UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES

"CAMPUS ARAGON"

"FUNDAMENTOS APLICADOS AL DESARROLLO DE UN SISTEMA DE INFORMACION PARA EL INSTITUTO MEXICANO DEL PETROLEO"

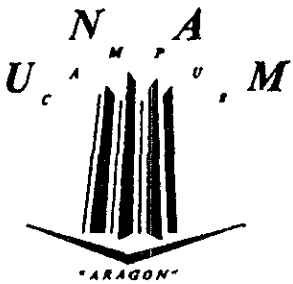
T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A :

MIGUEL ANGEL ROMERO MARTINEZ



ASESOR: ING. LILIA ENCISO GARCIA.

1998

AGRADECIMIENTOS

A Dios, por permitirme ver la luz del día de hoy.

A mi familia, por compartir conmigo todos aquellos momentos gratos y amargos, y de los cuales encuentro la razón de vivir.

A todos mis amigos y amistades, por los que mantengo una vida agradable y lejos de la melancolía. En especial a Chacha por compartir momentos importantes de mi vida.

A la UNAM Campus "Aragón" y al Instituto Mexicano del Petróleo por todo el apoyo brindado.

Y a la Ing Lília por encaminarme en este paso fundamental.

Mil gracias

MIKE.

" Fundamentos Aplicados al Desarrollo de un Sistema de Información para el Instituto Mexicano del Petróleo "

PAG.

III. Evaluación.

IV. Desarrollo.

INTRODUCCIÓN

Este trabajo de tesis está dirigido principalmente a gente que esta aprendiendo a desarrollar Sistemas de Información por primera vez, o a personas que pudieran tener poco o ningún adiestramiento formal en el desarrollo o la Ingeniería de Software Detrás de cualquier aplicación competente no sólo están los arduos conocimientos de la sintaxis de un lenguaje en particular, sino está toda una metodología a seguir que son aspectos importantes para su carrera como desarrollador de Software

Antiguamente cuando se asignaba una tarea a un programador (antes del 90), este se sentaba frente a su PC y comenzaba a codificar, pero eran tareas breves y códigos relativamente rápidos, por lo que en una noche se podrían llevar a cabo. Al hacerse más complejas las PC´s, los usuarios también incrementaron sus exigencias, las tareas eran más grandes al igual que el tiempo de codificación y todo parecía desbordarse Por todo esto es clara la necesidad de planear, se necesita aplicar toda una metodología al problema y no sólo un talento artístico indisciplinado o comercial

Ésta es una guía practica que abarca todos los pasos que se requieren para desarrollar Sistemas de Información competentes y a la altura de la actualidad, pues se apoya por un lado en la aplicación de la ingeniería para lograr resultados satisfactorios y por el otro crear con éxito verdaderas aplicaciones funcionales para *Windows*.

En el capítulo I se expone a la necesidad de información que requiere la alta gerencia como un punto importante para realizar un análisis antes del diseño final, además de tratar las diversas definiciones y tipos de Sistemas de Información. Algunas características para producir software de calidad son otro de los aspectos importantes que se consideraron.

El capítulo II está enfocado básicamente a los principios de diseño que Windows 95 propone para satisfacción de los usuarios y al diseño de pantallas que cumplan con este sello

En el capítulo III se trata a la ingeniería de software como la gran diferencia para producir software productivo y de calidad principalmente

En el capítulo IV se expone un análisis muy interesante para crear aplicaciones a 16 y 32 bits, además de las diferencias existentes entre los diversos modelos de bases de datos más importantes

El capítulo V destaca las características necesarias para implantar un Sistema de Información, así como las complejas características para construir un programa de instalación a la altura de los requerimientos de Windows 95

Y en el capítulo VI se manifiesta la importancia de documentar un Sistema de Información como soporte para su mantenimiento y uso

CAPÍTULO 1

FUNDAMENTOS PARA EL ANÁLISIS DE SISTEMAS DE INFORMACIÓN.

1.1. Definición y acción de un Sistema de Información.

Un "Sistema" es *un conjunto de elementos interrelacionados entre sí que cumplen con un objetivo en común*, siendo dichos elementos personas, cosas, datos, procedimientos, etc

Las personas que se encuentran relacionadas en el medio (computacional) saben que el enunciar una definición de "Sistema de Información" (SI) no es nada fácil, pues los Sistemas afectan a todos los niveles de una organización; puesto que la información es el recurso administrativo en esencia y constantemente surgen cambios, la información al día es una de las grandes preocupaciones del personal que toma las decisiones (gerencia) En cualquier compañía la información es la base de sus actividades, y por ello se deben desarrollar Sistemas para producirla y administrarla, asegurando así que esta sea exacta y confiable al momento de querer disponer de ella para presentarla en forma óptima

La economía en la era moderna está basada en primer lugar por el uso de la información, en segundo lugar por la producción y por último en la administración (naciones del primer mundo), muchas compañías se dedican a la producción de información pero lo importante es que lo hagan de manera eficaz y el éxito se reflejará sobre las que no lo hacen

La figura 1 1 muestra una gráfica de la importancia que tuvo la información durante algunos periodos de la humanidad

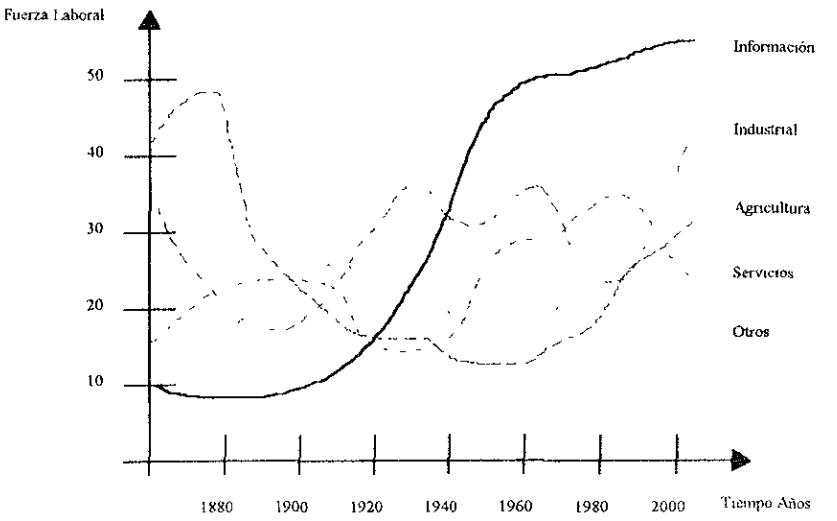


Figura 1.1 Evolución de la información

Un SI principalmente realiza tres actividades, la primera es recibir datos de fuentes internas o externas como elementos de entrada, la segunda es actuar sobre los datos para generar información y la tercera es reproducir la información para la diversidad humana

James A Senn en su libro "Sistema de Información para la Administración", define a los SI como *un conjunto de personas, datos y procedimientos que funcionan en conjunto* [1], por su parte George M Scott en su libro "Principios de Sistemas de Información" los define como *sistemas creados por analistas y administradores para llevar a cabo tareas específicas esenciales para el funcionamiento de la organización* [2], por mi parte y antes de enunciar cualquier definición de SI considero necesario hacer un preámbulo en la definición de "Información", ya que durante la historia de la humanidad este recurso ha estado alcanzando cada vez más su esplendor, pues la base de una nación fuerte y consolidada es aquella que hace de la misma, un uso racional e inteligente. Tomemos por ejemplo el caso de nuestro vecino del norte, ellos cuentan actualmente con la fuente más amplia y completa del Mundo como es la sede de las Naciones Unidas en Nueva York, en ella se tiene conocimiento de las zonas más ricas del planeta, de los puntos militares estratégicos, de conflictos internos de cada país, etc , y por si fuera poco cuentan con la Red de Información más poderosa del Planeta como la World Wide Web que es INTERNET, (la infraestructura y la inversión que se requirió para darle el surgimiento a esta fue impresionante) es importante darnos cuenta lo que puede hacer un

[1] James A Senn, "Sistema de Información para la administración", Pag 2

recurso como este por una nación Es por ello que considero a la información como un recurso del cual la humanidad se sirve para ampliar sus conocimientos, y por lo tanto la definición de SI es la siguiente:

"Es un conjunto de elementos (Hardware y Software) que controlan un flujo de información y la transforman en el aprovechamiento humano".

Los SI en la actualidad son utilizados en cualquier proceso y sector productivo, como por ejemplo en el sector industrial se utilizan en el control distribuido (proceso en donde interactuan las computadoras y la maquinaria), en la administración del personal y de sus recursos, etc ; en el sector empresarial se utilizan para apoyar las actividades de oficina, en procesos que influyen para la toma de decisiones, en transacciones, etc , en fin, definitivamente son la base de las actividades que se desarrollan en la sociedad

Un SI es el instrumento vital para la depuración de la información en todas las facetas del medio empresarial, logrando así los niveles de eficacia y eficiencia que se desea

[2] George M. Scott, "Principios de Sistemas de Información", Pag 4

1.2. Necesidades de información para la Alta Gerencia.

Los Sistemas de Información Gerenciales (SIG) básicamente se encargan de procesar y seleccionar la información requerida por los administradores o gerentes en apoyo a la toma de decisiones. Es muy importante diferenciar la información que un administrador o un usuario final necesita y, esto se puede determinar mediante un análisis de los requerimientos y responsabilidades para cada sector La necesidad de información ideal también es de gran importancia por ello, carecer de información vital puede ocasionar que los administradores cometan errores, pierdan oportunidades y con ello tengan graves problemas de rendimiento.

El tipo de información que se requiere en general es la que se agrega al conocimiento siendo relevante para la situación en la cual se aplicará, por ejemplo la información contable y la administrativa.

La información contable básicamente se aplica en la generación de estados financieros (balance general) o estados de resultados, en donde costos de personal, gastos de operación y distribución de gastos generales son funciones de la contabilidad administrativa que sustentan el control y la toma de decisiones por parte de la gerencia, y de supervisores que responden a cuestiones como el rendimiento de las operaciones y actividades de esa empresa. La información administrativa se considera como un subproducto del proceso de contabilidad y básicamente debe informar a los

administradores y no abrumarlos, puesto que los grandes ejecutivos no necesitan extensos detalles contables sino formas resumidas que muestren los resultados generales y puntos de mayor interés para comparar el rendimiento planeado contra uno real entre divisiones, departamentos, áreas de producción u otras dependencias.

Los administradores de alto nivel no cuentan con el tiempo y la paciencia para deliberar grandes cantidades de detalles aún cuando se llegasen a presentar en forma resumida, por ello requieren identificar con precisión los puntos fundamentales que demanden su atención.

La figura 1.2 muestra siete tipos de información necesarios para la gerencia.

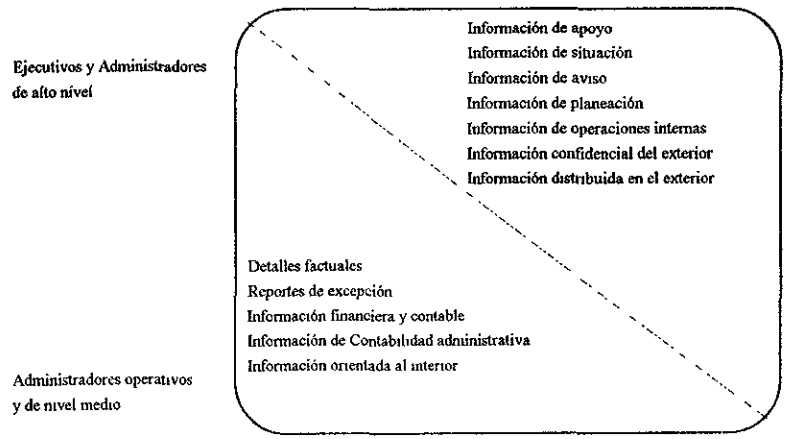


Figura 1.2 *La información administrativa difiere según la organización.*

La información de apoyo ayuda a los administradores en relación a situaciones actuales o niveles de logro para saber el rendimiento de acuerdo a las expectativas del área de interés. La de situación mantiene a los administradores al tanto de crisis y problemas presentes para aprovechar las oportunidades que podrían perderse si no se responde al momento. Por otro lado la de aviso indica los cambios que están ocurriendo por algunas oportunidades o por presagios de problemas futuros. Por su parte la de planeación *describe* los principales programas que a futuro se pretenden realizar. La información de operaciones internas refleja el desempeño de la organización y de sus empleados, se concentran como su nombre lo dice en las actividades internas de la organización. Los informes, rumores y opiniones respecto a la empresa se considera como información confidencial, *incluyen desde cambios en la industria, movimientos en el mercado* financiero, transformaciones político económicas, etc. Y por último la información difundida en el exterior es proporcionada por un alto ejecutivo el cual la revisa antes de transmitirla a los medios.

Algunas otras necesidades de información no se pueden predecir porque durante su desempeño los ejecutivos se encuentran con diversas oportunidades y problemas. Cabría reiterar que un SIG es información extensa y coordinada de subsistemas integrados que transforman los datos en una diversidad de formas para mejorar la productividad conforme a estilos y características para la gerencia

1.3. Productividad.

La productividad es *la aptitud para incrementar la eficiencia de un proceso* [3] siendo esta la fabricación de artículos, la venta, la habilidad de los administradores para dirigir sus actividades, etc.. Por otro lado la productividad se refiere a *la utilización eficiente de los recursos (insumos) al producir bienes y/o servicios.* [4] Un SI utilizado adecuadamente mejora la productividad aumentando el volumen del trabajo realizado y la velocidad con la que se ejecutan las transacciones. Cabría mencionar que la definición de este término varía ligeramente según la persona que la enuncie pues no es igual un economista que un político o administrador, o un líder sindical a un ingeniero o industrial.

1.3.1. Ingeniero y Administración de la Productividad.

Al convertirse la productividad en una preocupación primordial tanto de los administradores como de los ingenieros, se podría decir que esta se subdivide en dos ramas: *ingeniería de la productividad* y *administración de la productividad.*

- **Ingeniería de la productividad.-** Se ocupa de diseñar, desarrollar y mantener sistemas de medición, evaluación, planeación y mejoramiento de la productividad en empresas diversas

[3] James A. Senn, "Sistema de Información para la administración", Pag. 9

- **Administración de la productividad.-** Es un proceso administrativo formal en que intervienen todos los niveles de la administración y los empleados con el objetivo final de reducir el costo de fabricar, distribuir y vender un producto o servicio a través de una integración de las cuatro etapas del ciclo productivo a saber, medición, evaluación, planeación y mejoramiento de la productividad. Recalca que lo importante es recordar que deben comprometerse todos los responsables de hacer que la organización sea productiva.

1.3.2. Calidad de Software.

La calidad del Software se puede comparar de diversas maneras, pero esencialmente se basa en Corrección, Facilidad de Mantenimiento, Integridad y Facilidad de Uso.

Un programa debe funcionar correctamente, pues de lo contrario adquiere poco valor para los usuarios La acción de Corrección se podría medir por el número de defectos siendo estos una carencia verificada de conformidad con los requisitos, cabe señalar que estos son detectados por un usuario del programa durante un tiempo estándar de un año.

[4] David J. Sumanth, "Ingeniería y Administración de la Productividad", Pag. 4

La facilidad de mantenimiento, se refiere a la facilidad con la que se puede corregir un programa en caso de error, adaptarlo si su entorno cambia o mejorarlo si el cliente desea un cambio en los requisitos, la manera de poder medir esta acción es mediante el tiempo promedio entre los cambios. Es importante recalcar que ésta no es una manera directa de evaluar la facilidad de mantenimiento por lo que se tendría que utilizar una medida indirecta como la corrección

Integridad es un atributo, es la habilidad de un Sistema para resistir ataques contra su seguridad ya sea accidentales o intencionados, estos ataques se pueden realizar en cualquiera de los tres componentes del Software: programa, datos y documentos.

Por su parte para la facilidad de uso básicamente es que el programa sea amigable, pues de lo contrario generalmente fracasan, incluso si sus funciones son valiosas, pero además para todo esto hay 4 características que son: la habilidad intelectual y/o física requerida para aprenderlo, el tiempo requerido para llegar a ser eficiente en su uso, aumento neto en productividad captado cuando es utilizado por alguien eficiente, y la valoración subjetiva de la disposición de los usuarios hacia éste.

1.3.3. Modelo de Productividad Total.

El Modelo de Productividad Total (MPT) considera toda la producción y los insumos cuantificables, por lo tanto es una representación más exacta del panorama

económico real de una empresa. El control de las utilidades a través del uso de índices de productividad total es un beneficio tremendo para la alta gerencia. Por otro lado si se usa junto con medidas parciales puede guiar al administrador de una manera efectiva y se relaciona fácilmente con los costos totales.

El MPT se basa en una medida de productividad total y un conjunto de cinco medidas de productividad parcial, esta se puede aplicar en cualquier empresa que produzca u organización de servicio y se da por la siguiente fórmula:

MPT = Producción Directa Total (PDT)/ Insumos Directos Totales (IDT)

Los elementos de producción y los insumos directos de la fórmula se muestran en las figuras 1.3-a y 1.3-b respectivamente.

Aunque muchas veces este modelo no se aplica en la práctica, es conveniente saber que hay opciones a seguir en la solución de un problema.

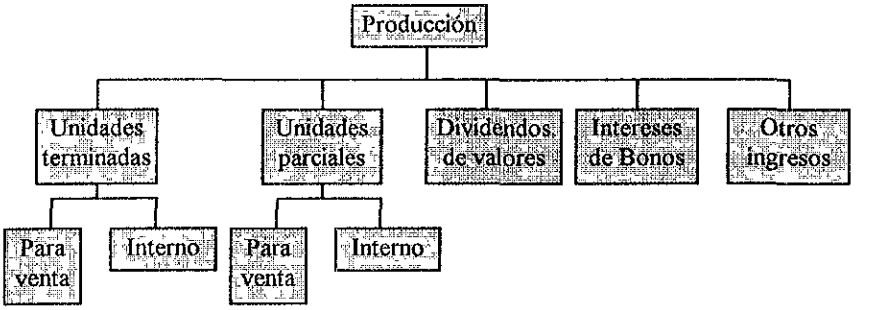


Figura 1.3-a Elementos de producción considerados en el MPT.

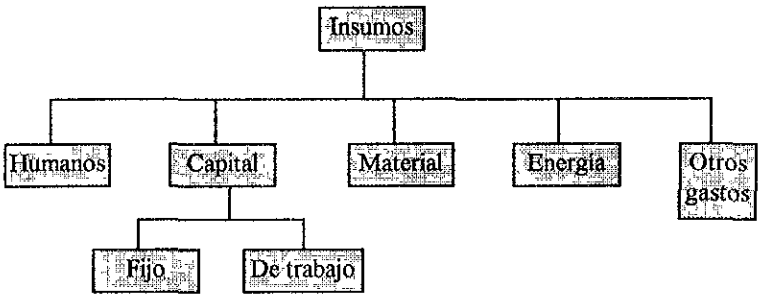


Figura 1.3-b Elementos de insumo considerados en el MPT.

1.4. Sistemas de Información de Mercadotecnia, Contabilidad y Administración de Bases de Datos.

Dentro de la extensa gama de tipos de sistemas, es conveniente destacar tres de ellos por considerarlos de mayor importancia para el peso de las actividades económicas.

En primer término los SI de Mercadotecnia, ya que proveen de ventajas competitivas al estructurarse correctamente y al apoyarse en un conjunto de subsistemas interrelacionados, siendo la mercadotecnia una actividad humana basada en informes que llegan de otras fuentes de información y cuya actividad fundamental es la de proveer y distribuir productos de cualquier índole. Entre los sistemas que se destacan figuran : los Especializados en Soporte de Ventas, los de Análisis de Ventas, Investigación e Inteligencia de Mercado y el de Planeación de Producto. Los Especializados en Soporte de Ventas dan a los vendedores información detallada de como son los inventarios actualizados Los de Análisis de Ventas se rigen por el control de pedidos proporcionando información acerca de líneas de productos, utilidades, lugares de venta y vendedores, dando origen al análisis de cuentas de clientes (que proporciona información para el sustento de la empresa). Los de Investigación e Inteligencia de Mercado tienen tal vez un peso mayor que el anterior porque se dirigen al análisis de información para identificar tendencias, cambios o amenazas con respecto a la organización. Y por otro lado el de Planeación de Productos que se basa en información recibida directa o indirectamente de

los demás SI antes mencionados pero proporciona también información acerca de costos de productos, demanda de clientes, precios de competidores, etc. y todo ello para apoyar a los gerentes a la fijación de costos para sus productos

Los SI de Contabilidad son una vértebra de casi todas las organizaciones pues estos alimentan a casi todos los demás subsistemas funcionales. Estos se clasifican en internos y externos; los internos no se rigen por principios de contabilidad aprobados generalmente, pero costos y otros factores ayudan a evitar que una organización se apoye en el desarrollo de sistemas de informes adicionales. El SI de Contabilidad suele sumar los costos en forma horizontal ya sea en una cuenta, registro o archivo cuando algún producto difiere del proceso de producción, y por otro lado suele basarce en la suma de costos y beneficios de manera jerárquica, es decir agrupa costos que son responsabilidad de un determinado nivel y a su vez con los de responsabilidad superior. Principalmente su propósito es el control de costos y la validación de los inventarios de los productos totales o parciales.

Por último los Sistemas de Administración de Bases de Datos están formados por un conjunto de Software que administra los archivos de base de datos (BD), estos deben ser capaces de accesarlos, actualizarlos, agregar y borrar registros, obtener datos requeridos, reorganizar la base reasignando espacio de almacenamiento para agilizar los procesos, etc . Estos sistemas realizan otras tareas de mantenimiento a las BD, pero digamos que un punto vital en ellos es la seguridad de la información puesto que

muchísimos usuarios accesan a esta y por lo consiguiente se requiere restringir o delimitar el acceso a los programas de aplicación o usuarios finales. Otro punto importante de la seguridad es que la centralización de toda la información en un archivo lógico, puede causar en algún tiempo daños o destrucción del mismo, es por ello que el sistema debe asegurar que la BD no se destruya total o parcialmente, y en caso de ser inevitable permitir la reconstrucción con exactitud y rapidez. Por otro lado el lenguaje de consulta y los costos de procesamiento son puntos que también se tendrían que tomar en cuenta, debido a que el lenguaje permite la preparación de reportes de manera óptima y eficaz, y los costos varían de acuerdo a la velocidad de procesamiento de la máquina en uso y al Administrador de BD (SYBASE, DBASE, ORACLE, FOX PRO, INFORMIX, etc.).

1.5. Conceptos de Sistemas de Información para la Administración de Bases de Datos.

Para entender con mayor exactitud la capacidad y el desarrollo de un SI de BD es necesario hacer hincapié en algunos términos de importancia.

- **Geografía de las BD**.- Con ello se refieren a la distribución física que se hace de estas para ser aplicadas en una empresa, puesto que se pueden emplear en un lugar o en varios.

- **Tecnología de BD** - Comprende varias características entre ellas la estructura de archivos para la asociación de registros internos y externos, la integración de archivos de funciones cruzadas para proceso automático, la independencia de programas y datos para actualizar y dar mantenimiento a las BD, estandarización de todo el sistema en cuanto a datos, formatos de registros, etc., un sólo administrador para los archivos de datos, un diccionario que contenga información acerca de datos y BD, memoria de acceso directo a gran escala para datos y BD, programas y equipos de comunicaciones para hacer que los usuarios tengan acceso simultáneo, técnicas de respaldo y recuperación de archivos, y por último el lenguaje de consulta que podría agilizar la búsqueda de información en línea.

- **BD para actividades de operación** - Se refiere a la congruencia que debe haber entre los elementos de información para dar mayor validez a los resultados, a la actualización y respaldo de las BD para hacer que la información sea más oportuna y disponible ,y a que la información sea compartida entre los usuarios.

- **BD para actividades Administrativas**.- Implica información estratégica para los administradores, problemas administrativos no muy comunes como la organización de datos en forma flexible para una o varias BD, a modelos

administrativos de cómputo para el apoyo de la gerencia, y a subsistemas de información para tareas clave.

Con estos términos es claro entender que un SI para la administración de las BD tienen lineamientos que son esenciales manejar.

1.6. Análisis del Sistemas de Información propuesto.

A continuación se describe el análisis del SI que se desarrollará y del cual al final de cada capítulo se le dará seguimiento

1.6.1. Antecedentes.

El Instituto Mexicano del Petróleo (IMP) es creado el 23 de agosto de 1965, por decreto presidencial, *se fundó como un organismo descentralizado, de interés público, con carácter preponderantemente técnico, con personalidad jurídica y patrimonio propios.* [5] El objetivo primordial es apoyar a Petróleos Mexicanos (PEMEX) en la solución de sus problemas tecnológicos y de recursos humanos para coadyuvar en el

[5] "Instituto Mexicano del Petróleo", http://www imp.mx/.

suministro de los hidrocarburos y sus derivados requeridos en el desarrollo y expansión de la infraestructura industrial del país. Su gente conformada por grupos de Geólogos, Geofísicos, Matemáticos, Físicos, Químicos, Electrónicos e Ingenieros Petroleros y Químicos, entre otros especialistas colaboran con colegas conocedores de los problemas centrales de la industria petrolera, petroquímica y química en el apoyo de las actividades de la industria y adelantar sus posibles demandas.

El IMP inicia sus actividades en cuatro edificios (tres con laboratorios y uno administrativo) y una nave de talleres rudimentarios Actualmente cuenta en la sede principal con 33 edificios (20 con laboratorios, cuatro naves de plantas piloto y talleres, y una torre administrativa), y en el Conjunto de La Reforma (en el estado de Hidalgo) con tres naves industriales de laboratorios. Cuentan también con cuatro zonas foráneas que comprenden 32 centros de capacitación, oficinas y laboratorios de otras dependencias del Instituto. [6]

La institución está ubicada en Eje Central Lázaro Cárdenas No. 152, colonia San Bartolo Atepehuacan CP 07730 apartado postal 14-805 en México Distrito Federal, inició sus actividades con 300 personas entre ellas técnicos que provenían de diferentes dependencias de PEMEX e investigadores de diversas instituciones del sector educativo; cabe señalar que por las demandas de proyectos y servicios solicitados por PEMEX el

[6] "Evolución Histórica del Instituto Mexicano del Petróleo", http //www.imp mx/evolución html.

personal alcanzó en el año de 1987 la cantidad de 6 mil efectivos. Actualmente el IMP debido a la reestructuración reciente de PEMEX y por consiguiente a la estructuración de los procesos administrativos y servicios que proporcionaban interna y externamente, se redujo la plantilla de efectivos a 4300 empleados.

1.6.2. Organigramas.

El IMP presenta una estructura organica compuesta por una Dirección General, tres Coordinaciones y seis Subdirecciones, como se muestra en la figura 1 4.

La División de Tecnología Aplicada en Sistemas de Control (Grupo de Desarrollo [GDD]) en donde se desarrollará el SI, se ubica en la Subdirección de Ingeniería compuesta por dos Unidades, una Unicota y siete Gerencias (ver figura 1.5); se trabaja principalmente para la Gerencia de Proyectos Industriales, *es un grupo especializado en el diseño de aplicaciones y la solución a problemas operativos de sistemas de control,* [7] cuenta con una plantilla de 10 Ingenieros, 4 Maestros en Ciencias (2 en Sistemas Digitales y 2 en Computación) y 5 Especialistas

[7] "IMP Grupo de Desarrollo en Instrumentación y Control -GDD-", http://192 100 181.165/.

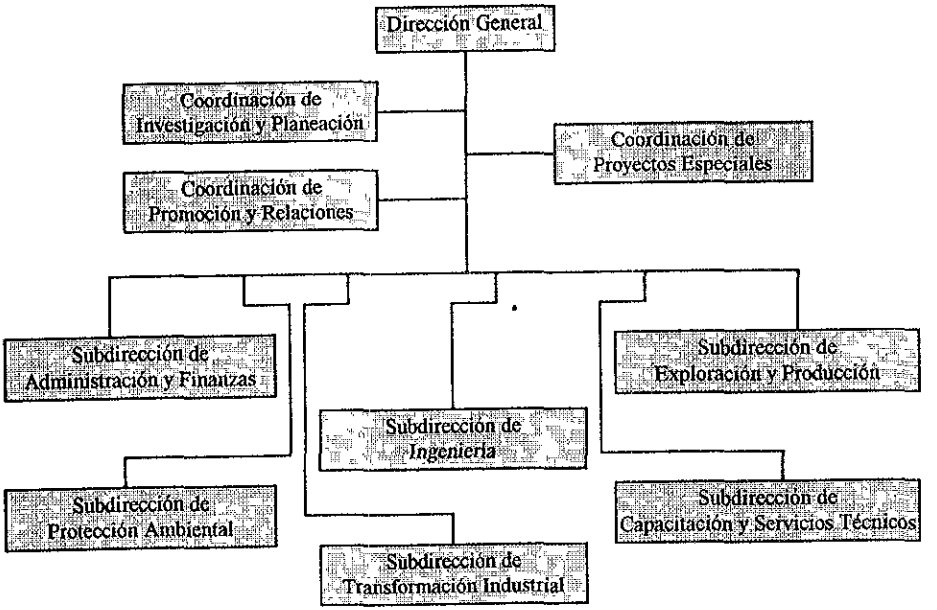


Figura 1.4 Estructura Organica del IMP.

El Sistema se instalará en la Plataforma de Compresión, que a su vez consta de cuatro módulos de compresión, uno de Servicios Generales y dos Turbogeneradores.

El Complejo Marino "Pol - A" se ubica en la sonda de Campeche aproximadamente a 90 Km de la Isla del Carmen, Campeche; consta de 5 plataformas

enlazadas que son Explotación, Producción, Enlace, Habitacional y Compresión Ver figura 1.6.

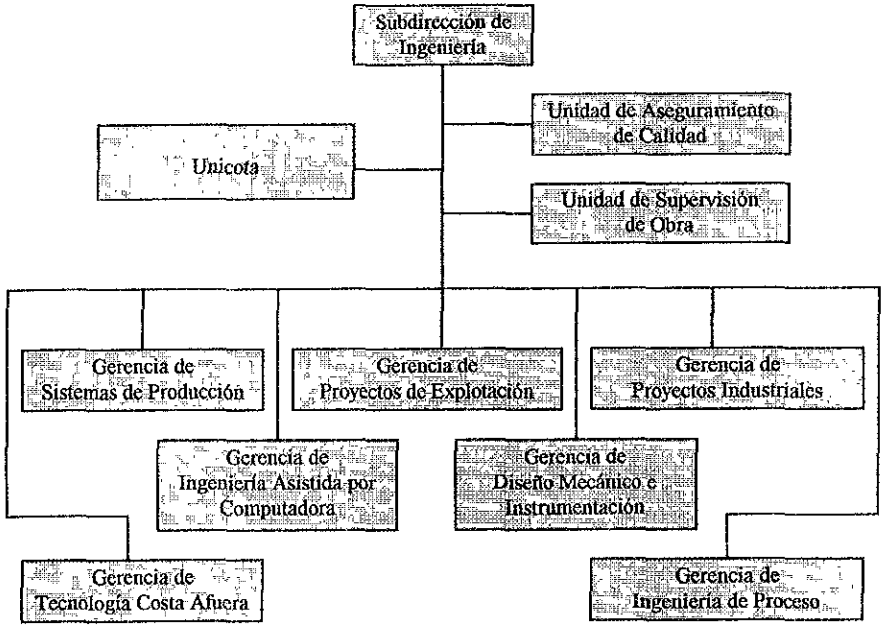


Figura 1.5 Subdirección de Ingeniería.

El módulo de servicios generales lo componen dos PLC's (Controlador Lógico Programable) 5/60 y un PLC 5/40 paro de emergencia. Cada módulo de compresión se compone por un PLC de proceso 5/40 redundante (Un PLC redundante se compone de otro auxiliar para evitar las fallas en el PLC principal, por lo que se tendrían dos PLC's

idénticos) y un PLC 5/40 crítico. Cada turbogenerador se compone de un PLC Genset 5/40 redundante. Haciendo un conteo se tendrían 3 PLC's para servicios generales, 12 para los módulos de compresión y 4 para los turbogeneradores, lo que nos da un total de 19 PLC's.

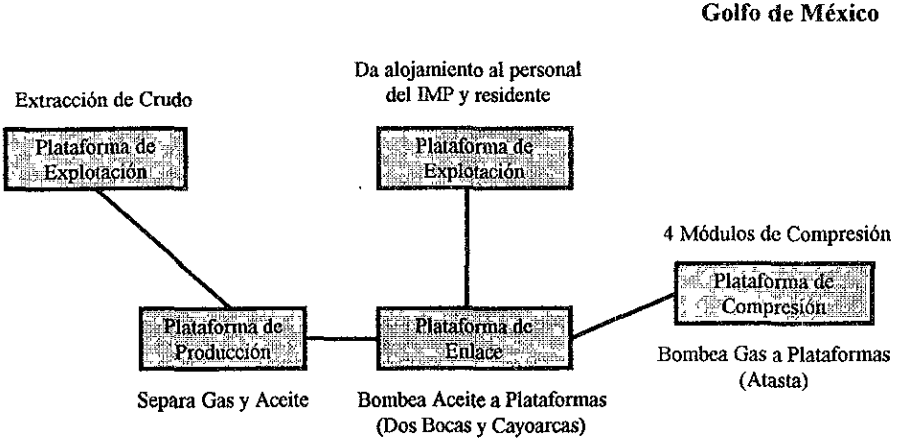


Figura 1.6 Complejo Marino Pol - A.

En cuanto a la red de computo instalada en esta Plataforma se cuentan con 6 estaciones de trabajo para servicios generales, 2 para cada módulo de compresión y una para cada turbogenerador. Por lo que se cuenta con 16 estaciones de trabajo, en la tabla 1.1 se da una breve descripción de las mismas.

ESTACIONES DE TRABAJO	DESCRIPCIÓN
Servicios Generales	
1	Servidor de Red Ethernet
1	Estación de seguridad para PLC 5/40 paro de emergencia
2	Estaciones de control para PLC 5/60
1	Estación de visualización para los módulos y turbogeneradores
1	Estación configurada para WonderWare Intouch (software de control)
Módulos de Compresión	
1	Estación principal de control de motores (INCOM - protocolo de comunicación), de redes de datos redundantes DH+ y de visualización del sistema gas combustible mediante el protocolo de comunicación Modbus
1	Estación redundante de monitoreo de vibración (QNX - programa protocolo de comunicación) de las turbinas de los módulo de compresión, de redes de datos redundantes DH+, de configuración fuera de línea para WonderWare Intouch al sistema Gas - Combustible mediante el protocolo de comunicación Modbus y sistema operativo UNIX
Turbogeneradores	
1	Estación principal de visualización por medio de WonderWare Intouch, interfaz a sistema de gas combustible por medio de Modbus a turbogenenador 1.
1	Estación de respaldo con la aplicación de visualización (WonderWare Intouch), interfaz al sistema de gas combustible (Modbus) del turbogenerador 2 e interfaz a la red QNX para análisis de vibración

Tabla 1.1. Descripción de la Red de Cómputo.

1.6.3. Trabajo de campo, Objetivos y Metas.

Los módulos de compresión de Pol - A tienen como función elevar la presión de los gases que provienen de la plataforma de producción, para su posterior transporte por los gasoductos de la Sonda de Campeche a tierra El **trabajo de campo** se basa en el proceso de compresión que consiste en elevar la presión de gas en tres etapas sucesivas, habiendo equipo para cada etapa que rectifica los efectos de la compresión como la elevada temperatura y formación de condensados de hidrocarburos El aumento de temperatura del gas comprimido se reduce por medio de ventiladores y a su vez para evitar que los compresores succionen condensados, ya que la entrada de líquidos al compresor puede resultar en la destrucción del equipo. En cada etapa hay instrumentos que supervisan el proceso, sobre todo el control de inestabilidad del compresor, ya que puede dañar el equipo al punto de destrucción debido a los movimientos axiales generados en el eje del mismo. Para el proceso de control antes citado se censa la presión de succión y descarga , y el flujo, se relacionan para obtener una señal de control que se envía a una válvula de control que recicla gases para mantener el compresor en operación segura. Por otra parte la potencia al compresor es suministrada por una turbina acoplada mecánicamente, la cual es instrumentada con sistemas redundantes para controlarla, ya que ésta es el equivalente a las usadas en los aviones de reacción. Otros procesos de control importantes son los de servicios auxiliares, como por ejemplo el sistema de aceites lubricantes y de sello, y el sistema de gas combustible

Se requiere consultar los dispositivos de campo para saber su conexión a módulos de entrada - salida (es decir la conexión desde campo hasta el tablero de control). En primer termino saber la ubicación física y el tipo de modulo que puede ser de entradas - salidas analógicas (4 - 20 mA) o de entradas - salidas discretas (24 VCD o 120 VCA); saber que tipo de barrera (son dispositivos usados para resguardar áreas peligrosas) y en cual está conectada una señal

Por otro lado para los diagramas de escalera, conocer el escalón y el programa donde se utiliza una señal (XREFRNCE.DBF) de campo, una instrucción o función especial.

Para el programa de control WonderWare Intouch (WWI) saber en qué pantalla gráfica se está utilizando determinada señal.

Detectar por medio del programa los problemas que puedan presentar los PLC's para rutinas de diagnostico, comunicación, CPU, etc , es decir fallas específicas del equipo. Hace falta un programa que diagnostique o auxilie en la solución de problemas de control que para el caso es definir sobre que programa y en que escalón se está utilizando determinada señal.

Se requiere poder detectar fallas al no tener una señal recibida; por medio del programa se desea detectar fallas y obtener todo el ruteo del PLC para realizar un chequeo físico y poder corregirlas posteriormente.

1.6.4. Datos a utilizar, fuente de información y características.

La información que se manejará principalmente es la siguiente:

- Información técnica de los PLC's de la familia 5 de Allen Bradley (5/60 y 5/40). Ver tablas 1.2 y 1 3 respectivamente.

- *Archivos generados por el software de control WWI, así como los archivos del* PLC Crítico (Entradas - Salidas Discretas y Entradas - Salidas Analógicas).

- Miscelánea de formatos e información del PLC-5 para referencias cruzadas.

Toda la información requerida fue proporcionada por Ingenieros del GDD a través de archivos de computadora (WWI y PLC Crítico), manuales y cuadernillos, y por Ingenieros de campo (especialistas de plataformas) por medio de entrevistas, manuales y cuadernillos.

No. Identificación	No. Catalogo	Descripción	Marca
PR1 1 - PR1 2	1785 - L60BK	Procesador de 64 K de mem. y 3072 punto de E - S	Allen Bradley
EA1 1 - EA1 10	1771 - NISK	Módulo de entradas analógicas de 8 canales 4 - 20 mA	Allen Bradley
SA1 1 - SA1 7	1771 - NOCK	Módulo de salidas analógicas de 8 canales 4 - 20 mA	Allen Bradley
ED1.1 - ED1.8	1771 - IBDK	Módulo de entradas discretas de 16 canales 24 VCD	Allen Bradley
SD1 1 - SD1 5	1771 - OBDK	Módulo de salidas discretas de 16 canales 24 VCD	Allen Bradley
FP1 1 - FP1 2	LZS - 250 -3	Fuente de poder regulada 24 VCD 12.5 A	Lambda
FR1 1 - FR1.24	1771 - P4RK	Fuente de poder redundante de 120 VCA y 8 A	Allen Bradley

Tabla 1.2. Sinopsis de la Información Técnica del PLC 5/60.

El SI a desarrollar debe representar al modulo de mantenimiento y del cual se debe restringir el acceso a través de un password único.

Para el PLC crítico se manejarán 9 tipo de archivos, 4 de conexión (I/O discretas e I/O analógicas), 3 de WWI, 1 para referencias cruzadas y 1 que lo genera WWI para referir el uso de las diversas pantallas de datos La relación entre el de referencias cruzadas y los de conexión se realizará por medio del campo DIR_PLC, el cual contiene información descriptiva del tipo de señal que se manejará en ese momento, es decir

información de la conexión física. La tabla 1.4 describe los diferentes tipos de señales que se tienen en los PLC's así como su valor correspondiente en bits.

No. Identificación	No. Catalogo	Descripción	Marca
PR2.1 - PR2.2	1785 - L60BK	Procesador de 64 K de mem. y 3072 punto de E - S	Allen Bradley
EA2.1 - EA2.10	1771 - NISK	Módulo de entradas analógicas de 8 canales 4 - 20 mA	Allen Bradley
SA2.1 - SA2.7	1771 - NOCK	Módulo de salidas analógicas de 8 canales 4 - 20 mA	Allen Bradley
ED2.1 - ED2.8	1771 - IBDK	Módulo de entradas discretas de 16 canales 24 VCD	Allen Bradley
SD2.1 - SD2.5	1771 - OBDK	Módulo de salidas discretas de 16 canales 24 VCD	Allen Bradley
FP2.1 - FP2.2	LZS - 250 -3	Fuente de poder regulada 24 VCD 12.5 A	Lambda
FR2.1 - FR2.24	1771 - P4RK	Fuente de poder redundante de 120 VCA y 8 A	Allen Bradley

Tabla 1.3 Sinopsis de la Información Técnica del PLC 5/40.

La relación entre los de conexión y los de WWI se dará por medio del campo ItemName que es información vital del PLC para saber la dirección digital correspondiente de la conexión física de los dispositivos.

Señal	Descripción	Valor en Bits
I	Entradas Discretas	1
O	Salidas Discretas	1
C	Contadores	(3 palabras) 48
T	Temporizadores	(3 palabras) 48
N	Enteros	16
F	Flotantes	32

Tabla 1.4 Señales de los PLC's.

De acuerdo a la información presentada en éste capítulo se puede apreciar que el SI no es directamente para la alta gerencia, pero sí se manejará información de apoyo que de una u otra manera afectará el rendimiento adecuado al área de interés, que para el caso es mantenimiento.

Se pretende incrementar la eficiencia del trabajo sistemático de los PLC's de plataformas, por lo que se estaría aplicando ingeniería de productividad.

El SI cumplirá con el atributo de integridad pues se protegerá el acceso a través de password y, por otro lado como se pretende desarrollar una pantalla amigable a la percepción del usuario en conjunto con una estimación de tiempo reducido para su uso

eficiente y la valoración con que éstos lo dispongan, se estaría generando software de calidad.

La administración de la BD estará a cargo de la máquina de Dbase por que es uno de los manejadores más populares, el lenguaje a utilizar para algunos casos será SQL por mantener así un estándar y todo el desarrollo del SI estará integrado por medio de Delphi, que es la herramienta visual basada en Pascal, que me permitiría obtener una aplicación de 16 o 32 bits (Para mayor detalle ver capítulo IV.)

CAPÍTULO 2

DISEÑO DE SISTEMAS DE INFORMACIÓN

2.1 Principios de Diseño de Bases de Datos.

Es importante reconocer el papel que juegan las BD en la actualidad, puesto que miles de ellas se emplean y de estas sólo unas cuantas están diseñadas para servir a los niveles medios y altos de la administración. Son varios los factores que influyen en esta incertidumbre, por un lado están los administradores que no enfocan el potencial de su uso, mientras que por el otro está el personal de procesamiento de datos que generalmente no pone atención en la posibilidad de su uso por los administradores medios y altos. Se nota un contrapunteo enorme, pues ninguno de los dos factores primordiales se llegan a comunicar como es debido, aunque cabría señalar que en este caso los administradores son los que deben tener el control en el desarrollo de las BD de la organización Hay cinco principios que se destacan por su relevancia en el desarrollos de las BD para la gerencia y son

- **Perspectiva Global.-** Al desarrollar una aplicación, se tiene que hacer como parte de todo el esquema y no de manera independiente, pues ello trae como consecuencias a *módulos inconsistentes y por consiguiente* el costo elevado no se deja esperar.

- **Diseño Secuencial.-** Hay veces que los proyectos erróneamente comienzan enfocándose en necesidades de información de la parte inferior de la organización por lo que las necesidades de la gerencia nunca recibirán atención adecuada. Un diseño secuencial permite reconocer explícitamente las necesidades de los administradores altos e intermedios y después las necesidades de operaciones.

- **Capacidad para obtener informes selectivos.-** La BD debe ser capaz de proporcionar informes para cada actividad administrativa y a su vez crear respaldos de los detalles por si son necesarios. Además no debe contener información de otra actividad administrativa mas que la relevante y necesaria.

- **Diversidad de BD para los diferentes niveles de administración.-** Las BD deben diseñarse por tipo de actividad administrativa y para cada nivel de *administración según los requerimientos*

- **Respetar los archivos convencionales.-** Al convertir archivos convencionales puede que no se este respetando el diseño secuencial antes mencionado, además se podrían estar creando archivo que no fueran compatibles por lo que se tendría como resultado un sistema más caro en términos de utilidad administrativa.

2.2 Fundamentos del Diseño de Software.

El diseño es el primer paso de la fase de desarrollo de cualquier producto o sistema de ingeniería, es "el proceso de aplicar distintas técnicas y principios con el propósito de definir un dispositivo, proceso o sistema con los suficientes detalles como para permitir su realización física". El objetivo es producir un modelo de una identidad que se construirá más adelante, este debe combinar la intuición y los criterios en base a experiencia de construir entidades similares, un conjunto de principios y/o heurísticas que guían la forma en la que se desarrolla el modelo, un conjunto de criterios para discernir sobre calidad y proceso de iteración que conduce a una representación del diseño final.

En la figura 2 1 se muestra el flujo de información desde que inicia la fase de diseño hasta la fase de prueba.

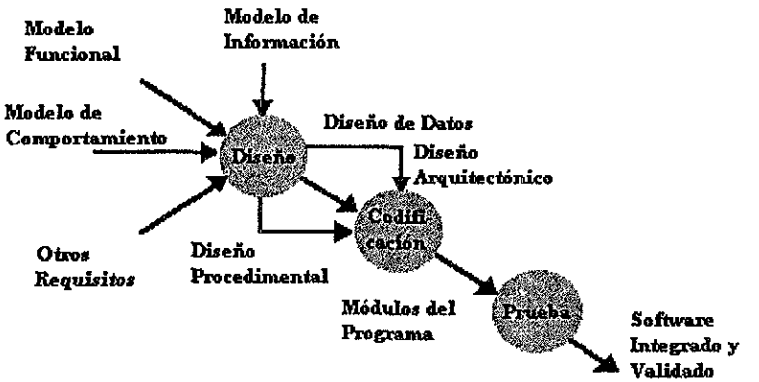


Figura 2.1. Flujo de Información.

La fase de diseño, codificación y prueba absorben el 75% o más del costo de la ingeniería de software, aquí es donde se toman decisiones que afectarán finalmente el éxito de la implantación del programa y la facilidad de mantenimiento que tendrá el software. La importancia del diseño se puede definir con una sola palabra, **"Calidad"**.

La figura 2.2 muestra esquemáticamente la importancia que tiene el presentar un proyecto con diseño, el cual es la base que sustenta a toda la aplicación, y otro sin éste, del cual no se le augura un futuro prometedor.

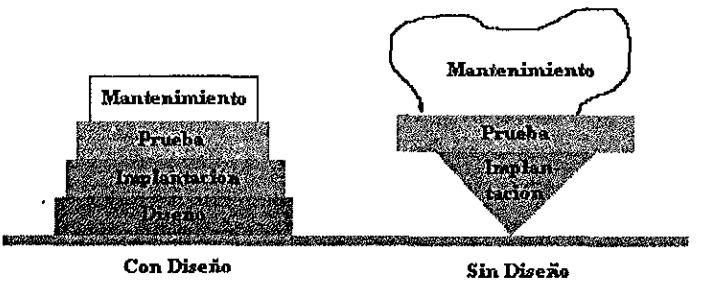


Figura 2.2. Importancia del Diseño

2.3 *Estructuración básica de un Sistema de Información.*

Un SI de computadora es *un conjunto u ordenación de elementos organizados para llevar a cabo algún método, procedimiento o control mediante el procesamiento de información* [8] . La figura 2.3 representa los elementos de un SI basados en PC's y posteriormente se dará una breve descripción de cada uno.

[8] Roger S. Pressman, "Ingeniería del Software, un enfoque practico", Pag. 140.

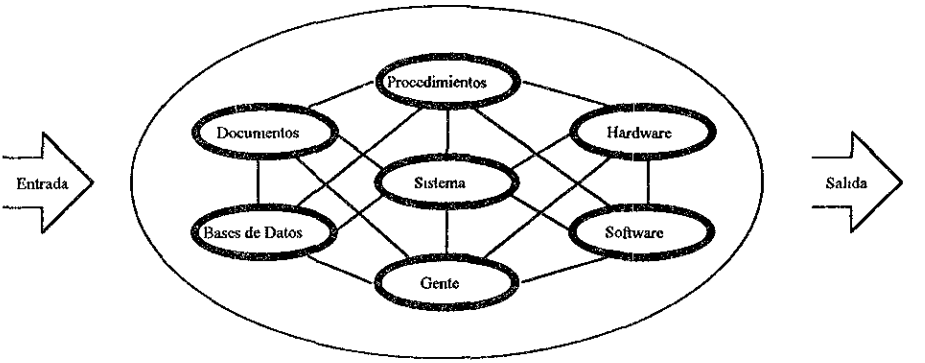


Figura 2.3. Elementos de un SI.

- **Software.-** Es todo programa de computadora, estructuras de datos y documentación asociada para realizar método lógico, procedimiento o control requerido

- **Hardware.-** Dispositivos Electrónicos y Electromecánicos que proporcionan capacidad de computación y las funciones del mundo exterior.

- **Gente.-** Individuos que son usuarios y operadores del Hardware y Software.

- **Bases de Datos.-** Colección grande y organizada de información que se accede mediante Software y que es parte integral en el funcionamiento del Sistema.

- **Documentación.-** Manuales, Documentos y otra información descriptiva que explica el uso y/o la operación del sistema

- **Procedimientos.-** Pasos para definir el uso específico de cada elemento del Sistema o el contexto en que reside

2.3.1. Programación Modular.

Es conveniente mencionar que antiguamente los programadores solían escribir un sólo programa de gran tamaño, por lo que el tiempo de depuración solía alargarse considerablemente. Ahora esta técnica de modularización promueve la división de los programas en módulos de tamaño razonable en donde cada uno realice una función de proceso. Cabe mencionar que estos módulos deben escribirse de la forma más fácil de entender y que el flujo de información que actúe sobre el entre al principio del modulo y continúe hasta el final sin perdida de información

Una de las principales ventajas que se obtiene con la utilización de esta técnica es que los programas en pequeños módulos facilitan la depuración y con ello la modificación del sistema. Además promueve la portabilidad de código, ya que en la actualidad los programadores que trabajan en un proyecto se apoyan para hacer de la programación una actividad de grupo en lugar de un esfuerzo individual. Esto se muestra en la figura 2 4, *donde un bloque robusto es comparado con módulos estructurados que en conjunto* resuelven la misma tarea.

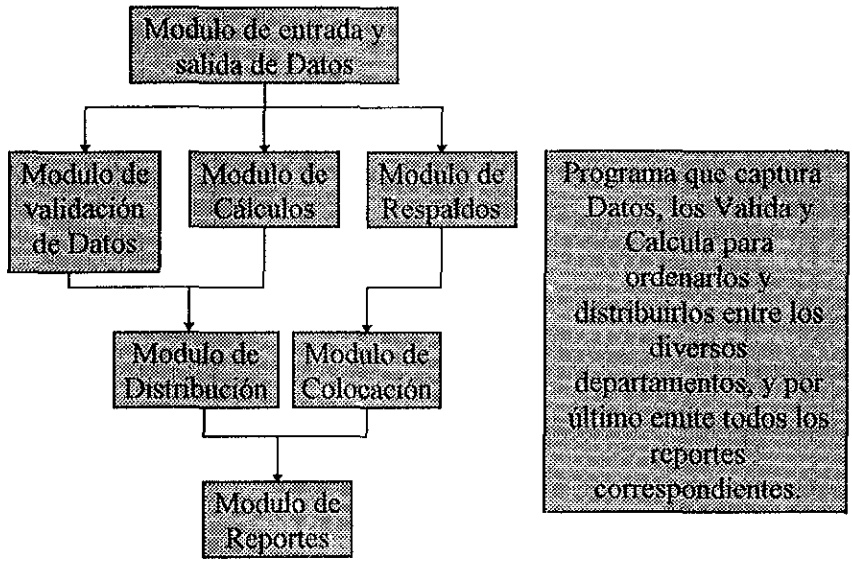


Figura 2.4. Programa Modular y No Modular.

2.3.2. Programación Orientada a Objetos. (POO.)

Este termino de programación no es nada nuevo, pues ya lleva algunos años empleándose en lenguajes de programación como Ada, SmallTalk, C++, Pascal de Borland en varias versiones, etc.. Esta forma de programar es un medio, es una herramienta que puede usarse para bien o para mal, es decir se puede dar el caso de obtener código orientado a objetos mucho peor de lo que pudiéramos haber construido en un código estándar, pero sí cabría recalcar que en la actualidad si un ingeniero de Software está a la par con la POO, este hecho hace que los gerentes de sistemas se enfoquen más en él.

En el mundo real nuestro lenguaje tiene dos componentes principales Sustantivos (Objetos) y Verbos (Operaciones), para que las aplicaciones se apeguen a la realidad el lenguaje de computación debe ser el mismo. La POO nos permite ver conceptos como una variedad de objetos. Se pueden representar las tareas que van a ser ejecutadas, la interacción y cualquier tipo de condiciones que deban ser observadas entre los objetos Se basa en conceptos que alguna vez aprendimos: objetos y atributos, clases y miembros, todo y parte.

Para entender un poco más sobre la POO se tratarán algunos términos que serán esenciales y breves para comprender el tema

2.3.2.1. Objetos y Clases.

Es importante hacer notar que no existe un consenso universal sobre los conceptos que sirven a la POO, pero a través de ideas clave es como lo lograremos

Un objeto es un elemento miembro de una clase o un conjunto de elementos mucho mayor, en la que se puede asociar un conjunto de atributos genéricos a cada miembro de la clase. Por ejemplo un objeto "***silla***", es un miembro (también se usa el término instancia) de una clase de objetos que denominamos "***muebles***", los atributos genéricos de esta clase podrían ser "***precio, dimensiones, peso, situación y color***" entre otros muchos posibles. Como lo ilustra la figura 2.5, en la que se observa este sencillo ejemplo

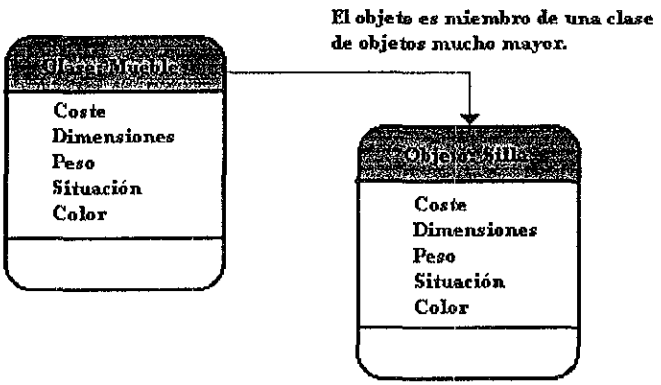


Figura 2.5. Clases y Objetos.

Una clase define las propiedades y atributos que describen las acciones de un objeto que pertenece a ésta Las clases son el elemento principal que permiten desarrollar la POO.

2.3.2.2. Propiedades.

Las propiedades se refieren a las clases en particular, es decir son atributos que se declaran para objetos de una clase y para las acciones asociadas con la lectura y escritura del atributo.

2.3.2.3. Herencia.

La herencia es una de las características más importantes de la POO, consiste en que las clases hijas tomen los atributos de su clase padre como son campos, métodos, propiedades y eventos, con ello además de tener los atributos de sus padres, la clase hija puede agregar nuevos componentes a lo que hereda y como resultado podríamos obtener una clase que tenga casi todas o todas las piezas fundamentales que usted necesita, y agregar nuevos objetos para adaptar esa clase exactamente a nuestros requerimientos.

La figura 2.6 muestra un esquema de lo que sucede en el proceso de herencia, en donde el objeto que hereda se le conoce como descendiente y el objeto del cual se hereda se le conoce como ascendiente.

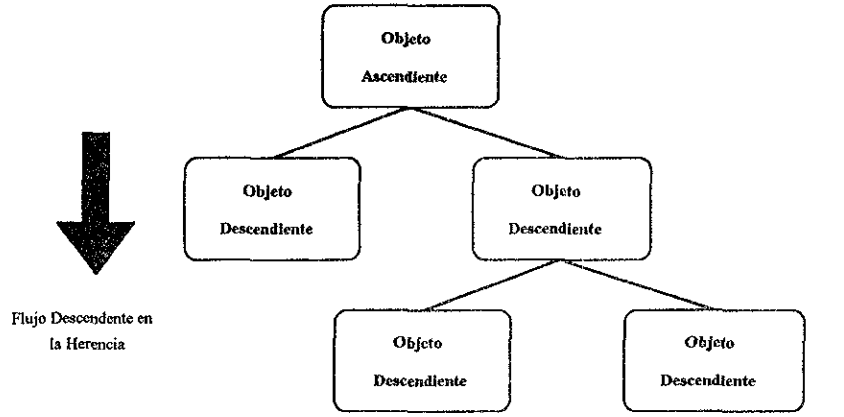


Figura 2.6. Herencia de los Objetos.

2.3.3. Menús y Submenús.

Los menús en conjunción con sus submenús es una manera adecuada de que los usuarios tengan un acceso ordenado y rápido a la funcionalidad de un programa por simple reconocimiento en vez de comandos, es decir con el simple hecho de seleccionar un letrero o elemento descriptivo a la funcionalidad deseada más que por la utilización de una combinación de teclas o comandos. Principalmente existen 3 tipos de menús que son los desplegables, los de cascada y los de aparición súbita.

Los menús desplegables son muy típicos en cualquier aplicación, consisten en una barra de menú que por lo regular está claramente visible en la parte superior de la aplicación, (siguiendo el contexto legado desde la aparición del Sistema operativo Macintosh y posteriormente por Microsoft Windows) estos menús consisten en elementos denominados títulos de menús, a los que al seleccionarlos proporcionan el acceso inmediato a los menús desplegables y estos a su vez a sus correspondientes submenús. La figura 2.7 muestra una barra de menú típica (menú desplegable) utilizada en aplicaciones desarrolladas a 32 bits.

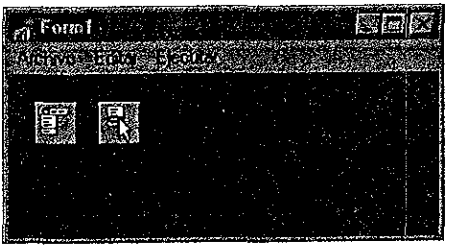
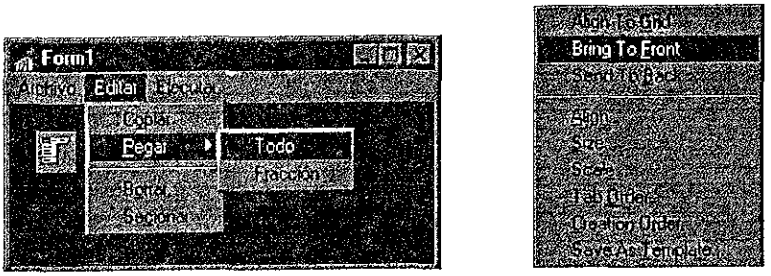


Figura 2.7. Barra de Menú.

Los menús de cascada se usan para minimizar la confusión de sobrecargar al usuario con demasiadas opciones en un sólo menú; son representativos de una variedad de opciones a seguir para ejecutar una acción. Y los menús de aparición súbita surgieron con la llegada de Windows 95 en la que por medio del botón derecho (o alterno) del ratón

ofrecen un conjunto de funciones relevantes que se pueden realizar sobre un objeto seleccionado. La figura 2.8 muestra un menú de cascada y otro de aparición súbita respectivamente.



(a) (b)

Figura 2.8. (a) Menú de cascada. (b) Menú de aparición súbita.

2.4. Principios de Diseño (Microsoft Windows ® 95) centrados en el usuario.

Es muy relevante la participación del usuario en el desarrollo de un SI, puesto que gracias a ella se logra hacer un diseño exitoso Aunque desde un punto de vista el departamento de servicios de información o el analista es el que esta a cargo del Sistema un enfoque muy orientado al usuario puede ocasionar desviaciones en el uso de la información, ello porque muchas veces el usuario no le dedica el tiempo suficiente y sus recomendaciones pudieran ser un tanto ambiguas. Una real participación requiere tiempo

para entender al Sistema y para que sus recomendaciones realmente prevalezcan; mucha gente prefiere apoyarse en el uso de prototipos, que son diseños a menor escala de la aplicación que se pretende terminar.

2.4.1. *Control.*

El usuario debe siempre sentir que tiene el control de lo que está sucediendo en la pantalla, ellos debieran tener siempre la sensación de que son los que inician la acción en vez de reaccionar a los caprichos de la computadora. Al efectuar un programa se requiere proporcionar un alto grado de automatización, por ello se debe asegurar que el usuario tenga control sobre todo el proceso. Por otro lado no todos los usuarios son iguales, cada uno tiene sus propias preferencias y necesidades, por esta razón hay que personalizar las aplicaciones.

Una buena aplicación es aquella que le dice al usuario lo que está sucediendo, o en que estado o modalidad se encuentra. La aplicación deberá ser tan interactiva como sea posible, deberá ser capaz de dar respuesta y no dejar que el usuario se quede preguntando qué es lo que pasa.

2.4.2. Direccionalidad.

Los usuarios deben manipular directamente los objetos de su entorno, la frase "una imagen vale más que mil palabras" es cierta puesto que es mucho más fácil recordar cómo luce algo que su sintaxis de comando, un ejemplo muy claro es el recordar que con el hecho de deslizar sobre la pantalla un determinado objeto y ponerlo o depositarlo sobre otro ejecuta una acción determinada (arrastro este objeto, lo dejo caer sobre este otro e imprime; esta es la forma en que piensa la mayoría de los usuarios). Se debe desarrollar Software de manera que sea fácil trabajar con él, dejando que los usuarios vean cómo las acciones que realizan afectan a los objetos que aparecen en su pantalla.

Cabe mencionar que una de las formas más directas en que los usuarios pueden interactuar con una computadora es con el uso de metáforas. Macintosh es una computadora popular gracias a las metáforas y éstas son las responsables en parte de su éxito, por ejemplo el concepto de carpeta tiene más sentido que un directorio o archivo, en un archivero hay carpetas y dentro de éstas se encuentran los documentos Las metáforas apoyan el concepto de reconocimiento por el usuario en vez de la recolección por el usuario, ya que por lo regular los usuarios pueden recordar el *significado asociado* con un objeto con más facilidad que con un comando.

2.4.3. Consistencia.

Este punto es de gran importancia en el desarrollo de una aplicación Windows, si todas las aplicaciones son consistentes en cómo presentan sus datos, en la forma en que interactúan con los usuarios, éstos pueden emplear su tiempo realizando tareas y no tratando de entender las diferentes formas en que su aplicación interactúa con ellos. Este punto se extiende a varias áreas de consideración·

- Asegurarse de que la aplicación a desarrollar actúe en forma muy familiar al sistema operativo Windows, así el usuario puede con facilidad transferir las habilidades aprendidas en Windows hacia su aplicación.

- Asegurarse de que su producto es consistente con sigo mismo, ya que por ejemplo si maneja "Ctrl+C" como un camino corto a otra pantalla, no use otro paradigma en otra pantalla (como "Ctrl+D").

- Asegurarse de que sus metáforas son consistentes, ya que por ejemplo si su icono de "Agujero Negro" es el mismo que el de "Cajón de Reciclado", los usuarios pensarían que también pueden recuperar documentos del "Agujero Negro".

2.4.4. Rectificabilidad.

En la mayoría de las aplicaciones bien escritas el tiempo empleado para explorar su funcionalidad (apretar botones para ver que pasa) rara vez representa un problema. Si se está a punto de ejecutar alguna acción y se aparece un cuadro de diálogo que alertará sobre el particular, aquí se puede presionar el botón de Cancel para abortar la misma, éste es el concepto de rectificabilidad. Los usuarios necesitan que se les advierta antes de cometer un error Todos cometemos errores, por ello se debe permitir solicitar una confirmación que se indique ya sea con el ratón o con el teclado para el caso de acciones destructivas que se hayan iniciado como errores.

2.4.5. Retroalimentación.

El no saber que es lo que la computadora se encuentra haciendo y que no se lo informa, daría pauta a enunciar una regla cardinal: "permita que los usuarios vean lo que sucede". Proporcione una retroalimentación hacia éste en forma periódica, puede usar una combinación de indicadores visuales o sonoros para hacer saber al usuario que está pendiente de él, además es importante que la retroalimentación sea cercana al punto en que los usuarios están trabajando. Algunos tips para la retroalimentación se basan en la generación estratégica de los mensajes de error y en la manipulación de la forma del cursor para indicar determinadas acciones entre otras

2.4.6. Estética y Sencillez.

La aplicación deberá ser visualmente atractiva al usuario, además del color del sistema para sus pantallas el diseño en sí de la misma es muy importante. La colocación de objetos y el número de estos en sus pantallas determinan que tan utilizables son. Si es posible siga la regla del siete, que consiste en dar al usuario sólo siete opciones (+/- dos), de cinco a nueve opciones provienen de una investigación sobre cuántas cosas puede comprender el cerebro a la vez, ya que con más de nueve opciones la gente tiende a confundirse.

El último principio de diseño es la sencillez, la aplicación debe ser fácil de aprender y utilizar. Se requiere un balance entre dos cosas que funcionen una en combinación con la otra, la primera es el acceso a todo el funcionamiento e información en la aplicación y la segunda es que continúen siendo sencillos la interfaz y el uso del producto, la buena aplicación equilibra estos dos principios y encuentra un justo medio. Por otro lado elabore sus pantallas sin mucho texto, cuando utilice etiquetas en los campos de captura trate de ser lo más breve posible por ejemplo para designar un campo de captura "Nombre" y no "Nombre del cliente", trate de usar el menor número posible de palabras para comunicar el *significado correctamente.*

Microsoft recomienda el concepto de divulgación progresiva, es decir se basa en presentar los datos conforme se vayan necesitando, por ejemplo en un programa de

Agenda Personal se podría mostrar en la pantalla principal el nombre y teléfono de la persona y el usuario tendría que presionar un botón que proporcionara el resto de la información sobre esa persona.

2.5. *Diseño de Pantallas.* (Especificaciones de Microsoft Developer Network CD-ROM, Microsoft Windows ® 95.)

Las imágenes visuales tienen un gran impacto en nuestras mentes y emociones, las imágenes que un diseñador de Software presenta a sus usuarios pueden tener diversos efectos, podrían ir desde la inspiración hasta la distracción, es por ello que es importantísimo diseñar las aplicaciones para inspirar al usuario sin distraerlo de la tarea que lo ocupa

Al momento de diseñar sus pantallas, puede valorar el gran cúmulo de información que investigaciones como la que ha desarrollado Microsoft en algunas décadas. Toda esta información forma parte de las especificaciones de diseño de Microsoft Developer Network (MSDN) CD-ROM o Windows 95 y se tratará a continuación.

2.5.1. Organización.

Hay seis principios organizacionales que MSDN señaló como importantes y que se tratarán en consecuencia.

2.5.1.1. Legibilidad y Flujo.

Este principio indica que organice el diseño para comunicar sus ideas de manera simple y directa con un mínimo de interferencia visual, para incrementar la legibilidad y minimizar la interferencia visual se tendrían que tomar en cuenta tres importantes puntos para diseñar un cuadro de diálogo o ventana

- ¿Se está presentando la idea de la forma más sencilla posible?
- ¿Puede el usuario recorrer este diálogo en la forma en que lo diseñé?
- ¿Todo lo que está en la ventana tiene una razón para estar ahí?

2.5.1.2. Estructura y Balance.

Este punto esta enfocado a la base sólida de los diseños; la estructura de la aplicación puede ser un tanto psicodélica si se quiere, pero deberá hacer referencia a cómo se conjunta la aplicación global Sin una buena estructura se carecería de orden y significado, la relación entre las pantallas y la información contenida en ellas juegan un papel en cómo el usuario final percibe la aplicación, se tendría que hablar de un balance en cuanto a la distribución de la información en esas pantallas. Si hay demasiada información en una pantalla y es insuficiente en otra, la aplicación parecerá ladeada o fuera de balance. La falta de estructura y balance hace más difícil la comprensión del usuario hacia esa interfaz.

2.5.1.3. Relación de elementos.

Es importante destacar visualmente las relaciones entre los elementos de la aplicación, por ejemplo si un botón amplía la información de una lista desplegable es importante que estos dos elementos estén conectados entre sí visualmente de modo que la relación sea obvia para el usuario, esto podría implicar que se mantenga el botón y la lista cerca uno de otra sobre la pantalla o bien enmarcarlos en el mismo control Si la pantalla no es más que un agrupamiento aleatorio de botones, así es exactamente como se verá.

2.5.1.4. Enfoque y Énfasis.

El enfoque se refiere a la identificación del tema o idea central sobre la que girará su pantalla, y por otro lado el concepto de énfasis se refiere a la elección de los controles o tema central para hacerlos sobresalir de modo que el usuario entienda qué cosa en la pantalla es de mayor importancia. Estos conceptos refuerzan también los de estructura y balance. Si su aplicación tiene un enfoque sólido, la estructura de ésta le parecerá fuerte al usuario y por lo consiguiente atractiva La aplicación debe enfocarse en realizar una tarea y hacerla bien, un buen ejemplo de ello es la aplicación WinZip todo lo que hace es comprimir y descomprimir archivos, no tiene un defragmentador de disco, un Explorer o un procesador de Textos integrados, se concentra en la tarea de comprimir archivos y lo hace bien, es decir la interfaz con el usuario es concisa, enfocada y hace su trabajo.

2.5.1.5. Jerarquía de la Información.

Este concepto se aplica tanto al diseño de pantallas como al de datos, usted debe decidir que información es la más importante y por lo tanto tenerla contemplada en la pantalla inicial, que información en una segunda pantalla y así sucesivamente Hay varios puntos que nos pueden ayudar a descifrar la jerarquía y son los siguientes:

- ¿Cuál es la información más importante para el usuario?

- ¿Qué necesita hacer el usuario en primer término, en segundo, etc ?

- ¿Qué es lo que debe ver el usuario en la primera pantalla, en la segunda, etc.?

- ¿Qué prioridades tiene el usuario, en cuanto a que la organización de pantallas le ayuda o perjudica y sobre qué parte le gustaría enfatizar?

2.5.1.6. Unidad e Integración.

Esta característica se relaciona en cuanto al modo de integrar la aplicación en un escritorio generalizado, así como la manera de interactuar con otras aplicaciones, si ésta permanece apartada podría estancarse. Para ello se debe tener una visión muy clara de lo que se está realizando, al seguir todos los lineamientos antes descritos se podrá obtener una aplicación muy semejante a una aplicación estándar.

2.6. Diseño del Sistema de Información propuesto.

De acuerdo a los principios de diseño de BD el SI se realizará como parte de todo un conjunto de información, por lo que se cumpliría con el principio de perspectiva global, y en cuanto a los archivos convencionales el SI utilizará archivos compatibles con Dbase, por ser un manejador muy popular en la mayoría de las PC's La estructura modular del SI se ilustra en la figura 2.9.

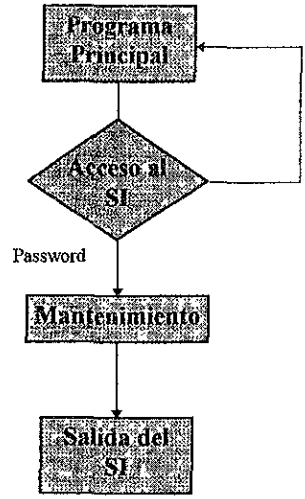


Figura 2.9. Módulos del SI propuesto

2.6.1. Objetivo de diseño.

El diseño del SI se basa en los principios que Windows 95 ha propuesto, principalmente el de Control, pues es fundamental proyectar al cliente las acciones que ocurren en pantalla, otro punto importante es la Consistencia, puesto que la manera de presentar los datos influirá sobre el uso de la aplicación; pero en general todos los principios tratados son esenciales en cualquier diseño.

Los objetivos del modulo Programa principal (PP) son la presentación y la coordinación del acceso a la aplicación El objetivo del modulo de Password es restringir el acceso a los usuarios del SI. Los objetivos del modulo de Mantenimiento son consultar los dispositivos de campo para saber la ubicación física (conexión) y el tipo de modulo que puede ser de entradas - salidas (E/S) analógicas o discretas hasta el tablero de control, por otro lado deberá proporcionar para los diagramas de escalera, el escalón y el no de programa en donde se utiliza una señal de campo, y por último indicar que tipo de barrera tiene conectada una señal. El objetivo del modulo de Salida del SI es como su nombre lo dice proporcionar a los usuarios una la salida directa del sistema.

2.6.2. Elementos de diseño.

El flujo de información que se maneja en el PP es sencillo, ya que lo único que se controla es la presentación del programa y la preparación de la aplicación, como se describe en la figura 2.10.

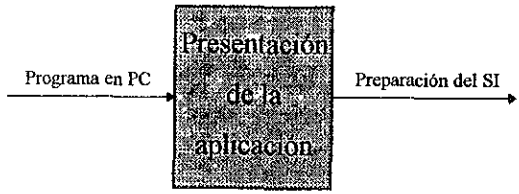


Figura 2.10. *Flujo de información del PP.*

El modulo de Password presenta un flujo de información sencillo pero a la vez importante, ya que al presentarse la petición de entrada al sistema, este la restringe si la clave no es correcta y regresa al modulo PP, la figura 2.11 ejemplifica dicha acción.

El flujo de información que se utiliza en Mantenimiento es muy complejo, ya que en base a la selección de un Símbolo (conocido como TAG) y un tipo de archivo (los de conexión), es la manera de relacionar y desplegar la información sobre la pantalla de

consulta; este procedimiento se esquematiza en la figura 2.12. Por otro lado el flujo de datos que se lleva a cabo en la Salida del SI es clara y concreta, puesto que lo único que se requiere es la petición de salida y la orden para ejecutarla, como se ejemplifica en la figura 2.13.

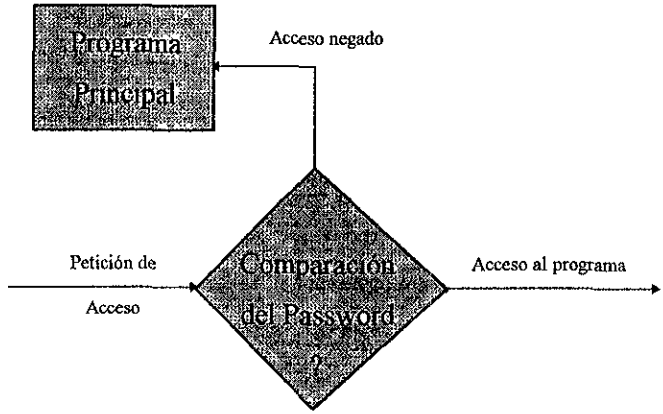


Figura 2.11. *Flujo de información de Acceso al SI.*

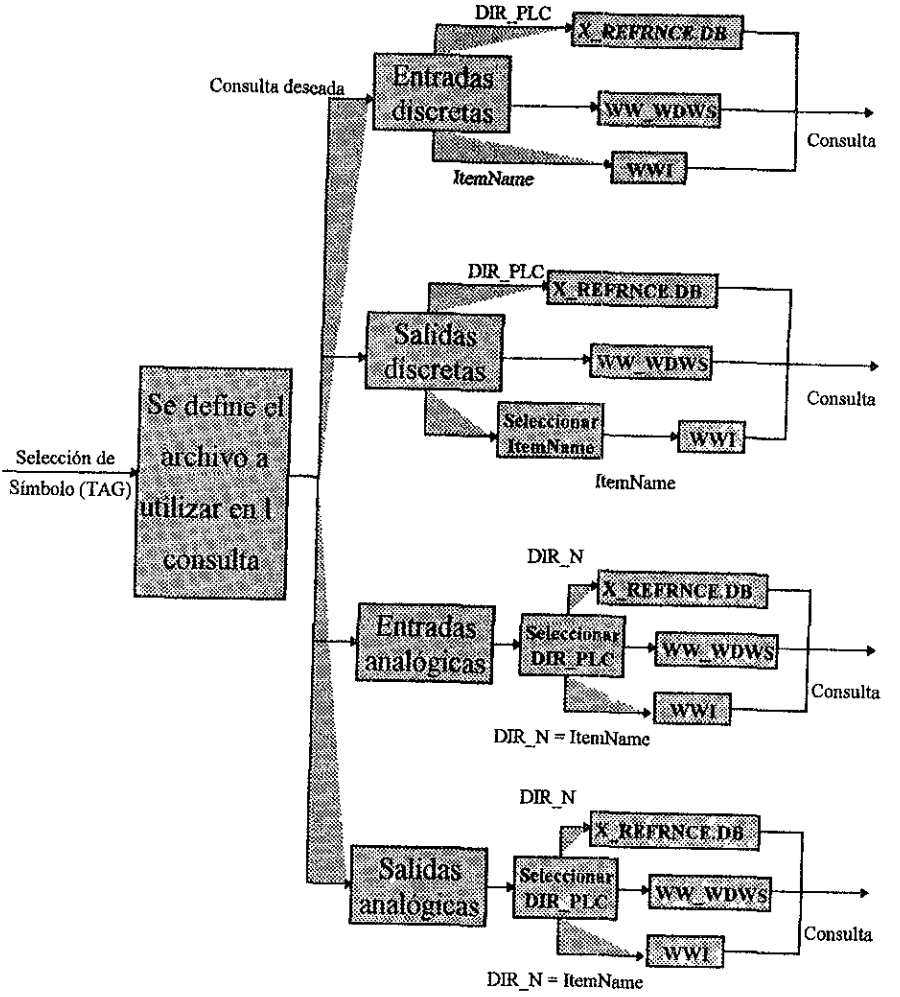


Figura 2.12. Flujo de información de Mantenimiento

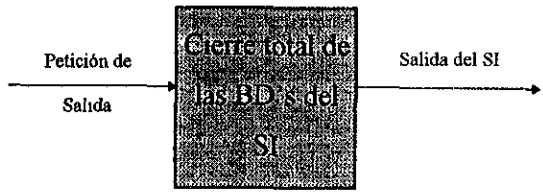


Figura 2.13. Flujo de información de Salida del SI.

2.6.3. Diseño de entradas y salidas.

Se desea diseñar 3 pantallas, una para la presentación del sistema, otra que sería la pantalla de Password's y la otra para desplegar toda la información que se requiere consultar. La primer pantalla se genera como resultado de una respuesta afirmativa para ejecutar el programa en la PC, como se esquematiza en la figura 2 14

La pantalla de Password's acepta diferentes cadenas de caracteres y las compara contra otra que es usada por el programa como llave única de acceso, como resultado a dicha acción se obtiene el regreso a la pantalla anterior o el acceso a la pantalla de Mantenimiento, esto se ejemplifica en la figura 2.15.

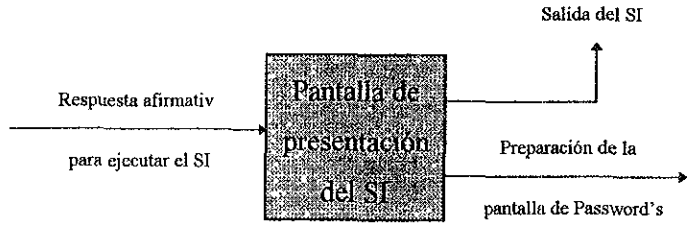


Figura 2.14. E/S de la pantalla de presentación.

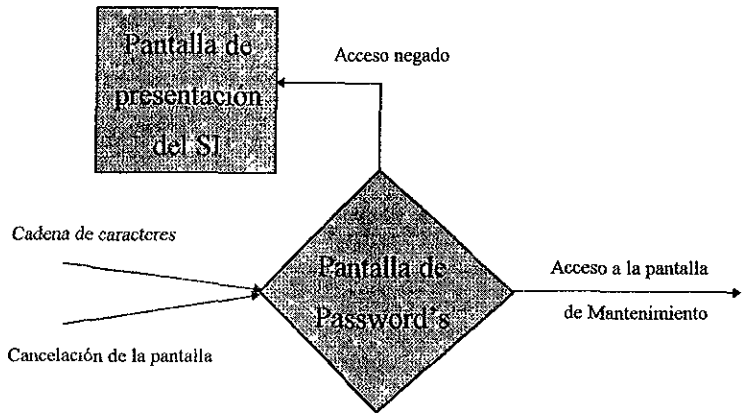


Figura 2.15. E/S de la pantalla de Password

La pantalla de Mantenimiento es la estructura del SI, ya que a través de ella se logrará el objetivo de éste. La pantalla requiere saber el tipo de archivo inicial con el cual se abocará a la tarea de desplegar la información pertinente de los datos que se piden, por otro lado puede desplegar información de las pantallas de WWI con el simple hecho de requerirla. Para mayor detalle ver la figura 2.16.

Figura 2.16. E/S de la pantalla de Mantenimiento.

2.6.4. Diseño de las base de datos.

Para diseñar las BD's se requirieron de archivos del programa de control WWI, de archivos basados en manuales como por ejemplo el de referencias cruzadas del PLC Crítico, el del modulo A de compresión de la plataforma POL - A del PLC Crítico, entre otros. En la tabla 2.1 se listan los diferentes tipos de archivos con su procedencia respectiva

Archivo	Procedencia
IAMOD4.DBF	WWI
IDMOD4 DBF	WWI
MRMOD4.DBF	WWI
MIMOD4.DBF	WWI
MDMOD4.DBF	WWI
DIMOD4 DBF	WWI
DDMOD4.DBF	WWI
DMMOD4.DBF	WWI
DRMOD4.DBF	WWI
VARMOD4.DBF	WWI
XREFRNCE. DBF	Referencias cruzadas del PLC crítico
B-00,R-0 DBF	Modulo A de compresión de la plataforma POL-A
B-00,R-1. DBF	Modulo A de compresión de la plataforma POL-A
B-00,R-2. DBF	Modulo A de compresión de la plataforma POL-A
B-00,R-3 DBF	Modulo A de compresión de la plataforma POL-A
B-00,R-4. DBF	Modulo A de compresión de la plataforma POL-A
B-00,R-5. DBF	Modulo A de compresión de la plataforma POL-A
B-00,R-6. DBF	Modulo A de compresión de la plataforma POL-A
B-00,R-7. DBF	Modulo A de compresión de la plataforma POL-A
B-01,R-0. DBF	Modulo A de compresión de la plataforma POL-A
B-01,R-1. DBF	Modulo A de compresión de la plataforma POL-A
WW_WDWS NDX	WWI

Tabla 2.1. Descripción de los tipos de archivos.

Los nombres de los archivos de WWI describen el tipo de datos que contienen, es decir las primeras 2 o 3 letras indican el tipo de información que manejan y el complemento define que los archivos provienen del modulo 4 del PLC (MOD4) con excepción del último que sólo es de referencia; la tabla 2.2 describe los tipos de datos para cada uno de estos archivos.

Archivo	Tipo de datos
IAMOD4	Análogos Indirectos
IDMOD4	Discretos Indirectos
MRMOD4	Reales de Memoria
MIMOD4	Enteros de Memoria
MDMOD4	Discretos de Memoria
DIMOD4	Enteros DDE
DDMOD4	Discretos DDE
DMMOD4	Mensajes DDE
DRMOD4	Reales DDE
VARMOD4	Variables de diferentes fuentes
WW_WDWS	Referencia archivos WINxxxxx.WIN de pantallas

Tabla 2.2. Tipo de información de archivos WWI.

Los nombres de los archivos del Modulo A de compresión de la plataforma POL - A al igual que los de WWI indican el lugar físico en donde se encuentran y también el tipo de información que contienen; la letra B (B-0x) indica el número de chasis o bastidor y la

R (R-x) el número de grupo o ranura de conexión dentro del PLC. La tabla 2.3 contiene el tipo de información y modulo para estos.

El archivo de referencias cruzadas contiene información descriptiva de todos los símbolos que se utilizarán, así como del número de programa y escalón en donde se ocupa una señal para los diagramas de escalera.

Archivo	Tipo de señal	Tipo de modulo
B-00,R-0	Entradas Discretas	1771-IBD
B-00,R-1	Entradas Discretas	1771-IBD
B-00,R-2	Entradas Discretas	1771-IBD
B-00,R-3	Entradas Discretas	1771-IBD
B-00,R-4	Entradas Discretas	1771-IBD
B-00,R-5	Salidas Discretas	1771-OQ16
B-00,R-6	Salidas Discretas	1771-OQ16
B-00,R-7	Salidas Analógicas	1771-NOC
B-01,R-0	Entradas Analógicas	1771-IFE
B-01,R-1	Entradas Analógicas y de Frecuencia del HSDE	1771-IFE

Tabla 2.3. Tipo de información de archivos WWI

CAPÍTULO 3

EVALUACIÓN

3.1. Ingeniería de Software.

El software de un Sistema se basa en programas, datos y documentos que conforman el software de la aplicación y el software del Sistema El software de la aplicación implanta los procedimientos requeridos para realizar las funciones de procesamiento de la información y el software de Sistema implanta funciones de control que permitan al software de la aplicación comunicarse con diversos elementos del software.

La ingeniería de software (IS) es una disciplina para el desarrollo de software de alta calidad para sistemas de computadora. La figura 3 1 ilustra los pasos genéricos del proceso de IS, las distintas partes de la misma ilustran los pasos que se deben llevar a cabo

y las diferentes representaciones del software que se derivan según se evoluciona del concepto a la realización

Una definición dada por Fritz Bauer es la siguiente

"El establecimiento y uso de principios de ingeniería robustos, orientados a obtener Software económico que sea fiable y funcione de manera eficiente sobre máquinas reales". [9]

La IS surge de la ingeniería de sistemas y de hardware, está compuesta por una serie de pasos que abarcan métodos, herramientas y procedimientos, que son las bases para construir software de alta calidad de una forma productiva, a estos pasos se les denomina frecuentemente "paradigmas de la ingeniería del software". Pero tres de los principales paradigmas son el ciclo de vida clásico (ingeniería de sistemas, análisis, diseño, codificación, prueba y mantenimiento), construcción de prototipos y modelo en espiral; aunque el proceso de desarrollo de software contiene tres fases genéricas independientes del paradigma de ingeniería elegido que son: definición, desarrollo y mantenimiento

[9] Roger S Pressman, "Ingeniería del Software, un enfoque practico", Pag. 154.

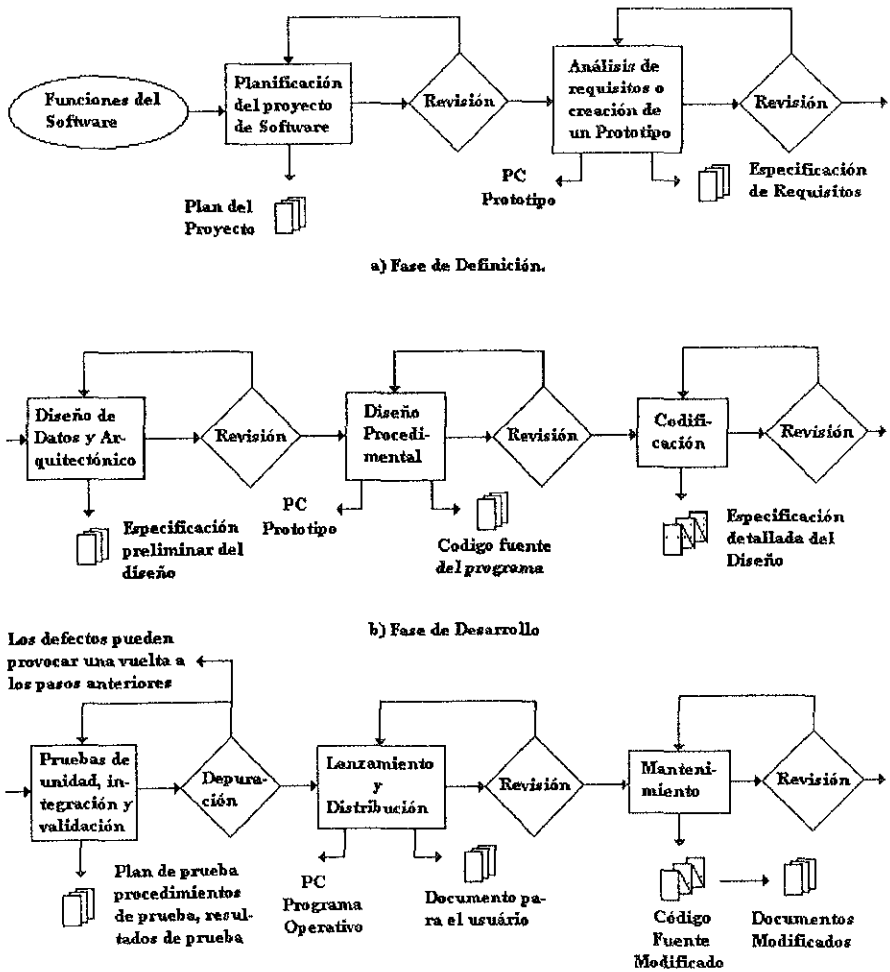


Figura 3.1 Ingeniería de Software

Un ingeniero de Software es muy distinto de un programador; un programador sólo se dedica a codificar el planteamiento de un proyecto en marcha, mientras que un ingeniero de software habla con el usuario potencial y lista sus necesidades, posteriormente escribe lo que se debe de hacer para construir su diseño, es entonces y sólo entonces cuando el ingeniero de software comienza a escribir el código. Es por ello que la IS es el enfoque disciplinado para la programación.

3.1.1 Metas y Principios.

Siempre que se requiera desarrollar una aplicación en primera instancia se necesita tener una meta. Una meta directa del desarrollo de software consiste en hacer que el producto terminado coincida con la especificación de requisitos. Por otro lado cabe mencionar que lo único constante es el cambio, la aplicación pasará la mayor parte de su ciclo de vida en la fase de mantenimiento, por lo que es importante que se tenga un conjunto de metas que trasciendan al cambio, siendo que las cuatro reglas generales de la IS son. modificabilidad, eficiencia, confiabilidad y comprensibilidad.

- **Modificabilidad.-** Es seguro que durante el ciclo de vida de su aplicación ésta requiera de modificaciones varias veces, algunas como resultado de un error que usted o su cliente descubrieron, otras por solicitud de una característica o mejora solicitada por su(s) usuario(s), por ello es importante que el trabajo que se realizó no sucumba cuando se le requiera hacer un cambio. Se debe poder

hacer un cambio a la aplicación sin mover los cimientos, esta meta implica un cambio controlado, en el que ciertas partes de la aplicación cambien mientras que otras permanezcan igual, cosa que no es nada fácil, puesto que muchos lenguajes no soportan esto y son muy sensibles a un cambio. El proceso a tomar en cuenta para hacer un cambio en la aplicación es realizar el mismo cambio en la documentación, agregando el nuevo requisito del usuario al documento de requisitos, luego rastreando ese requisito a través de su documento de diseño y por último haciendo el cambio al código. Todo lo anterior ayudará a que su *diseño sea más estable.*

- **Eficiencia.-** La aplicación debe usar los recursos óptimos disponibles, dado que muchas aplicaciones comparten recursos en memoria y se ejecutan a un tiempo en la PC. Refiriéndose como recursos al tiempo y espacio, es decir la aplicación pudiera tener requisitos para ejecutarse en un marco de tiempo dado, si se trata de reunir datos sensibles al tiempo o información de red en vivo. Un requisito de tiempo podría significar que usted dejará algunos ciclos de CPU para alguien más, la aplicación debe ser buena compartiendo ese tiempo con otras aplicaciones.

- **Confiabilidad.-** Esta meta es muy importante en especial si la aplicación es responsable de alguna función crítica. Las aplicaciones que operan por largos periodos sin intervención de alguien deben ser capaces de recuperarse de

problemas automáticamente. El problema principal se enfoca al costo elevado de alguna falla para permitir algo que no sea la más alta confiabilidad del software.

- **Comprensibilidad.-** Para que la aplicación sea comprensible debe ser fácil de entender, esta es una meta difícil en un Sistema complejo; esta meta surgió de la POO y una forma de hacerla alcanzable es asegurarse de que el diseño de sus aplicaciones e instrumentación refleje el mundo real. Si los objetos de la aplicación son modelos de objetos del mundo real, relacionarlos en nuestra mente resultará más fácil. Otra forma en que el código resulta comprensible es gracias a un nivel básico de legibilidad, por el uso de buenas técnicas de codificación, el estilo y la incorporación de comentarios para que el producto final sea muy comprensible.

Se deben definir un conjunto de principios para obtener las metas antes descritas; principalmente se enuncian siete y de los cuales se trataran a continuación.

- **Modularidad y localización.-** Estos principios de derivan fácilmente de la estructura de clase Las clases son inherentemente modulares y la localización se refiere a mantener los módulos organizados de manera lógica, en donde cada grupo debe ser un grupo de código lógicamente relacionado Al ir construyendo

el código, si se escriben pequeños módulos autónomos, éstos serán fácilmente transportables a otros proyectos de Software.

- **Abstracción y ocultamiento de información.-** Estos principios tienen un buen apoyo en las clases. Por ejemplo en Delphi las palabras reservadas *private* (privada) y *protected* (protegido) le permiten dos niveles de soporte para ocultamiento de información a objetos externos de una clase en particular. La abstracción se maneja a través del ocultamiento de información y a través de la interfaz/instrumentación de los archivos de clase Sólo permita que se haga pública la naturaleza abstracta de la clase, los detalles de la instrumentación permanecen ocultos.

- **Confirmabilidad.-** Esto se logra mediante una combinación de una verificación de tipos de datos y la construcción de módulos que puedan ser probados. La verificación de tipos es la manera en que el compilador confirma que una variable se usa adecuadamente. Los módulos examinables permiten al programador hacer pruebas lógicas de precisión en módulos individuales; recuerde que es mucho más fácil construir programas pequeños libres de error, que programas grandes libres de error Básicamente este principio implica que se deben dividir las aplicaciones en módulos que puedan ser probados.

- **Uniformidad y terminalidad.-** Estos principios pertenecen por completo al dominio del programador El código es más fácil de leer y corregir si se escribe y comenta de una manera uniforme; los módulos terminados cuando fueron escritos originalmente por lo regular no necesitan reescribirse o aumentarse cuando surgen nuevas necesidades al llevar una uniformidad. Por su parte la terminalidad es una condición para estar libre de error, es difícil de alcanzar, pero si se propone llevar a cabo este principio puede maximizar la adaptabilidad de su código y minimizar los cambios que se necesiten más adelante, ello porque los cambios son una acción a que los errores aparezcan con mayor frecuencia en la aplicación.

3.2 Tipos de archivos.

Es importante mencionar los tipos de archivos que se manejan en estos casos pues dependiendo del tipo que se use es la manera de accesar la información por ejemplo, un archivo de texto no contiene el mismo formato de información que uno de BD o un archivo binario es completamente diferente a uno secuencial, etc.; además cambia también el tipo de dato que puede contener ya sea numérico o alfanumérico, o la capacidad de información, etc

Un archivo es *un conjunto organizado de datos que se almacenan en disco duro, disco flexible, CD-ROM, cinta u otros medios de almacenamiento* [10], para las BD se requiere crear, leer y escribir archivos, pero también se tienen muchos otros usos. Cabe mencionar que los archivos tienen propiedades que manifiestan su configuración. La Tabla 3.1 lista los tipos de atributos que puede tener un archivo.

Atributo de Archivo	Descripción
Sólo Lectura	Permite leer los archivos, pero no actualizarlos o eliminarlos.
Oculto	Evita que el archivo aparezca en los listados normales de directorio.
Sistema	Marca al archivo para uso del sistema y evita su visualización en una lista de directorio.
Respaldo	Este atributo está apagado si el archivo fue respaldado.

Tabla 3.1. *Atributo de archivos.*

Todo archivo tiene un byte de atributo para almacenar su configuración y cada configuración se guarda en uno de los ocho bits que conforman al byte. Sólo se usan seis bits, dos para etiquetas de directorio y volumen, y cuatro para los atributo de la tabla 3 1,

[10] Osier, Grobman y Batson, "Aprendiendo DELPHI en 21 días", Pag. 353.

los cuales se activan y desactivan por medio de programas que usan o crean archivos Por lo tanto el byte de atributo se descifra con el siguiente mapa de bits. (Ver Tabla 3.2)

Byte de Atributo	Mapa de bit
Bit 0	Bit de sólo lectura
Bit 1	Bit de archivo oculto
Bit 2	Bit de archivo de sistema
Bit 3	Volumen ID bit
Bit 4	Bit de subdirectorio
Bit 5	Bit de respaldo
Bit 6	Sin uso
Bit 7	Sin uso

Tabla 3.2. Mapa de Bits de un archivo.

Se destacan dos tipos básicos de archivos que son los de texto y los binarios, hay varias formas de poder almacenar o dar formato a los datos en estos tipos de archivos pero siempre son de una de estas categorías. Cada tipo de archivo tiene sus ventajas y desventajas, lo que puede funcionar para una aplicación puede no hacerlo en otra, por ello se requiere saber que tipo de archivo satisfacerá nuestras necesidades.

Un archivo de Texto está compuesto de caracteres ASCII puros, regularmente los datos se almacenan y recuperan en forma secuencial, es decir línea por línea, y cada línea

termina con caracteres de control de carro (Carriage Return - $D) y de alimentación de línea (Line Feed - $A). Si se planea trabajar con datos en forma secuencial y no necesita saltar a varias ubicaciones en el archivo, uno de Texto podría ser una buena opción; por otro lado el hacer una búsqueda en archivos grandes o realizar muchos cambios puede resultar muy tedioso e ineficiente puesto que la información se maneja secuencialmente, pero en muchas situaciones como en la exportación de datos a formato estándar éste tipo de archivo sería el indicado.

El otro tipo de archivo es el Binario, en donde se ubican todos los archivos que no son de texto y por consiguiente contienen información binaria mediante código ASCII, que a diferencia de los de texto cualquier archivo abierto como binario incluyendo de texto, de programa, de mapa de bits, etc , es fácil de leer, por lo que la responsabilidad de manejar los datos es grande. Existen dos categorías para estos archivos, una es la de Archivos de Tipo, en donde uno decide el formato o estructura de este y el tipo de dato que contendrá como puede ser enteros, reales, cadenas de caracteres, etc La otra categoría es la de Archivos sin Tipo, que es un tanto más flexible para manipular archivos puesto que en ellos es posible saltar a cualquier ubicación, cambiar un byte o un bloque completo, almacenar los datos y cerrar el archivo, etc., debido a que no se tiene una estructura rígida de almacenamiento de información, pero el único inconveniente es que para manipular los datos es necesario determinar exactamente la parte del archivo en donde se desea trabajar y por ello se recomienda cierta precaución.

3.3 Factores que afectan la productividad.

Las causas del problema de productividad es un tema complejo y los resultados de investigaciones varían debido a diferencias tanto de puntos de vista económicos, periodos de estudio, hipótesis y preguntas diversas que los investigadores se plantean, es conveniente presentar algunas razones que se citan con más frecuencia.

- **Inversión.-** Existe una fuerte correlación entre la inversión y el mejoramiento de la tasa de productividad, ya que cuando aumenta la inversión aumenta la productividad y a su vez crea un mayor porcentaje de mercado captado, una tasa baja de introducción de productos, alta capacidad de utilización, etc.

- **Razón capital/trabajo.-** El retraso en la productividad del trabajo está ligada íntegramente a las reservas de capital que en ese momento se tengan

- **Investigación y desarrollo.-** No todos están de acuerdo en que los gastos en investigación y desarrollo repercutan necesariamente en el mejoramiento de la productividad.

- **Utilización de la capacidad.-** Es el tiempo en que las plantas están operando y la cual es una razón muy ligada a la productividad del trabajo

- **Reglamentación del gobierno.-** Se refiere a la reglamentación para proporcionar equilibrio entre el progreso industrial y las metas sociales deseadas, como un medio ambiente limpio y lugares de trabajo. El dinero invertido en cumplir con reglas innecesarias casi siempre se distrae de la investigación y el desarrollo útiles. Una reglamentación excesiva causa retrasos e incertidumbre haciendo más difícil satisfacer los criterios ordinarios de inversión o inflación.

- **Vida de planta y equipo.-** Se relaciona con la vida útil promedio de toda la industria que conforma a las empresas, debido a que si esta se alargara mucho podría ser un indicativo de falta de modernización, y con todo ello una baja considerable en la productividad de las mismas.

- **Costos de Energía.-** Este punto se agudiza en los incrementos del costo de energía, ya que cualquier empresa y en mayor peso a las que se dedican al procesamiento de información, este recurso es fundamental. A pesar de un aumento en la productividad parcial, si los insumos de energía se elevan con mayor rapidez, el efecto neto puede ser un aumento en los costos de producto final.

- **Mezcla de la fuerza de trabajo.-** Debe existir un equilibrio entre las diversas fuerzas laborales de las empresas, debido a que muchas veces crece el número de personas dedicadas a una actividad en especial, mientras que la capacitación orientada a mayores habilidades no se desarrollan al mismo paso, y ello repercute sobre las cifras de producción por hora.

- **Ética del trabajo.-** Esto se relaciona con la estimación del tiempo que se pasan los empleados desarrollando sus labores, ya que muchas veces el número real de horas es siempre menor que las horas que se pagan o viceversa.

- **Temor a la perdida de empleo.-** Siempre que se mejoran las técnicas para el mejoramiento de la productividad en las empresas existe una tremenda resistencia. Los empleados de muchas organizaciones no comparten las ganancias de la productividad y siempre tendrán ciertas preocupaciones sobre los motivos de los administradores, por lo que se encuentran recelosos de las mejoras a la productividad cuando no ha habido suficiente comunicación antes de que se pongan en marcha las mejoras. Cabría mencionar que cuando se introdujeron las computadoras este era uno de los principales problemas.

- **Influencia sindical.-** Este factor de investigación indica que muchas veces los sindicatos proporcionan cierta influencia negativa sobre varios factores de las empresas incluyendo a la productividad. (Ver tabla 3.3.)

- **Administración.-** Se sabe que el papel de la administración en la disminución de la productividad es uno de los factores más importantes y a continuación se darán unas cuantas razones del porqué :

  - 35 % de la pérdida de productividad se debe a una pobre planeación y programación del trabajo.

  - 25 % de la pérdida de productividad se debe a instrucciones dadas a los empleados fuera de tiempo o con falta de claridad

  - 15 % de la pérdida de productividad es por falta de habilidad para ajustar la cantidad de personal y sus obligaciones durante los periodos pico y los de holgura.

  - 25 % de la pérdida de productividad restante se debe a mala coordinación en el flujo de materiales, falta de disponibilidad de herramientas necesarias, tiempo de traslados excesivos y, falta de

supervisión en el tiempo de inicio y terminación de las labores de los trabajadores principalmente.

Efecto en las empresas	Grandes (%)	Medianas (%)	Pequeñas (%)
Reducción en la productividad	26	23	14
Reglas de trabajo inflexibles	17	12	7
Salario y prestaciones inflexibles	15	12	19
Menor lealtad a la empresa	13	12	5
Precios más altos	12	18	31

Tabla 3.3. Efectos negativos que los empresarios citan con mayor frecuencia

Entre los factores que afectan el crecimiento de la productividad, se encuentran también las metas a corto plazo de ganancias y el deseo de producir aún a costa de la calidad **La Calidad y la Productividad van juntas**. Si el producto no se realiza bien la primera vez, imagine los costos extras innecesarios para reprocesarlos y, la pérdida de la buena voluntad del cliente y del porcentaje de mercado que ocasiona

3.4 Recursos para nuevos Sistemas.

Es importante que para la evaluación de cualquier Sistema se necesite destacar los recursos que a lo largo del desarrollo del mismo se requerirán. Ello porque siempre se debe prever el uso de nuevas o mejoras en sus aplicaciones. En la mayoría de las instalaciones maduras de cómputo existen más demandas de servicios de lo que los recursos disponibles pueden ofrecer Por otro lado se debe prever el presupuesto para el desarrollo de nuevos Sistemas, ya que éste sólo es una parte de todo el presupuesto total asignado a cualquier departamento.

Las responsabilidades de un departamento de servicios de información son muchas y variadas, por un lado uno de los primeros aspectos es operar los Sistemas existentes, ya que estos se deben procesar rutinariamente. Otro punto es el mantenimiento, ya que este también ocupa recursos, pues en donde existen diversos Sistemas en operación se requiere dar mantenimiento y reparación a los programas Como se había mencionado las mejoras a los Sistemas existentes son solicitadas por los usuarios, éstas modificaciones ocasionan cambios en los programas y algunas veces requieren de un nuevo equipo de cómputo. Y de una manera drástica el desarrollo de un nuevo Sistema requerirá por lo regular de una mayor cantidad de recursos.

Se ha tratado muchas veces que el mantenimiento y las mejoras requieren alrededor del 50% del esfuerzo de programación en la mayoría de las compañías, sin

embargo existen principalmente dos categorías de recursos de servicios de información que son personal y máquinas Las capacidades de las máquinas son indispensables para operar y desarrollar los SI Los diferentes tipos de SI requieren equipos diferentes, ya que por ejemplo una aplicación robusta de BD requiere dispositivos de almacenamiento de datos, mientras que un Sistema en línea necesita equipo para comunicaciones y terminales.

Desde el punto de vista humano algunos recursos de los servicios de información no se pueden intercambiar entre los diversos puestos, por lo que a continuación se describen brevemente.

- **Operador:** personal capacitado para operar la computadora y su equipo periférico.

- **Personal administrativo:** procesa manualmente todo lo relacionado a entradas y salidas.
- **Capturista:** transcriben datos a información legible por la máquina.

- **Programadores de Mantenimiento:** reparan errores en programas que dirigen a las computadoras y son responsables de mejoras a Sistemas actuales.

- **Analistas de Sistemas:** trabajan con usuarios en la definición de *especificaciones de nuevos Sistemas*

- **Programadores de aplicación:** convierten las especificaciones de un Sistema en programas de computadora necesarios para procesar los datos y producir la *salida deseada.*

- **Programadores de Sistema:** trabajan con el software de control de la computadora y se encuentran en instalaciones

- **Gerentes:** se emplean para diferentes funciones tales como operación y diseño de Sistemas para el caso de que el departamento sea suficientemente grande.

Deben coincidir los requerimientos de servicio en relación a los recursos. A excepción de que alguna aplicación se vaya a eliminar, un departamento de SI debe contener. equipo, operadores, personal administrativo, programador de Sistema y programadores de mantenimiento. Para las mejoras y el desarrollo de nuevos SI los recursos que se pueden disponer son: analistas de Sistemas, programadores de aplicación, los gerentes necesarios y la capacidad de máquina

La capacidad de máquina se puede aumentar, pero este hecho normalmente toma varios meses en obtener e instalar un nuevo equipo de cómputo Por otro lado implicaría la contratación de nuevo personal, limitándose así la rapidez con que los nuevos empleados puedan ser integrados a la nueva organización y ser productivos. Por todo lo anterior se concluye que a corto plazo poco es lo que puede realizarse para incrementar los recursos dedicados al desarrollo de los nuevos Sistemas dentro de una organización Cabe mencionar que podrían utilizarse recurso adicionales al adquirir paquetes de aplicación y/o servicios de consultoría externos a la empresa y a largo plazo tal vez los usuarios quedarían satisfechos con los recursos dedicados al desarrollo del nuevo SI

3.5 Análisis Costo - Eficacia.

El análisis Costo - Eficacia es un factor que puede influir en el éxito o fracaso de un SI, en esta evaluación se totalizan los costo que produjeron el desarrollar la aplicación y se comparan con la eficiencia que producen los beneficios monetarios resultantes; este análisis muchas veces hace reflexionar respecto a lo que se está haciendo pues si los costos son muy elevados y resulta que la eficiencia del SI es inferior a la esperada, el resultado deberá ser un rediseño a la aplicación para aumentar la productividad al procesar el trabajo con el menor incremento en el costo

3.6 Evaluación del Sistema de Información propuesto.

3.6.1. Selección de Tecnología.

Como ya se ha mencionado en capítulos anteriores se seleccionó a Delphi como la principal herramienta de desarrollo para la realización de este proyecto. Delphi es un Pascal Visual con valores agregados, es decir nos da las sólidas bases del Object Pascal de Borland más las características de construcción de aplicaciones visuales para Windows. El principal beneficio de utilizar Delphi como lenguaje de desarrollo orientado a objetos es que es un lenguaje de alto nivel de definición, en donde se pueden realizar aplicaciones ya sea de 16 o de 32 bits. Por otro lado es una herramienta fácil de autodocumentar, aspecto que hace la posibilidad de leer el código de arriba hacia abajo como si fuera la simple lectura de un libro. Una aplicación Delphi siempre será en ultima instancia un programa ejecutable o un grupo de ejecutables, sólo cuando se realiza una aplicación con BD's se requieren de archivos DLL's (Dynamic Link Library) y algún otro tipo de archivo si se utilizó para la misma. Un proyecto Delphi consta de formas, unidades, opciones, configuraciones, recursos, etc.. La tabla 3.4 muestra los archivos que son creados por Delphi, por el compilador y otros que pueden utilizarse para una aplicación cualquiera

Del hardware utilizado para la operación del SI dependerá la optimización de los resultados esperados, porque no es lo mismo utilizar una PC con procesador 486 a utilizar una Pentium, o bien utilizar un disco duro con espacio libre reducido (que muchas veces

retarda los procesos de información) a utilizar uno con suficiente capacidad de almacenamiento, es por ello que en la tabla 3 5 menciono el hardware sugerido para un funcionamiento adecuado.

Utilidad	Extensión	Descripción
Archivo de proyecto	.drp	Guardar información sobre formas y unidades, parte principal de la aplicación
Archivo de unidad	.pas	Almacena código
Archivo de forma	dfm	Archivo binario con información relativa a sus formas
Archivo de opción de proyectos	dfo	Contiene configuraciones de opciones del proyecto
Archivo de recursos	res	Archivo binario que contiene un icono que usa el proyecto
Archivos de respaldo	.-dp, .-df, -pa	Copias de seguridad o respaldo del proyecto, forma y unidades respectivamente
Archivo ejecutable	.exe	Archivo que ejecuta la aplicación final
Archivo objeto de unidad	.dcu	Versión compilada de los archivos de unidad para vincularse con el ejecutable
Archivos de vinculación	dll	Biblioteca de vinculación dinámica
Archivos de ayuda	.hlp	Archivos Windows estándar de ayuda
Archivos de imagen o gráficos	bmp, .ico, wmf	Archivos usados para construir aplicaciones atractivas y amigables con el usuario

Tabla 3.4. Archivos que utiliza Delphi en proyectos

Hardware típico	Hardware mejorado
PC con procesador 486	PC con procesador Pentium
10 MB de espacio libre en disco duro	20 MB de espacio libre en disco duro
8 MB en RAM	16 MB en RAM
Tarjeta de Red específica del equipo (Ethernet)	Tarjeta de Red específica del equipo (Ethernet)
Windows 3.1, 3.11 o Windows 95	Windows 95

Tabla 3.5. Hardware sugerido.

El lenguaje de consulta utilizado es SQL (Structure Query Language [Lenguaje de consulta estructurada]), que es un lenguaje de alto nivel y por lo regular es considerado como estándar entre plataformas de BD's (puede utilizar archivos compatibles con Dbase, Paradox, Excel, INTRBASE, Fox Pro, Oracle, etc.).

3.6.2. Alcance del Sistema de Información.

Se piensa que este SI tendrá una vida útil muy prolongada, pero dependerá del tiempo en que dure funcionando el sistema de PLC's 5/40 y 5/60 en la plataforma Aunque el equipo se acaba de instalar, este sistema de PLC's vino a sustituir a otro sistema de control llamado BENDIX (se habla de un tiempo no mayor a un año), el departamento en

el cual laboro todavía proporciona mantenimiento a dicho sistema, por lo que si partimos de esta idea y tomamos en cuenta la vida útil de este, entonces estaríamos hablando de entre 10 y 15 años aproximadamente.

3.5.3. Recursos y capacidades.

Los recursos necesarios para llevar a cabo la aplicación son variados. En primer lugar se contemplaron los requerimientos de Hardware y Software como por ejemplo la versión que se utilizará de Delphi puesto que hay 3 distintas (Delphi Desktop que es capaz de conectarse con Dbase y Paradox a través del dispositivo de BD de Borland, Delphi Developer capaz de conectar cualquier fuente de datos por medio de un manejador ODBC y Delphi Cliente / Servidor que proporciona conectividad con manejadores de alta velocidad de 32 bits y servidores SQL de BD's como SyBase y Oracle), las características de la PC para instalarlo, un procesador de palabras para realizar todo el trabajo escrito, etc., Otro factor importante a contemplar es el tiempo que se le dedicará a cada una de las actividades a desarrollar, el apoyo o asistencia de otros desarrolladores o personal que colabore tanto en la captura como en el suministro de información importante, etc , son recursos variados que también se deben definir. Todos los recursos antes mencionados se listaron en la tabla 3 6

Software	Hardware	Varios
Delphi C/S o	PC 486DX/2 o	Colaboradores de apoyo en la
Delphi Developer	PC Pentium	captura de las BD's, para la
Word for Windows	Disco Duro de 500 MB	información especializada y para el
Paint Brush	libres	desarrollo del SI
Corel Draw	16 MB en RAM	1 Año de tiempo aproximadamente
Excel	Impresora	Papelería (hojas, lápices, etc.)
Dbase III plus	Discos Flexibles de 3 ½ "	*Bibliografías y manuales*
Windows 95 o		especializados en la materia
Windows 3.11		

Tabla 3.6. Recursos y capacidades necesarias.

CAPÍTULO 4

DESARROLLO

4.1. Desarrollar aplicaciones de 32 Bits.

Para comprender los recursos disponibles en un ambiente de programación a 32 Bits se requiere tener un conocimiento básicos de conceptos importantes del sistema operativo, ya que Windows 95 y NT poseen ciertas características muy poderosa que pueden utilizarse y que son necesarias para hacer aplicaciones consistentes y poderosas

Una cantidad determinada de bits describe la cantidad de datos que la computadora puede mover en una sola instrucción, de ahí que multiplicar dos enteros de 32 bits en un sistema operativo de 16 se lleve muchas más instrucciones que un sistema de 32 El resultado es que el sistema de 32 bits es mucho más rápido. Por otro lado una de las principales ventajas de Windows 95 es el nuevo modelo de memoria. En Windows 3.X y en DOS se limitaba a una compleja instrumentación de memoria consistente en segmentos

de 64K, por lo que esto limitaba la cantidad de memoria a piezas de datos de este tamaño. Windows 95 utiliza un modelo de memoria plana en donde cada programa o proceso tiene un espacio virtual de memoria, es decir, realiza la tarea de mapear la memoria virtual de un proceso en localidades físicas de memoria y además presenta la funcionalidad requerida para el intercambio de memoria al disco, a fin de proporcionar más memoria virtual que la que está físicamente instalada en la máquina.

La optimización de un código nativo de 32 bits (como el que genera Delphi) da por resultado código que, se ha comprobado es de tres a cuatro veces más rápido que un código de 16 bits Por otro lado los tipos de datos de 32 bits aprovechan el direccionamiento de memoria de 32 bits en Windows 95, por lo que ahora las aplicaciones pueden acceder información de hasta 2 GB de memoria virtual, volviendo obsoleta la barrera de segmentos de 64 Kb implícita con los tipos de 16 bits.

4.2. Bases de Datos.

Comenzaremos con una pequeña definición de BD la cual es la siguiente: *una BD es una colección de datos formateados regularmente a la que más de una persona tiene acceso y/o que se emplea para más de un propósito.* [11] Las BD pueden ser una colección

[11] R. Frost, "Bases de Datos y Sistemas Expertos", Pag 53

grande de datos similares, o bien constar de varios datos relacionados; lo anterior suena un poco confuso, pero un buen ejemplo de una colección grande de datos podrían ser los datos relacionados con las reservas de asientos en las líneas aéreas, y un ejemplo de varios datos relacionados podrían ser los datos que se utilizan en un sistema de contabilidad integrado

Para comprender la funcionalidad de una BD es necesario conocer la definición de algunos términos que comúnmente se mencionan en éste medio, y son los siguientes·

Datos: Información que se puede almacenar como texto o gráficas en un campo.

Campo: Una porción discreta de datos en una tabla.

Columna: Un conjunto vertical de campos en una tabla

Renglón o fila: Un conjunto horizontal de campos en una tabla, también conocido como registro.

Tabla: Un conjunto de campos de datos en una cuadrícula de columnas y renglones.

Base de Datos: Una tabla o grupo de tablas relacionadas entre sí, por lo regular designadas por un alias

Alias: Es un nombre de código para una BD común a la cual se le aplica una conexión.

Conexión: Es una ruta totalmente calificada que informa a un componente ligado a datos sobre la ubicación física de la cual requiere información.

La figura 4.1 muestra gráficamente algunos de estos términos antes descritos para las BD's.

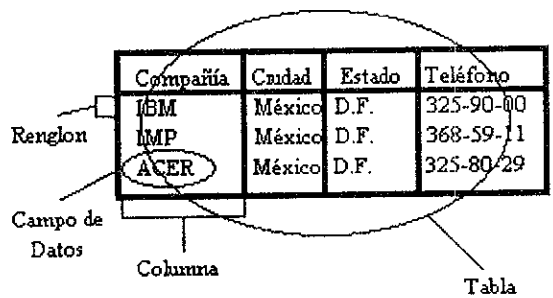


Figura 4.1 Partes de una BD

Como ya se mencionó, una BD proporciona un medio genérico para almacenar datos Un dispositivo de BD proporciona el mecanismo para manipular y visualizar datos en la BD Sin BD los programadores se verían forzados a escribir rutinas complejas para manipular archivos siempre que hubiera necesidad de accesar datos de una manera eficiente. En esencia el programador sería responsable de crear tanto la BD como el manejador de ésta

4.2.1 Sistemas de Bases de Datos.

Un sistema de BD *es un conjunto de recursos cuya responsabilidad es la de almacenar datos, mantener la seguridad de la BD reforzando las medidas de privacidad e integridad proporcionando la reserva necesaria para prever fallas en equipos o programas, y proporcionar a los usuarios el uso de rutinas necesarias de entrada y salida para acceder a la BD cuando se requiera.* [12] Toda BD tiene un punto de arranque en cuanto a su diseño se refiere, por lo que una descripción abstracta y general de esa parte del universo que los datos de la BD van a representar se le llame "esquema conceptual", la cual contiene:

- **Una lista de los tipos de *entidades* implicados:** empleados, departamentos, máquinas, etc.

[12] R. Frost, "Bases de Datos y Sistemas Expertos", Pag. 55

- **Una lista de las *relaciones* importantes entre los tipos de entidades:** empleados que trabajan en departamentos, máquinas que se construyen a partir de piezas, etc..

- **Una lista de las *restricciones de integridad* que se aplican:** si el empleado A dirige al B y el empleado A trabaja en el área C, por lo que se deduce que el empleado B debe trabajar en el área C.

Un esquema conceptual describe a la realidad y no a los datos, por ello no se refiere a vías de acceso, restricciones de privacidad y características similares Un esquema conceptual puede emplearse al comienzo del proceso de diseño para integrar los intereses de los diversos usuarios; es una descripción útil para la comunicación con usuarios no técnicos por ello está libre de jerga técnica, por otro lado ayuda al diseñador a construir un sistema de BD más duradero y una vez diseñada la BD, el esquema conceptual puede utilizarse para introducirlo a usuarios potenciales siendo importante si se considera que el sistema ha de transferirse utilizándose en otro sitio. Cabe mencionar que uno de los problemas más difíciles al crear un modelo conceptual es la obtención de un consenso de todos los usuarios potenciales del sistema al cual representará el área de aplicación correctamente.

El esquema de BD es una descripción de los datos que están almacenados en la base y que especifica los elementos de los datos almacenados y el camino de acceso proporcionado entre estos. Este esquema contiene especificaciones de restricciones e integridad, es en cierta manera similar al conceptual pero se enfoca a una descripción de datos en vez de la realidad, aunque no especifica cómo se almacenan en la realidad o cómo se facilitan los caminos de acceso, hace una descripción independiente de la instalación, por ello se le conoce también como esquema logicial. Antes de especificar el esquema de la BD, el diseñador del sistema de la base debería realizar las siguientes tareas.

- Construir el esquema conceptual
- Identificar los datos requeridos
- Analizar los datos.
- Especificar los grupos de datos de entrada/salida.

Al construir el esquema conceptual el diseñador está en condiciones de identificar las áreas de aplicación representadas por datos y la forma de presentarse ante los usuarios. Teniendo identificados los datos el siguiente paso implica una definición y

clasificación de los mismos. Como resultado de esto obtendríamos lo que comúnmente llamamos "diccionario de datos", el cual permite al diseñador registrar las siguientes propiedades de cada uno de los datos: nombre, sinónimos, descripción de la entidad del mundo real que representa, tipo de datos (entero, real, cadena, etc.), formato, restricciones de acceso y las relaciones con otro datos. Construido el diccionario de datos para lograr la especificación de la interface del usuario final se debe determinar cómo se presentarían los grupos de datos para la entrada y salida del sistema. Este ultimo punto no es sencillo, ya que el diseñador necesita racionalizar los requerimientos del usuario con el fin de construir un conjunto mínimo de requerimientos

4.2.2 El Modelo de Bases de Datos Relacionales.

La mayoría de las nuevas BD de hoy son Bases de Datos Relacionales (BDR). Una BDR almacena información en tablas lógicas formadas por filas y columnas, a estas tablas se les denomina tablas de BD. Por ejemplo las Tablas 4 1-a y 4.1-b, donde a través de la relación del campo en común, que para este caso es "Categoría", se obtiene una ampliación de datos para una consulta.

Clave	Nombre	Categoría	Teléfono	Prom.
185-34-1324	Juan Vázquez	Nuevo Ingreso	789-45-61	7.8
456-21-1567	Jorge Pérez	Nuevo Ingreso	132-45-67	8.9
258-25-4568	Tomás Cruz	Nuevo Ingreso	789-25-36	9.9
147-58-7894	Bart Sánchez	Segundo Año	456-45-78	10
159-45-7895	Eduardo Mata	Tercer Año	366-66-76	5.5

Tabla 4.1-a. Tabla Maestra.

Categoría	Base	Aulas
Nuevo Ingreso	789-45-61	1 a 5
Segundo Año	132-45-67	5 a 10
Tercer Año	789-25-36	10 a 15
Cuarto Año	456-45-78	15 a 20
Quinto Año	366-66-76	20 a 24

Tabla 4.1-b. Tabla Secundaria.

A grandes rasgos *las BDR se usan para almacenar, manipular y accesar datos, siendo estos almacenados en tablas formadas por columnas y filas lógicas,* [13] y por medio de estas debe haber información relacionada en cada campo

[13] Osier, Grobman y Batson, "Aprendiendo DELPHI en 21 días", Pag. 390

4.2.3 Modelos de Bases de Datos.

Como alguna vez se mencionaba hay una diversidad de aplicaciones que utilizan BD, e inclusive en la actualidad una sola aplicación suele usar más de un tipo de BD porque los volúmenes de datos son muy distintos y suelen accesarce por más de una aplicación. Y por otro lado hay aspectos que necesitan relacionarse mediante BD a las que tienen acceso más de una persona. A menudo el poder, complejidad y funcionalidad de un sistema de BD se relaciona con su diseño físico e infraestructura, pero a continuación exploraremos tres de las clases principales de diseño de BD. independientes, de archivos compartidos y cliente/servidor.

4.2.3.1 Independientes.

Estas son las más sencillas de manejar pues se pueden ignorar varios aspectos. Este tipo tiene su BD almacenada en un sistema de archivo local y el manejador de BD para accesarla reside en la misma máquina. Una BD Independiente pura el diseñador no tiene que preocuparse del manejo de concurrencia, la condición en la que dos personas intentan modificar un mismo registro a la vez, pues esto no sucede En general este tipo de BD no se usa para una aplicación que necesita. un poder de cómputo excesivo pues el tiempo de proceso se utilizará en la manipulación de datos y no estará del todo disponible para la aplicación Estas Bases son útiles en el desarrollo de aplicaciones que se distribuyen a muchos usuarios, en donde cada uno mantiene una BD por separado Cabe mencionar que

con la aplicación de BD que incluye Delphi se pueden crear y manipular las BD Paradox y Dbase, y estas pueden actuar a la vez como Independientes o bien de Archivos Compartidos.

4.2.3.2 De Archivos Compartidos.

Este tipo de BD es casi igual a una independiente a excepción de que pueden ser accesadas por diversos usuarios a través de una red. Este hecho permite una mayor accesibilidad ya que la BD puede ser manipulada y accesada por diferentes máquinas. Otra ventaja de este tipo de Bases es que no hay nociones preconcebidas de la Red. Al manejador de BD no le importa si se ejecuta Novell, Banyan, Microsoft NT o cualquier otro Sistema Operativo de Red, debido a que ve la BD simplemente como un archivo. Una situación en la que pudiera no ser conveniente este tipo de BD es cuando se necesita realizar un elevado volumen de cómputo y accesos simultáneos a ésta. La mejor solución para ello es emplear una BD Cliente/Servidor

4.2.3.3 Cliente/Servidor.

Utilizar este tipo, es una buena solución para accesar una BD; para este caso una máquina dedicada o servidor se encarga de realizar el acceso a la BD para un grupo de clientes. En un sistema de archivos compartidos una simple consulta haría que la máquina solicitante se detuviera mientras hace un recorrido de los datos, sin embargo para este tipo

de BD el cliente solicita al servidor que realice la tarea específica. Depende entonces del cliente decidir si desea esperar el resultado o hacer algo más interesante. Por otro lado el servidor se optimiza para manejar las solicitudes de la manera más rápida posible. Aunque las ventajas son muchas, en este tipo de arquitectura también existen desventajas; con frecuencia las soluciones cliente/servidor son mucho más caras que las de archivos compartidos, además el software de cliente/servidor necesita de un protocolo, como TCP/IP, para llevar a cabo una conversación. Aunque esto es a menudo flexible, es también una configuración y una función administrativa adicional.

4.3. Desarrollo del Sistema de Información propuesto.

4.3.1 Organización de la BD.

La BD del SI será del tipo Independiente debido a que se pretende que la aplicación se distribuya a varios usuarios y se utilicen archivos por separado. Lo anterior con la finalidad de que la aplicación funcione adecuadamente con el simple hecho de reemplazar los archivos de BD más actuales, cabe mencionar que el programa se desarrollará protegiendo contra escritura las BD's cuidando así la integridad de las mismas.

El esquema conceptual de la BD indica que el universo del SI lo constituye información proveniente de 4 tipo de señales del PLC Crítico que son: E/S Discretas y Analógicas. Cada una de estas señales tendrá un dato de referencias cruzadas con el cual se pueda asociar para obtener el número de programa y el escalón correspondiente en los diagramas de escalera Por otro lado a cada una le corresponderá una serie de valores con los cuales se asocien a información de archivos provenientes de WWI con la que se podrá saber su utilidad en pantallas (las pantallas son diversas aplicaciones dentro del programa de control WWI).

El esquema lógico para esta BD esta basado en el modelo de datos relacional, en el que todo el peso de información recaería en los datos de "dirección del PLC" (DIR._PLC) y en el "nombre del Item" (ITEMNAME). Los archivos que se listan en la tabla 4.2 son los que finalmente serán utilizados en la aplicación y de los cuales se describirán con mayor detalle en la sección siguiente.

Los datos que se manejan son íntegramente descriptivos de la conexión física de las diferentes señales tratadas, por ejemplo la dirección de la señal al PLC, el No. de cable que se utiliza en la conexión con este, la descripción de la aplicación de la señal, el símbolo que la representa, entre otras

Archivo	Descripción	Procedencia
B-00,R-0.DBF	Entrada Discreta	Modulo A de compresión de Pol-A
B-00,R-1.DBF	Entrada Discreta	Modulo A de compresión de Pol-A
B-00,R-2.DBF	Entrada Discreta	Modulo A de compresión de Pol-A
B-00,R-3.DBF	Entrada Discreta	Modulo A de compresión de Pol-A
B-00,R-4.DBF	Entrada Discreta	Modulo A de compresión de Pol-A
B-00,R-5.DBF	Salida Discreta	Modulo A de compresión de Pol-A
B-00,R-6.DBF	Salida Discreta	Modulo A de compresión de Pol-A
B-00,R-7.DBF	Salida Analógica	Modulo A de compresión de Pol-A
B-01,R-0.DBF	Entrada Analógica	Modulo A de compresión de Pol-A
B-01,R-1.DBF	Entrada Analógica y del HSDE	Modulo A de compresión de Pol-A
DDMOD4.DBF	Discretos DDE	WWI
DRMOD4.DBF	Reales DDE	WWI
XREFRNCE.DBF	Referencias cruzadas	Referencias cruzadas del PLC Crítico
WW_WDWS.NDX	Anida información de pantallas	WWI

Tabla 4.2. Archivos a utilizar en el SI.

4.3.1.1 Diccionario de datos.

Este diccionario está basado en la lista de archivos contenidos en la tabla 4.2, en el que se indica con letras negrillas el nombre o nombres de los archivos, y en seguida el análisis de cada campo con su tipo de dato (α para datos alfanuméricos y # para datos numéricos) en la parte inferior a esta

B-00,R-0.DBF, B-00,R-1.DBF, B-00,R-2.DBF, B-00,R-3.DBF y B-00,R-4.DBF

TAG/POSICION/DESCRIPCIO/CABLE_CAMP/B_L._CAMP
α, 11 #, 11 α, 27 α, 14 α, 11

ID_BARRERRA/OPERACION/B._L _PLC/CABLE_PLC/CON._A_PLC
α, 16 α, 12 α, 10 α, 11 α, 11

DIR._PLC/POSICION_1/IDENTIFICA/ITEMNAME/TIPO_MODUL
α, 11 α, 11 α, 15 α, 11 α, 13

B-00,R-5.DBF y B-00,R-6.DBF

TAG/POSICION/DESCRIPCIO/CABLE_CAMP/B_L._CAMP
α, 14 #, 11 α, 34 α, 15. α, 11

ID_BARRERRA/B._L _FUS/FUSIBLE/CABLE_PLC/CON._A_PLC
α, 12 α, 13 α, 11 α, 11 α, 11

DIR._PLC/POSICION_1/IDENTIFICA/DIR_1/DIR_2/ACCES_NAME
α, 11 α, 11 α, 15 α, 12 α, 11 α, 14

TIPO_MODUL
α, 12

B-00,R-7.DBF

TAG/POSICION/DESCRIPCIO/CABLE_CAMP/B_L _CAMP/ID_BARRERA
α, 11 #, 11 α, 23 α, 14 α, 11 α, 13

B._L._TERMINAL/CABLE_TERM/TERMINAL_1/CON._A_PLC/DIR._PLC
α, 14 α, 10 α, 12 α, 11 α, 11

POSICION_1/IDENTIFICA/ARCHIVO_DE/CALCULOS/ITEMNAME
α, 11 α, 15 α, 11 α, 11 α, 11

TIPO_MODUL
α, 11

B-01,R-0.DBF

TAG/POSICION/DESCRIPCIO/CABLE_CAMP/B_L._CAMP/ID_BARRERA
α, 11 #, 11 α, 25 α, 15 α, 12 α, 11

B.L._HT16/CABLEHT16/CON.HT16/CON.HT16_A/CABLE_A_PL
α, 11 α, 9 α, 11 α, 11 α, 10

CON._PLC/DIR._PLC/POSICION_1/IDENTIFICA/ARCHIVO_DE
α, 11 α, 11 α, 11 α, 11 α, 15

CALCULOS/ITEMNAME/TIPO_MODUL
α, 11 α, 11 α, 11

B-01,R-1.DBF

TAG/POSICION/DESCRIPCIO/HSDE/CABLE_A_AI/CON._AIRPA
α, 12 #, 11 α, 23 α, 8 α, 9 α, 18

CON._SAL._/CABLE_A_PL/CON._A_PLC/DIR._PLC/POSICION_1
α, 11 α, 10 α, 11 α, 11 α, 11

IDENTIFICA/ARCHIVO_DE/CALCULOS/ITEMNAME/TIPO_MODUL
α, 15 α, 11 α, 11 α, 11 α, 14

DRMOD4.DBF

DDEREAL	GROUP	COMMENT	ENGUNITS	INITIALVAL	MINEU
α, 34	α, 11	α, 46	α, 11	#, 13	#, 13

MAXEU	MINRAW	MAXRAW	CONVERSION	DDEACCESSN
#, 12	#, 13	#, 12	α, 13	α, 20

ITEMUSETAG	ITEMNAME	READONLY
α, 20	α, 13	α, 11

DDMOD4.DBF

DDEREAL	GROUP	COMMENT	ENGUNITS	INITIALVAL	MINEU
α, 34	α, 11	α, 46	α, 11	#, 13	#, 13

MAXEU	MINRAW	MAXRAW	CONVERSION	DDEACCESSN
#, 12	#, 13	#, 12	α, 13	α, 20

ITEMUSETAG	ITEMNAME	READONLY
α, 20	α, 13	α, 11

XREFRNCE.DBF

DIR_PLC	TAG	ESCLNES
α, 10	α, 14	α, 66

El archivo con extensión NDX no presenta un diccionario de datos pues es un archivo de texto. Para fines del SI todos los archivos de entradas discretas se agruparon en un archivo llamado CERO.DBF, los de salidas en otro llamado UN.DBF, los de entradas analógicas en otro llamado CUATRO.DBF y los de salidas en DOS.DBF. No todos los campos de los archivos son relevantes, a continuación se analizarán brevemente.

En los archivos de conexión la información importante se encuentra en los siguientes campos.

TAG — o símbolo para indicar una señal dentro del diagrama de escalera.

DESCRIPCIO — descripción breve de la acción a ejecutar en el PLC.

DIR._PLC — dirección que tendrá la señal en el PLC.

POSICION_1 — indica el número de chasis o bastidor y grupo o ranura de conexión.

IDENTIFICA — tipo de PLC que contiene la señal.

ITEMNAME — dato que direcciona información a WWI.

En cuanto a los archivos de salidas discretas los campos más relevantes son.

DIR_1 — primera dirección que tendrá la señal para direccionarse a WWI

DIR_2 — segunda dirección que tendrá la señal para direccionarse a WWI.

ACCES_NAME — indica la procedencia del PLC respecto de la segunda dirección

Los campos más relevantes de las E/S analógicas son

ARCHIVO_DE — contiene la dirección del bloque al cual se destinó la señal.

CALCULOS — información que indica la aplicación de una fórmula o calculo para la señal

ITEMNAME — dato que expresa el resultado (en unidades de ingeniería) de una señal calculada y que servirá de enlace con WWI.

Con respecto a los archivos de WWI los campos más relevantes a emplearse son.

:DDEDISC, :DDEREAL — Datos que representan directamente una señal en pantallas de WWI y por medio de los cuales se desprenden valores discretos y reales respectivamente.

El campo más relevante empleado en el archivo de referencias cruzadas es

ESCLNES — Información que corresponde a los diagramas de escalera, donde se ubican a las señales por módulos. Nomenclatura: [No. de programa]Escalón. Cabe mencionar que el dato del escalón puede tener una diagonal, ello indica signo negativo

Cabe recalcar que los campos más importantes de todos los archivos utilizados para la aplicación son: DIR._PLC e ITEMNAME, porque en ellos se encuentra todo el enlace de información que se requiere. Los archivos del Modulo A de compresión de Pol - A son producto de información en tablas por lo que se tienen que convertir a BD.

4.3.2 Desarrollo del Software.

Como se mencionó anteriormente el SI se desarrollo en Delphi y a continuación se presentará el listado completo. Primero se presenta el listado principal y posteriormente el de cada una de las pantallas:

"Programa Prj1"

```
program Prj1;

uses
  Forms,
  Pr1 in 'PR1.PAS' {Form1},
  Pasw in 'PASW.PAS' {PasswordDlg},
  Mant in 'MANT.PAS' {Form3};

{$R *.RES}

begin
  Application.CreateForm(TForm1, Form1);
  Application.CreateForm(TForm3, Form3);
  Application.CreateForm(TPasswordDlg, PasswordDlg);
  Application.Run;
end.
```

"Presentación del SI"

```
unit Pr1;
```

```
interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, ExtCtrls, StdCtrls, Pasw,

type
  TForm1 = class(TForm)
    Image1: TImage,
    Button1: TButton;
    Label1: TLabel,
    Label2: TLabel;
    ProgramIcon: TImage,
    Label3: TLabel;
    Image3: TImage;
    Label5: TLabel;
    Image6: TImage;
    Label8: TLabel;
    Image8: TImage;
    Image11: TImage;
    Timer1: TTimer;
    Panel1: TPanel,
    procedure Button1Click(Sender. TObject);
    procedure Image2MouseMove(Sender. TObject, Shift  TShiftState; X,
      Y: Integer);
    procedure Image8MouseMove(Sender: TObject; Shift  TShiftState; X,
      Y: Integer);
    procedure Image11MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Label8Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject),
    procedure FormCreate(Sender: TObject);
    procedure FormDestroy(Sender. TObject),
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1,

implementation

{$R *.DFM}

procedure TForm1 Button1Click(Sender: TObject),
{ESTE PROCEDIMIENTO DIRECCIONA A LA FORMA DE PASSWORD Y POSTERIOR-
MENTE PERMITE UNA SALIDA DIRECTA DEL PROGRAMA }
begin
    Button1 Visible  = False,
```

```
    PasswordDlg  = TPasswordDlg.Create(Self);
    PasswordDlg.ShowModal;
    Label3.Visible := True;
    if MessageDlg('Deseas Salir del Programa.',mtInformation,
    [mbYes, mbNo], 0) = mrYes then
    begin
       Release;
       Application.Terminate
    end else
       Label3.Visible := False
end;

procedure TForm1.Image2MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
{ESTE SEGMENTO PERMITE EL INTERCAMBIO TOTAL DE IMAGENES Y EL CAM-
BIO DE COLOR DE TODOS LOS LETREROS, PARA PONER LA PANTALLA EN
CONDICIONES INICIALES.}
begin
    Image8.Visible := False;
    Image11.Visible := False;
    Image3.Visible := True;
    Image6.Visible := True;
    Label5.Font.Color := clYellow,
    Label8.Font.Color := clYellow
end;

procedure TForm1.Image8MouseMove(Sender: TObject; Shift: TShiftState, X,
  Y: Integer);
{ESTE SEGMENTO PERMITE EL INTERCAMBIO DE IMAGENES Y EL CAMBIO DE
COLOR EN EL LETRERO CORRESPONDIENTE.}
begin
    Image3 Visible := False;
    Image8.Visible := True;
    Label5.Font.Color := clRed
end;

procedure TForm1.Image11MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
{ESTE SEGMENTO PERMITE EL INTERCAMBIO DE IMAGENES Y EL CAMBIO DE
COLOR EN EL LETRERO CORRESPONDIENTE }
begin
    Image6 Visible := False;
    Image11.Visible := True,
    Label8 Font.Color := clRed
end;

procedure TForm1.Label8Click(Sender: TObject);
{ESTE SEGMENTO HABILITA EL 3ER. LETRERO Y PREPARA LA SALIDA DEL
PROGRAMA.}
begin
    Label3.Visible  = True,
```

```
    if MessageDlg('Deseas Salir del Programa ',mtInformation,
    [mbYes, mbNo], 0) = mrYes then
    begin
       Release,
       Close
    end else
    begin
       Label3.Visible .= False
    end;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
{ESTE PROCEDIMIENTO HABILITA AL CAPTION DEL PANEL1 CON LA HORA DE
LA COMPUTADORA EN USO.}
begin
   Panel1.Caption  = TimeToStr(Time)
end;

procedure TForm1.FormCreate(Sender: TObject);
{ESTE PROCEDIMIENTO INICIALIZA LA APLICACION.}
begin
      MessageDlg('Inicio de programa ', mtInformation,[mbOk], 0),
end,

procedure TForm1 FormDestroy(Sender: TObject);
{ESTE SEGMENTO CIERRA LA APLICACION.}
begin
      MessageDlg('Fin de programa.', mtInformation,[mbOk], 0);
end;

end.
```

"Password's"

```
unit Pasw,

interface

uses WinTypes, WinProcs, Classes, Graphics, Forms, Controls, StdCtrls,
 Buttons, Messages, SysUtils, Dialogs, Oper, Mant, Pants;

type
 TPasswordDlg = class(TForm)
  Label1: TLabel;
  Password: TEdit;
  OKBtn: TBitBtn;
  CancelBtn: TBitBtn,
  procedure Button1Click(Sender  TObject);
  procedure OKBtnClick(Sender. TObject);
```

```
  procedure FormClose(Sender  TObject; var Action· TCloseAction),
 private
  { Private declarations }
 public
  { Public declarations }
 end,

var
 PasswordDlg: TPasswordDlg;

implementation

{$R * DFM}

procedure TPasswordDlg.Button1Click(Sender: TObject),
{ESTA FRACCION DE CODIGO EJECUTA LA FORMA DE "PASSWORD"}
begin
   passworddlg.showmodal
end;

procedure TPasswordDlg.OKBtnClick(Sender: TObject),
{ESTE PROCEDIMIENTO CHECA LOS CARACTERES QUE SE INTRODUCEN COMO
PASSWORD'S Y LOS COMPARA CON EL QUE ESTA DEFINIDO.
SI ENCUENTRA UN PASSWORD IGUAL, SE GENERA EL MENSAJE CORRESPONDIEN-
TE Y SE EJECUTA LA FORMA, SI NO SE GENERA TAMBIEN UN MENSAJE QUE
INDICA DICHA ACCION.}
var
  f : string;
begin
   f := Password.Text;
   f := AnsiUpperCase(f);
   if f = 'DOS' {and ((Sender as TLabel).Tag = 1)} then
   begin
      MessageDlg('Acceso a Mantenimiento ', mtInformation,
      [mbOk], 0);
      Password.Text := '';
      Form3 := TForm3.Create(Self);
      Form3.ShowModal;
   end else
   begin
      MessageDlg('Password Incorrecto      !', mtError,
      [mbOk], 0);
      Password.Text := '';
   end
end;

procedure TPasswordDlg.FormClose(Sender  TObject,
 var Action  TCloseAction),
begin
   Release;
   Close
```

```
end;

end.
```

"Mantenimiento"

```
unit Mant;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  StdCtrls, Forms, DBCtrls, DB, DBTables, Mask, ExtCtrls, Grids, DBGrids,
  Buttons, Dialogs;

type
  TForm3 = class(TForm)
    ScrollBox: TScrollBox;
    Label1: TLabel;
    DataSource1: TDataSource,
    Panel2: TPanel;
    Query1: TQuery;
    Query2: TQuery;
    DataSource2: TDataSource;
    Panel1: TPanel;
    BitBtn1: TBitBtn;
    Bevel2: TBevel;
    Image1: TImage;
    Bevel1: TBevel;
    Bevel3: TBevel;
    DBGrid1: TDBGrid;
    DBGrid2: TDBGrid;
    Query3: TQuery;
    DataSource3: TDataSource;
    Bevel4: TBevel;
    DBGrid3. TDBGrid;
    GroupBox1: TGroupBox;
    Label6: TLabel;
    Label3: TLabel;
    ESCAL. TDBEdit,
    Label2: TLabel;
    DIRPLC: TDBEdit;
    DIRWWI: TDBEdit;
    Label7: TLabel,
    Label4: TLabel;
    Label8: TLabel;
    DBEdit5: TDBEdit;
    DESCRIP  TDBEdit;
    TAG: TComboBox;
```

```
    RadioGroup1: TRadioGroup;
    Bevel5: TBevel;
    Panel3: TPanel;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Label9: TLabel;
    PROCED2: TDBEdit;
    PROCE: TDBEdit,
    procedure BitBtn1Click(Sender  TObject);
    procedure TAGChange(Sender: TObject),
    procedure RadioGroup1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender. TObject);
    procedure Button3Click(Sender. TObject);
    procedure q1(var arch_o  string);
    procedure q2(var camp: string);
    procedure q3(var camp: string);
    procedure llena_combo(var arch_o: string);
  private
    { private declarations }
  public
    { public declarations }
  end;

var
  Form3: TForm3;
  dir1, dir2, dir3: boolean;

implementation

{$R *.DFM}

procedure TForm3.BitBtn1Click(Sender: TObject);
{PREPARA Y EJECUTAA EL CIERRE DE LA APLICACION. (NIVEL 2.)ASI CO-
MO EL CIERRE DE LAS CONSULTAS }
begin
    Image1.Visible := False;
    MessageDlg('Nivel Cerrado.', mtInformation, [mbOk], 0);
    Query1.Close,
    Query2.Close;
    Query3.Close;
    Release,
    Close
end;

procedure TForm3 TAGChange(Sender: TObject);
var
  arch_o, camp. string,
```

```
begin
  Query1 SQL Clear,Query2.SQL Clear,Query3.SQL Clear;
  if RadioGroup1 ItemIndex = 0 then
  begin
    arch_o  = 'CERO', q1(arch_o);
    camp := Query1.Fields[10].AsString; q2(camp);
    camp .= Query1 Fields[13].AsString; q3(camp);
  end else
  if RadioGroup1 ItemIndex = 1 then
  begin
    if dir1 or dir2 then
    begin
      arch_o := 'UN'; q1(arch_o);
      camp := Query1 Fields[10] AsString, q2(camp),
      if dir1 then
      begin
        camp := Query1 Fields[13] AsString, q3(camp);
      end else
      begin
        camp .= Query1 Fields[14] AsString, q3(camp);
      end;
    end else
    begin
      MessageDlg('Debes seleccionar un botón .'+
      '(DIR_1 o DIR_2)', mtInformation, [mbOk], 0);
      Panel1.Caption := 'Selecciona DIR_1 o DIR_2';
      Button1.Enabled := True; Button2.Enabled := True;
    end,
  end else
  if RadioGroup1 ItemIndex = 2 then
  begin
    arch_o  = 'CUATRO'; q1(arch_o);
    if dir1 or dir2 or dir3 then
    begin
      if dir1 then
      begin
        camp := Query1 Fields[12].AsString; q2(camp), q3(camp);
      end else
      if dir2 then
      begin
        camp := Query1 Fields[13].AsString, q2(camp), q3(camp),
      end else
      begin
        camp := Query1 Fields[14].AsString, q2(camp); q3(camp),
      end,
    end else
    begin
      MessageDlg('Debes seleccionar un botón '+
      '(DIR_1, DIR_2 o DIR_3)', mtInformation, [mbOk], 0),
      Panel1 Caption  = 'Selecciona DIR_1, DIR_2 o DIR_3';
      Button1.Enabled  = True, Button2 Enabled  = True,
```

```
        Button3.Enabled := True,
      end
    end else
    if RadioGroup1.ItemIndex = 3 then
    begin
      arch_o  = 'DOS'; q1(arch_o);
      if dir1 or dir2 or dir3 then
      begin
        if dir1 then
        begin
          camp .= Query1 Fields[13].AsString; q2(camp), q3(camp);
        end else
        if dir2 then
        begin
          camp  = Query1.Fields[14].AsString, q2(camp), q3(camp),
        end else
        begin
          camp := Query1 Fields[15] AsString; q2(camp); q3(camp);
        end;
      end else
      begin
        MessageDlg('Debes seleccionar un botón '+
        '(DIR_1, DIR_2 o DIR_3)', mtInformation, [mbOk], 0);
        Panel1 Caption := 'Selecciona DIR_1, DIR_2 o DIR_3';
        Button1.Enabled := True; Button2.Enabled := True,
        Button3.Enabled := True;
      end,
    end;
end;

procedure TForm3 RadioGroup1Click(Sender· TObject),
var
  a, b, c, d, resp: char,
  arch_o· string;
begin
    dir1 := False; dir2 .= False, dir3 := False;
    TAG.Items.Clear,
    PROCED2.DataField := '', PROCED2.Visible .= False,
    Label9.Visible := False;
    Form3.FormCreate(Sender);
    Panel1.Caption .= 'Selecciona un TAG',
    if RadioGroup1 ItemIndex = 0 then resp := 'a' else
      if RadioGroup1.ItemIndex = 1 then resp .= 'b' else
        if RadioGroup1.ItemIndex = 2 then resp := 'c' else
          if RadioGroup1.ItemIndex = 3 then resp .= 'd';
    case resp of
    'a' · begin
          arch_o  = 'CERO', llena_combo(arch_o),
        end;
    'b' · begin
          PROCED2.DataField  = 'ACCES_NAME';PROCED2 Visible  = True,
```

```
        Label9.Visible := True;
        arch_o := 'UN', llena_combo(arch_o);
      end,
  'c' : begin
        arch_o := 'CUATRO'; llena_combo(arch_o),
      end;
  'd' : begin
        arch_o := 'DOS'; llena_combo(arch_o),
      end,
  end;
end;

procedure TForm3.FormCreate(Sender: TObject),
begin
  Query1.SQL.Clear, Query2.SQL.Clear, Query3 SQL Clear,
  Button1.Enabled .= False, Button2 Enabled  = False;
  Button3.Enabled := False,
  Panel1.Caption := 'Selecciona un tipo de Señal';
end;

procedure TForm3.Button1Click(Sender: TObject),
begin
  dir1 := True; dir2 := False; dir3  = False,
  if (RadioGroup1.ItemIndex = 2) or (RadioGroup1 ItemIndex = 3)
  then Panel1.Caption := 'Señal en archivo de datos' else
  Panel1.Caption := 'Primer procedencia';
  TAGChange(Sender);
end;

procedure TForm3.Button2Click(Sender: TObject),
begin
  dir2 := True, dir1 := False; dir3 := False,
  if (RadioGroup1.ItemIndex = 2) or (RadioGroup1.ItemIndex = 3)
  then Panel1.Caption := 'Señal producto de un calculo' else
  Panel1 Caption  = 'Segunda procedencia';
  TAGChange(Sender);
end;

procedure TForm3.Button3Click(Sender· TObject);
begin
  dir3  = True, dir2 := False; dir1 .= False,
  Panel1.Caption := 'Señal en unidades de ingeniería';
  TAGChange(Sender);
end;

procedure TForm3.q1(var arch_o. string);
var
  simb1: string,
begin
  simb1 := TAG Items[TAG.ItemIndex],    {TAG}
  Query1 SQL.Add('SELECT * FROM '+arch_o);
```

```
    Query1.SQL.Add('WHERE TAG = '''+simb1+''');
    Query1.Open;
end,

procedure TForm3.q2(var camp  string),
begin
    Query2.SQL.Add('SELECT * FROM XREFRNCE');
    Query2.SQL Add('WHERE DIR_PLC = '''+camp+''');
    Query2.Open;
end,

procedure TForm3.q3(var camp· string),
begin
    Query3.SQL.Add('SELECT * FROM DRMOD4'),
    Query3.SQL.Add('WHERE ITEMNAME = '''+camp+''');
    Query3.Open;
end;

procedure TForm3.llena_combo(var arch_o: string);
begin
    Query1.SQL.Add('SELECT * FROM '+arch_o);
    Query1.Open;
    while not Query1.EOF do
    begin
        TAG.Items.Add(Query1.Fields[0] AsString),
        Query1.Next;
    end;
    Query1.First;
end;

end.
```

4.3.3 Pantallas.

La primer pantalla presentada en la figura 4.2 es la del PP, en la que el control fue la idea fundamental, ya que a través de esta se dará acceso a la siguiente (Mantenimiento) y se proporcionará la salida del SI, la legibilidad y fluidez fue otro punto importante, pues presenta la idea formal y sencilla, sin intervenciones exageradas de interferencia visual.

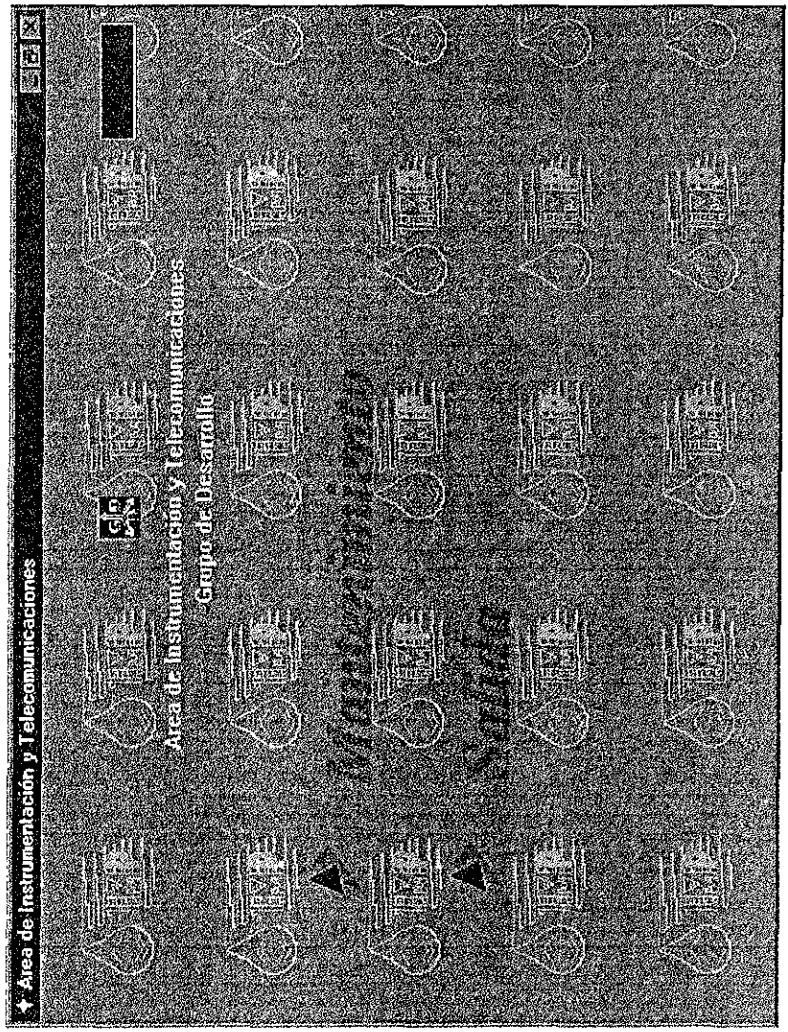


Figura 4.2 Pantalla del PP.

La figura 4 3 muestra la pantalla de acceso al SI, la característica que la define principalmente es su estructura, ello por la distribución de los elementos que la componen denotándose un orden y balance.

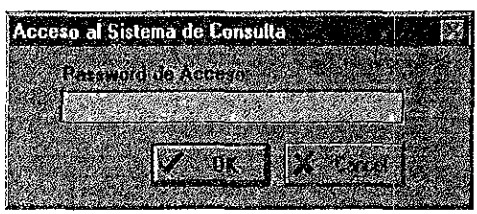


Figura 4.3 Pantalla de acceso

No fue nada fácil construir la pantalla de mantenimiento, pues la cantidad de elementos utilizados (aproximadamente 53 objetos) requerían de una compleja coordinación, combinada con una diversidad de información. La figura 4.4 ilustra la pantalla de mantenimiento, en la que el enfoque y énfasis son la mejor definición para ella. Otras de las características que sobresalen es la consistencia ya que se expresa en la interacción que presentan los objetos con los usuarios, la retroalimentación expresada en la generación estratégica de mensajes, la estética que hace una presentación visual agradable, etc.

En general se trató de cubrir por completo los principios de diseño centrados en el usuario y los diseños de pantallas(Para mayor detalle ver CAP II.), en donde el control fue una de las características principales pues se requiere poner al usuario por delante.

4.3.4 Almacenamiento y respaldos.

Toda la aplicación se almacenó en un sólo archivo ejecutable, aunque como el programa utilizó BD's requiere de otros archivos especiales para correr en cualquier PC, (Para mayor detalle ver Cap. III) los archivos requeridos para el SI se listan en la tabla 4.3. Cabe mencionar que esta lista es independiente de los archivos de la tabla 4 2, ya que en ella sólo se listaban archivos de BD

El árbol de todos los archivos utilizados en la aplicación debe ser el siguiente, puesto que como se manejaran archivos que contienen información de una configuración definida previamente, es necesario que por el momento se siga el mismo modelo, aunque cabe mencionar que a la posteridad se puede cambiar la configuración inicial por alguna otra con el uso de la aplicación proporcionada en el archivo BDECFG.EXE:

C: \ PROGRAMA \ BASES \ {TODAS LAS BASES DE DATOS .DBF}

C: \ D \ IDAPI \ {TODOS LOS ARCHIVOS .DLL, .CFG, Y DBECFG}

Archivo	Tamaño	Descripción
PRJ1 EXE	739 KB	Ejecutable del SI
IDAPI01.DLL	386 KB	Habilita la máquina de BD de Borland
IDR10009.DLL	21 KB	Habilita la máquina de BD de Borland
ILD01.DLL	47 KB	Habilita la máquina de BD de Borland
IDAPI CFG	8 KB	Configura la máquina de BD de Borland para la PC
IDQRY01 DLL	396 KB	Habilita al manejador de consultas de SQL
IDBAT01.DLL	103 KB	Habilita al manejador de consultas de SQL y procesamiento por lotes
IDPDX01 DLL	243 KB	Habilita al manejador de BD para PARADOX (se aconseja cargarlo para aplicaciones SQL)
IDDBAS01.DLL	313 KB	Habilita al manejador de BD para DBase
BDECFG.EXE	352 KB	Configura la máquina de BD de Borland (utilidad de ayuda)
BDECFG.HLP	73 KB	Redefine los aliases establecidos (opcional)

Tabla 4.3. Archivos requeridos por el programa (no son los de BD).

El respaldo del conjunto de archivos presentados en la tabla 4.3 asciende alrededor de los 2.7 MB, ello considerando la integridad de los mismos, es decir sin comprimirlos, puesto que en la actualidad existen aplicaciones que podrían hacerlo hasta en un 40%, 50% o más, con lo que obtendríamos un solo archivo de 1.3 MB aproximadamente, y esto nos hace pensar en un diskette de 3 ½ " de 2HD. En este desarrollo no se tomaron en cuenta las DB's puesto que el programa está hecho para usar BD independientes, por lo que el tamaño del conjunto de archivos sólo es un estimado, y este podría ascender como base a no más de 300 KB sin comprimir. Concluyendo esta sección, para almacenar al SI

se requieren de aproximadamente 3 MB de espacio en disco duro y para el respaldo de dicha información comprimida se requieren de 2 diskettes de 3 ½ " de 2HD (aproximadamente 1.5 MB libres).

CAPÍTULO 5

IMPLANTACIÓN

5.1. Pruebas.

En las pruebas principalmente lo que se hace es verificar toda la lógica del programa en conjunto con la funcionalidad esperada. En este, que es uno de los últimos pasos para generar un buen SI la participación del usuario es fundamental, ya se había mencionado en capítulos anteriores la importancia que juega este factor para el desarrollo de los mismos. Por lo general en esta etapa los usuarios deben generar datos de prueba, es decir datos con errores y que cubran tantas condiciones diferentes como sea posible, con ello se obtiene la revisión de errores adecuada. Al probar un programa en forma individual sólo se asegura la funcionalidad de la ruta más utilizada, aunque los programadores faciliten las pruebas al codificar claramente, no existe un programa completamente depurado, por lo que el propósito de estas es reducir la frecuencia y severidad para detectar los errores

5.2. Programa de Instalación.

En 1981 IBM presenta su IMB PC y el Sistema Operativo PC-DOS, después de esto surgieron otros productos pero a consecuencia de éste; en ese entonces instalar un programa no representaba más que copiar archivos de un disco a un subdirectorio y correr el archivo ejecutable o en lote. Algunos otros programas contenían programas de instalación que simplemente lo que hacían era, copiar por usted los archivos en un subdirectorio; al correr el tiempo los usuarios podían seleccionar opciones que se agregaban a archivos de configuración (.dat, .cfg y otros por el estilo). No existía una norma para configurar e instalar programas, cada programador tenía su propia forma de registrar las configuraciones del usuario y otra información necesaria para el programa. Cuando usted requería limpiar el disco duro con frecuencia tenía problemas para determinar que archivos eran necesarios y además los archivos AUTOEXEC.BAT y CONFIG.SYS contenían líneas aplicables a programas en particular.

Windows por su parte trae consigo los archivos .ini y los programas SETUP.EXE, incluso proporcionó funciones API (Interfaz de Programación de Aplicaciones *[Aplication Programing Interface]*), las cuales se podían emplear para crear, leer y escribir archivos .ini con facilidad. Este desarrollo condujo a seguir ciertas normas en las convenciones de nombres, métodos de acceso e incluso en la ubicación de archivos; cabe señalar que aunque esto aún no es suficiente los archivos .ini también tienen sus propios problemas, ya que algunos programadores utilizan los archivos WIN.INI y SYSTEM.INI, mientras que

otros usan archivos .ini específicos para su propia aplicación y hay quienes usan ambos. Todos los archivos .ini y aplicaciones Windows hacen aún más compleja la depuración de un disco duro, en DOS usted tiene que eliminar el directorio de la aplicación y su contenido, y tal vez algunas líneas en los archivos AUTOEXEC.BAT y CONFIG.SYS. Por su parte en Windows si usted borra los archivos y directorios de una aplicación, a menudo quedarán residuos en los archivos WIN.INI y SYSTEM.INI, sobre el cual se instalo la aplicación. Esto causará desorden, confusión y efectos impredecibles en Windows, por ello ahora hay aplicaciones que vigilan la instalación y le permiten desinstalar o revocar la instalación de los programas aunque no siempre se pueda confiar en que funcionen, puesto que muchos programas Windows comparten DLL's y otros recursos, así tendríamos que, para que funcionen adecuadamente los programas de desinstalación deben instalarse siempre antes que otras aplicaciones. Debido a que prácticamente no pueden tener una base de datos completa de cada programa Windows, no podrían desinstalarse correctamente cada programa a menos que monitorearan su instalación.

Windows NT y Windows 95 contienen una BD que almacena información sobre la configuración de Hardware del sistema, de Windows y de aplicaciones Windows que se llama Registry. *Registry es una BD jerárquica utilizada por Windows para almacenar configuraciones de Hardware y Software del Sistema, así como configuraciones de*

programa y de usuario. [14] Para aplicaciones sobre Windows 95, Microsoft requiere que los desarrolladores utilicen la BD Registry, que observen una serie de normas y que proporcionen programas completos de instalación y desinstalación. Esto es bueno porque a través de ello se tendría un método ordenado y fácil para instalar y desinstalar el Software.

5.2.1 Características principales.

Estos son requerimientos técnicos de Windows 95, que deben seguir los desarrolladores al crear Software y Hardware. Cabe señalar que estos se actualizan constantemente así que tendríamos que estar en contacto con ellos periódicamente. (*Programmer's Guide to Microsoft Windows 95 y Windows Interface Guidelines for Software Design.* Títulos de Microsoft Press.) Es importante mencionar que cuando se escriba una aplicación que cubra con los requisitos se tendría que hacer una solicitud a Microsoft antes de poder exhibir el logotipo. No es necesario hacer aplicaciones que cumplan con todos los requisitos del logotipo Windows 95 Si desarrolla Freeware (Software gratuito), Shareware (Software compartido o de remuneración voluntaria por su uso) o Software personal, podría no valer la pena el esfuerzo, pero si decide más tarde comercializar su aplicación no le costaría mucho trabajo conseguir la autorización para ostentar el logotipo

[14] Osier, Grobman y Batson, "Aprendiendo DELPHI en 21 días", Pag 631

Microsoft clasifica las aplicaciones en cuatro categorías principalmente:

- Aplicaciones basadas en archivos

- Aplicaciones no basadas en archivos que se ejecutan exclusivamente en modo total, es decir sin ventanas.

- *Utilerías, que incluyen* las que sirven para el manejo de archivos y discos, y a los programas de detección de virus y limpieza o eliminación de los mismos

- Herramientas de desarrollo.

Los requisitos que a continuación se mencionan son aplicables a todo diseño

- La aplicación debe utilizar la API de Win32

- La aplicación debe apegarse, para la interfaz del usuario, a la apariencia e impresión sensible de Windows 95.

- La aplicación debe ser probada y ejecutada con éxito tanto en Windows 95 como en Windows NT y debe ser ajustada según sea necesario para compensar las diferencias entre las características de ambos.

- La aplicación debe poder manejar nombres de archivos largos y mostrarlos de manera adecuada en cuadros de diálogo, controles, etc..

- La aplicación debe soportar la característica de Plug and Play. (conéctese y úsese)

Hay otra cantidad de requisitos relacionados con el uso de OLE (*Object Linking and Embedding,* vinculación e incrustación de objetos), MAPI (*Messaging Application Programming Interface,* interfaz de programación de aplicaciones mensajeras), y UNC (*Universal Naming convention,* convención universal de nomenclatura) Por otro lado la aplicación debe proporcionar como ya se había mencionado, programas de instalación y desinstalación, además de usar la BD Registry en lugar de los archivos WIN.INI y SYSTEM.INI.

Para escribir un instalador y desinstalador sencillo que cumpla con los requisitos de Windows 95, las recomendaciones que hace Microsoft son:

- Proporcionar una GUI (*Graphical User Interface*, interfaz gráfica de usuario) estándar de Windows.

- Proporcionar la capacidad de desinstalar con seguridad la aplicación y los archivos relacionados.

- Verificar la configuración de Hardware y Software.

- Verificar el espacio disponible en disco duro.

- Copiar la aplicación y los archivos requeridos a disco duro, creando subdirectorios según se requiera.

- Modificar Registry y los archivos que sea necesario para hacer que la aplicación funcione correctamente.

- Proporcionar opciones de instalación a los usuarios, como la instalación típica, la compacta, la personalizada, etc

- Proporcionar configuraciones comunes por omisión.

- *Solicitar un disco sólo una vez durante la instalación.*

- Solicitar oportunamente por el indicador en pantalla y por sonido el siguiente disco.

- Exhibir un indicador de avance que muestre al usuario qué tanto se ha realizado del proceso de instalación.

- Ofrecer al usuario la capacidad de cancelar la instalación antes de terminar.

- Llevar un registro de todos los archivos instalados y de los cambios en el sistema del usuario.

- Efectuar la limpieza en el caso de cancelar la instalación o de que se ejecute el programa de desinstalación

- Eliminar cualquiera de los archivos, entradas Registry, atajos de acceso y otros cambios efectuados durante la instalación

Estos son los requisitos más importantes, pero recuerde que si se desea realizar alguna aplicación apegada a la actualidad se tendría que verificar con Microsoft respecto a la última información Por lo anterior se aprecia que no es nada fácil construir un instalador y desinstalador que satisfaga los requisitos de logotipo de Windows 95, pues también se tendrían que tomar en cuenta la compresión de archivos, el almacenamiento de valores como la trayectoria en registry para hacer útiles las entradas, la creación de una carpeta o grupo de estas con accesos rápidos o iconos, etc , en fin escribir un instalador competente es un tema apto para la realización de un sólo proyecto.

5.3. *Depuración.*

Generalmente cuando realizamos un programa en ensamblador o en cualquier lenguaje normalmente cometemos errores Estos errores se podrían clasificar principalmente en dos categorías que son errores de sintaxis y errores de programación (o lógico) Los errores de sintaxis generalmente son notificados al compilar o ensamblar el código del programa, en caso de que este fuera muy grave, el compilador o ensamblador interrumpe el proceso y no permite que se genere el archivo ejecutable Para corregir este

tipo de error se debe entrar al editor del programa y arreglar el código en la línea indicada. Este tipo de errores son los más fáciles de corregir Y por otro lado un programa puede ser compilado o ensamblado correctamente sin mostrar errores, y aún así no funcionar como se pretendía, a lo sucedido se le conoce generalmente como error de programación (o error en la lógica de programación), siendo estos los más difíciles de localizar que los de sintaxis. Este último tipo de error se genera cuando el programador hace una suposición incorrecta en la implantación del código, como por ejemplo al no extender el rango del bucle de control para incluir los límites adecuados, la verificación de una variable equivocada en un proceso de toma de decisión, entre otros

5.3.1 Complemento para el "Trabajo Perfecto".

Por lo expresado en el punto anterior la depuración la defino como.

"El proceso por el cual las diversas metodologías de programación se sirven para corregir cualquier tipo de error dentro de los programas".

Sobre todo de los errores lógicos, ya que es el error más difícil de detectar y por consiguiente de corregir; así todo este proceso hace de cualquier programa el tan deseado "trabajo perfecto"

5.4. Puesta en marcha.

La necesidad de una puesta en marcha antes de instalar los SI es necesaria debido a que a través de esta se señalan ciertos aspectos que no se habieran detectado por ejemplo, que el diseño original del SI estuviera defectuoso y tenga tantos errores que nadie confíe en lo que el sistema produce, otro aspecto es la interrelación de este con el usuario ya que si no es la adecuada causaría una descontinuación en su uso real, tomando en cuenta que la operación de un sistema requiere aspectos de largo plazo después de que este ha sido diseñado e instalado, etc.. Se debe considerar la puesta en marcha como parte del proceso de diseño, en general a los últimos pasos, pues el desarrollo de un nuevo SI es para cambiar los procedimientos existentes del proceso anterior de información Algunos autores piensan que la puesta en marcha destaca el aspecto de largo plazo, es decir *considerándola como parte del proceso que comienza* con la primera idea del sistema y termina cuando este ha sido integrado con éxito.

Son muchos los factores que influyen para una puesta en marcha exitosa pero algunos de los más importantes son los que den por resultado altos niveles de uso, como por ejemplo las actitudes favorables por parte de los usuarios, pues dan lugar a las acciones y si son positivas aumentaran los niveles de utilización y satisfacción del SI, otro aspecto es la calidad técnica pues afecta directamente las actitudes del usuario y simplifica su uso Una estrategia para prevenir fallas en los SI junto con la puesta en marcha es

reconocer al diseño como una actividad de cambio planeado y para obtener el éxito requieren cambios de comportamiento por parte de los usuarios.

5.5. Implantación del Sistema de Información propuesto.

5.5.1 Pruebas de la puesta en marcha.

El SI no se ha instalado en la plataforma de compresión Pol-A, porque se encuentra en la etapa de pruebas para la puesta en marcha y de las cuales algunas de ellas son las siguientes:

- Correcta lectura de información con BD más grandes
- Intercambio de archivos para observar su funcionamiento.
- Completar la transformación de información de Excel para las BD's, puesto que los archivos de las señales obtienen su información mediante tablas.

- La correcta obtención de información mediante pruebas manuales; es decir monitoreando los datos obtenidos del SI con la información de los manuales y con los datos de pantallas de WWI.

- Poner el SI ante la mayor variedad de usuarios en el GDD

- Velocidad de proceso ante diversas computadoras.

- Hacer pruebas en plataformas.

CAPÍTULO 6

MANTENIMIENTO

6.1. Elementos principales en la documentación de un programa.

Todo SI, programa o aplicación se debe documentar, puesto que toda la información que se contenga en esta sección servirá de guía para la utilidad que se pretenda proporcionar y ésta será aplicable tanto para los programadores encargados de dar mantenimiento, como para todos los usuarios que lo utilicen.

6.1.1 Seudocódigo y Algoritmo.

Estos elementos son de gran utilidad para el mantenimiento de cualquier SI, pues en ambos se encuentra la descripción paso a paso de la funcionalidad de la aplicación. Son sencillos de leer y comprender porque se estructuran con lenguaje natural, es decir lejos de tecnisismos y palabras rebuscadas Aunque ambos son muy semejantes y su objetivo sea el

mismo, existen diferencias muy marcadas, por ejemplo en un seudocódigo si existe una sintaxis definida ya que se construyen con estructuras de control (*IF - THEN - ELSE, DO WHILE*, ETC. [SI - ENTONES - SI NO, HAZ MIENTRAS]) mientras que los algoritmos son enunciado completamente redactados en español (para nuestro caso). Es importante mencionar que estos dos elementos son muy útiles cuando se aplican a módulos o fracciones del programa, pero cuando la complejidad aumenta es recomendable utilizar alguna otra alternativa como por ejemplo un diagrama de flujo.

6.1.2 Diagramas de flujo.

Los diagramas de flujo son una representación visual de todas las operaciones que se realizan en una aplicación, es el medio más común y utilizado por los programadores, habrá personas que tal vez no sepan como construir un seudocódigo o algoritmo pero un 99% reconocen la simbología de los diagramas de flujo. En la figura 6.1 se muestra la simbología ANSI más común de dichos diagramas

6.1.3 Manuales de usuario.

Los manuales de usuario deben ser siempre el ultimo documento de las ultimas versiones del SI, ya que en ellos se debe contener toda la información que resuma tanto el funcionamiento del mismo como los requerimientos del Software haciendo de esta referencia la mejor ayuda para el usuario después de que se ha iniciado la operación de

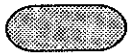
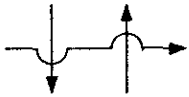
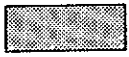
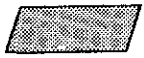
SIMBOLO	NOMBRE	FUNCIÓN
	Terminal	Representa el principio o el fin de un programa
	Líneas de flujo	Representa el flujo de la lógica. Las prominencias de la flecha horizontal indican que éste pasa sobre las otras sin conectarse con las líneas verticales.
	Proceso	Representa una acción a ejecutarse.
	Entradas/salidas	Representa la entrada o la salida de información.
	Decisión	Representa una decisión a la cual tendrá como resultado, dos caminos a elegir
	Conector en una misma página	Representa una interrupción en el flujo de datos e indica la continuación en otro punto dentro de la misma página
	Conector de fin de página	Representa la continuación del flujo de datos en otra página, este se usa cuando el diagrama no cabe en una misma página

Figura 6.1. Símbolos ANSI más comunes.

este. Como los manuales por lo regular son para consultarse al principio de la aplicación y cuando se tengan dudas o problemas, se requiere que esta información sea de buena

calidad y así los usuarios podrán contestarse sus propias preguntas sin necesidad de llamar al departamento de servicios de información reduciendo así los conflictos potenciales.

Se recomienda que estos manuales se anexen en carpetas de argollas, ello con la finalidad de facilitar la actualización según la evolución del SI a lo largo del tiempo. Los índices detallados permiten que la documentación sea fácil de usar en los momentos difíciles (cuantas veces no ha salido de apuros consultando un manual que contiene un gran contexto informativo en su índice) Un análisis completo de la entrada, salida y lógica del proceso es parte fundamental de estos manuales pues en ella se explica prácticamente todo el funcionamiento de las aplicaciones. Por ultimo uno de los componentes más *importantes y fundamentales* que no se deben omitir en estos documentos son las listas de condiciones de error y acciones a tomar para cada uno de los casos tratados, ya que muchas veces estas *listas son de gran utilidad y logran ahorrar mucho tiempo y dinero* a la vez

6.2. Mantenimiento del SI propuesto.

Como el SI desarrollado durante el transcurso de todo este documento aunque es una aplicación terminada, no es el producto final, por ello sólo se presenta la información más relevante. En la figura 6.2 se presenta el diagrama de flujo del SI, en la figura 6.3 al módulo de password y en las figuras 6.4-A, B, C, D, E y F al módulo de Mantenimiento.

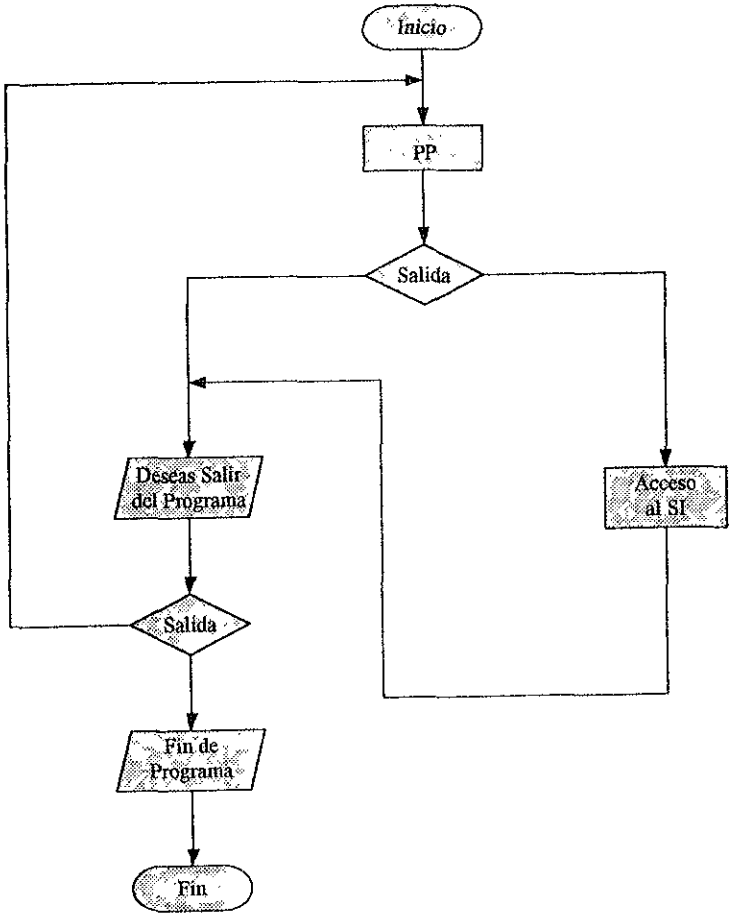


Figura 6.2. Diagrama de flujo del SI

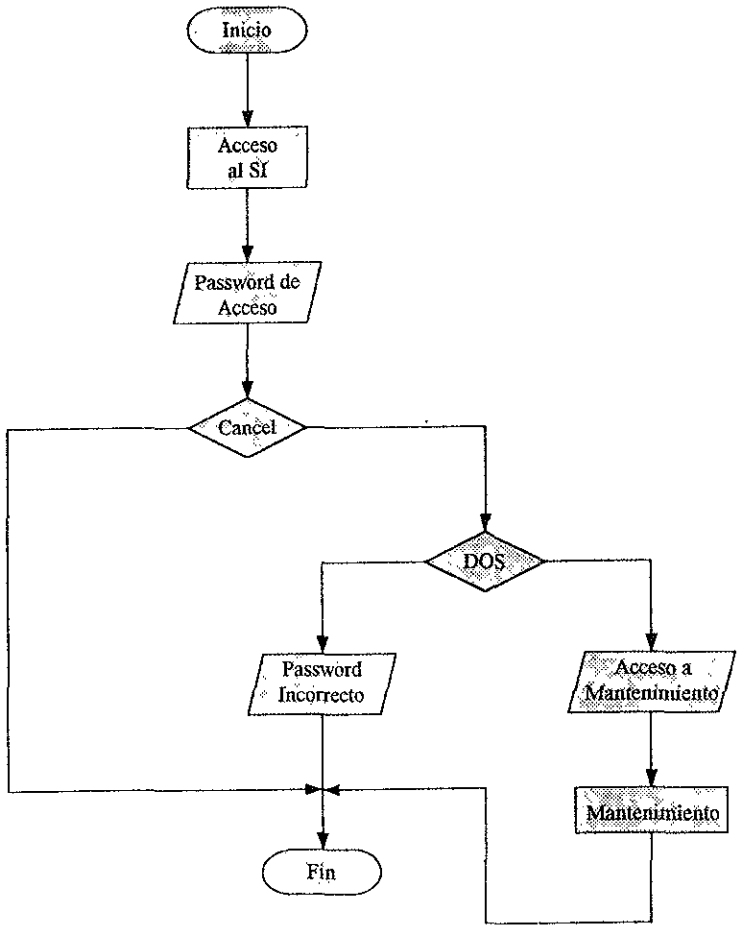


Figura 6.3. Diagrama de flujo del módulo de Password's.

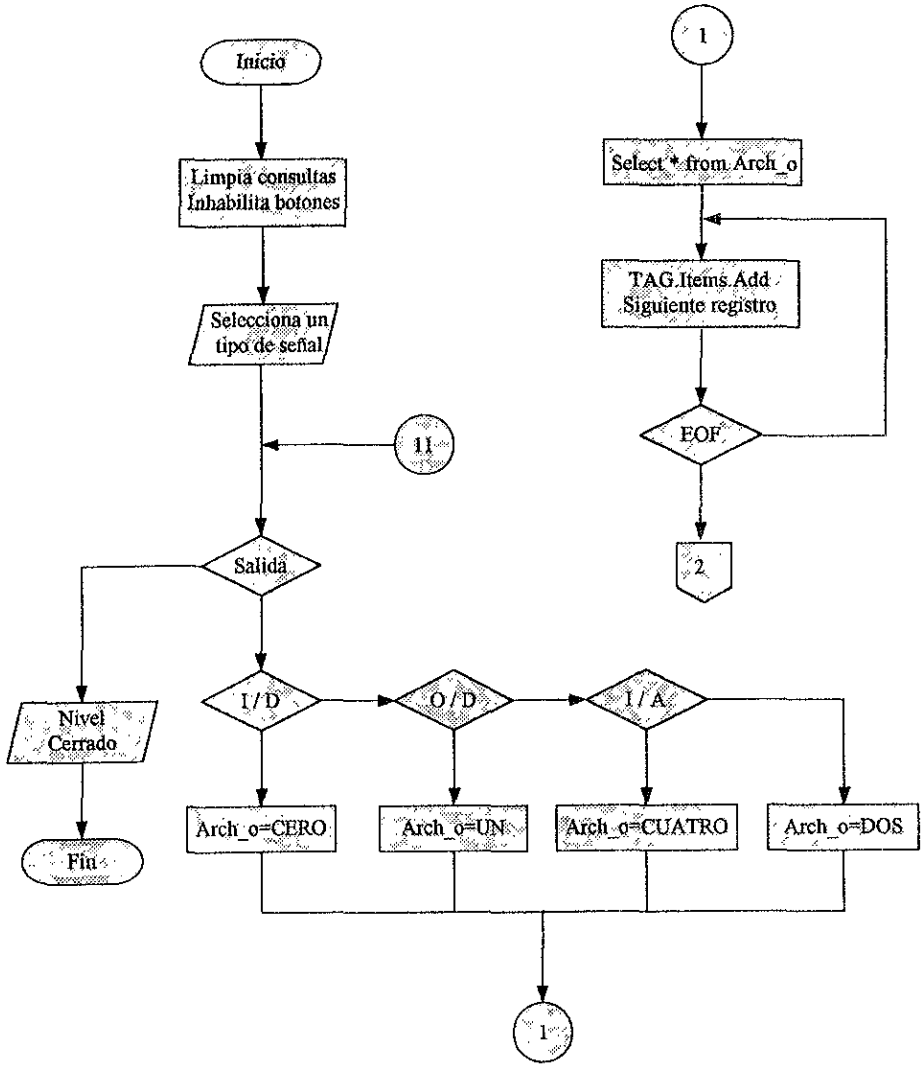


Figura 6.4-A. Diagrama de flujo del modulo de Mantenimiento

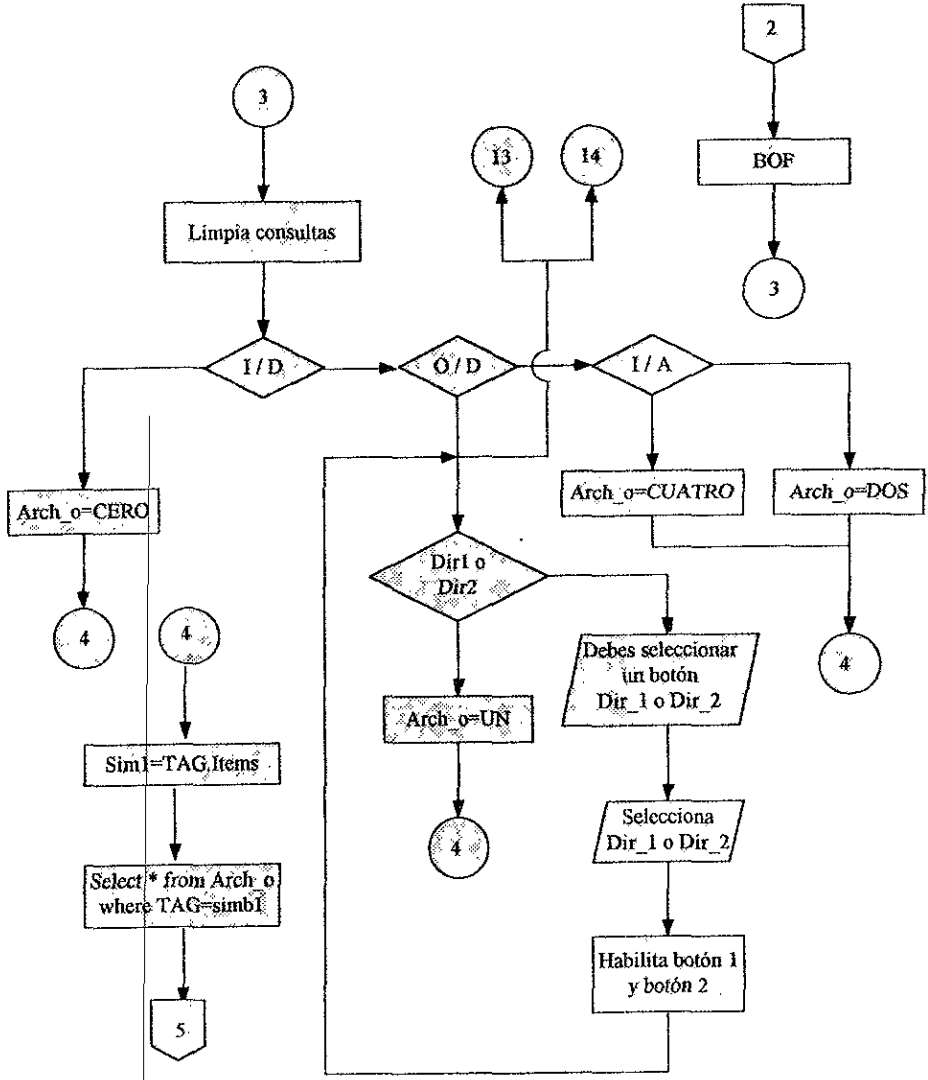


Figura 6.4-B. Diagrama de flujo del modulo de Mantenimiento

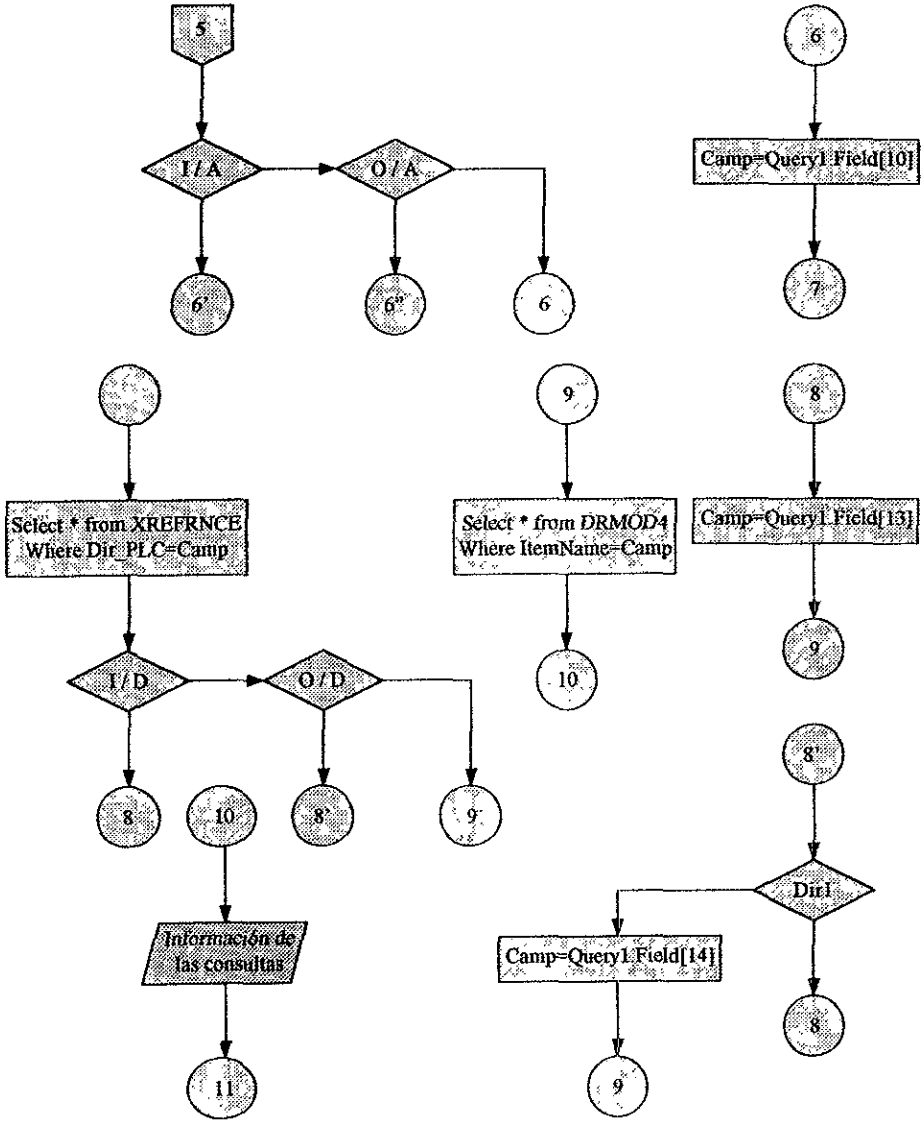


Figura 6.4-C. Diagrama de flujo del modulo de Mantenimiento

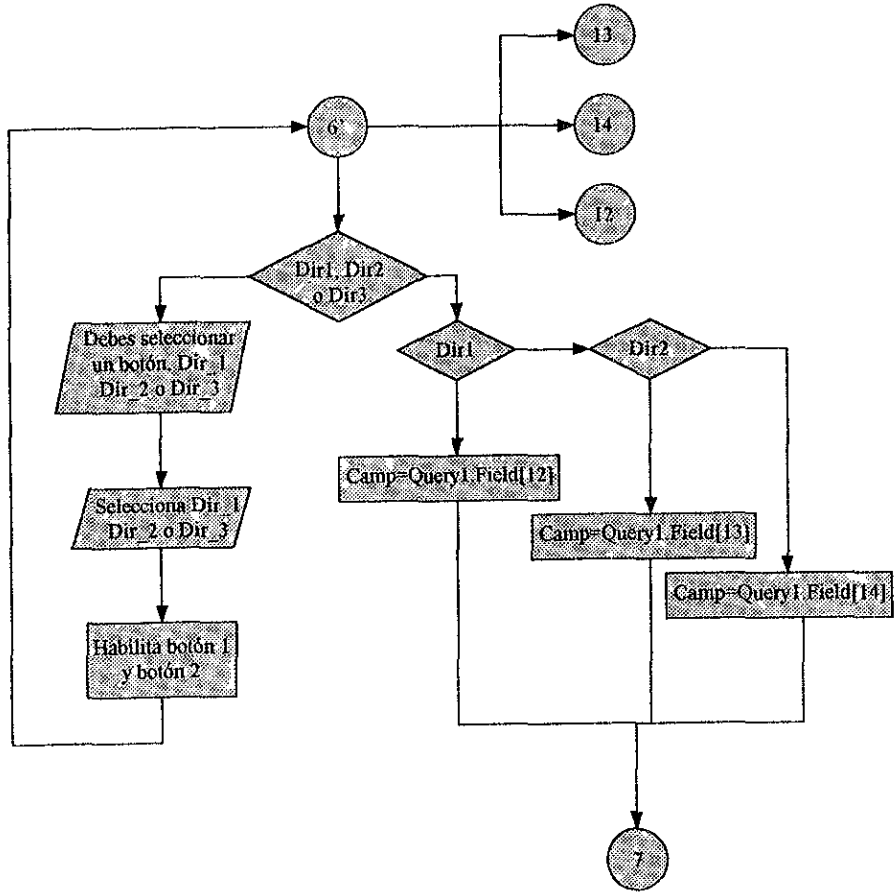


Figura 6.4-D. Diagrama de flujo del modulo de Mantenimiento

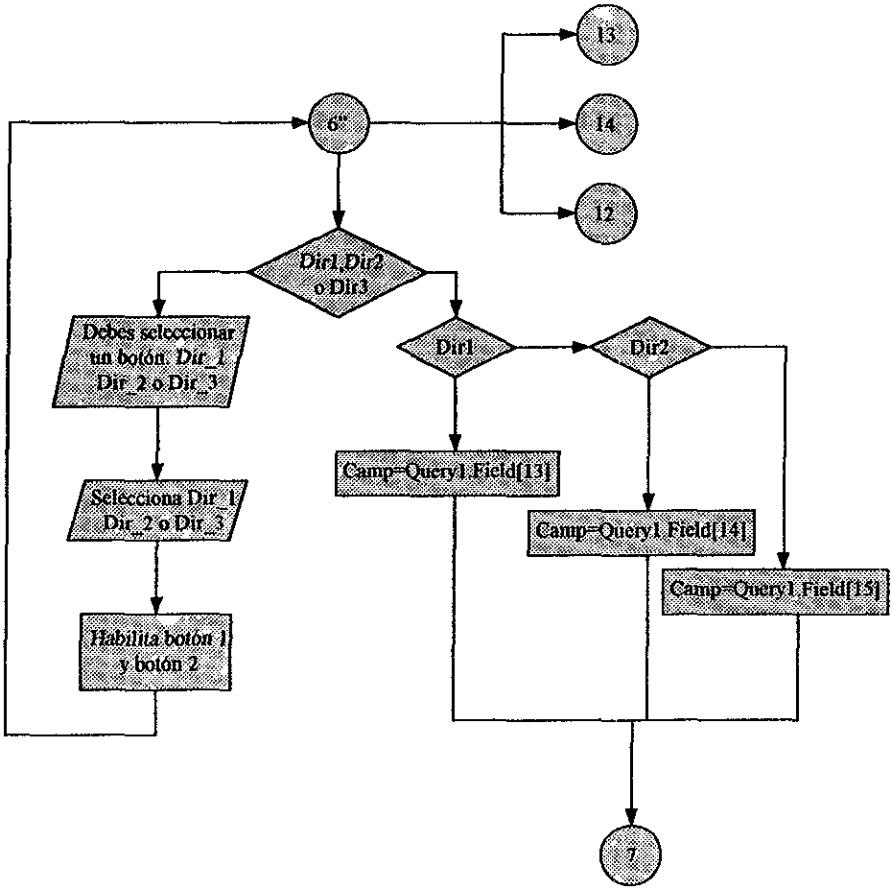


Figura 6.4-E. Diagrama de flujo del modulo de Mantenimiento.

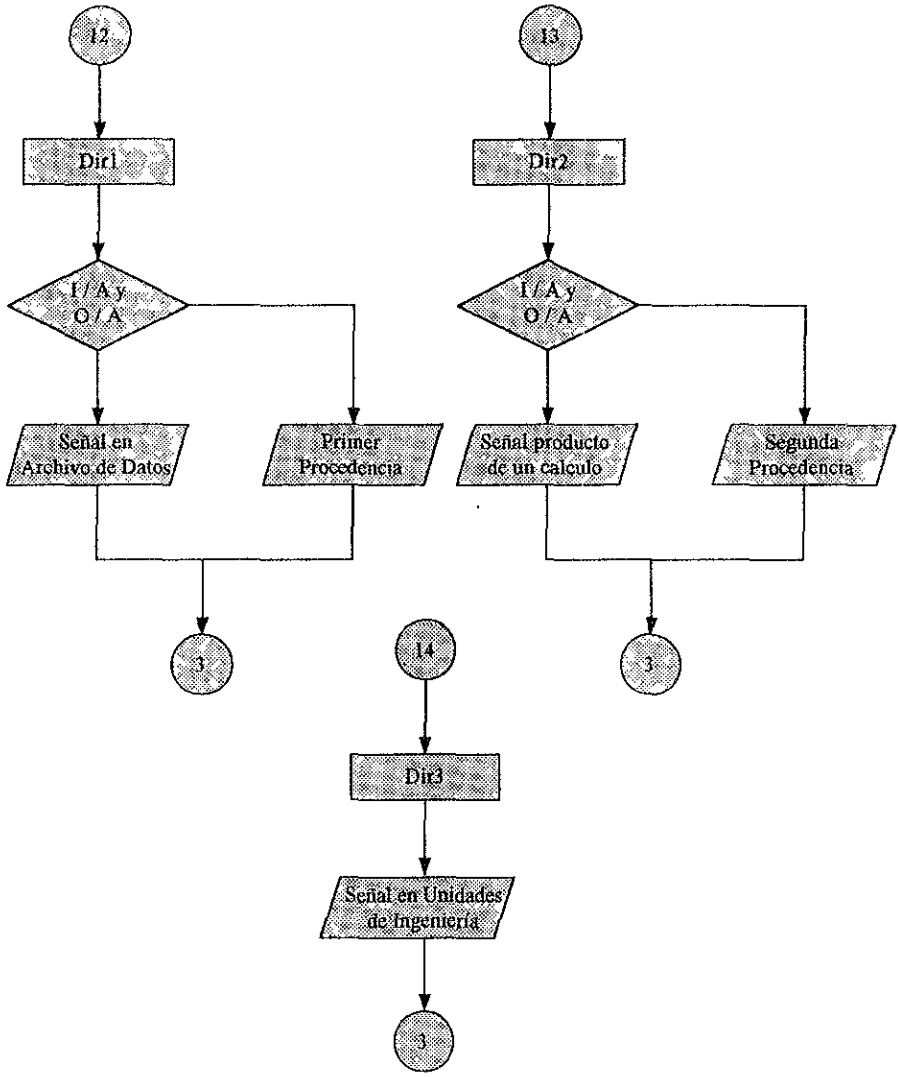


Figura 6.4-F. Diagrama de flujo del modulo de Mantenimiento

CONCLUSIONES

Se presentaron los fundamentos principales para construir un SI competente y a la par de nuestros días, destacándose la importancia de aplicar a la IS como una de las poderosas herramientas y disciplinas para desarrollar software de excelencia.

Se expuso de manera extensa los principios de diseño centrados en el usuario en conjunto con el diseño de pantallas que Windows 95 propone para desarrollar software de calidad.

Se describió y documentó la importancia que juegan los programas de instalación para el desarrollo de software final y comercial a la altura de los requerimientos de Windows 95.

Se dio una breve explicación del trabajo de campo que se lleva a cabo en los complejos marinos para entender las posibilidades de desarrollar software capaz de ayudar en la consulta de los dispositivos de campo

Se diseñó un software capaz de proporcionar información proveniente de los PLC's de campo, en donde a través de la interacción de varias BD's nos proporciona información respecto a su ubicación física y al tipo de señal que se maneja. El SI no se ha instalado en la Plataforma del complejo marino porque aunque es un producto terminado, no es la aplicación final y se encuentra en la etapa de pruebas para la puesta en marcha

El SI desarrollado proporciona una lógica de escalera, en la que nos permite saber por un lado la procedencia de la señal, es decir si proviene del PLC Crítico, del PLC de Proceso o de ambos, y también los escalones y programas de referencias cruzadas en donde se utilizan.

El SI nos indica la utilización de las señales indicándonos si es producto de un cálculo, si es parte de un bloque de señales o si se encuentra en unidades de ingeniería. También proporciona la información correspondiente para determinar en qué pantalla gráfica de WWI se está utilizando la señal o el grupo de ellas.

ABREVIATURAS

API	Application Programming Interface (Interface de Programación de Aplicaciones)
BD	Base de Datos
C / S	Cliente / Servidor
CPU	Central Processing Unit (Unida Central de Procesamiento)
E / S	Entrada / Salida
GDD	Grupo De Desarrollo
GUI	Graphical User Interface (Interfaz Gráfica de Usuario)
HD	High Density (alta Densidad)
IDAPI	Integrated Database Application Programming Interface (Interfaz Integrada de Programación de Aplicaciones de Bases de Datos)
IMP	Instituto Mexicano del Petróleo

IS	Ingeniería de Software
MAPI	Messaging Application Programming Interface (Interfaz de Programación de Aplicaciones Mensajeras)
MPT	Modelo de Productividad Total
MSDN	Microsoft Developer Network
SI	Sistema de Información
SIG	Sistema de Información Gerenciales
OLE	Object Linking and Embedding (vinculación e incrustación de Objetos)
PEMEX	Petróleos Mexicanos
PC	Personal Computer (Computadora Personal)
PLC	Programming Controler Logic (Controlador Lógico Programable)
POO	Programación Orientada a Objetos
PP	Programa Principal
SQL	Structure Query Lenguage (Lenguaje de Consulta Estructurada)
UNC	Universal Naming Convention (Convención Universal de Nomenclatura)
WWI	WonderWare Intouch

BIBLIOGRAFÍAS

Birnes J., "Enciclopedia McGraw-Hill de la Programación de Microcomputadoras, Lenguajes y Sistemas Operativos", McGraw-Hill Interamericana de México, México 1990.

Blank and Tarkin, "Ingeniería Económica", McGraw-Hill Interamericana de México, México 1988

Bob Svacina, "Understanding Hazardous Area Sensiting", Turck Inc , Mineapolis MN, E.U., 1994.

Brab George J., "Computadoras y Sistemas de Información en los Negocios", Ed Interamericana, México 1983.

Chapra y Canale, "Introducción a la Computación para Ingenieros", McGraw-Hill Interamericana de México, México 1989.

Frost R , "Bases de Datos y Sistemas Expertos (Ingeniería del Conocimiento)", Ed Díaz de Santos S A, Madrid, España 1989.

Grupo Industrial del Atlántico S.A. de C V , "Sistema Digital de Monitoreo y Control de la Plataforma de Compresión Pol-A, Manual de Operación y Mantenimiento, Plataforma de Compresión de Gas Pol-A", Cd. del Carmen, Campeche, México 1997.

Joyanes Aguilar Luis, "Turbo Pascal 5 5, Manual de Bolsillo", McGraw-Hill Interamericana de España, Madrid, España 1991.

Lucas Henry C., "Concepto de los Sistemas para la administración", McGraw-Hill Interamericana de México, México 1985.

Manning Michelle M., "Delphi 2, Guía Oficial de Borland", Prentice Hall Hispanoamericana, México 1996

Osier, Grobman y Batson, "Aprendiendo DELPHI™ 2 en 21 días", Prentice Hall Hispanoamericana, México 1996

Pressman Roger S., "Ingeniería de Software, un enfoque práctico", McGraw-Hill Iberoamericana, España 1993.

Scott George M., "Principios de Sistemas de Información", McGraw-Hill Interamericana de México, México 1990.

Senn James A , "Sistema de Información para la Administración", Ed Iberoamericana, México 1990

Sumanth David J., "Ingeniería y Administración de la Productividad", McGraw-Hill Interamericana de México, México 1990.

"Aprendiendo DELPHI™ 2 en 21 días", http //www prentice.com.mx

"Evolución Histórica del Instituto Mexicano del Petróleo",
http://www.imp.mx/evolución.html.

"IMP Grupo de Desarrollo en Instrumentación y Control -GDD-",
http://192.100.181 165/.

"Instituto Mexicano del Petróleo", http·//www.imp mx/.
"Información técnica de los PLC's de la familia 5 de Allen Bradley (5/60 y 5/40)", Allen Bradley, E.U., 1997

"Manual de Operación del Sistema Digital de Monitoreo y Control de la Plataforma de Compresión Pol - A", México 1995.

"Miscelánea de formatos e información del PLC-5 para referencias cruzadas", Plataforma de Compresión, Campeche, México 1997