CAPÍTULO V

5. DISEÑO E IMPLEMENTACIÓN

Este capítulo incluye los detalles del diseño utilizado para llevar a cabo este proyecto y las particularidades al momento de la implementación.

5.1. ARQUITECTURA DEL SISTEMA

Este sistema maneja una arquitectura de tres capas con tres niveles lógicos:

1. La capa de ***presentación*** ,
2. La capa de ***negocio***
3. La capa de ***datos*.**

Capa de Presentación simplemente contiene los componentes de interfaz de usuario, es decir los formularios del sistema

Capa de Negocio es el "puente" entre un usuario y los servicios de datos, ya que responden a peticiones del usuario (u otros servicios de negocios) para ejecutar una tarea de este tipo. Cumplen con esto aplicando procedimientos formales y reglas de negocio a los datos relevantes. Cuando los datos necesarios residen en un servidor de base de datos, garantizan los servicios de datos indispensables para cumplir con la tarea de negocios o aplicar su regla. Esto aísla al usuario de la interacción directa con la base de datos.

El nivel de servicios de negocios es responsable de:

- Recibir la entrada del nivel de presentación (Formularios).
- Interactuar con los servicios de datos para ejecutar las operaciones de negocios para los que la aplicación fue diseñada a automatizar en este caso, el procesamiento de consultas de estudiantes, materias, etc).
- Enviar el resultado procesado al nivel de presentación.

.

Capa de Datos se suele incluir tan sólo la base de datos, aunque a veces también un conjunto de componentes que quedan entre los del

negocio y los de la base de datos. Haya o no componentes en esta capa, la base de datos deberá tener implementada alguna lógica del negocio en procedimientos almacenados (Store Procedures) y disparadores (Triggers).

5.1.1 Diseño de la base de datos

Tan importante como obtener los datos y disponer de un medio para guardarlos, es que dicho medio nos permita extraer, filtrar, organizar y seleccionar conforme a nuestras necesidades. La solución a este problema la proporcionan las aplicaciones denominadas manejador de base de datos

Un manejador de base de datos se puede describir, como un contenedor de información, que organiza la misma en base a una serie de reglas. Dicha información puede ser manipulada mediante un conjunto de instrucciones que permitirán al usuario consultar y modificar los datos contenidos.

La administración de un manejador de datos, por otra parte, se puede definir como el conjunto de labores cuyo objetivo es conseguir un rendimiento óptimo del sistema de base de datos, de forma que la información esté en todo momento disponible y

con el menor tiempo de espera posible para el usuario que la solicita.

Bases de datos en SQL Server 2000.

Para el presente trabajo se utilizara el DBMS “Microsoft SQL Server 2000”, el cual es un manejador de base de datos relacionales compuesto por un conjunto de elementos, que se integran con los sistemas operativos NT de Windows (Windows NT 4.0 Workstation y Server, Windows 2000 Profesional, Server y Advanced Server y Windows XP Professional), y el resto de la familia de productos empresariales de Microsoft, BackOffice, para proporcionar un entorno avanzado de proceso de datos, dentro de una arquitectura cliente-servidor, (en próximos apartados trataremos el concepto cliente-servidor).

El rendimiento conseguido por SQL Server 2000 al ejecutarse en sistemas Windows NT, ediciones Server, es excelente, debido a la mencionada orientación cliente-servidor, y a los componentes específicamente desarrollados en estos sistemas operativos para la ejecución de SQL Server.

Algunas de las ventajas del trabajo conjunto entre SQL Server y los sistemas NT de Windows se enumeran a continuación:

- SQL Server aprovecha las características multiproceso de Windows NT, utilizando todos los procesadores instalados para optimizar el manejo de datos.

- El sistema de seguridad de SQL Server 2000 está conformado por un usuario creado en la base de datos. De manera que el usuario inicia su sesión en el sistema y automáticamente se conecta con SQL Server, ye establece una relación de confianza en la que SQL Server verifica si el usuario y la clave de acceso es correcto.

- Para las labores de supervisión del funcionamiento, SQL Server aprovecha el Visor de sucesos del sistema operativo para insertar sus propios mensajes, unificando en un sólo lugar el sistema de avisos. De igual modo, utiliza el Monitor del sistema de Windows NT para aspectos relacionados con el rendimiento de las base de datos.

- Sobre la disponibilidad inmediata de los datos en casos de fallo del servidor, SQL Server aprovecha las capacidades de clustering de que dispone Windows NT Server, de forma que si en un sistema se han instalado dos servidores en clúster, SQL Server realizará el cambio al servidor de respaldo en el caso de que se produzca una caída del principal.

SQL Server puede ejecutarse en una amplia variedad de sistemas operativos. Dependiendo del sistema, podrá actuar como cliente o servidor. Los sistemas a los que se proporciona capacidad de servidor son:

- Windows NT en cualquiera de sus ediciones (Server,y Workstation) y Windows 9x. En cuanto a los sistemas para los que SQL Server dispone de elementos de cliente están los antes mencionados en el aspecto de servidor más Windows 3.x, MS-DOS, Macintosh y UNIX.

- Finalmente, fuera del ámbito de sistemas, también puede ejecutarse como cliente dentro de los navegadores de Internet.
- En cuanto a la escalabilidad del motor de datos, puede manejar desde pequeñas base de datos en modo local a grandes base de datos con conexiones de miles de usuarios y más de un terabyte de capacidad de almacenamiento.

Una base de datos está formada por una serie de elementos, también denominados objetos de la base de datos, que permiten organizar la información, relacionarla con otros objetos de la base de datos, mantener su integridad, etc.

Entre los objetos más importantes podemos destacar los siguientes:

• Tabla.

• Índice.

• Vista.

• Procedimiento almacenado.

• Función o rol.

• Desencadenador.

El sistema desarrollado para esta tesis necesita de una base de datos con una estructura en la que se han implementado Procedimientos almacenados para todos lo procesos que impliquen llamadas a base de datos, es decir que las sentencias de consultas, inserciones, modificaciones y eliminación de registros no se encuentran en el código de VB, sino en la base de datos aumentando notablemente el rendimiento y la optimización de procesos ya que todo debe estar centralizado en el DBMS.

La estructura de la base de datos mostrada en la figura 5.1, incluye lo siguiente:

Tablas primarias:

Ordenes.- Tabla donde se almacenan las ordenes que el usuario haya ingresado

Tablas de relación:

***Tests*.-** Tabla donde se almacenan los todos los tests que la Industria utiliza

***Flautas*.-** Tabla donde se almacenan las clase de flautas que la Industria utiliza

***TetsFlautas*.-** Tabla donde se encuentra todas las combinaciones posibles de los Tests y Flautas que en la producción Industrial se utiliza

***Corrugadoras*.-** Tabla donde se almacenan los nombres de las Corrugadotas que la Industria posee

***CorrugadorasFlautas*.-** Tabla donde se encuadra la asignación de las Flautas existentes a las Corrugadotas de la Industria

***Papeles*.-** Tabla donde se almacenan los tipos de papeles que en la producción industrial se utilizan

***Clientes*.-** Tabla donde se almacenan los clientes que la Industria tiene registrados

***Encargados*.-** Tabla donde se almacenan los encargados que la Industria posee

***Mercados*.-** Tabla donde se almacenan todos los tipos de mercados a los que la Industria se dirige

***Productos*.-** Tabla donde se almacenan todos los tipos de productos que la Industria maneja en su producción

Usuarios.- Tabla donde se almacenan los usuarios registrados para el uso del sistema, ya sea de tipo Administrador o No Administrador.

***Programaciones*.-** Tabla donde se guardan los encabezados de todas la programaciones, resultado del sistema y seleccionadas por el usuario.

***DetallesProgramaciones*.-** Tabla donde se guardan los detalles de todas la programaciones, resultado del sistema y seleccionadas por el usuario.

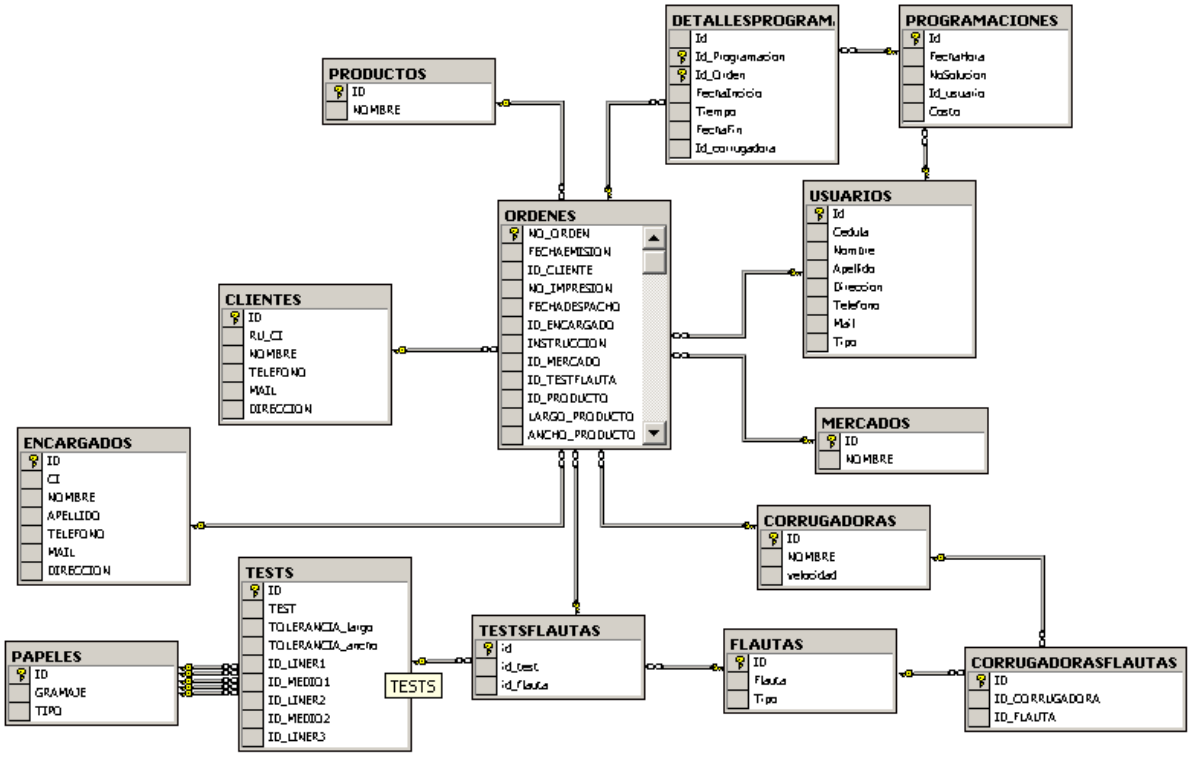

Figura 5.1. *Estructura de la base de datos.*

5.1.2 Diseño de la interfaz del sistema

A continuación presentamos el diseño de la interfaz del sistema propuesto para interactuar con el usuario. Se define la forma de realizar las consultas y la información mostrada, resultados del sistema, así como también la distinción de opciones de acuerdo al tipo de usuario.

Pantallas del Sistema:

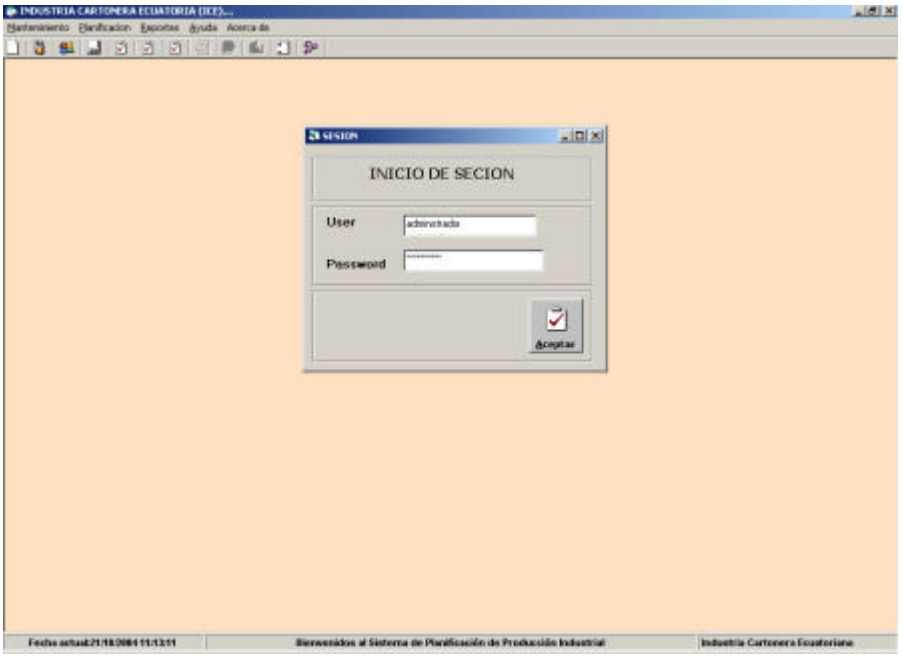

Figura 5.2. *Pantalla de Ingreso.*

En cuanto a la información de utilidad para el usuario Administrador, se encuentra en los formularios del menú de mantenimientos ya sea para:

- Cliente
- Encargado
- Producto
- Mercado

Todos los mantenimientos se efectúan de la misma manera así que mostraremos uno de ellos, el de Cliente

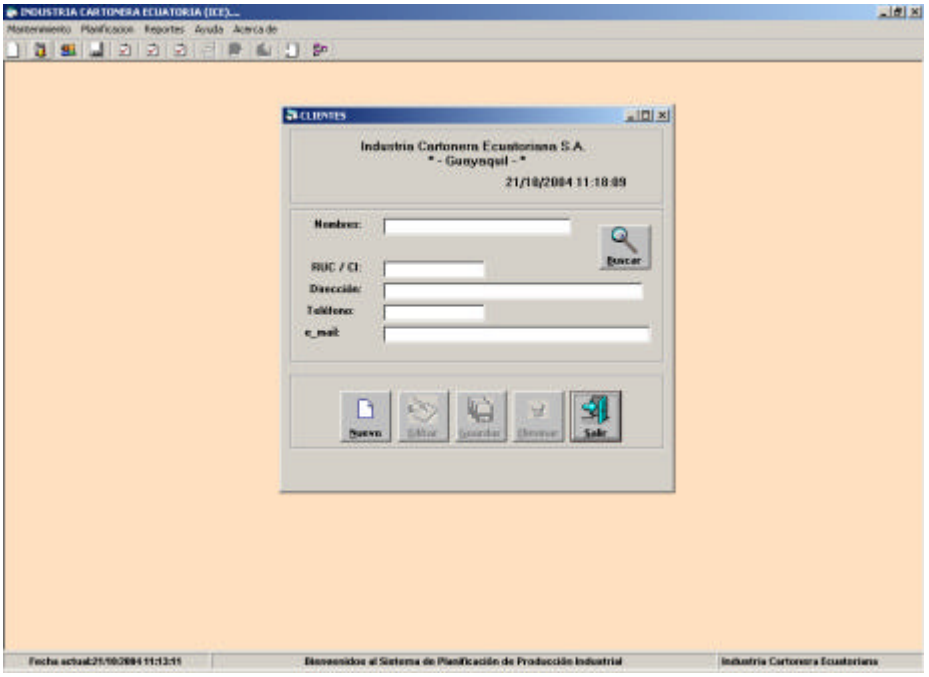

Figura 5.3. *Pantalla de Mantenimiento de Cliente*

Al hacer click en botón Nuevo le permite ingresar un cliente y al dar clic en el botón Guardar insertara en la base de datos el nuevo cliente.

El botón Buscar podrá visualizar una lista de clientes que constan en la base de datos le permitirá seleccionar uno y le mostrara los datos de dicho cliente en el formulario ya sea tan solo para consultar la información, o actualizar algún dato con el botón Actualizar o eliminar al cliente con el botón Eliminar

En este menú de Mantenimientos, si el usuario es No Administrador tendrá ciertas restricciones que no le permiten ingresar nuevos registros actualizar o eliminar registros existentes, tan solo consultar . En el menú de Reportes este usuario tendrá acceso a cada una sus opciones ya que solo son reportes.

La Programación de Ordenes existentes solamente le compete al usuario Administrador que son los encargados y responsables de planificar la programación de las ordenes dentro de una Producción. Esta sección se encuentra en la opción Programar dentro del Menú Planificación del Sistema.

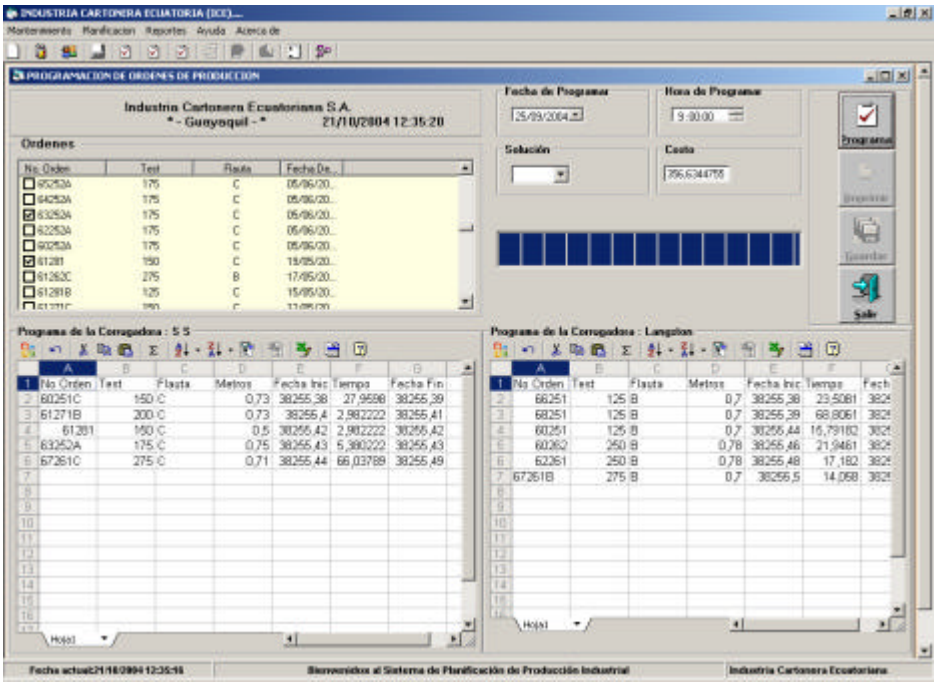

Figura 5.4. *Pantalla de Programación de Ordenes*

El usuario podrá visualizar en un list view todas las órdenes que dentro de la Base tienen un estado de Abiertas, es decir que aun no han sido programadas, de manera que el usuario tiene la opción de escoger las ordenes que desea programar, una vez seleccionadas, debe ingresar la fecha y hora en que desea que inicie la programación, finalmente da click en el botón Programar y los resultados de cada máquina denominados Solución Optima serán visualizados en los Web Components correspondientes a cada máquina; y además podrá observar el costo o tiempo que demorara en producir las ordenes bajo esa programación

Pero además el usuario tiene la opción escoger las otras soluciones que fueron aceptadas antes de la optima, según el algoritmo. Así el podrá determinar que solución le conviene de acuerdo al costo. Y dando clic en el botón Imprimir podrá visualizar las soluciones impresas y guardarlas en la Base de Datos con el botón Guardar. De manera que si posteriormente desea revisar las programaciones anteriores puede hacerlo en la opción Programas del Menú Reportes del Sistema.

5.2. DESARROLLO DEL SISTEMA

En esta sección se va a explicar los detalles concernientes a la implementación del sistema, la cual incluye lo referente a los lenguajes de programación utilizados tanto en VB como en el manejador de base de datos (Procedimientos almacenados).

5.2.1 Programación del lado de la aplicación

Cabe resaltar que éste sistema esta desarrollado en Visual Basic 6.0

Conexión a la base de datos

Puede utilizar la opción Orígenes de Datos ODBC de Herramientas Administrativas del Panel de control (Administrar Orígenes de Datos ODBC) para agregar, configurar o quitar DSN de usuario, DSN de sistema, DSN de archivo; para el desarrollo de este sistema hemos agregado un DSN de sistema con el nombre de DSN_ICE que esta configurado con una conexión a la base de datos del sistema que tiene el nombre de ICE, con user = sa y sin password., la misma que debe ser restaurada con el backup que se adjunta en el CD del sistema de la tesis que tiene el nombre de backup_ICE.

Para la conexión a la Base de Datos se crea un objeto recordset, este no es mas que una forma de obtener registros de la fuente de datos, para poder identificar el origen de la información

Para crearlo se debe agregar la referencia Microsft Data Formatting Object en VB dentro del menú Project en References

En el Anexo A se encuentra la programación modular que contiene los objetos recordset y de conexión, las variables publicas y todos los procedimientos y funciones, que se implementaron para el desarrollo de este trabajo.

Para el desarrollo de esta tesis hemos trabajado también en memoria para la implementación del Algoritmo Recocido Simulado, es decir que tan solo la Solución Inicial que el algoritmo utiliza para empezar la simulación fue implementada en procedimiento almacenado del motor de la BD. Pero en VB se implemento las Listas Doblemente Enlazadas dentro de una programación modular de Clases, la misma que se encuentra en el Anexo B.

5.2.2 Programación del lado de la base de datos

Este tipo de programación se la realiza en el motor de base de datos, en este caso SQL Server 2000, mediante la utilización de procedimientos almacenados.

Procedimientos almacenados

Los procedimientos almacenados son conjuntos de sentencias en lenguaje Transact SQL que pueden almacenarse en el propio servidor. Los procedimientos almacenados de SQL Server, son mas potentes, porque permiten almacenar funciones y procedimientos compuestos por varias instrucciones, introducir saltos, bucles, etc.

Ejecutar un procedimiento dentro del Query Analayzer

Par ejecutar el procedimiento dentro del Analizador de Consular Query Analayzer se utiliza el comando EXECUTE o su abreviación EXEC.

```
EXEC asigna_programacion
```

Ejecutar un procedimiento dentro de VB

Para ejecutar un procedimiento almacenado desde VB, a través de un objeto conn (variable de conexion), se utilizan las siguientes sentencias:

```
conn.Execute ("asigna_programacion")
```

Llamar a un procedimiento dentro de VB

Para llamar a un procedimiento almacenado desde VB, a través de un objeto Recorsedt, se utilizan las siguientes sentencias:

```
Set rst = conn.Execute("programacion 1")
```

En el Anexo C, se muestra la creación de uno de los procedimientos almacenados que se implementaron para el desarrollo de este proyecto de tesis

5.3. PRUEBAS DEL SISTEMA

En esta sección se muestra la fase de prueba por la que ha tenido que pasar el proyecto de la tesis, ya que es debido esta fase el sistema ha ido mejorando tanto en su interfaz (para que esta sea amigable y fácil para el usuario), como en las soluciones (para que estas sean soluciones óptimas en las que el usuario pueda confiar).

Como se mencionó en el capítulo 3, para lograr que las soluciones sean óptimas es muy importante experimentar con los parámetros que el algoritmo Recocido Simulado requiere, es decir ajustar o calibrar los parámetros.

5.3.1 Calibración de los Parámetros

Los parámetros a calibrar del Algoritmo Recocido Simulado, la explicación y demostración de sus fórmulas se mencionaron en el capítulo 3. A continuación se nombra los parámetros, utilizados para la implementación del algoritmo:

- Temperatura Inicial $T_0 = 0.005 * C(S_{inicial})$
- Velocidad de Enfriamiento (Número de iteraciones) $L(T)$, éste parámetro es el más importante, ya que es el criterio de parada del algoritmo. Para obtener una solución óptima, $L(T)$ debe ser al menos igual a $card(\Omega)^2$, es decir que el algoritmo haría una búsqueda exhaustiva; pero el tiempo de ejecución sería demasiado largo, y el objetivo de este trabajo es implementar un algoritmo que busque una solución rápida y óptima, de manera que hemos experimentado con $L(T)$ hasta ajustarlo a nuestra conveniencia.

$$L(T) = 5 * n \quad ; n = \text{Número de Pedidos}$$

$L(T)$	$C(S_{óptima})$
$2 * n$	321.7356395
$3 * n$	321.7356395

$4*n$	321.73563955
$5*n$	321.7356395

- Decrecimiento de la Temperatura $\alpha(T) = kT, 0 < k < 1$; para nuestro caso hemos escogido un $k \in [0.8,8.99]$ aleatorio para cada iteración
- Temperatura Final $T_f = \frac{\varepsilon}{\ln(card(\Omega))}$; pero así mismo como se menciono en el parámetro $L(T)$ estaríamos haciendo una búsqueda exhaustiva; de manera que hemos ajustado este parámetro a la siguiente aproximación:

$$T_f = \approx \frac{\varepsilon}{\ln(n!)} \approx \frac{\varepsilon}{n/2}$$