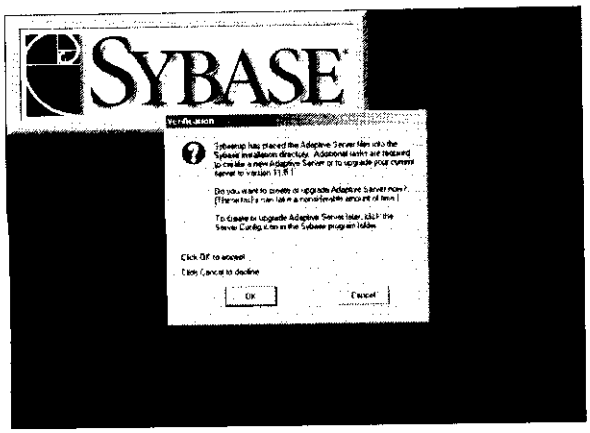
A continuación se revisa en pantalla por última vez todos los elementos a ser instalados, presionamos el botón Install, dando comienzo a la instalación.



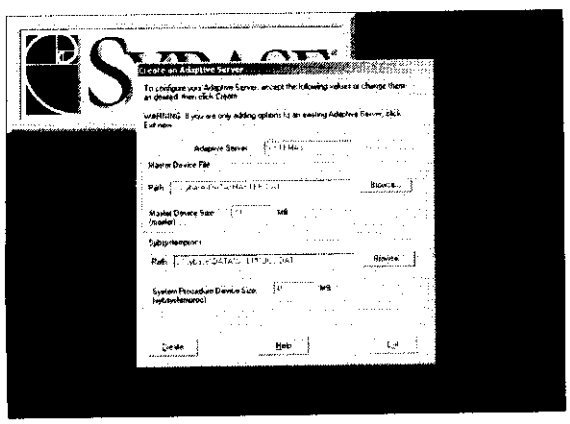
Una pantalla va detallando el porcentaje de la instalación.



Una vez terminada la instalación del software se nos pide una confirmación para la creación del Motor de la Base de Datos, si damos CANCEL se da por terminado la primera parte de la instalación.

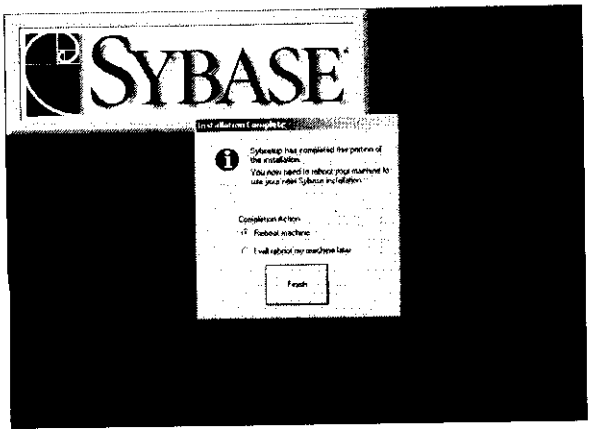


Si escogemos la opción OK aparece la siguiente pantalla, en donde se detalla el nombre del servidor, el tamaño de la base de datos MASTER y la SYBPROCS que son la parte fundamental del motor de SYBASE, ya que estas almacenan toda la información tanto de la configuración como de los procesos que esta se irán realizando. Seleccionamos CREATE.

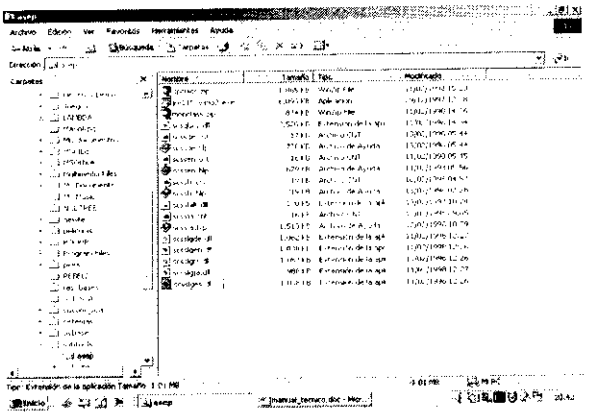


Una vez terminado el proceso de creación de las bases de configuración y procesos SYBASE nos pregunta si deseamos reiniciar el equipo y dar así por terminado la instalación.

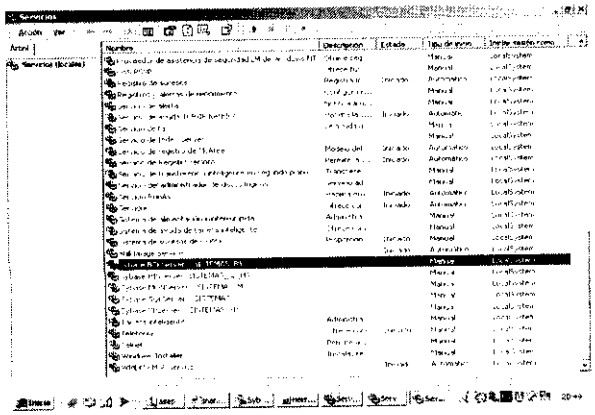
Nota: Para que todas las configuraciones realizadas durante la instalación tengan efecto se debe reiniciar el equipo.



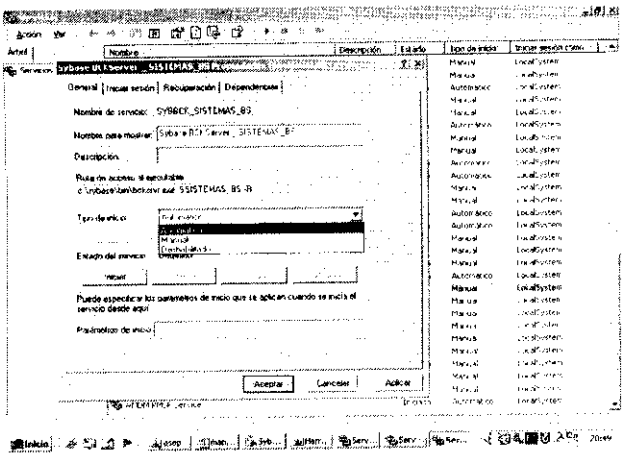
Una vez reiniciado el sistema se debe crear una copia de un archivo de librería dinámica llamado **scsslgen.dll** y renombrarlo a **scsslges.dll,** esto es para que se pueda ejecutar el **SYBASE CENTRAL**, que es el programa principal para el manejo del motor de la Base de Datos



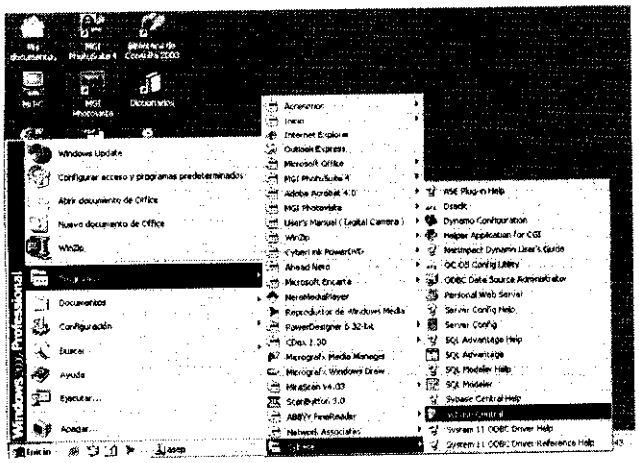
Para que el motor de la base de datos se levante automáticamente cada vez que se reinicie el servidor debemos configurar los servicios dentro del panel de control, herramientas administrativas, servicios; en donde se debe seleccionar SYBASE BCKServer y el SQLServer, que son el motor del backup de la base de datos y motor de la base de datos respectivamente.



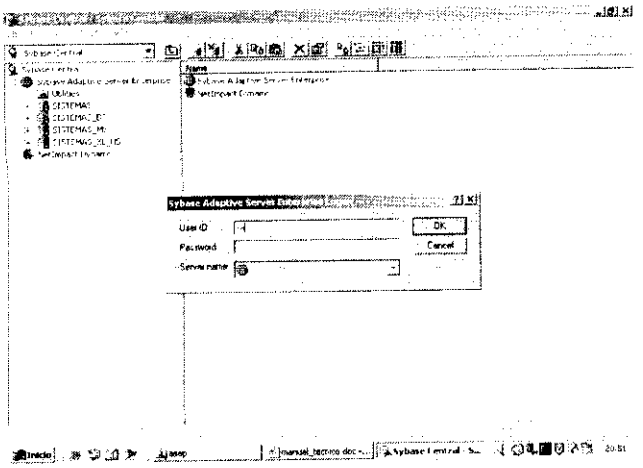
Para poner en forma automática estos servicios se debe presionar click derecho en el servicio, seleccionar tipo de inicio automático, presionar aceptar. Volver a proceder de la misma forma con los servicios que requieran este mismo tratamiento.



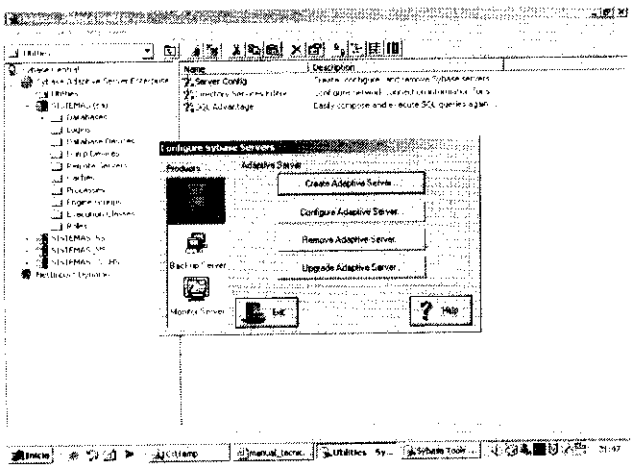
Para ejecutar el SYBASE CENTRAL, nos trasladamos a través de la barra de tareas, menú inicio, programas, sybase, en donde se encuentra el acceso directo del programa.

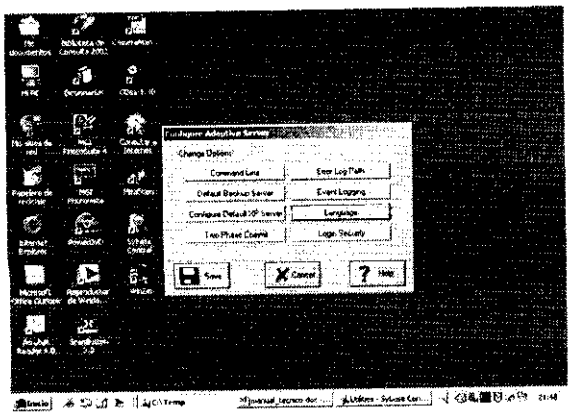


Una vez que se ejecuta el programa, se muestra la siguiente pantalla de presentación:



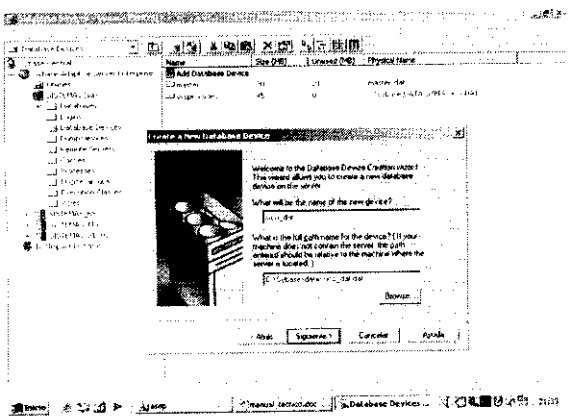
Antes de proceder con la creación de la base de datos, se debe configurar aspectos de lenguaje que formaran parte de las políticas del sistema, como son si se utilizará o no caracteres especiales, tildes y otros.



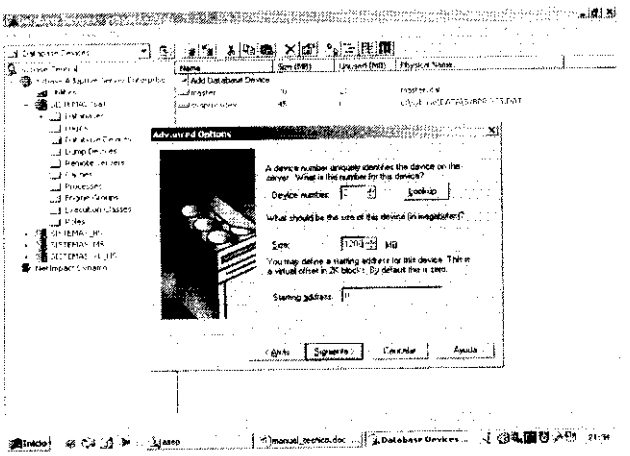


En la opción languaje debemos escoger Sort Order (set default), la opción Dictionary Order, case insensitive, este permite cualquier introducción de información, ya sea esta mayúscula o minúscula, tildes y otros. Después de hacer los cambios respectivos, grabamos y aceptamos, luego de lo cual para que tengan efecto se debe reiniciar el equipo.

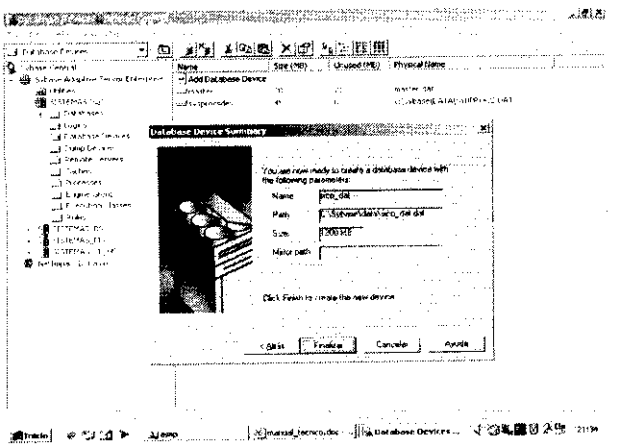
Una vez reiniciado el equipo se ejecuta el SYBASE CENTRAL y procedemos a crear los Data Devices que van a ser las unidades físicas en donde se almacenará toda la información de la base de datos.



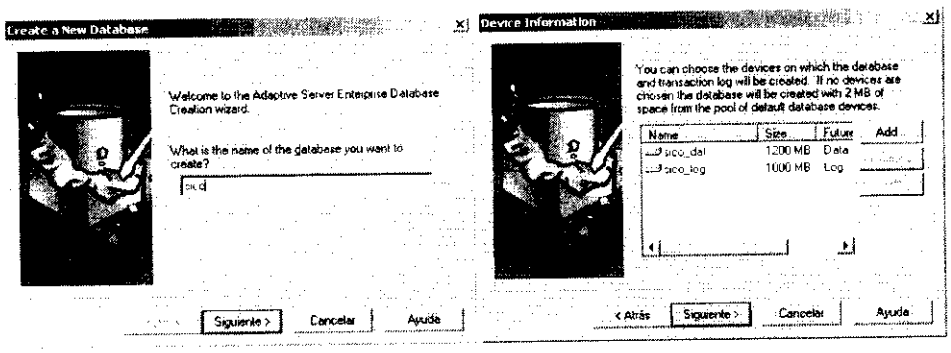
Para la creación de cada uno de los devices debemos tomar en consideración el tamaño que podrá alcanzar la base de datos realizando un cálculo estimado y haciendo una proyección para periodo de tiempo, esto nos dará como resultado el valor estimado en MegaBytes que se requerirán dentro de esta.



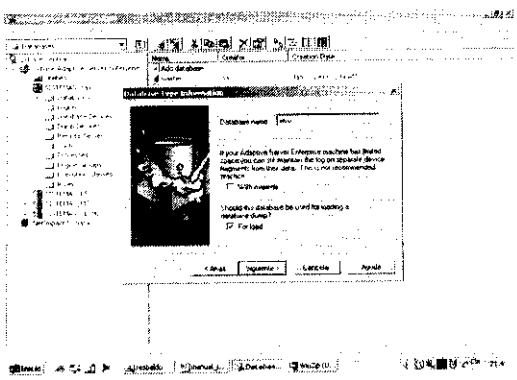
Una vez creado el device de datos procedemos de la misma forma con el device del log que no es otra cosa sino donde se almacenarán temporalmente todos los procesos que se realizaran contra la base de datos.



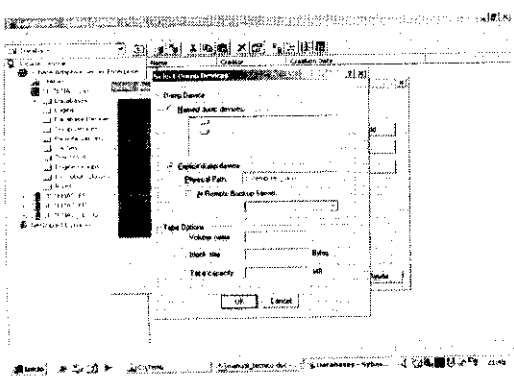
Terminado este proceso continuamos con la creación de la base de datos, dándole un nombre y seleccionando los tamaños que se utilizarán tanto para almacenamiento de datos y almacenamiento temporal de transacciones.



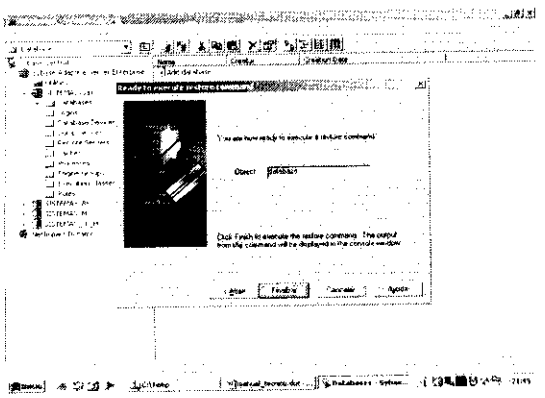
Hay una opción que se debe tomar muy en cuenta y es la FOR LOAD, esta se debe utilizar si se requiere subir un respaldo con todas las configuraciones de la base de datos respaldada, como es usuarios, grupos y los respectivos permisos a las tablas de la base. Si se quiere crear una base de datos nueva simplemente pasamos por alto estas selecciones.



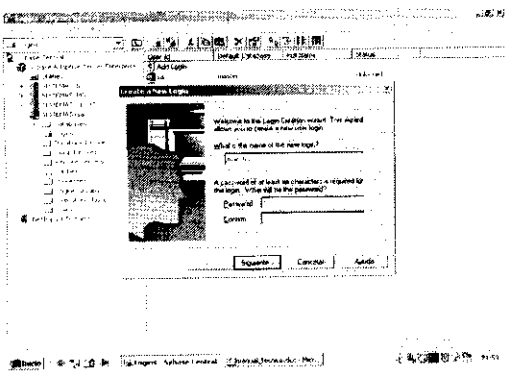
Para levantar un respaldo simplemente debemos seleccionar en la base de datos donde se quiere subir el mismo, click derecho, restore y seleccionar el origen de donde se encuentra en el respaldo.



Una vez que el respaldo es terminado de subir debemos fijarnos en el último mensaje que este despliega, donde aparece ON Line, esto quiere decir que el proceso termino satisfactoriamente, caso contrario debemos leer los diferentes mensajes que irán apareciendo y consultarlos con los manuales para ver si son o no factibles de solucionar.



Otro proceso que es muy importante es la creación de los usuarios y los respectivos permisos que a estos se les asignaran para el uso de las diferentes tablas de la base de datos. Adicionalmente se puede manejar grupos de usuarios, los mismos que facilitan de gran forma la utilización de permisos con diferentes usuarios que realicen procesos similares en la base de datos.



Para concluir con la instalación del sistema, debemos copiar la aplicación en una carpeta anteriormente definida en este caso la hemos llamado SIDIMMOP, donde irán todos las librerías del sistema incluida la carpeta BMP donde se encuentran los logotipos del sistema y algunos iconos anexados en la aplicación.

Luego creamos un icono de acceso directo hacia el escritorio de Windows, con la finalidad de dar facilidad al usuario para ejecutar el sistema.

MANUAL DE INSTALACIÓN.

Requerimientos Mínimos.

Requisitos de Hardware.

El sistema ha sido diseñado para trabajar en un ambiente Cliente.
Las características óptimas de hardware se detallan a continuación:

- Procesador Pentium o superior.
- RAM de 32 Mb
- Espacio en Disco Duro 60 Mb
- Drive 3 ½.
- USB
- Teclado.
- Mouse.
- Monitor SVGA.
- Unidad de CD – ROM.

Requisitos Software

Software

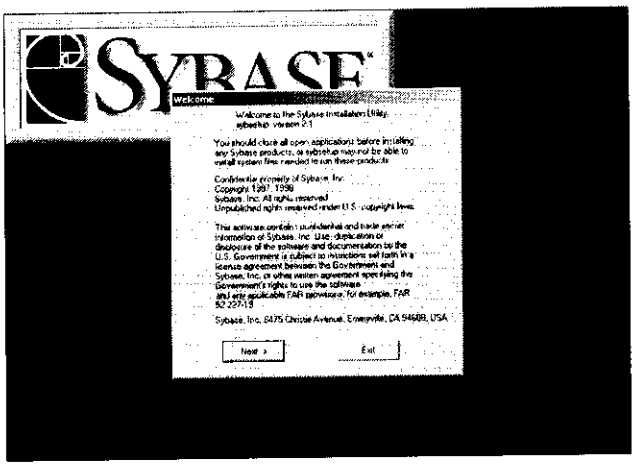
- Windows 2000
- Sybase 11.5 Cliente (RunTime).

Instalación de SIDIMMOP

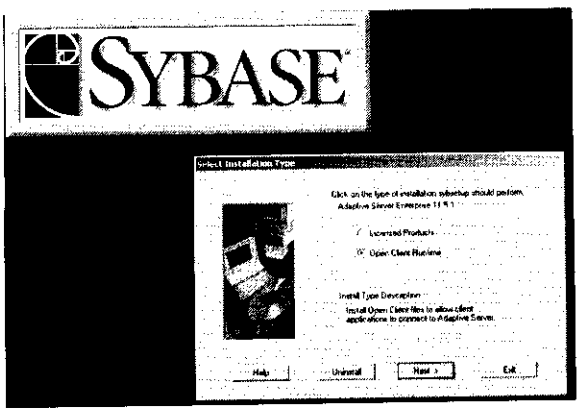
A continuación se detalla el proceso a realizar para la instalación del sistema en la estación de trabajo de cada una de las secciones:

Instalación cliente de SYBASE

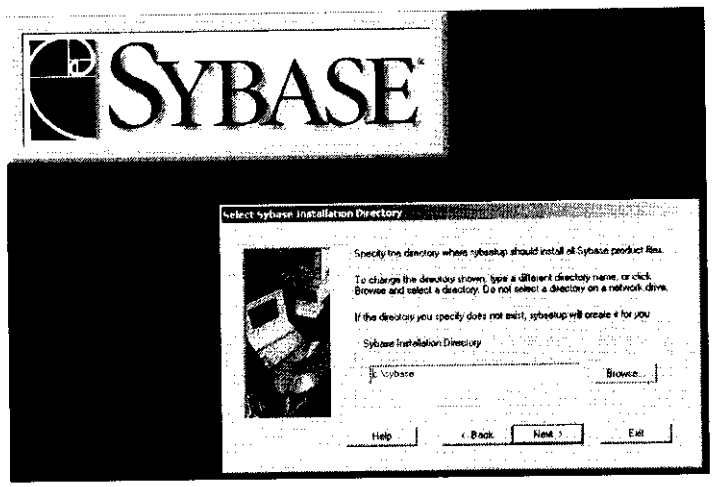
Desde el CR-ROM ejecutamos el programa **SETUP.EXE**, aparece la pantalla de presentación, presionamos **NEXT** para continuar o **EXIT** para salir de la instalación.



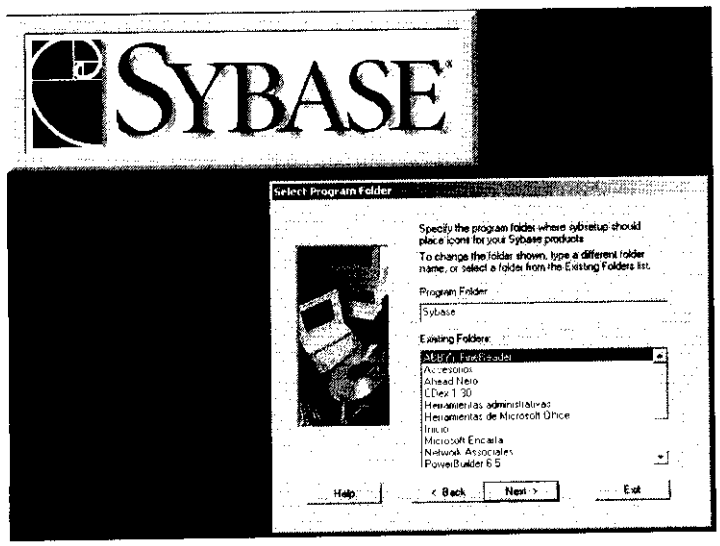
Seleccionamos la opción **Open Client Runtime,** presionamos **NEXT**



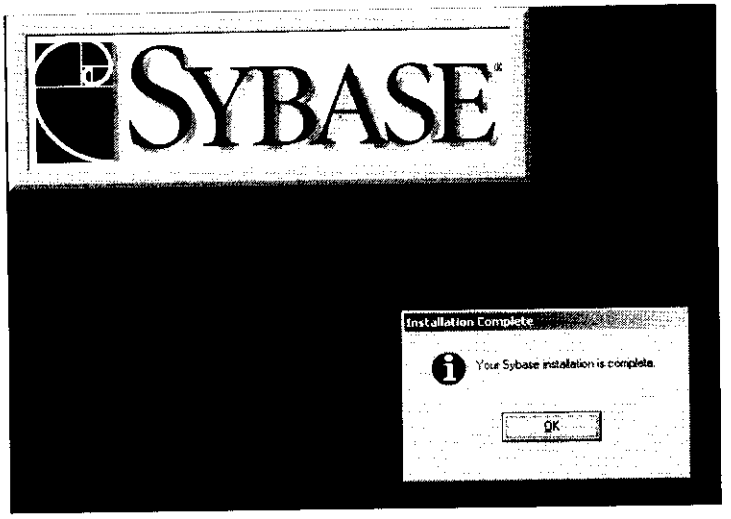
Escogemos la ubicación donde se instalarán los archivos de configuración, por defecto aparece **C: \SYBASE,** si se requiere se cambia el destino, caso contrario presionamos **NEXT.**



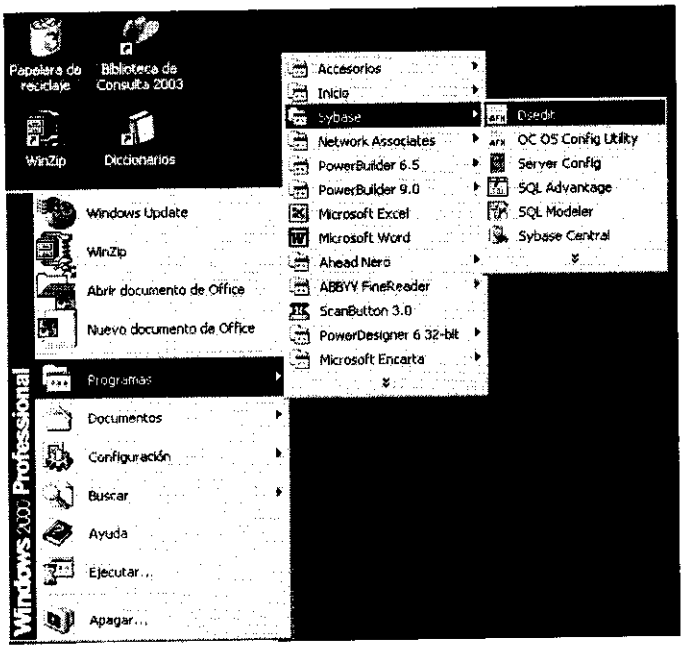
El programa de instalación muestra una opción para el nombre de la carpeta contenedora de los archivos que van a ser instalados, para continuar se presiona NEXT.



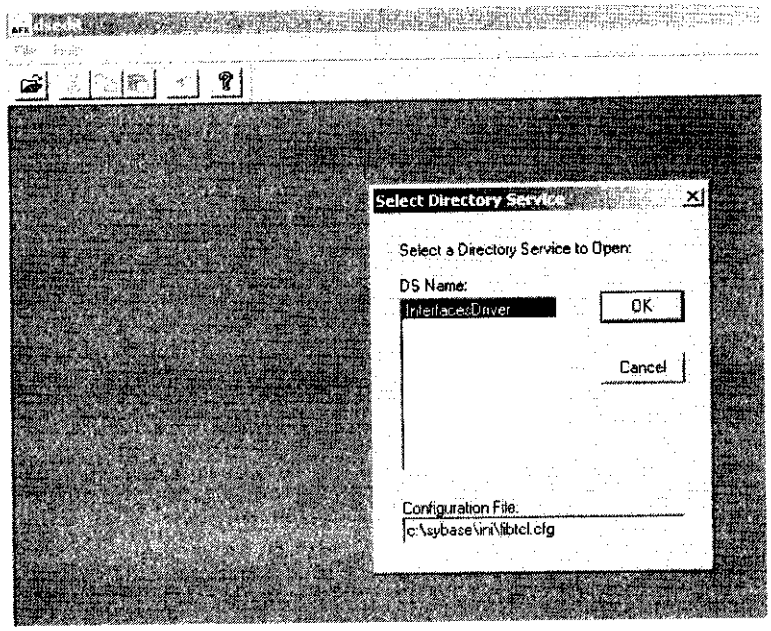
Una vez terminada la copia de los archivos de instalación se deberá proceder a reiniciar el equipo, para que los cambios tengan efecto.



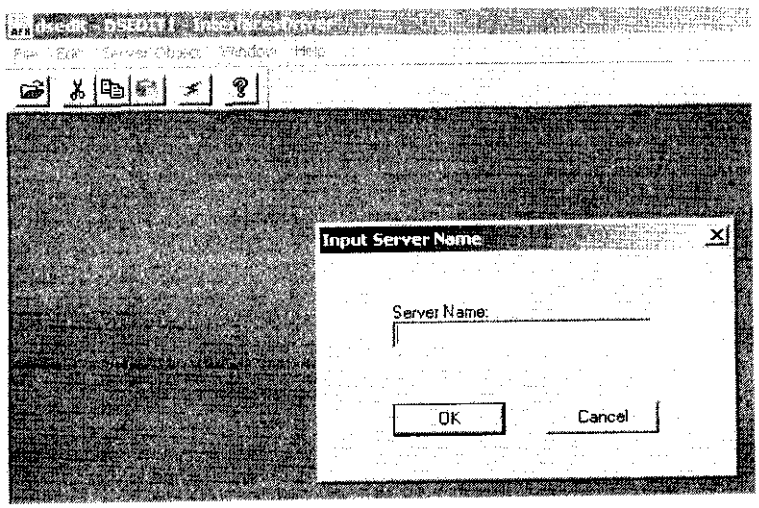
Una vez reiniciado el computador, nos dirigimos a INICIO, PROGRAMAS, SYBASE y escogemos DSEDIT, para continuar con la configuración del cliente.



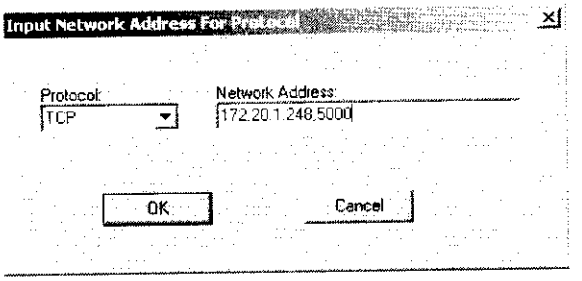
Una vez ejecutado el programa aparece la siguiente pantalla, presionar OK.



Se debe crear el nombre del servidor (en este caso es la misma maquina del cliente) donde el cliente se enlazará para la utilización de la Base de Datos, presionar la combinación de teclas CTRL.+A o a su vez acceder a la opción del menú **Server Object** y escoger **Add...,** entrar el nombre y presionar OK.

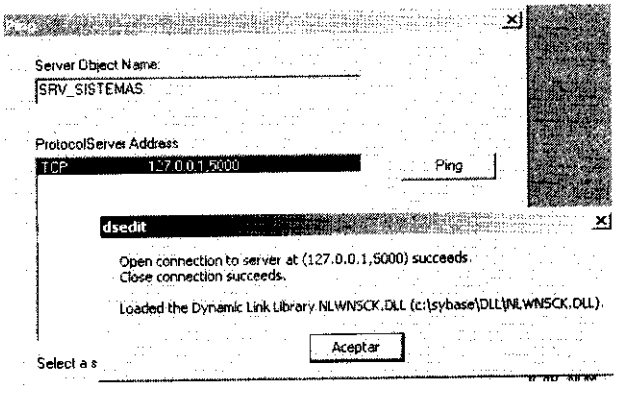


Nota: Esta configuración está por defecto en 5000 cuando se instala el motor de la Base de Datos por primera vez.

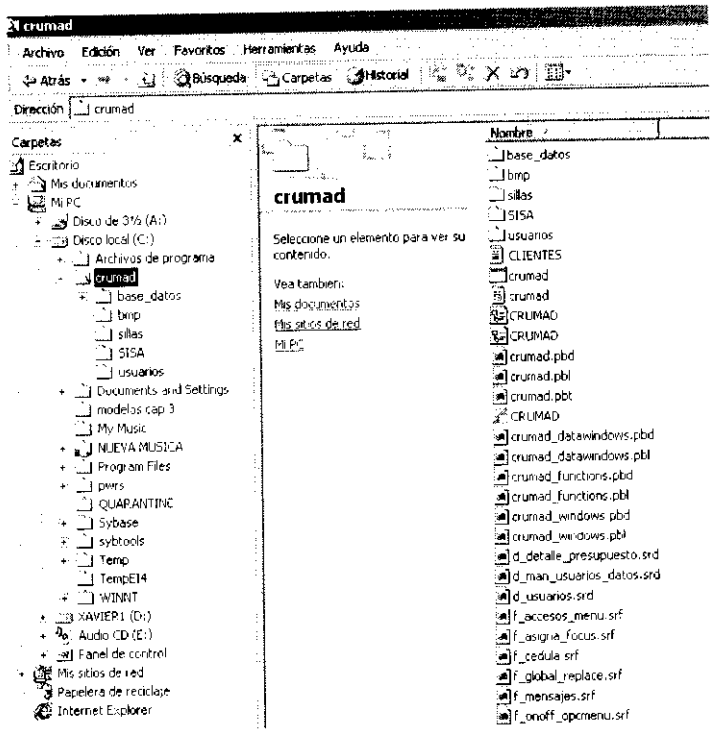


Debe presionar OK. Una vez realizado el proceso.

Para verificar el enlace se presiona el botón , este a su vez ejecuta un comando que verifica la conexión con la Base de Datos y devuelve el mensaje **succeeds** si la conexión está correcta, caso contrario un **fails** es mostrado.



Determinado que hay conexión con la base de datos cerramos el **Dsedit** y se procede con la creación de la carpeta **CRUMAD,** dentro de esta las subcarpetas **BMP** y **CRUMAD_EXE,** está última contendrá la aplicación compilada y el programa ejecutable.



Para finalizar el proceso, se debe crear un ICONO de acceso directo a la aplicación en el escritorio WINDOWS, se recomienda el icono [ ] y el nombre **SIDIMMOP,** para una rápida identificación del usuario.

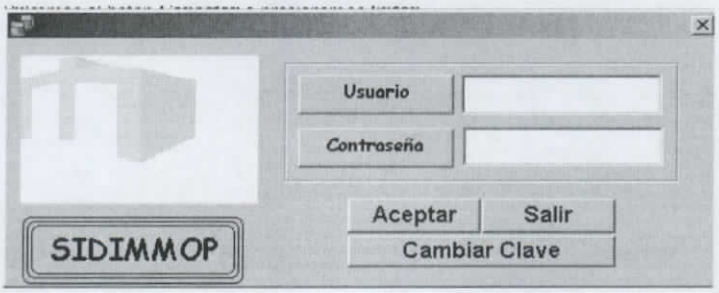
MANUAL DE USUARIO

Inicio de Sesión en el Sistema SIDIMMOP

Para iniciar una sesión en SIDIMMOP damos doble clic en el icono que se muestra a continuación ubicado en el escritorio de Windows.



Aparece la siguiente pantalla en la cual ingresamos el nombre y la clave del usuario. Pulsamos el botón **Aceptar** o presionamos **Enter.**



Si uno de los parámetros no es correcto se despliega un mensaje de que el usuario está incorrecto o que la clave suministrada es incorrecta, o que no existe conexión a la base de datos.

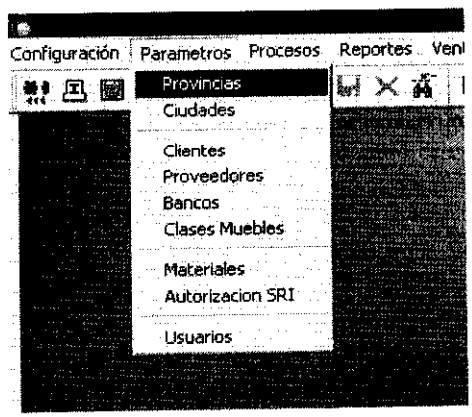
Una vez ingresado al Sistema aparece la siguiente pantalla, en la que tenemos disponible la opción de menú **SIDIMMOP**, la que debemos seleccionar para ingresar a la pantalla con el nuevo menú del sistema opción SIDIMMOP.



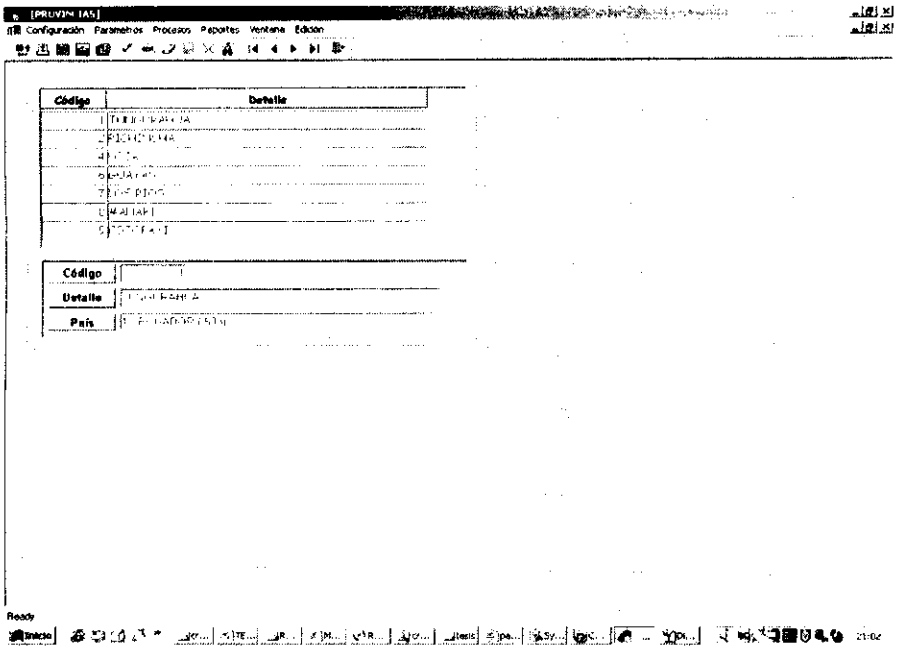
Aquí se presenta claramente las diferentes opciones del programa, con las cuales se trabajará, para ello se debe que seleccionar una de las opciones disponibles del mismo. A continuación se describe cada una de las opciones disponibles del sistema.

Parámetros Provincias

Para efectuar este proceso se ingresa a la opción Parámetros, Provincias.



Aparece la pantalla que se ilustra a continuación.



En esta pantalla seleccionamos la opción **NUEVO** y se ingresa la siguiente información:

- Detalle
- País

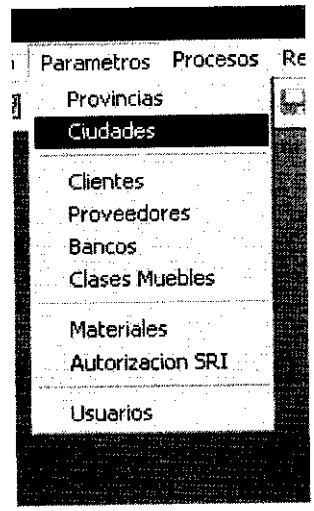
Además de esto el sistema se encarga de completar los campos correspondientes a:

- Código

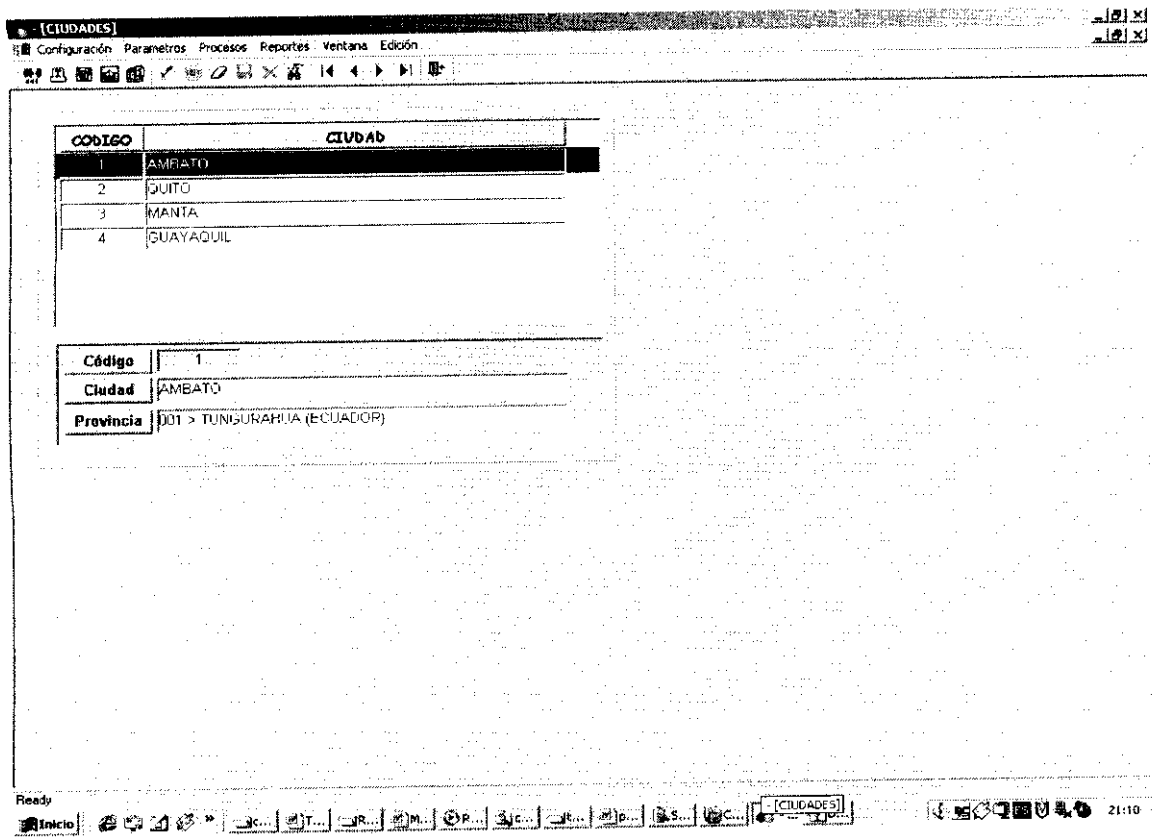
Con estos datos se almacena la información mediante la opción **GUARDAR.** El número de código que genera el sistema no debe coincidir con los antes generados.

Parámetros Ciudades

Para efectuar este proceso se ingresa a la opción Parámetros, Ciudades.



Aparece la pantalla que se ilustra a continuación.



En esta pantalla seleccionamos la opción **NUEVO** y se ingresa la siguiente información:

- Ciudad
- Provincia

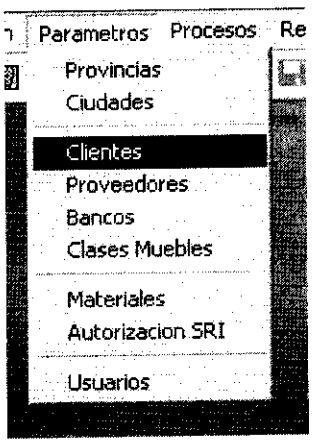
Además de esto el sistema se encarga de completar los campos correspondientes a:

- Código

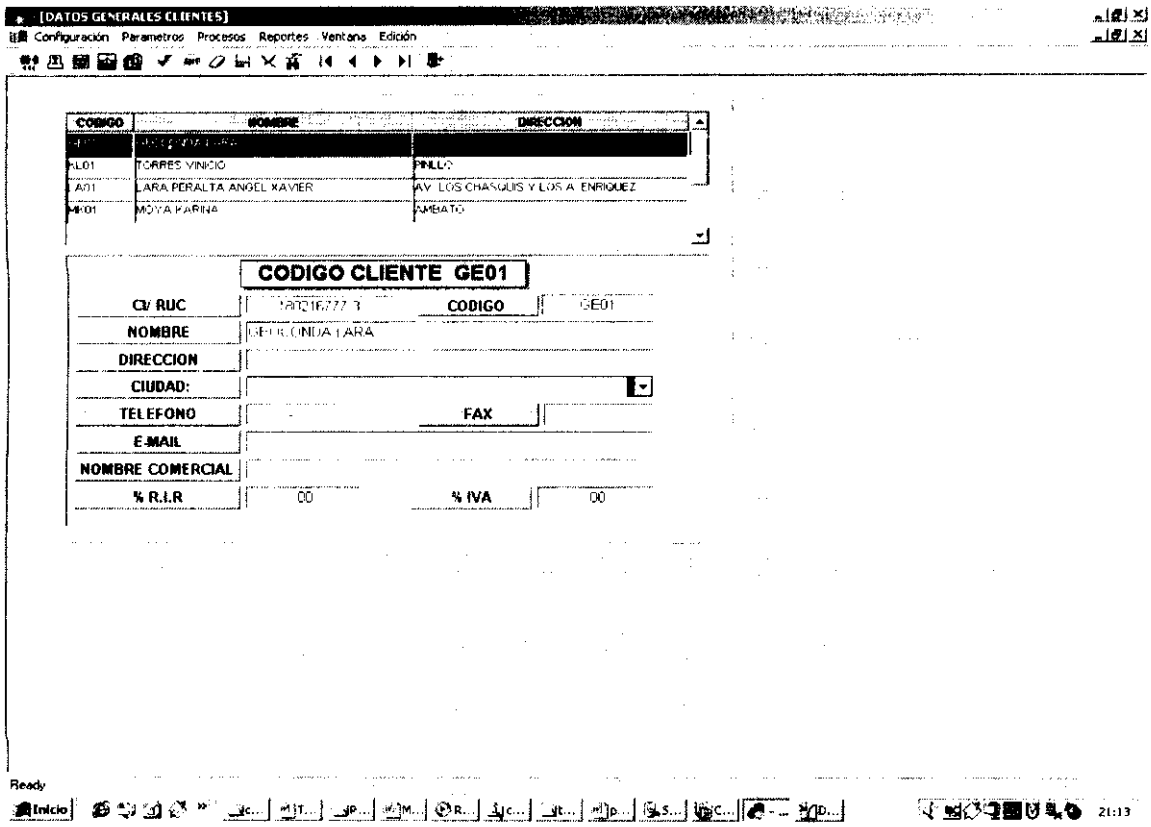
Con estos datos se almacena la información mediante la opción **GUARDAR.** El número de código que genera el sistema no debe coincidir con los antes generados.

Parámetros Clientes

Para efectuar este proceso se ingresa a la opción Parámetros, Clientes.



Aparece la pantalla que se ilustra a continuación.



En esta pantalla seleccionamos la opción **NUEVO** y se ingresa la siguiente información:

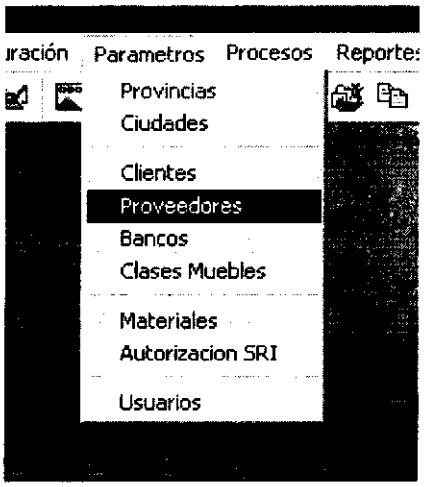
- CI / RUC
- Nombre
- Dirección
- Ciudad
- Teléfono
- Fax
- E-mail
- Nombre Comercial
- % retención de renta

- % retención de IVA
- Código

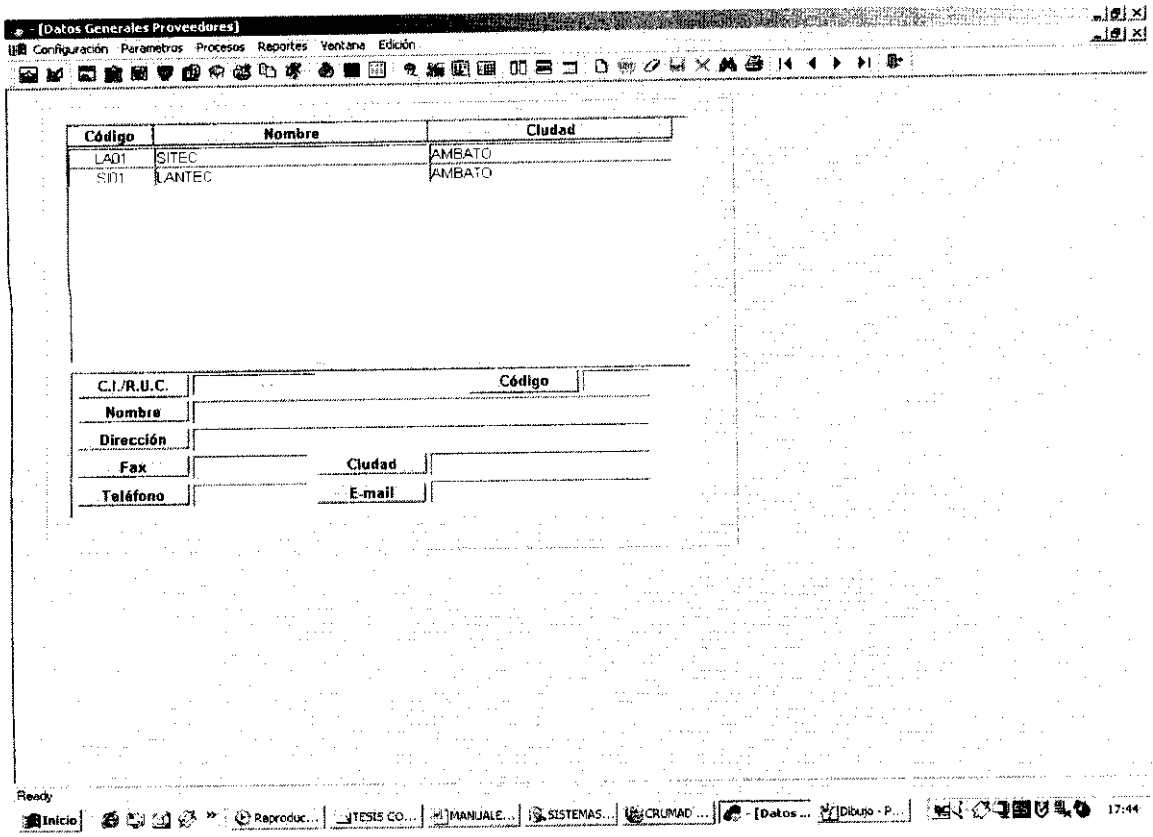
Con estos datos se almacena la información mediante la opción **GUARDAR.** El número de código que genera el sistema no debe coincidir con los antes generados.

Parámetros Proveedores

Para efectuar este proceso se ingresa a la opción Parámetros, Proveedores



Aparece la pantalla que se ilustra a continuación.



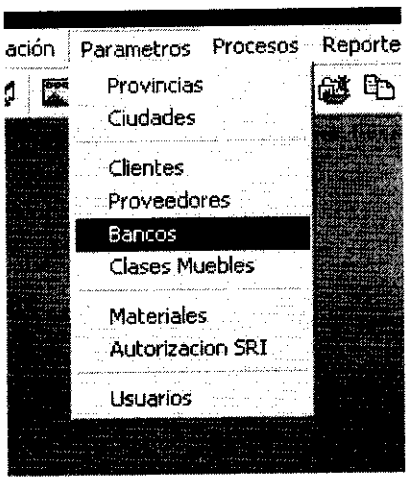
En esta pantalla seleccionamos la opción **NUEVO** y se ingresa la siguiente información:

- CI / RUC
- Nombre
- Dirección
- Ciudad
- Teléfono
- Fax
- E-mail
- Código

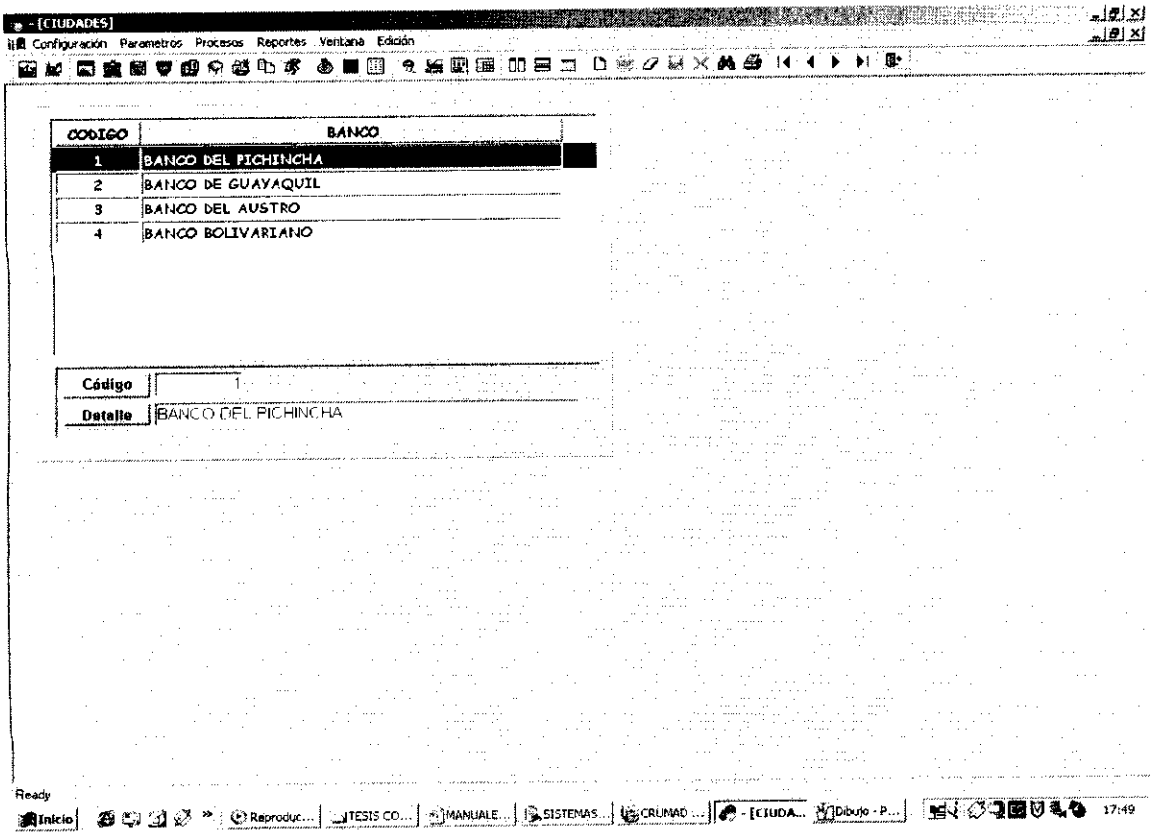
Con estos datos se almacena la información mediante la opción **GUARDAR.** El número de código que genera el sistema no debe coincidir con los antes generados.

Parámetros Bancos

Para efectuar este proceso se ingresa a la opción Parámetros, Bancos



Aparece la pantalla que se ilustra a continuación.



En esta pantalla seleccionamos la opción **NUEVO** y se ingresa la siguiente información:

- Detalle

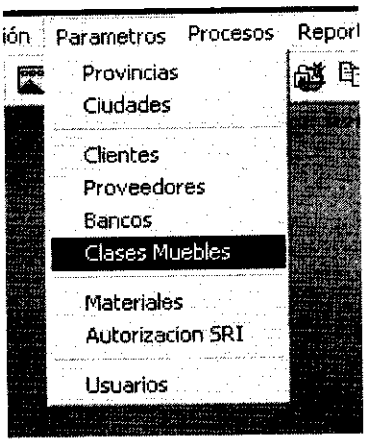
Además de esto el sistema se encarga de completar los campos correspondientes a:

- Código

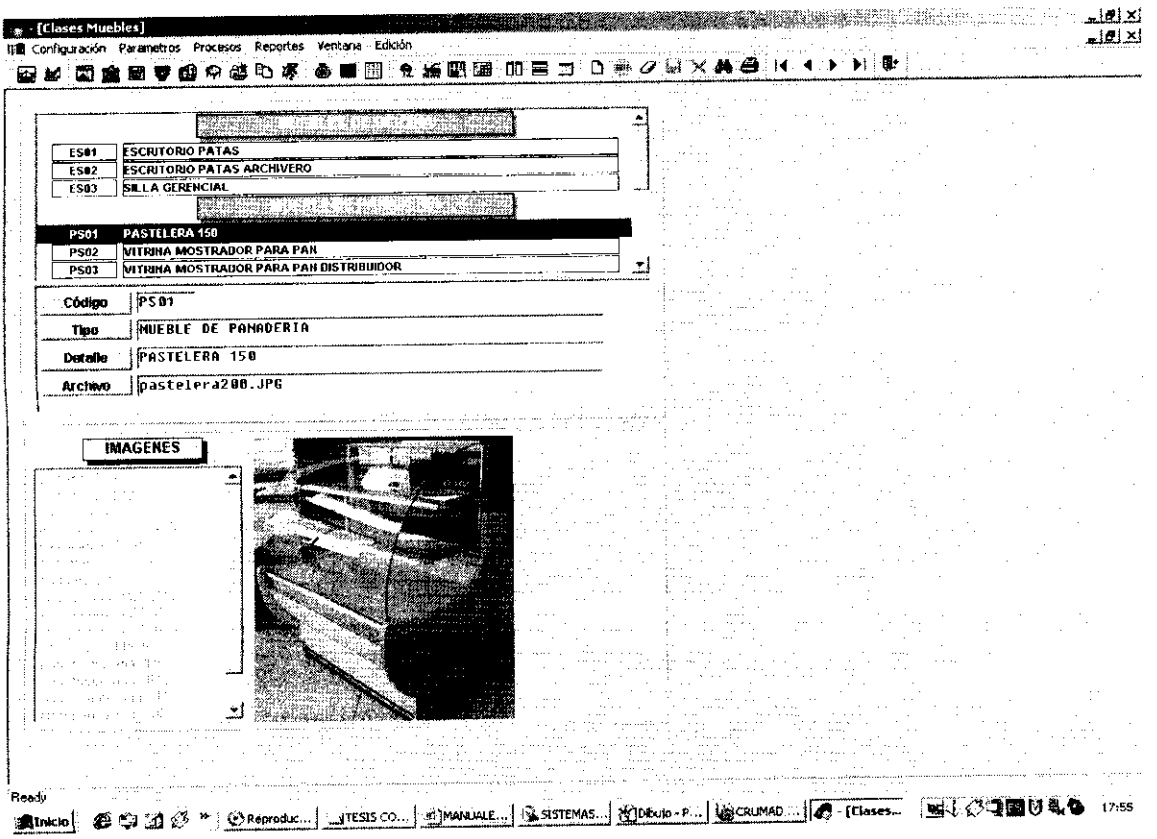
Con estos datos se almacena la información mediante la opción **GUARDAR.** El número de código que genera el sistema no debe coincidir con los antes generados.

Parámetros Clases de Muebles

Para efectuar este proceso se ingresa a la opción Parámetros, Clases de Muebles



Aparece la pantalla que se ilustra a continuación.



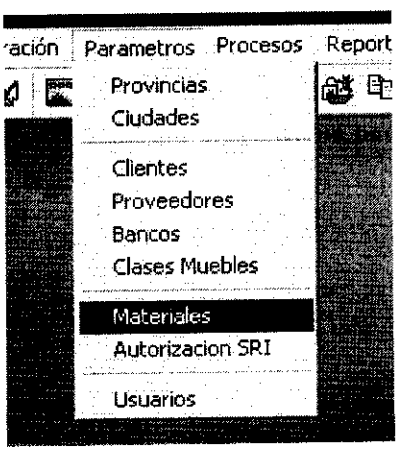
En esta pantalla seleccionamos la opción **NUEVO** y se ingresa la siguiente información:

- Tipo
- Detalle
- Código

Debemos seleccionar en el listado de imágenes el modelo de mueble que vamos a crear para que el sistema se encargue de completar el campo de la dirección del ARCHIVO.

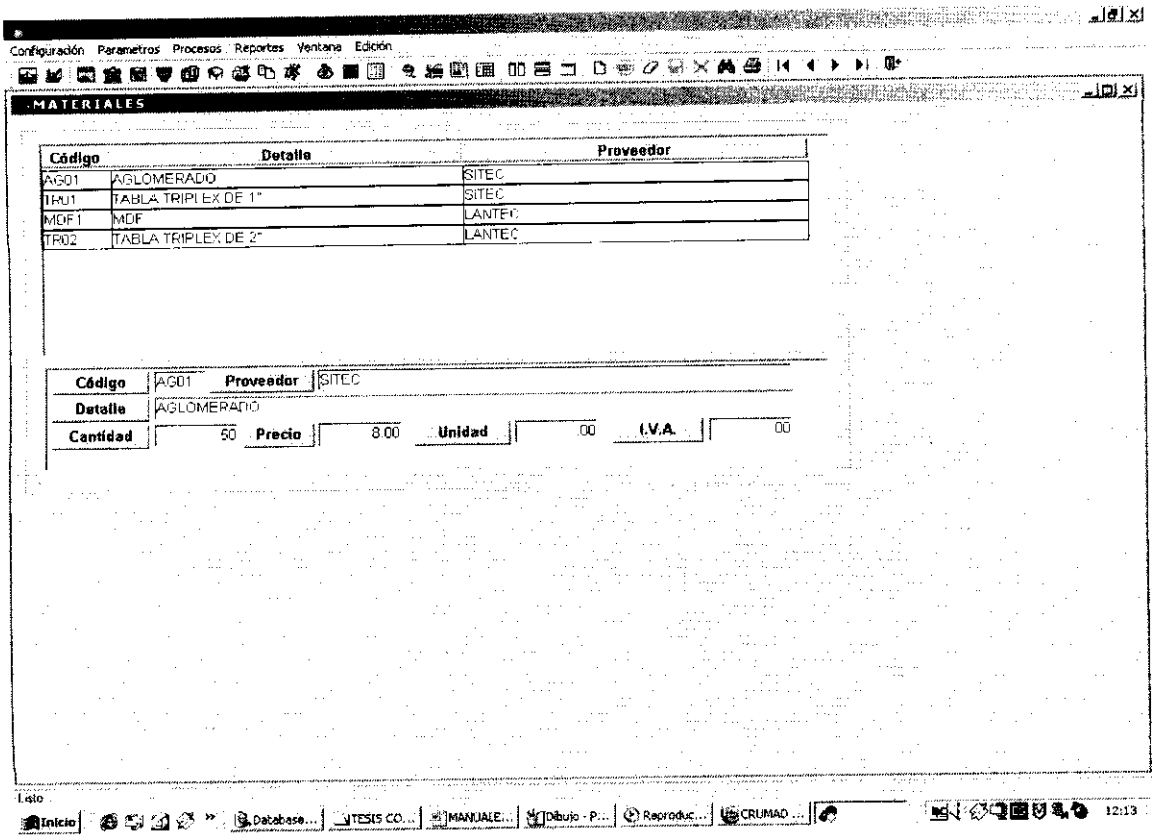
Con estos datos se almacena la información mediante la opción **GUARDAR.** El número de código que genera el sistema no debe coincidir con los antes generados.

Parámetros Materiales



Para efectuar este proceso se ingresa a la opción Parámetros, Materiales

Aparece la pantalla que se ilustra a continuación.

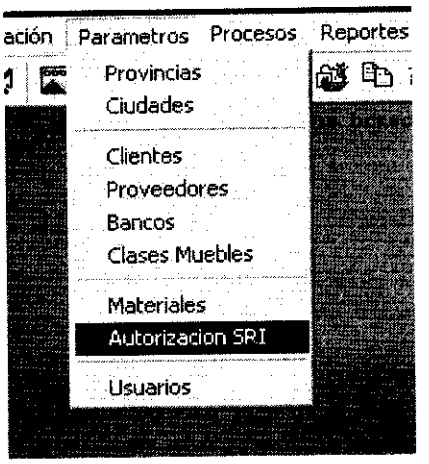


En esta pantalla seleccionamos la opción **NUEVO** y se ingresa la siguiente información:

- Código
- Proveedor
- Detalle
- Cantidad
- Precio

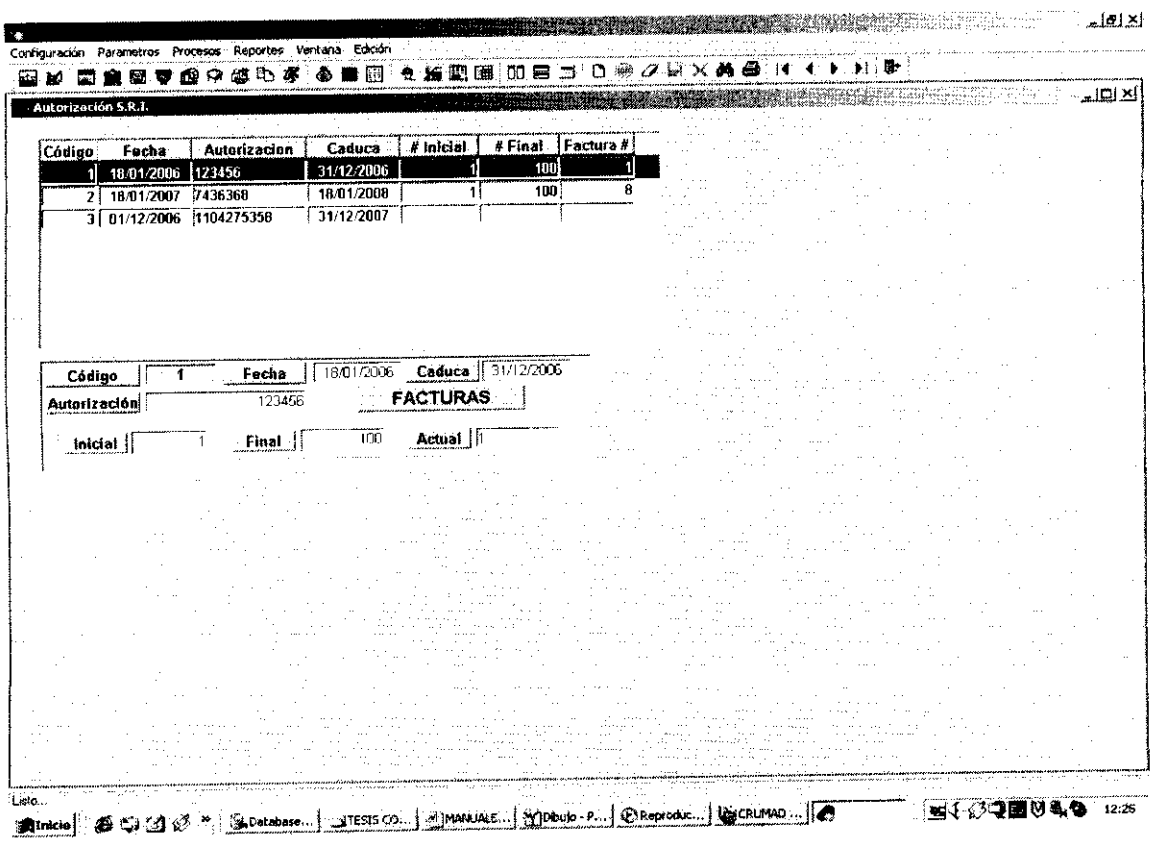
Con estos datos se almacena la información mediante la opción **GUARDAR.** El número de código que genera el sistema no debe coincidir con los antes generados.

Parámetros Materiales



Para efectuar este proceso se ingresa a la opción Parámetros, Autorización SRI

Aparece la pantalla que se ilustra a continuación.



En esta pantalla seleccionamos la opción **NUEVO** y se ingresa la siguiente información:

- Fecha de Autorización
- Fecha de Caducidad
- # Autorización
- Rango inicial
- Rango Final

Además de esto el sistema se encarga de completar los campos correspondientes a:

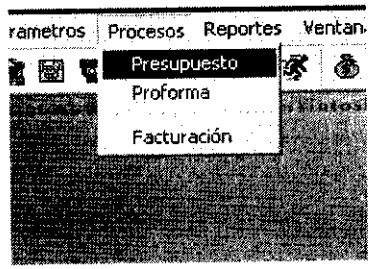
- Código

Con estos datos se almacena la información mediante la opción **GUARDAR.** El número de código que genera el sistema no debe coincidir con los antes generados

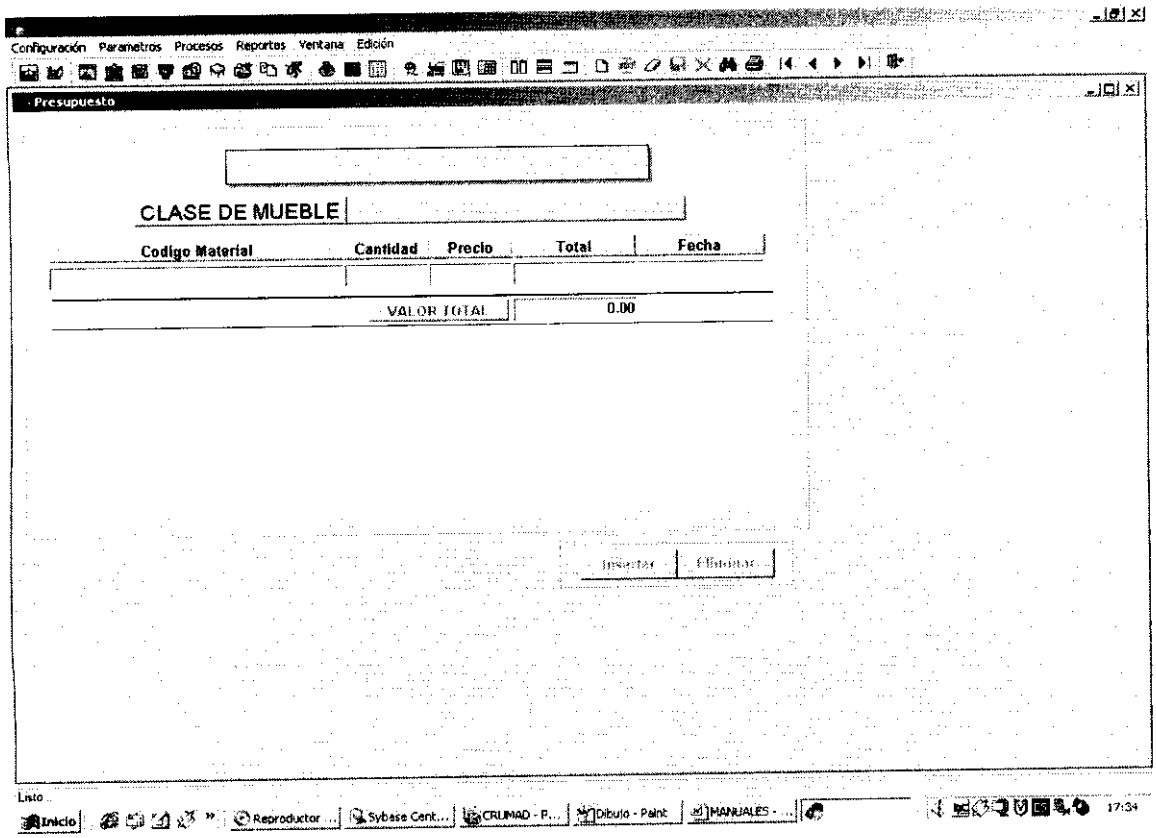
PROCESOS

Proceso Presupuesto

Para efectuar este proceso se ingresa a la opción Procesos, Presupuesto; o en la barra de herramientas damos click sobre el icono.



Aparece la pantalla que se ilustra a continuación.



Este proceso nos permite obtener el presupuesto de un mueble modular en base al parámetro materiales que ingresamos anteriormente.

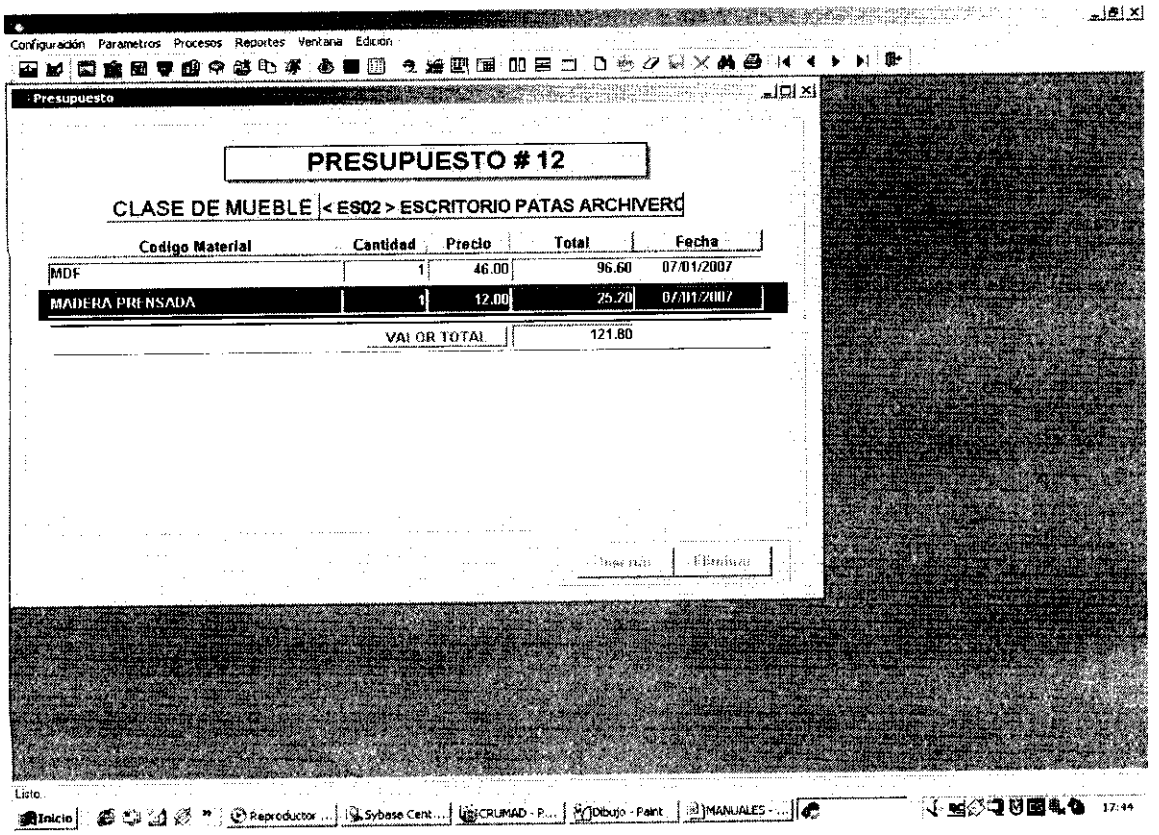
En esta pantalla seleccionamos la opción **NUEVO,** automáticamente se nos genera un número de presupuesto y se ingresa la siguiente información:

- La clase de mueble que vamos a presupuestar

- Código de material que se utiliza para la fabricación de dicho mueble.
- Cantidad
- Precio

Además de esto el sistema se encarga de completar los campos correspondientes a:

- Total
- Fecha

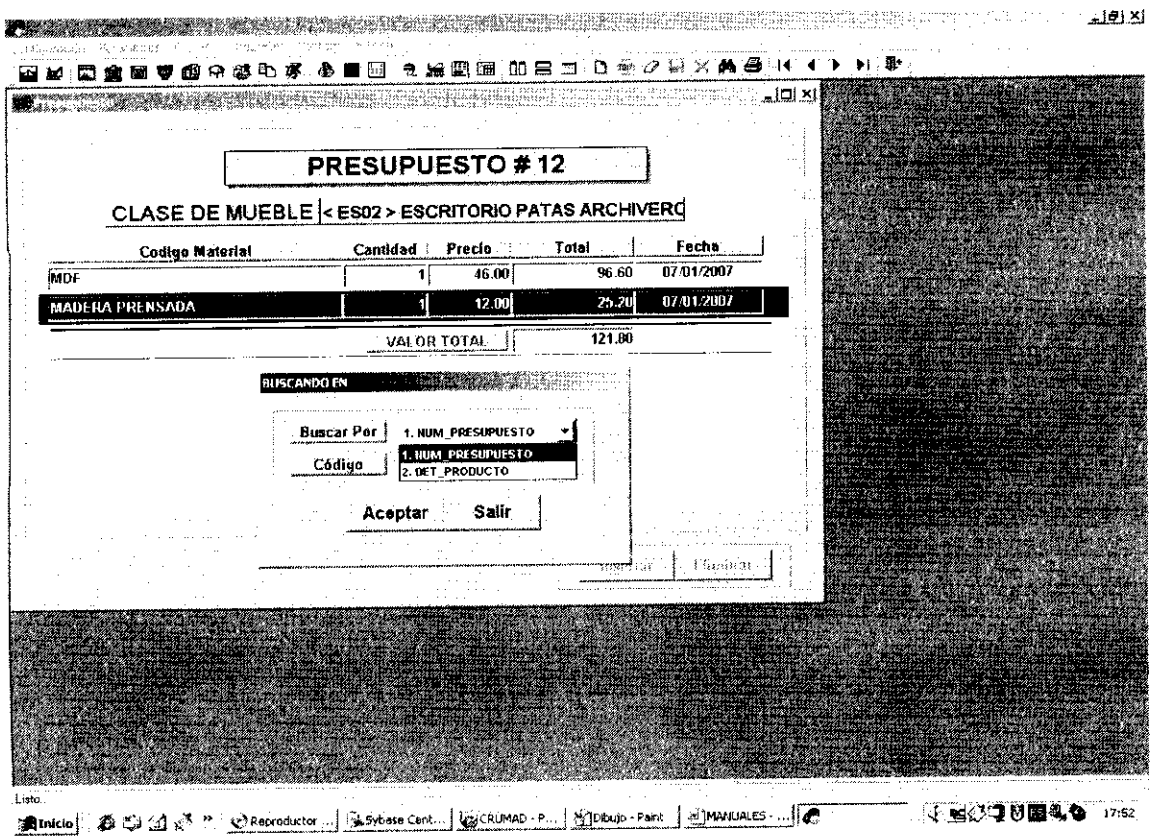


Si todo está correcto se procede a guardar esta información mediante la opción **GUARDAR**, caso contrario se procede a la actualización y su posterior almacenamiento. Este presupuesto es el valor de un solo mueble modular que nos va a servir posteriormente para los siguientes procesos.

Además en este proceso tenemos las opciones de Editar, Borrar, Imprimir, Buscar; tanto en la barra de herramientas como dentro del menú **Edición**.

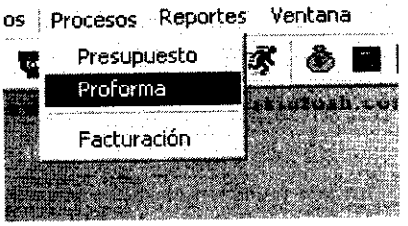
En la opción de buscar tenemos búsqueda por:

- Numero de presupuesto
- Detalle del producto

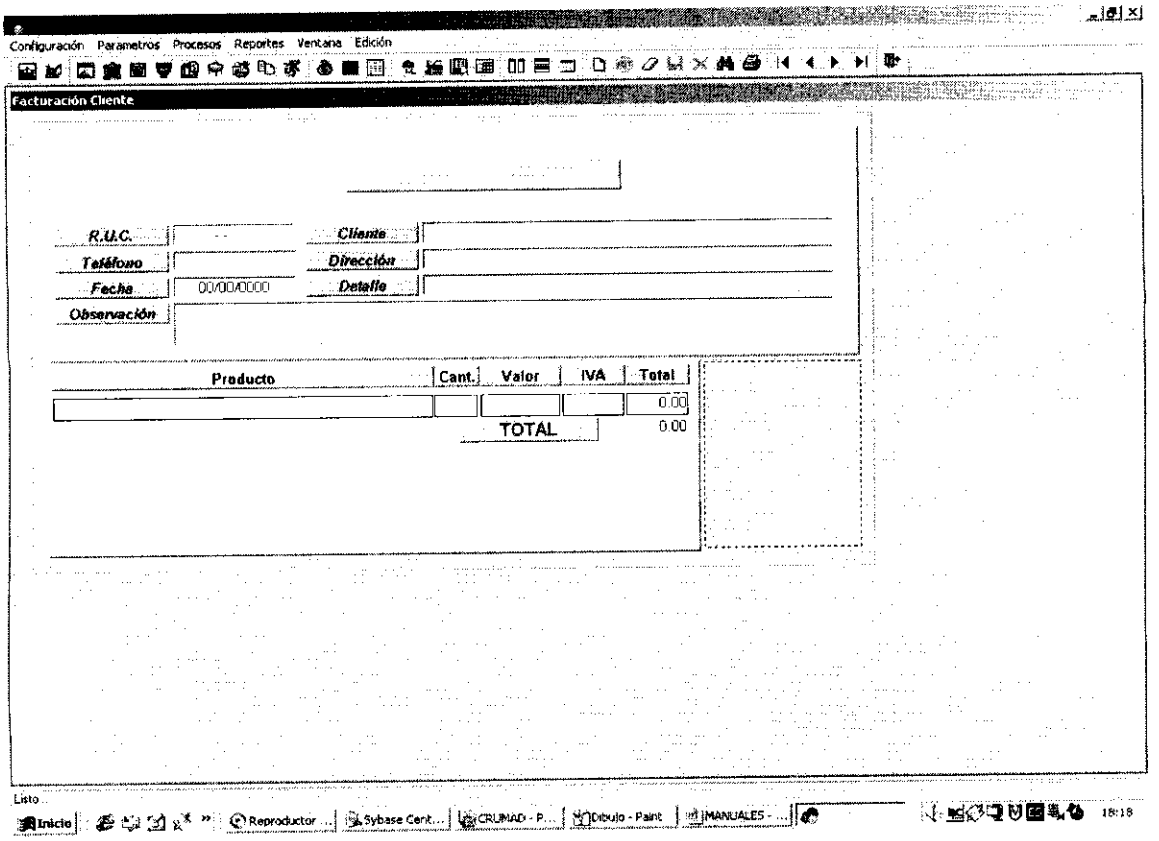


Proceso Proforma

Para efectuar este proceso se ingresa a la opción Procesos, Proforma; o en la barra de herramientas damos click sobre el icono■.



Aparece la pantalla que se ilustra a continuación.



Este proceso nos permite obtener la proforma para un determinado cliente de varios muebles modulares que fueron presupuestados en el proceso anterior.

En esta pantalla seleccionamos la opción **NUEVO,** automáticamente se nos genera un número de proforma y se ingresa la siguiente información:

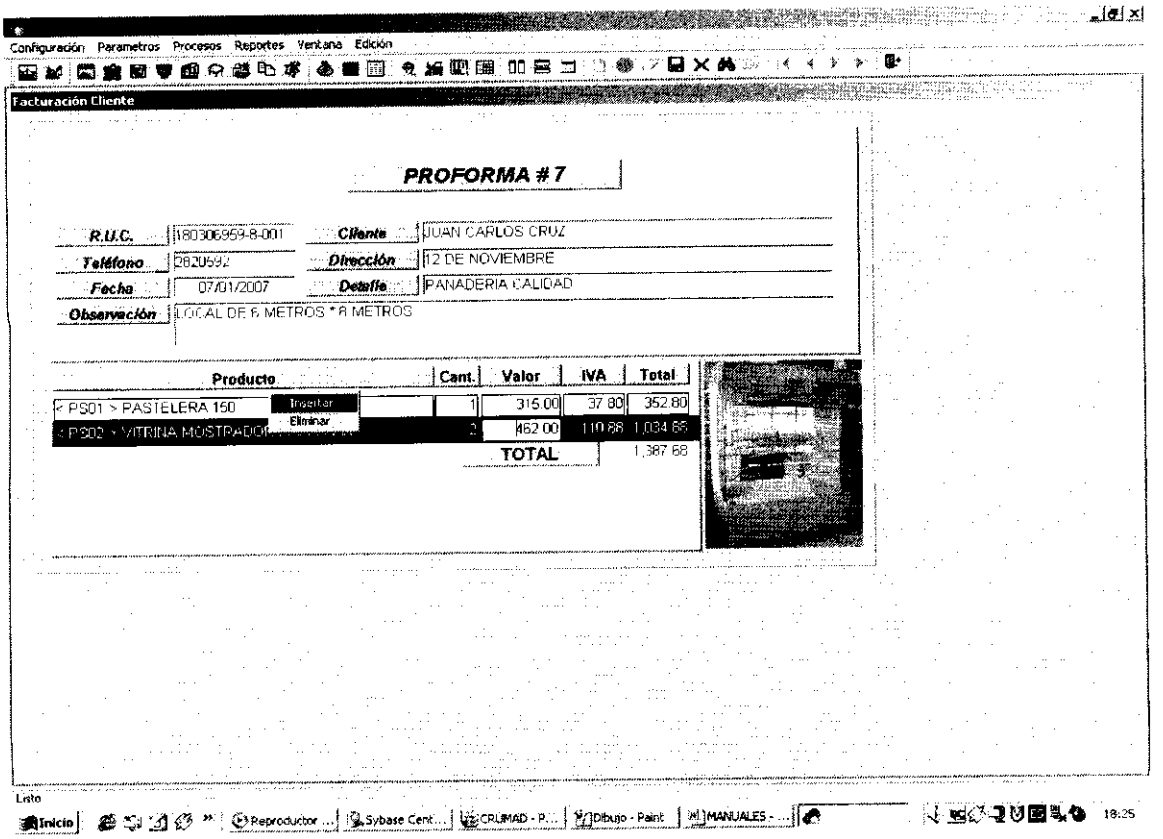
- RUC

- Cliente, si el cliente consta en nuestra base de datos de **Clientes** nos recupera automáticamente toda la información que tenemos de este cliente, caso contrario la ingresamos y se almacenara posteriormente cuando la guardemos en la base de clientes.
- Detalle
- Observación
- Producto
- Cantidad
- Valor

Además de esto el sistema se encarga de completar los campos correspondientes a:

- Fecha
- IVA
- Total

Dando click derecho sobre la columna de producto insertamos más productos.

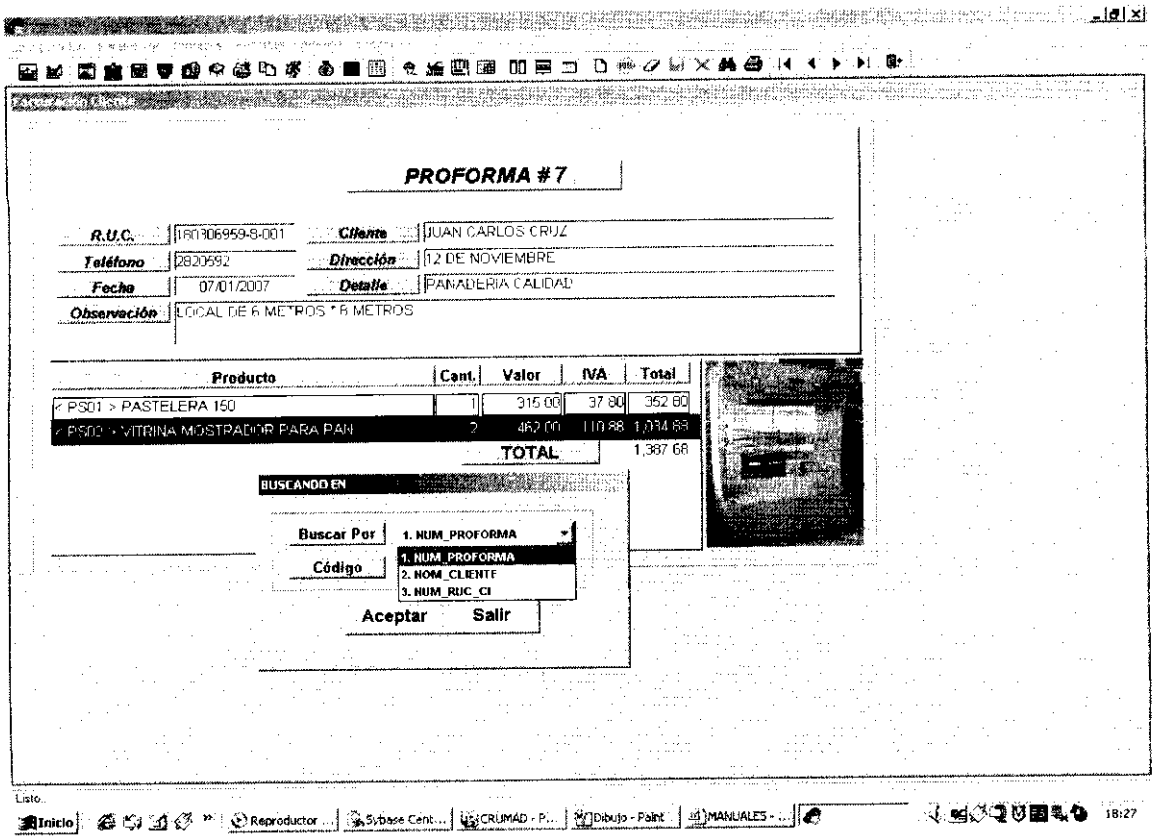


Si todo está correcto se procede a guardar esta información mediante la opción **GUARDAR**, caso contrario se procede a la actualización y su posterior almacenamiento. Esta proforma es el valor de varios muebles modulares que nos va a servir posteriormente para el siguiente proceso.

Además en este proceso tenemos las opciones de Editar, Borrar, Imprimir, Buscar; tanto en la barra de herramientas como dentro del menú **Edición**.

En la opción de buscar tenemos búsqueda por:

- Numero de proforma
- Nombre de Cliente
- RUC o Cedula del cliente

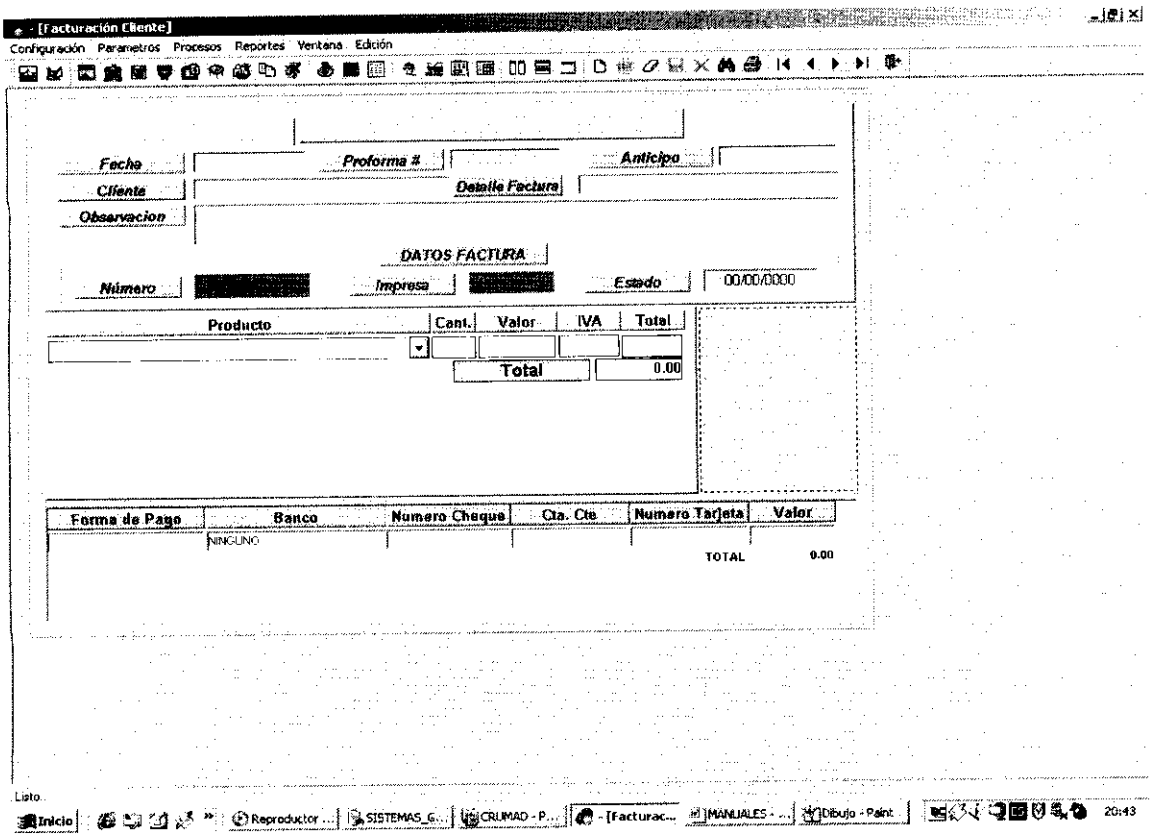


Proceso Proforma

Para efectuar este proceso se ingresa a la opción Procesos, Proforma; o en la barra de herramientas damos click sobre el icono .



Aparece la pantalla que se ilustra a continuación.

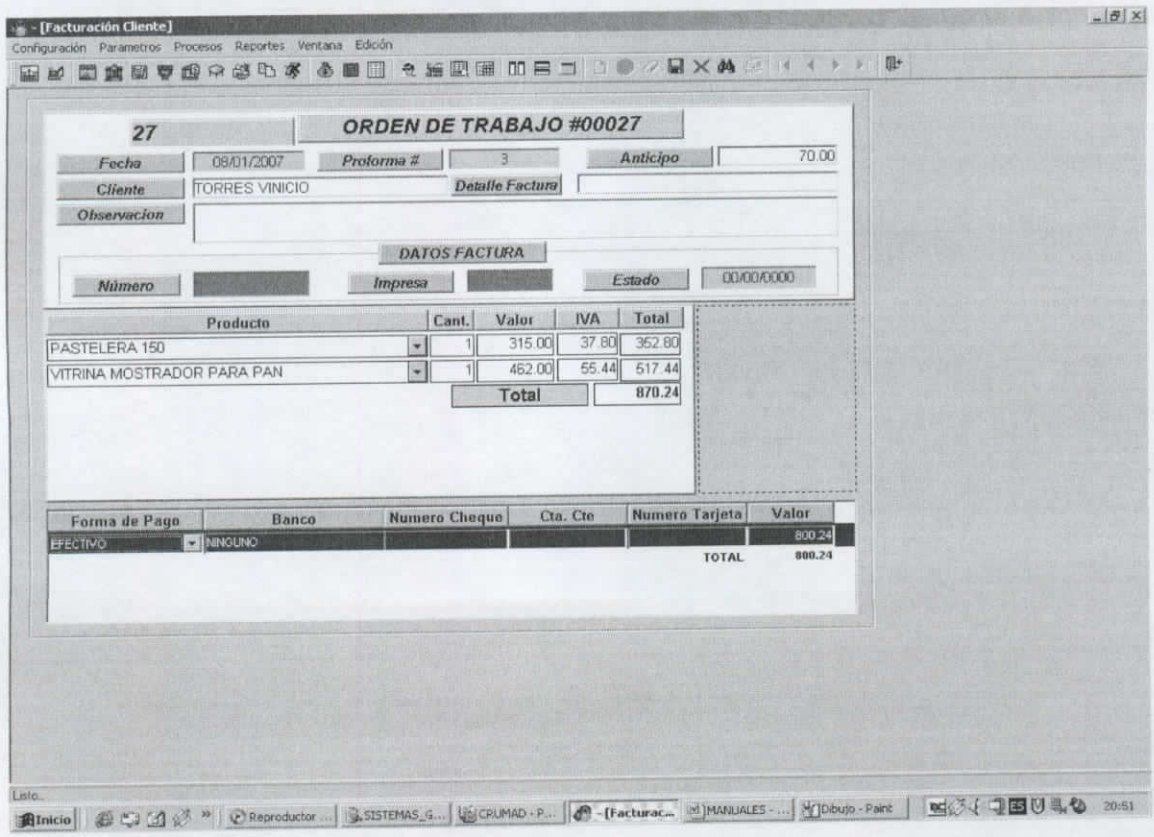


Este proceso nos permite recuperar proformas realizadas anteriormente para proceder a la respectiva facturación, o antes cuenta con una opción que es la Orden de Trabajo donde nos permite registrar el anticipo y también nos permite imprimir esta orden con opciones de edita si hay cambios.

Una vez realizada la orden de trabajo se procede a la opción de facturación donde una vez impresa la factura no se podrá realizar modificaciones al documento.

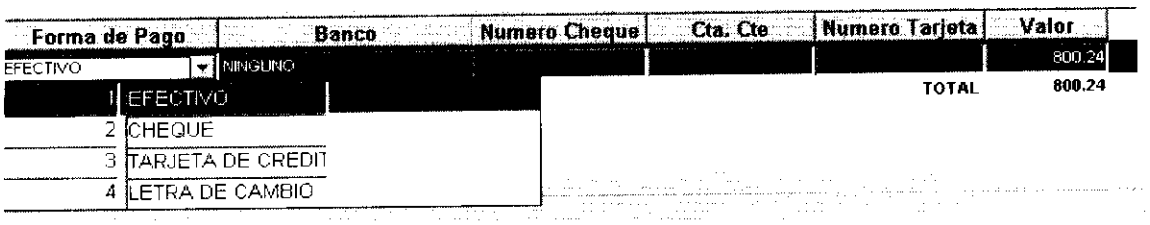
En esta pantalla seleccionamos la opción **NUEVO,** automáticamente se nos genera un número de Orden de trabajo y se ingresa la siguiente información:

- # de Proforma en caso de que queramos ingresar una proforma anteriormente generada.
- Valor de anticipo
- Cliente, si el cliente consta en nuestra base de datos de **Clientes** nos recupera automáticamente toda la información que tenemos de este cliente, caso contrario la ingresamos y se almacenara posteriormente cuando la guardemos en la base de clientes.
- Detalle
- Observación
- Producto
- Cantidad
- Valor



- Formas de Pago, en esta opción tenemos las formas de pago:

- Efectivo
- Cheque
- Tarjeta de crédito
- Letra de cambio



Además de esto el sistema se encarga de completar los campos correspondientes a:

- Fecha
- IVA
- Total

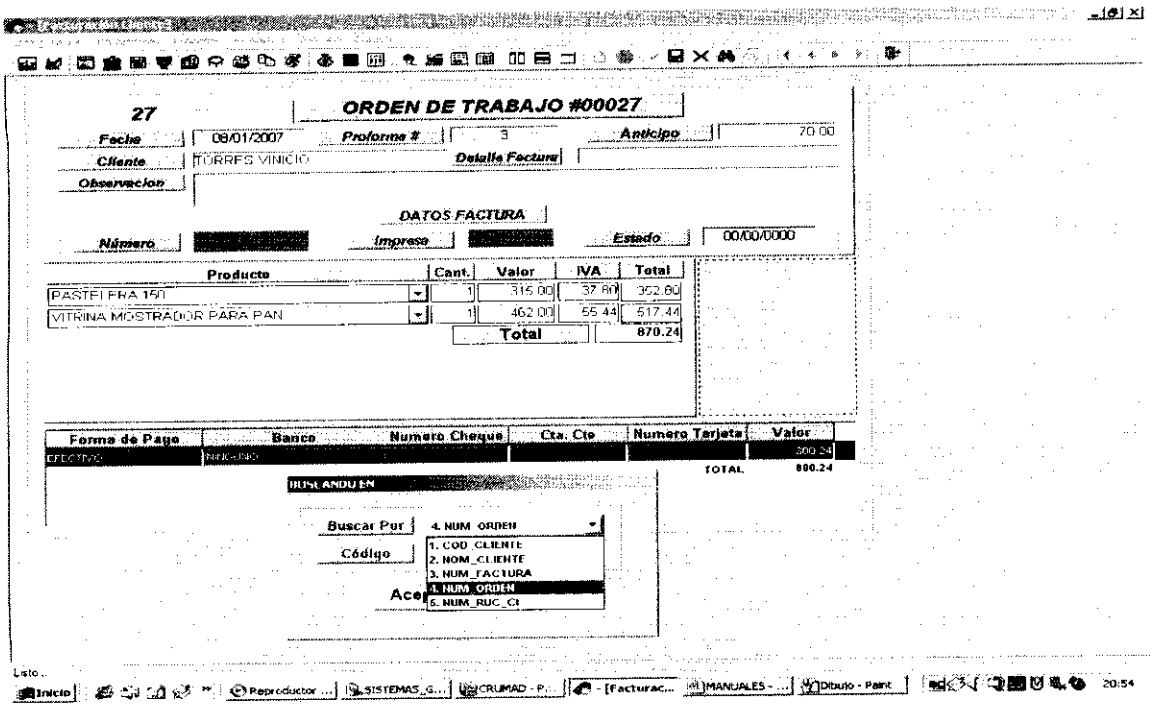
Si todo está correcto se procede a guardar esta información mediante la opción **GUARDAR**, caso contrario se procede a la actualización y su posterior almacenamiento. Este documento es el valor que se deberá pagar el momento en que el cliente acepta cualquier proforma que se realizo en el proceso anterior.

Además en este proceso tenemos las opciones de Editar, Borrar, Imprimir, Buscar; tanto en la barra de herramientas como dentro del menú **Edición**.

En la opción de buscar tenemos búsqueda por:

- Código del cliente
- Nombre de Cliente
- RUC o Cedula del cliente
- Numero de Factura

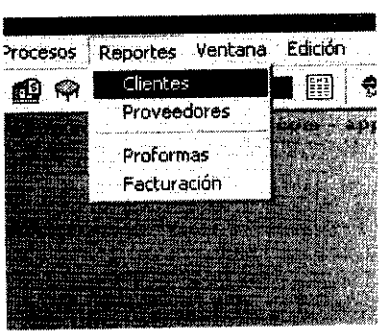
- Numero de Orden de Trabajo



REPORTES

Reporte de Clientes

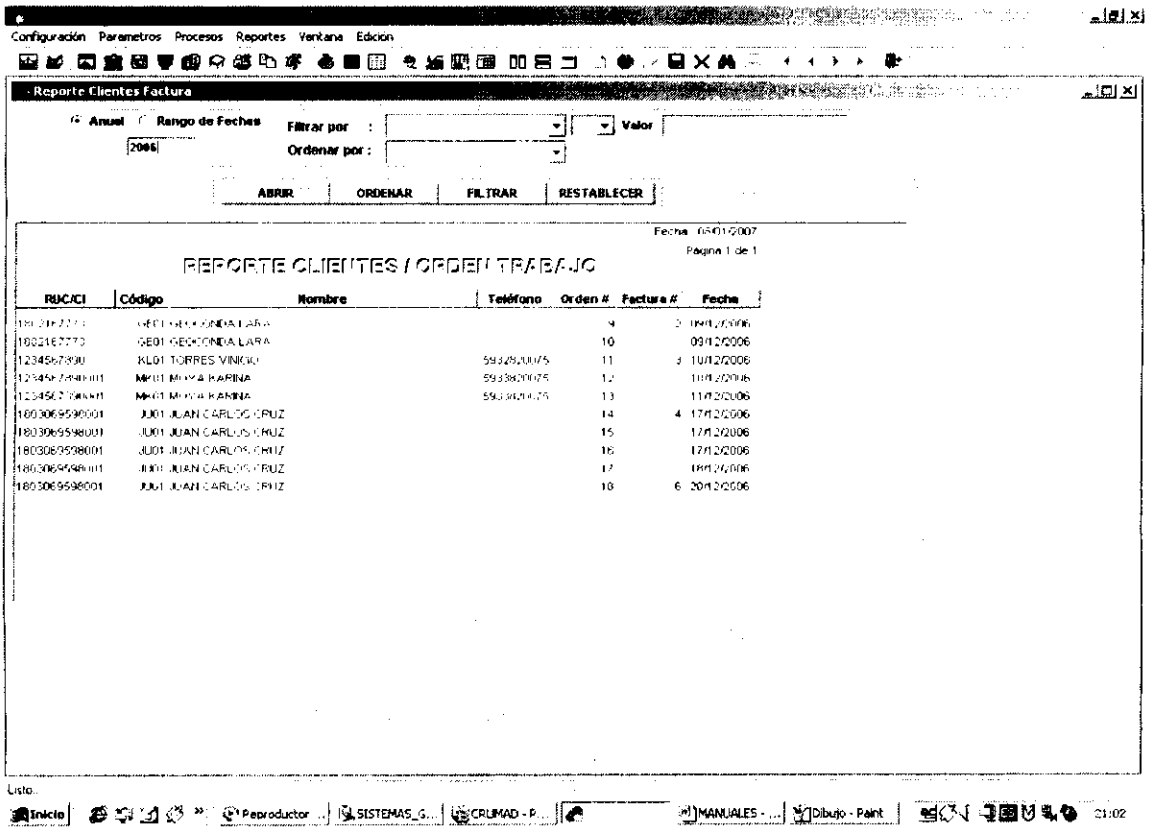
Para efectuar este proceso se ingresa a la opción Reportes, Clientes; o en la barra de herramientas damos click sobre el icono .



Este es un resumen de todos los Clientes que constan en nuestra base de datos a los que se les ha efectuado una determinada orden de trabajo y/o se les emitió una factura.

Todos y cada uno de los reportes tienen una misma característica y es el uso de rangos de fechas y un botón **Abrir o Enter** para generarlos.

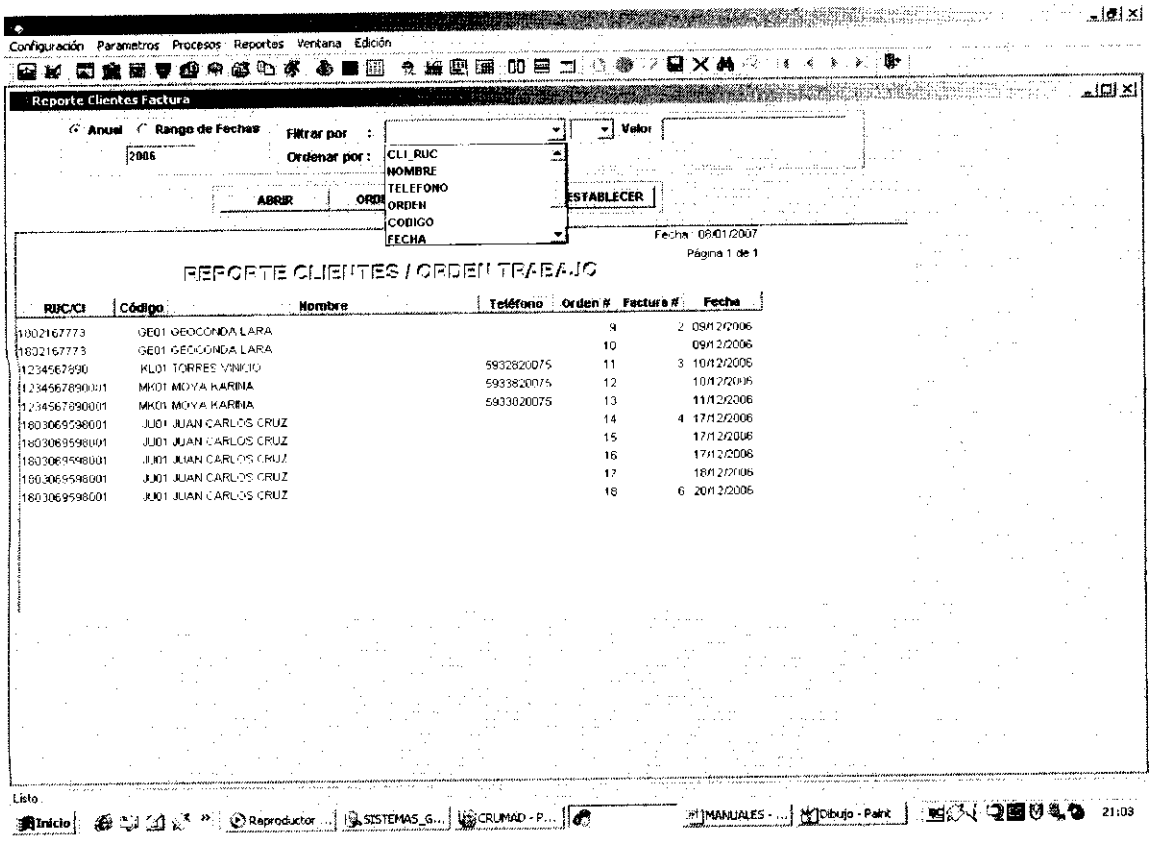
Una vez ingresado el Rango de fechas o el año en caso de reporte anual nos despliega la pantalla que se ilustra a continuación.



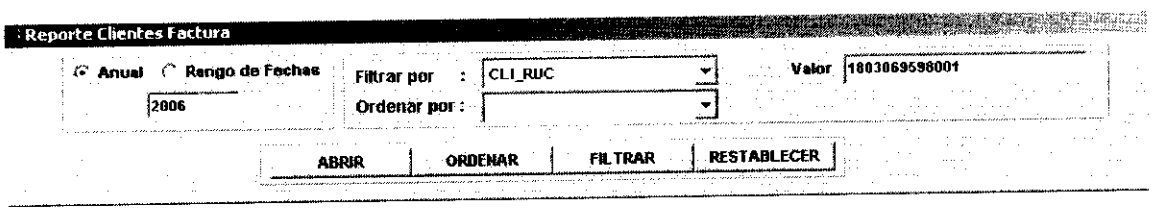
Si queremos realizar un filtro de los datos que aparecen en la pantalla lo podemos hacer por:

- RUC o Cedula
- Nombre

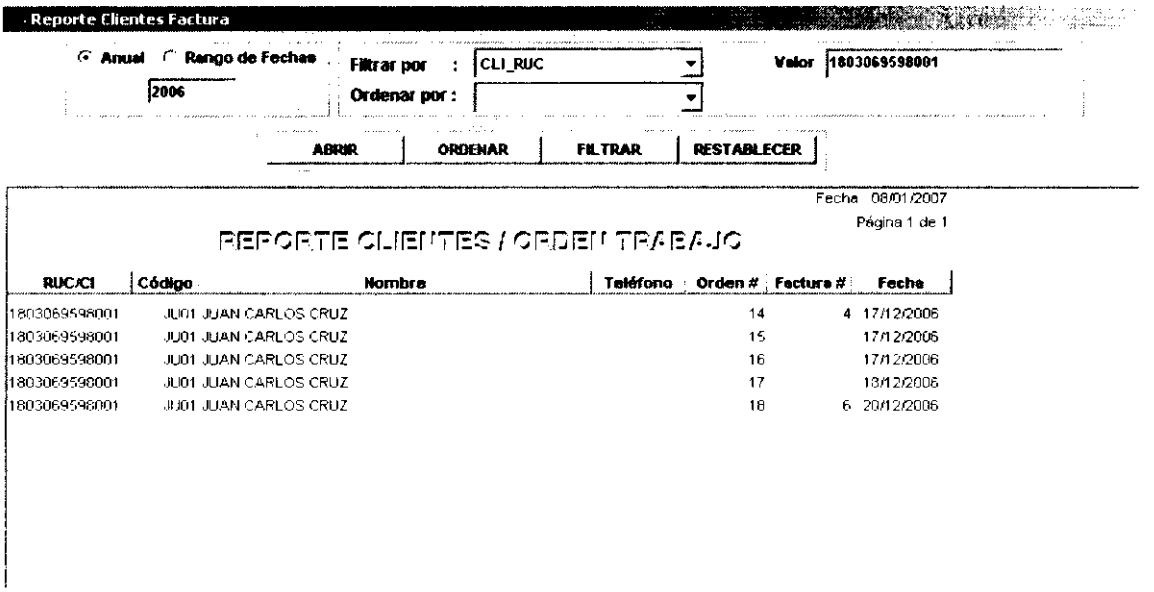
- Teléfono
- Orden de trabajo
- Código
- Fecha
- Factura



Seleccionamos el tipo de filtro que vamos a efectuar e ingresamos el valor correspondiente de la siguiente manera:

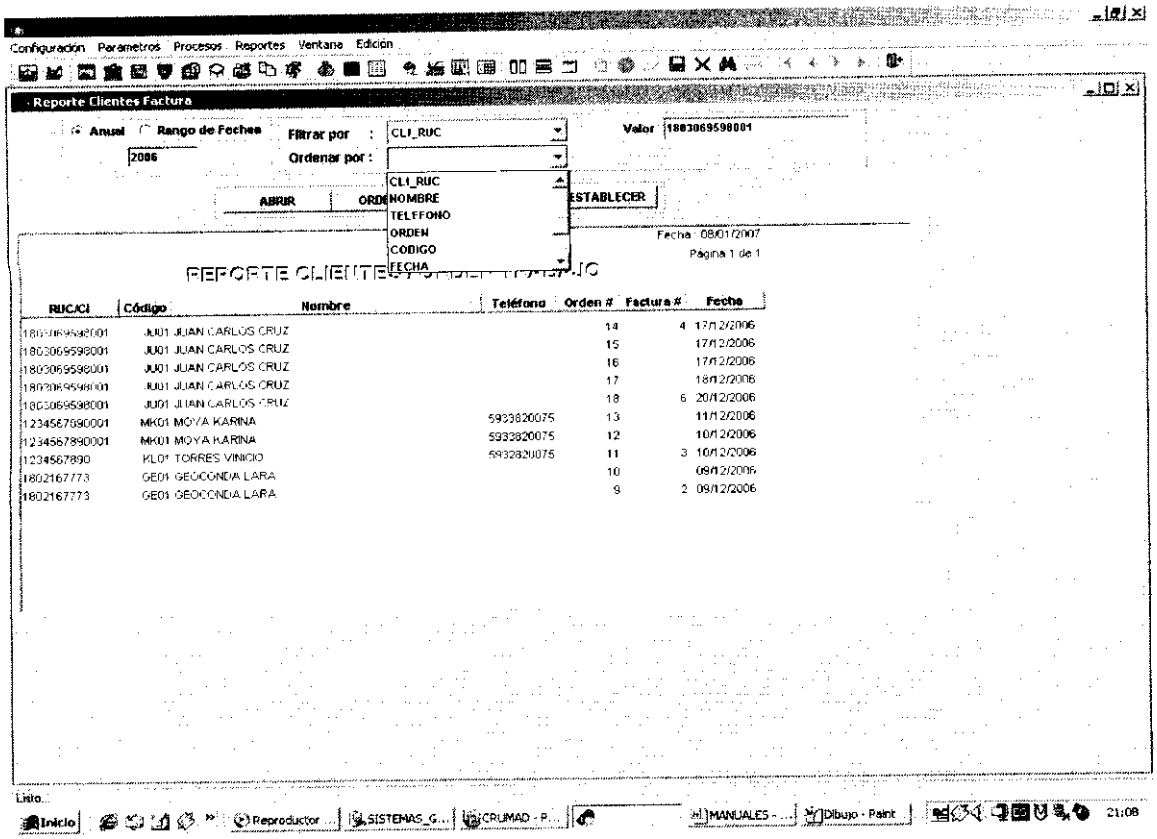


A continuación presionamos filtrar o Enter y nos filtra por el motivo y el valor que elegimos así:

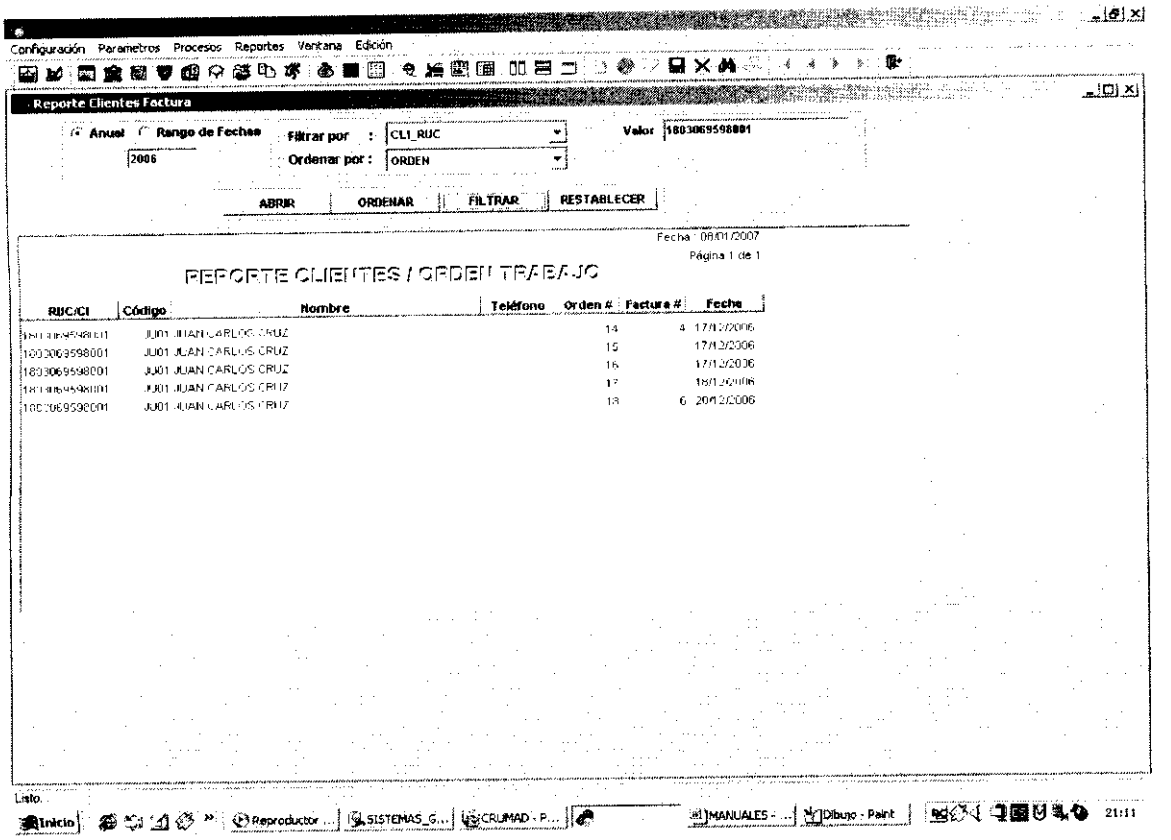


La última opción de los reportes es la de ordenar las opciones de orden son por:

- RUC o Cedula
- Nombre
- Teléfono
- Orden de trabajo
- Código
- Fecha
- Factura

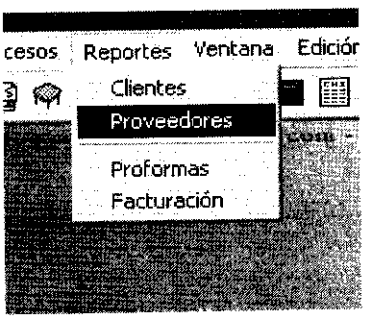


Se nos ordenara por el motivo que hayamos elegido:



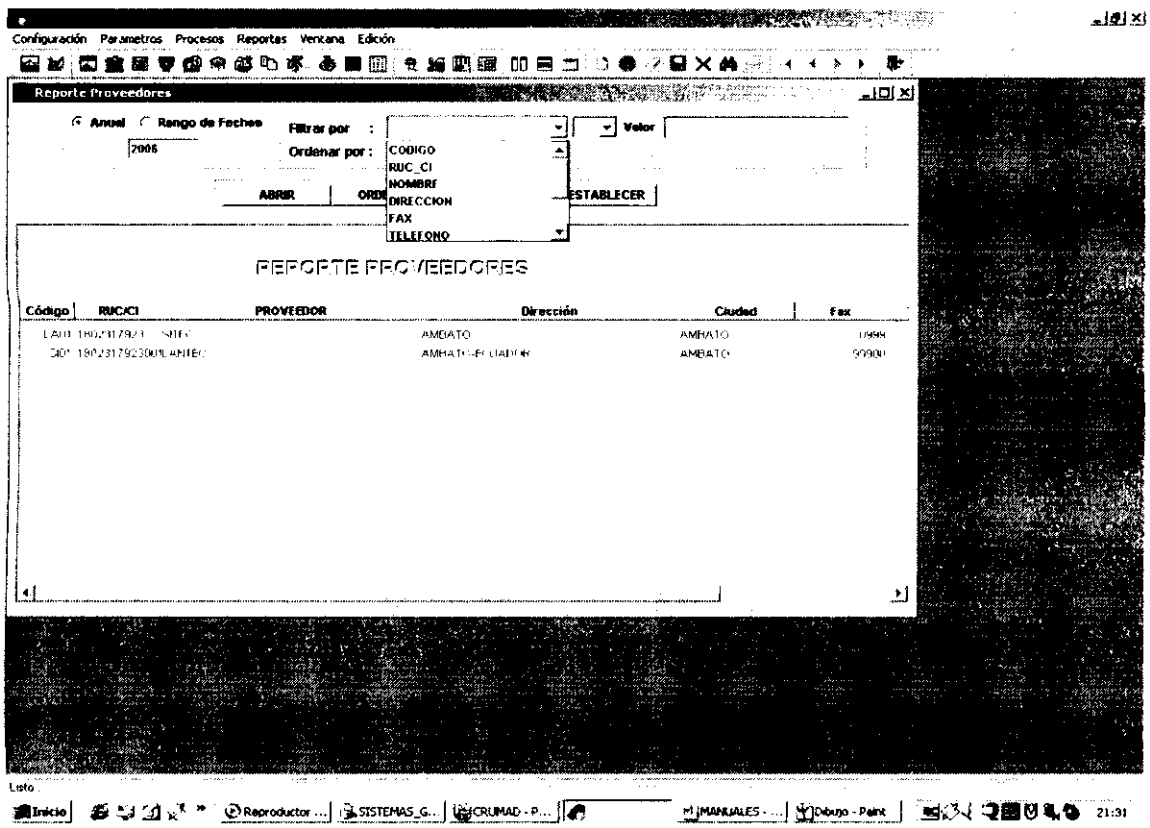
Reporte de Proveedores

Para efectuar este proceso se ingresa a la opción Reportes, Proveedores; o en la barra de herramientas damos click sobre el icono .

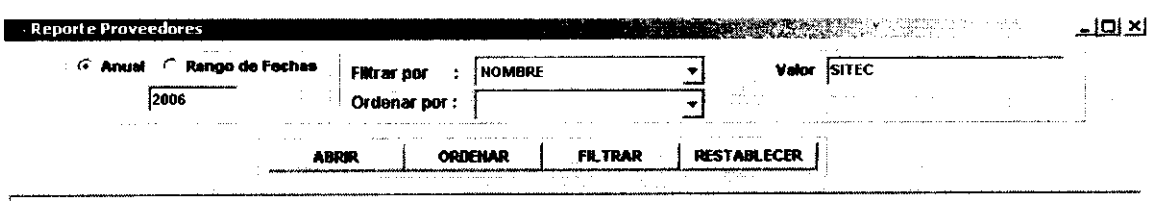


Este es un resumen de todos los Proveedores con los que cuenta la empresa que constan en nuestra base de datos a los que se les ha efectuado una determinada compra.

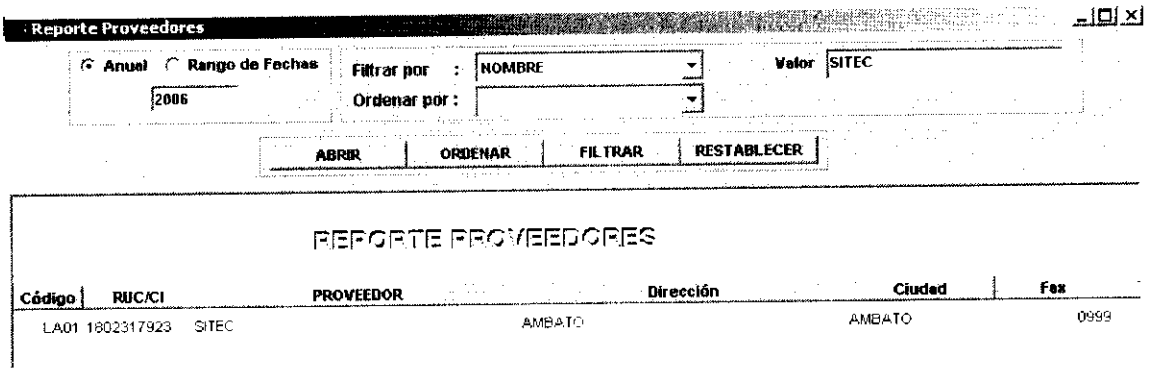
- Dirección
- Fax
- Teléfono
- E mail
- Ciudad



Seleccionamos el tipo de filtro que vamos a efectuar e ingresamos el valor correspondiente de la siguiente manera:

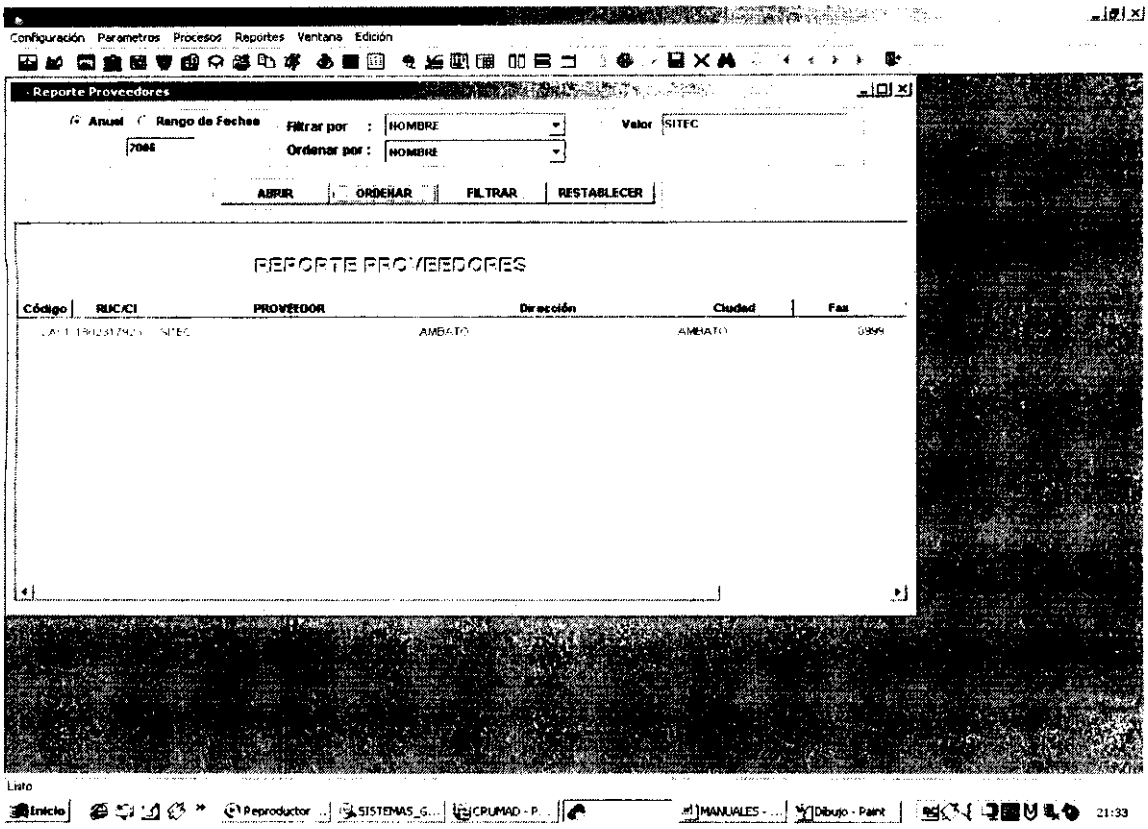


A continuación presionamos filtrar o Enter y nos filtra por el motivo y el valor que elegimos así:



La última opción de los reportes es la de ordenar las opciones de orden son por:

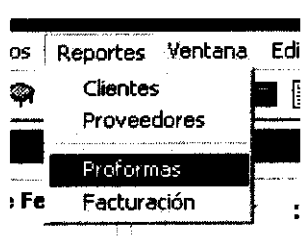
- Código
- RUC o Cedula
- Nombre
- Dirección
- Fax
- Teléfono
- E mail
- Ciudad



Se nos ordenara por el motivo que hayamos elegido.

Reporte de Proformas

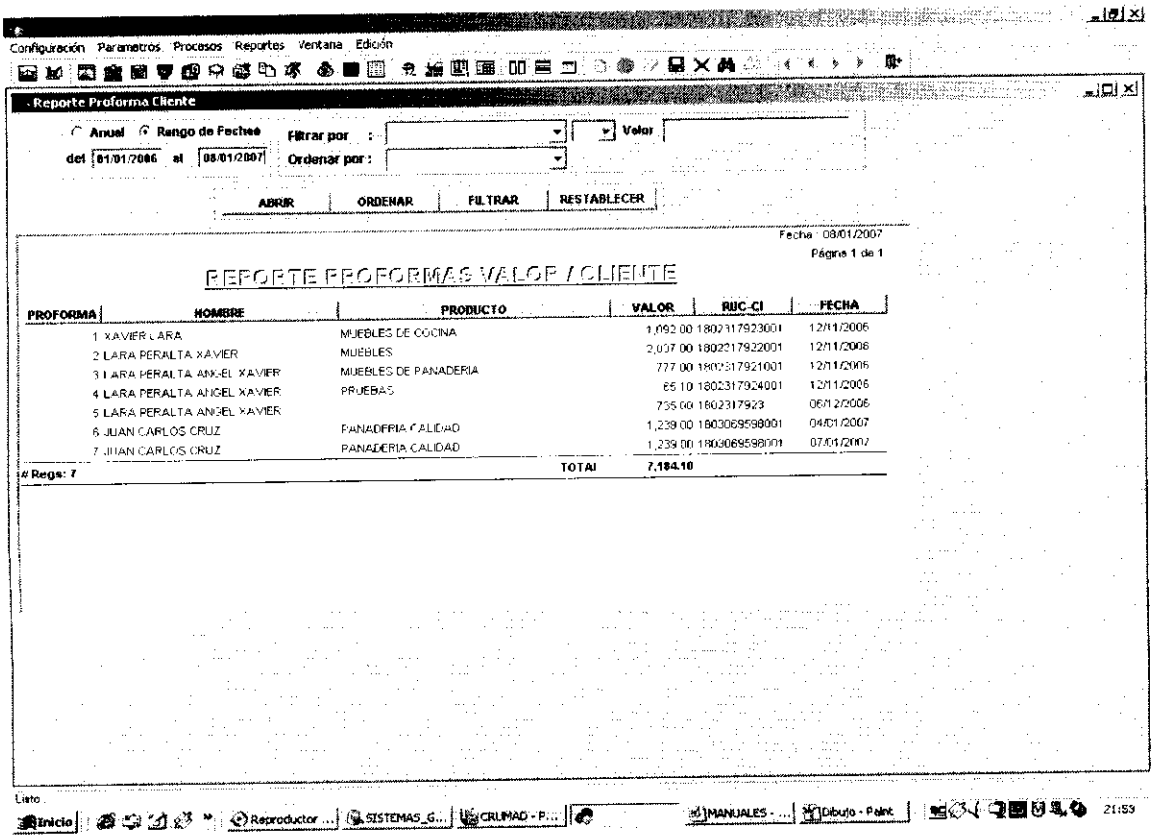
Para efectuar este proceso se ingresa a la opción Reportes, Proformas; o en la barra de herramientas damos click sobre el icono.



Este es un resumen de todas las proformas realizadas a determinados clientes en donde nos especifica el cliente al cual se le realizo la proforma, el producto y el valor total de la proforma, y la fecha de realización de la misma.

Todos y cada uno de los reportes tienen una misma característica y es el uso de rangos de fechas y un botón **Abrir o Enter** para generarlos.

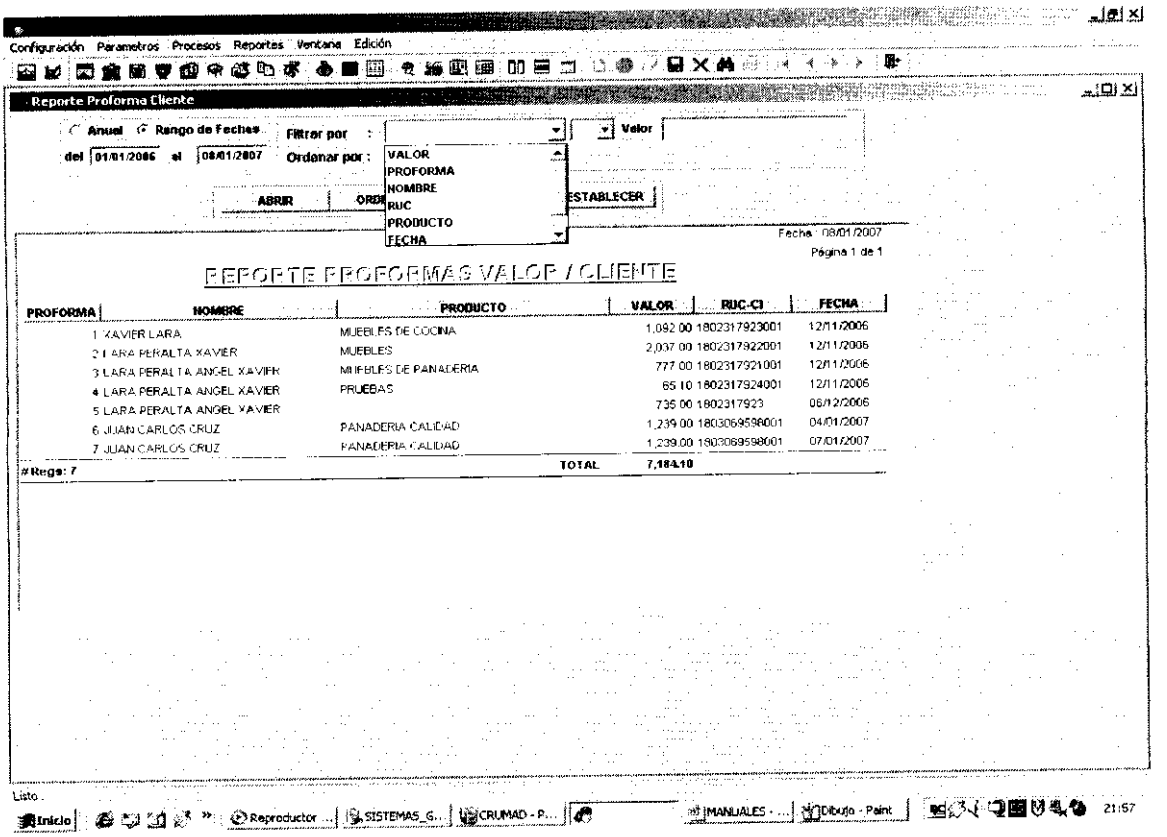
Una vez ingresado el Rango de fechas o el año en caso de reporte anual nos despliega la pantalla que se ilustra a continuación.



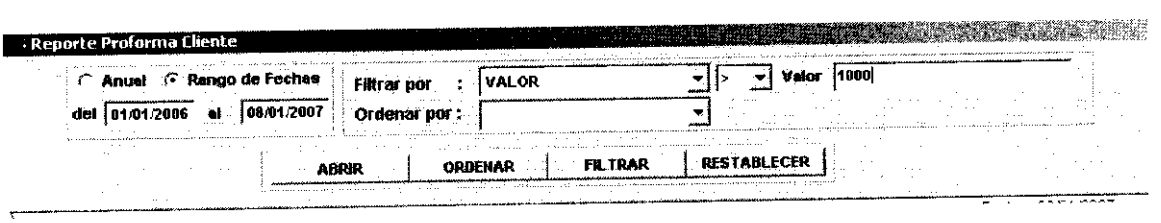
Si queremos realizar un filtro de los datos que aparecen en la pantalla lo podemos hacer por:

- Valor

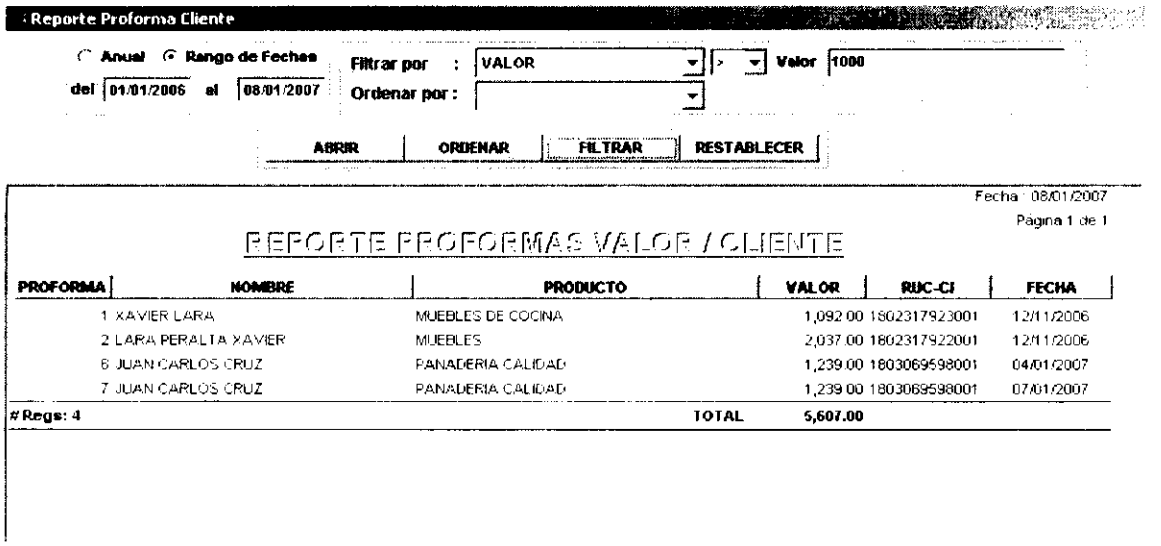
- Proforma
- Nombre
- RUC o Cedula
- Producto
- Fecha



Seleccionamos el tipo de filtro que vamos a efectuar e ingresamos el valor correspondiente de la siguiente manera:

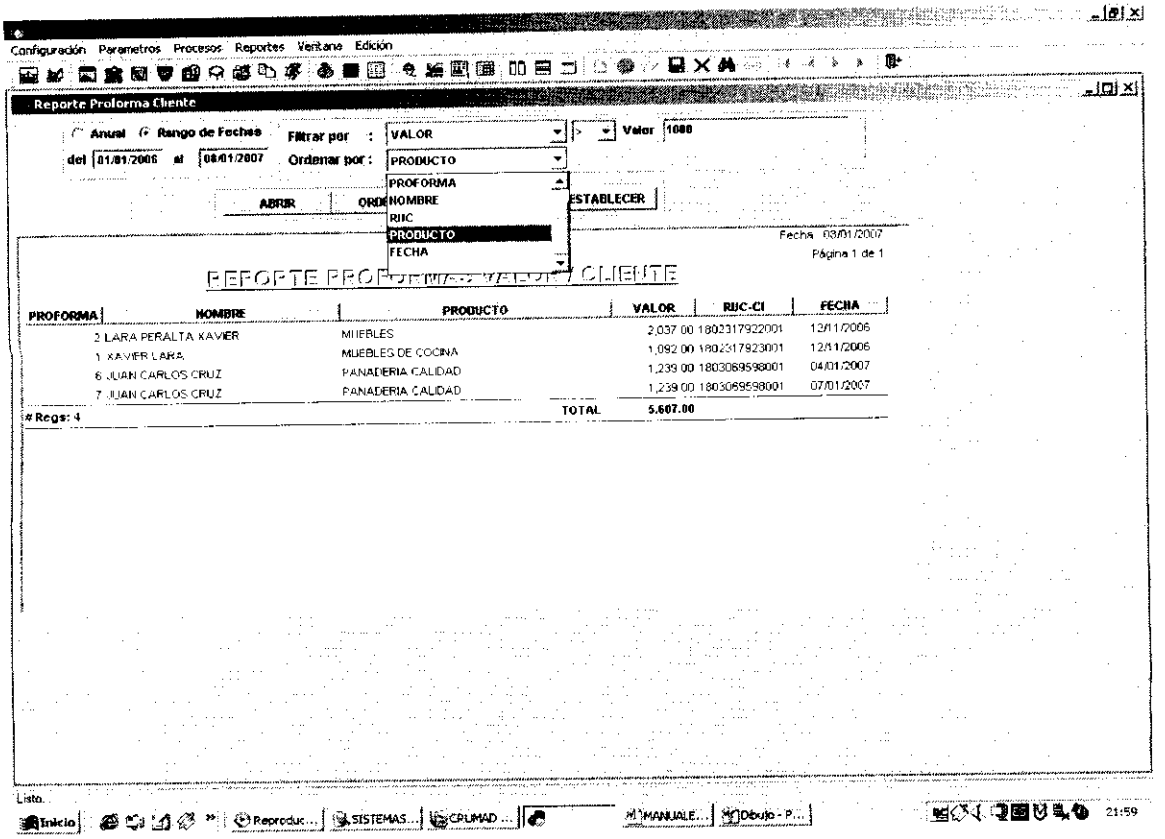


A continuación presionamos filtrar o Enter y nos filtra por el motivo y el valor que elegimos así:

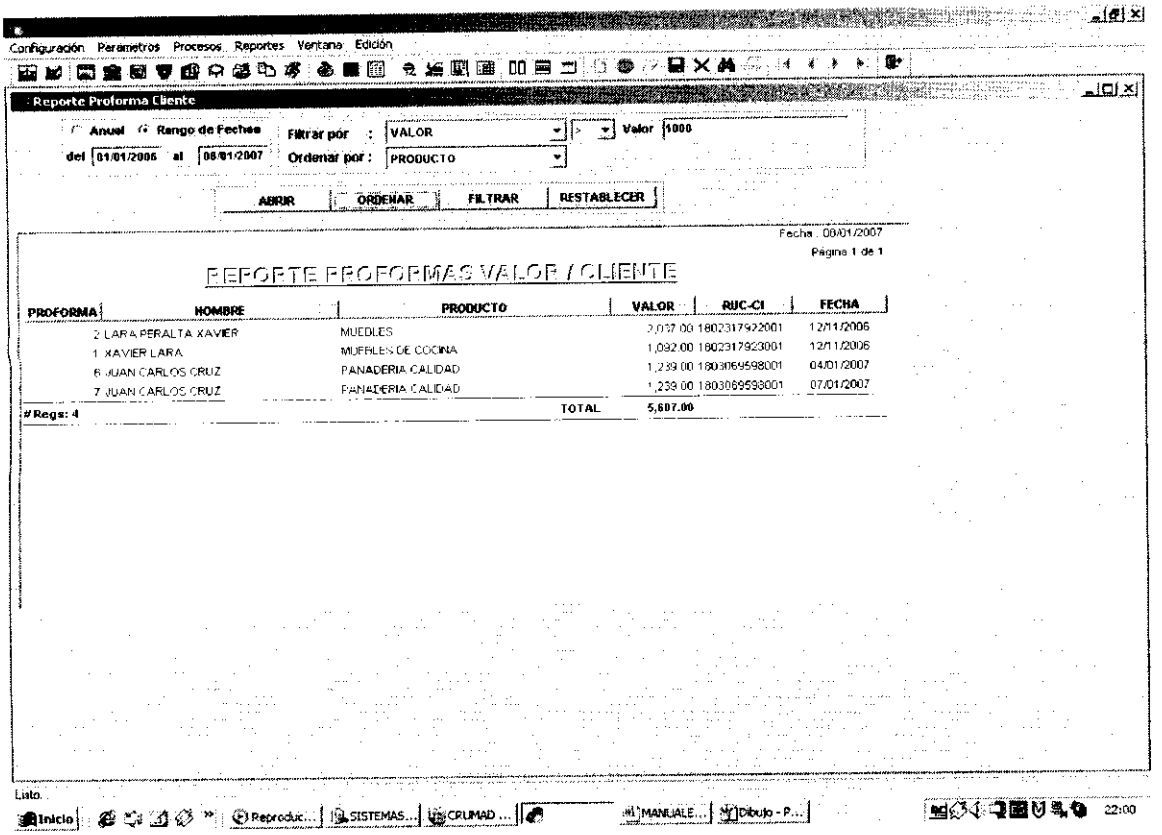


La última opción de los reportes es la de ordenar las opciones de orden son por:

- Valor
- Proforma
- Nombre
- RUC o Cedula
- Producto
- Fecha

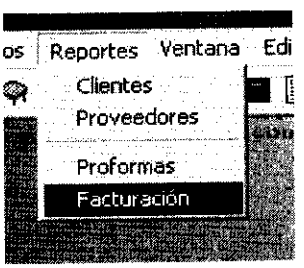


Se nos ordenara por el motivo que hayamos elegido:



Reporte de Facturación

Para efectuar este proceso se ingresa a la opción Reportes, Facturación; o en la barra de herramientas damos click sobre el icono .



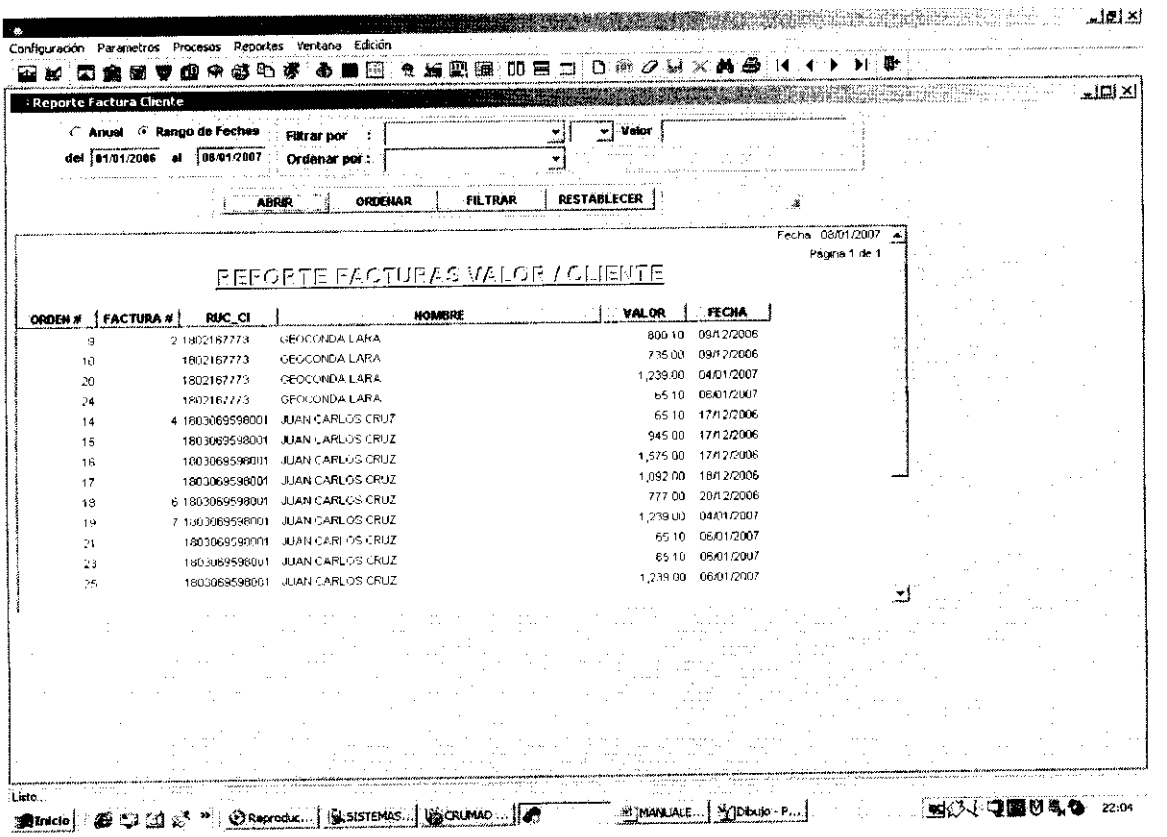
Este es un resumen de todas las Ordenes de trabajo y facturas realizadas a determinados clientes en donde nos especifica el cliente al cual se le realizo la Orden o se le facturo,

el numero de orden, el numero de factura, RUC, Valor total de la orden y/o factura y la fecha de realización de las mismas.

Además nos totaliza los valores de las órdenes y/o facturas realizadas en los periodos seleccionados y en los filtros que realicemos.

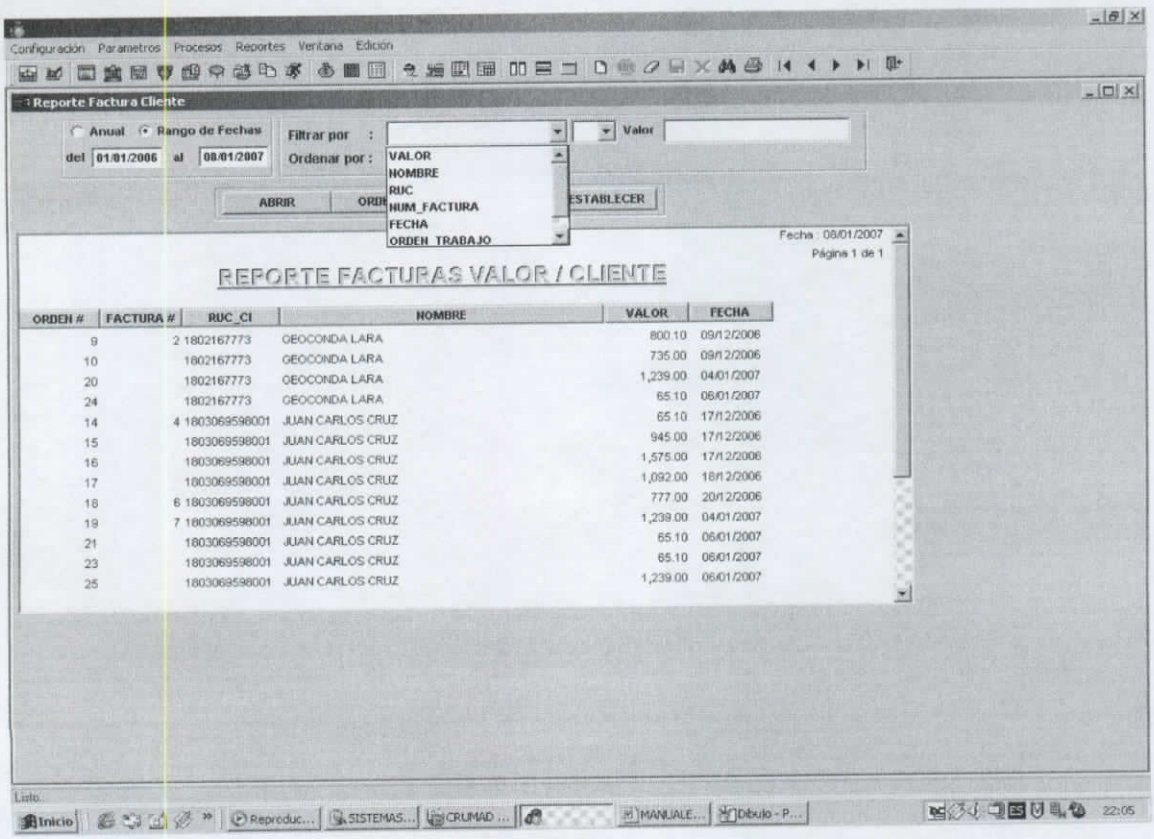
Todos y cada uno de los reportes tienen una misma característica y es el uso de rangos de fechas y un botón **Abrir o Enter** para generarlos.

Una vez ingresado el Rango de fechas o el año en caso de reporte anual nos despliega la pantalla que se ilustra a continuación.

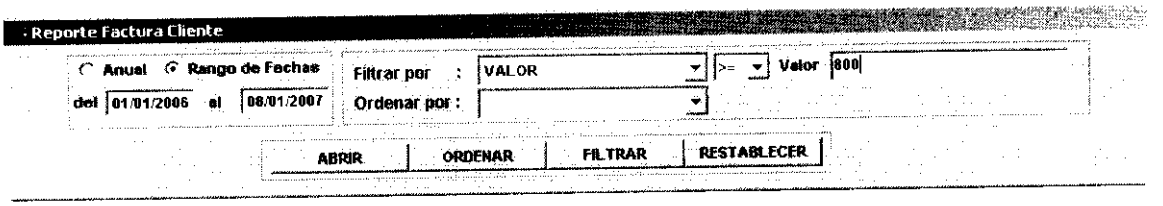


Si queremos realizar un filtro de los datos que aparecen en la pantalla lo podemos hacer por:

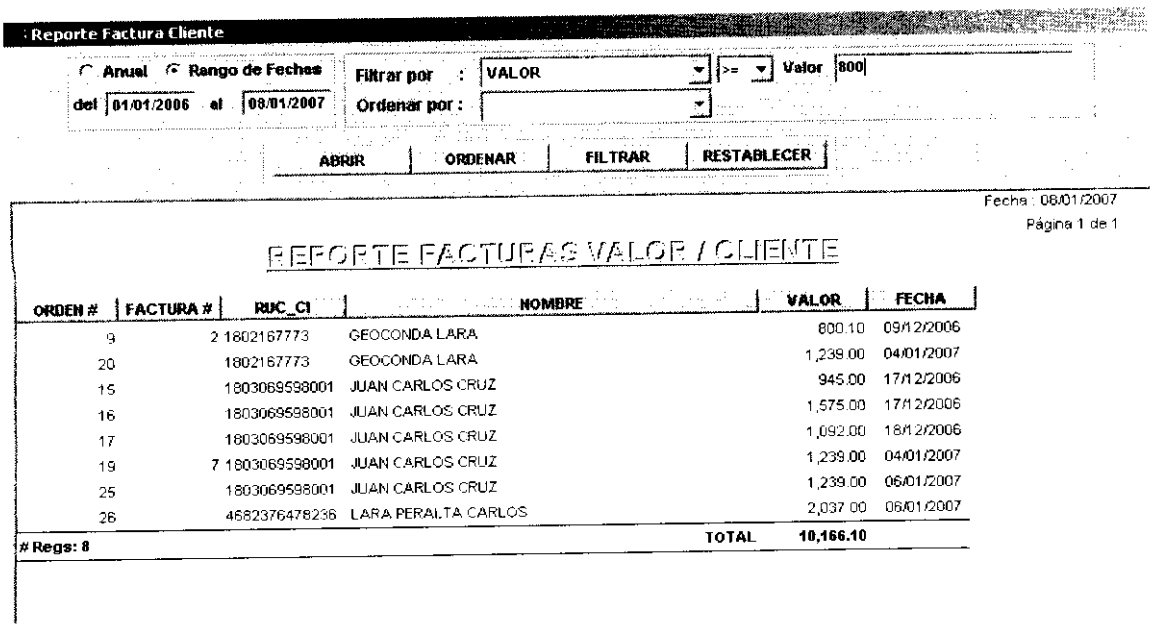
- Valor
- Nombre
- Número de factura
- RUC o Cedula
- Número de orden
- Fecha



Seleccionamos el tipo de filtro que vamos a efectuar e ingresamos el valor correspondiente de la siguiente manera:

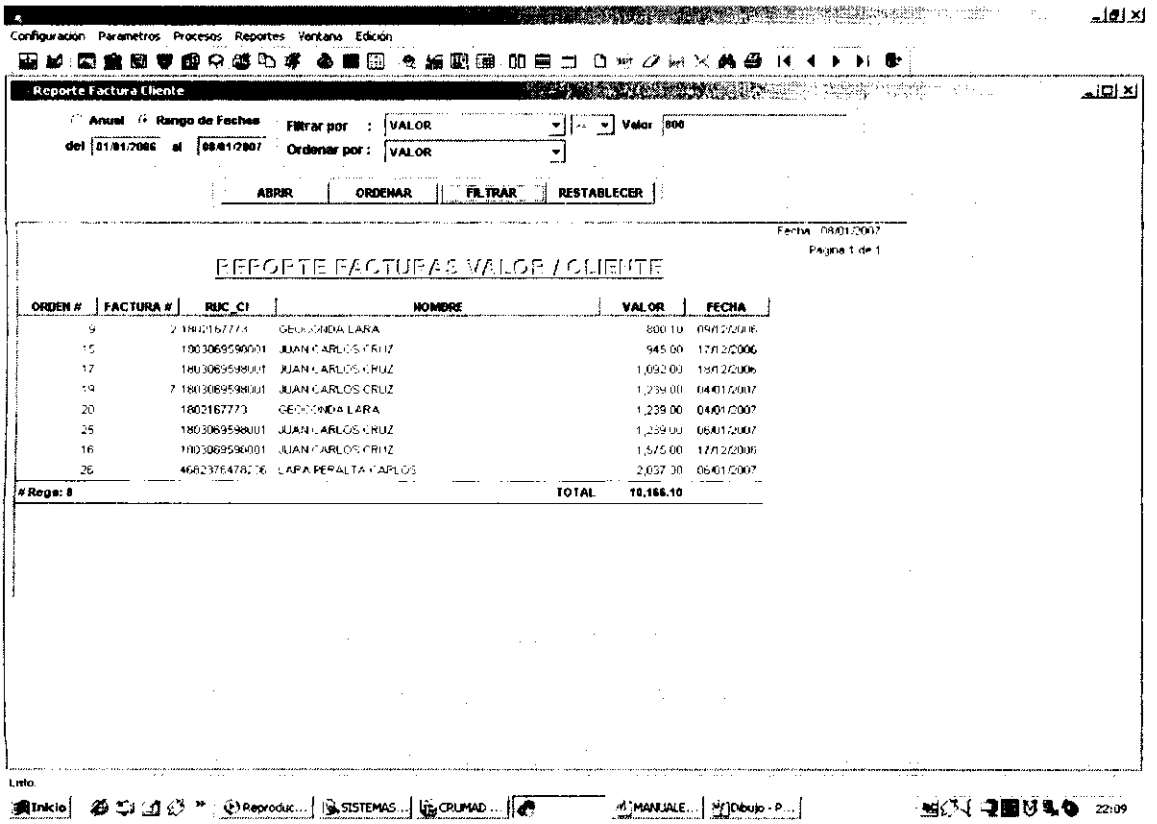


A continuación presionamos filtrar o Enter y nos filtra por el motivo y el valor que elegimos así:

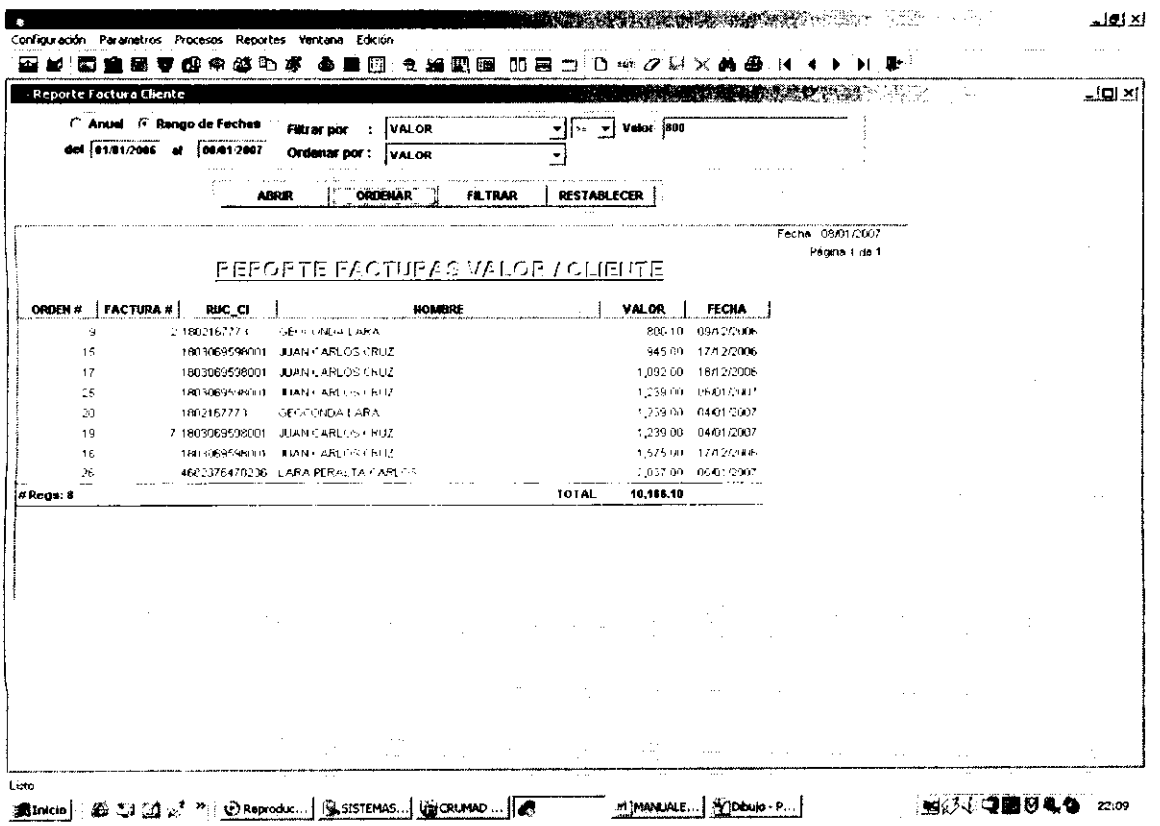


La última opción de los reportes es la de ordenar las opciones de orden son por:

- Valor
- Nombre
- Número de factura
- RUC o Cedula
- Número de orden
- Fecha



Se nos ordenara por el motivo que hayamos elegido:



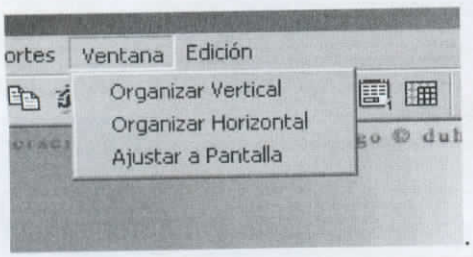
Adicionalmente el sistema posee una opción para ajuste de ventanas.

Para ello se ingresa a la opción Ventana, y elegimos el tipo de ajuste que deseamos efectuar a la pantalla; o en la barra de herramientas damos click sobre los iconos:

 Ajuste Vertical

 Ajuste Horizontal

 Ajuste de Pantalla



PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
SEDE AMBATO
BIBLIOTECA