UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

SISTEMATIZACION DEL TIPO DE SOPORTE TECNICO PARA MICROCOMPUTADORAS, MEDIANTE UN SISTEMA EXPERTO.

T E S I S

Que para obtener el Título de
INGENIERO EN COMPUTACION
p r e s e n t a:

OSCAR MONROY SOLIS

MEXICO, D. F. 1993

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

INDICE 1

CAPITULO 4. ELABORACION DEL SISTEMA EXPERTO.

INTRODUCCION

El Tratado de Libre Comercio (TLC), que firmará nuestro país con Estados Unidos y Canadá, sin duda constituye modernizar el aparato productivo así como alcanzar niveles superiores de desarrollo correspondientes a procesos productivos más eficientes, competitivos y directamente vinculados con la calidad de la Ingeniería en México.

Ingeniería que hoy en día se apoya cada vez más en los equipos de cómputo como herramientas eficaces para la ágil y oportuna toma de decisiones, factor vital del éxito o fracaso en la operación de las organizaciones modernas.

La Inteligencia Artificial como área de las Ciencias de la Computación, ofrece gran potencial para el planteamiento y resolución de problemas en una amplia variedad de campos: Representación del Conocimiento, solución heurística de problemas, sistemas Expertos, robótica, redes neuronales, programación automática, entre otros.

El presente trabajo expone una de las áreas más importantes dentro del estudio de la Inteligencia Artificial: Los Sistemas Expertos, enfocados primordialmente a la solución pragmática de los problemas científicos, donde la experiencia humana juega un papel fundamental para su desarrollo.

El objetivo propuesto consiste en elaborar un Sistema Experto que realice las funciones de Soporte Técnico para el diagnóstico y control de problemas de software al utilizar microcomputadoras del tipo PC compatibles. Idea que surge como una necesidad real en múltiples organizaciones, cada una con su propia problemática: infraestructura, recursos humanos, escasez presupuestal, eficiencia, etc; pero todas ellas con un factor común : tiempos de respuesta oportunos y eficaces ante cualquier problema técnico que se presente en sus equipos informáticos.

Para tal efecto, se consideró que la base de información a tratar fuera la más útil, la más conocida o estándar en el medio con el fin de que los resultados obtenidos fueran tangibles y directamente aplicables, pero a su vez se procuró que dicha información fuera lo menos voluminosa por tratarse de un trabajo donde el enfoque académico es primordial. Cumplimiento de esta selección fueron los sistemas operativos MS-DOS versión 5.0 y Windows versión 3.1.

Como herramienta de programación se utilizó el lenguaje C++ Orientado a Objetos con la característica de permitir analizar un problema en la composición de subgrupos entendidos cada uno de ellos como un objeto que puede ser tratado y aplicado particularmente, así como el polimorfismo que admite utilizar un nombre para diversos propósitos relacionados pero ligeramente diferentes y la herencia donde un objeto puede adquirir las propiedades de otro, es decir se puede clasificar. Y el lenguaje Clipper en la generación de reportes, captura de información de la Ficha de Servicio, catálogos y mantenimiento al Sistema.

Para alcanzar lo anterior los diferentes elementos que se consideraron en forma Conceptual, del Análisis y Desarrollo del Sistema están agrupados en cuatro Capítulos mismos que se enuncian en forma breve a continuación.

En los Capítulos uno a tres se hace una revisión conceptual, primero el Capítulo uno: Estructuras de Datos; respecto a la Programación, Recursividad, etc, como herramienta en las Estructuras de Control de programas; el Capítulo dos: Bases de Datos (modelo entidad relación); como manejo y administración correcta de la información y el Capítulo tres: Inteligencia Artificial, donde se examinan las Técnicas básicas para la resolución de problemas y representación del conocimiento; aunando esencialmente en la parte del Area de Sistemas Expertos.

Finalmente el capítulo cuatro : Elaboración del Sistema Experto; comprende las fases de : Análisis, Diseño, Desarrollo y Depuración del Sistema; así como parte del listado de la programación.

Este trabajo es un Sistema que pretende ser una herramienta empresarial para mejorar la calidad y la disponibilidad del crecimiento que requieren los profesionales en la toma de decisiones.

CAPITULO I

ESTRUCTURA DE DATOS

INTRODUCCION

Dentro de las Ciencias de la Computación el ámbito de las Estructuras de Datos es importante tanto en el estudio, como en la aplicación en las computadoras. Los resultados exitósos de los Sistemas tienen mucho que ver con la elección de las Estructuras de Datos (que son tan importantes como la elección de los algoritmos).

De hecho, se necesita un conocimiento de los algoritmos al aplicar los datos para hacer una acertada elección de las Estructuras de Datos. Recíprocamente, la elección de buenos algoritmos requiere el conocimiento de las Estructuras de Datos usadas.

Es por eso que se pretende presentar en este capítulo las Estructuras de Datos Elementales asi como las Estructuras de Datos Compuestas, ya que en algunas partes de la programación nos serán de gran ayuda, al reflejar los requerimientos de las operaciones a ejecutar sobre los datos.

I. ESTRUCTURA DE DATOS

Se define una Estructura de Datos como un conjunto de elementos que conservan relaciones trascendentes entre si y cuya organización esta en función del tipo de acceso y recuperación de los elementos.

Las Estructuras de Datos se clasifican básicamente en 2 tipos: Las Estructuras de Datos Elementales (Incorporadas al lenguaje) y las Estructuras de Datos Compuestas (Definidas por la habilidad del programador), que se subdividen a su vez en listas lineales y listas NO lineales.

Para implementar cualquiera de las Estructuras de Datos mencionadas se lleva a cabo, por un lado reservar memoria para la estructura de datos y por el otro determinar el código que ha de generarse para la función de acceso.

En la práctica, el cómo esté configurada la memoria es considerado por el que escribe el compilador; es decir la Arquitectura del procesador determina la configuración de la Memoria; 8088, 80286, 80386 etc.

II. ESTRUCTURAS DE DATOS ELEMENTALES

Las Estructuras de Datos Elementales son aquellas que se encuentran incorporadas al lenguaje de programación, esto quiere decir, que no necesitamos preocuparnos en su implementación ya que basta especificarlas para su uso; pueden ser números enteros, números reales, caracteres, arreglos, etc. Este tipo de estructuras de datos se ha estandarizado en los lenguajes de programación. Por ejemplo como programador uno puede usar este tipo de estructuras de datos sin tener que saber nada respecto a su implementación, al igual que un carpintero usa un cepillo eléctrico sin saber nada de electricidad.

Es decir las Estructuras de Datos Elementales son estructuras de datos incorporadas al lenguaje y el programador pudiera que nunca supiera como realmente se lleva a cabo este acceso ya que es transparente para el, pero definitivamente es recomendable consultar bibliografía para conocer como se lleva a efecto.

II.1. NUMEROS ENTEROS

La representación en la computadora de un número entero se realiza por medio de el lenguaje binario (0 y 1) ó lo que es lo mismo de base 2 debido a la tecnología que se utiliza en la memoria y sobre todo porque en la generalidad de representar cualquier caracter bastan 2 : 0 y 1, para representar cualquier cosa por larga que sea, es decir aunque se necesiten muchos bits en realidad solo tenemos 2 diferentes.

Los números enteros pueden ser positivos o negativos, para representarlos en la computadora se utiliza el dígito binario de la extrema izquierda para el signo, si este es 0 será negativo, y si es 1 será positivo.

Por ejemplo :

```
 127 en binario es  0 0 0 0 0 0 0 0  0 1 1 1 1 1 1 1
-127 en binario es  1 0 0 0 0 0 0 0  0 1 1 1 1 1 1 1
                    |
                   bit de signo.
```

En la mayoría de las computadoras (incluyendo las basadas en

la familia del procesador 8086) usan aritmética de complemento a dos, lo que hace que la representación de -127 sea diferente. Sin embargo el uso del bit de signo es el mismo. Un número negativo en forma de complemento a dos tiene todos los bits del número invertidos y un uno sumado al número. Ejemplo -127 en complemento a dos aparece como sigue :

-127 en binario 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1

II.2. NUMEROS REALES

Dentro de la representación de números reales existen algunas aplicaciones que involucran el manejo de cantidades o muy pequeñas o mucho muy grandes. Por ejemplo para obtener una precisión de 27 dígitos decimales, se requiere de aproximadamente 90 dígitos binarios.

El método alterno es el de la notación en punto flotante :

S EXPONENTE MATISA

S = Signo de la matisa.

EXPONENTE = Número entero en complemento a dos.

MATISA = Como número entero en magnitud y signo.

El número de bits para el exponente y la matisa depende de la magnitud y de la precisión de los números a representar.

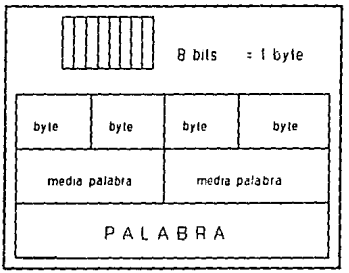
Por ejemplo :

float 32 bits 3.4E-38 a 3.4E+38

II.3. REPRESENTACION DE CARACTERES

La representación de caracteres en una computadora se hace a través de códigos; dependiendo de la computadora será el código utilizado. Existen varios códigos, de entre los más usuales tenemos el BCD Binary Code Decimal, que necesita 6 bits para representar cualquiera de los 63 caracteres (2^6 = 64 - 1 = 63) diferentes que tiene; el código EBCDIC Extended Binary Code Decimal Interchange Code que requiere 8 bits (2^8 = 256 - 1 = 255) para representar 255 caracteres y finalmente el ASCII, American Standard Code for Information Interchange también de 8 bits (2^8 = 256 - 1 = 255) para representar 255 caracteres.

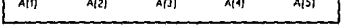
Por ejemplo si 8 bits forman 1 byte, 2 bytes media palabra y 4 bytes 1 palabra; puesto que 8 bits se usan para representar un caracter, seria ineficiente almacenar un caracter por palabra, en este caso varios caracteres son empaquetados en una sola palabra; algunas veces la palabra es de 2 bytes, dependiendo de la arquitectura del computador.



II.4. REPRESENTACION DE ARREGLOS

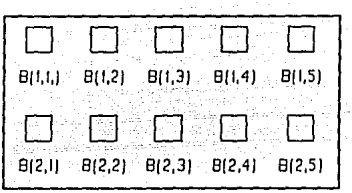
Dentro de las Estructuras de Datos incorporadas al lenguaje se encuentra una muy importante, el arreglo (array) . Existen dos tipos de arreglos utilizados, el arreglo unidimensional y el arreglo bidimensional.

El arreglo unidimensional consta de un conjunto de elementos ordenados, es decir elementos que tienen un orden en su colocación y que son finitos; al arreglo se le llama con un nombre y a cada elemento del arreglo se le asocia un índice que le fue asociado :



Cuando se declara el arreglo se le "dice" al compilador cuántas celdas se necesitan para representar el arreglo y el compilador asigna las celdas de memoria a las variables en un orden secuencial.

El arreglo bidimensional consta de un conjunto de elementos ordenados ahora en 2 dimensiones, es decir como una matriz; de hecho el arreglo bidimensional representa la estructura de datos ideal para declarar datos en forma de matriz de renglones y columnas, cada dimensión nos representa una relación; también se le asocia un nombre, y a cada elemento un par de índices que nos dará el elemento deseado.



III. ESTRUCTURAS DE DATOS COMPUESTAS

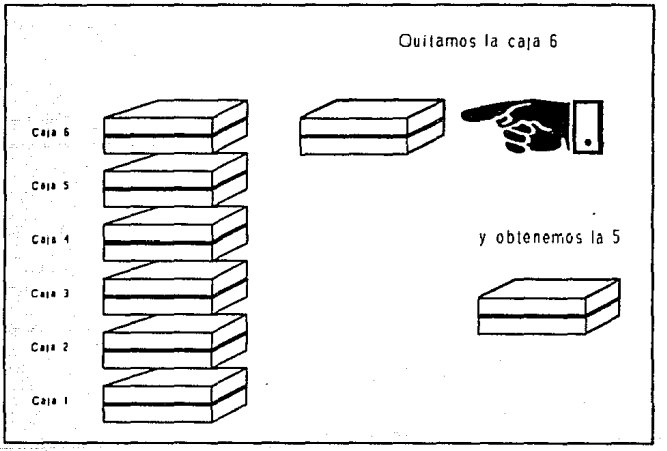
A diferencia de las Estructuras de Datos Elementales (Incorporadas al lenguaje), las Estructuras de Datos Compuestas requieren de la habilidad del programador y se subdividen en : Listas lineales y Listas NO Lineales; estas no se suministran en los lenguajes de programación de propósito general.

Las listas lineales de las que hablaremos son : la Pila, Cola, Cola doble y Lista Circular que constituyen los estándares en las estructuras de datos construidas por el programador, es decir, el programador especifica un conjunto de procedimientos y funciones que afectan las funciones de entrada; lo mismo sucede con las estructuras de datos NO lineales, como son los árboles binarios, una estructura de datos más compleja, cuya relación entre sus nodos puede ser mayor a una dimensión.

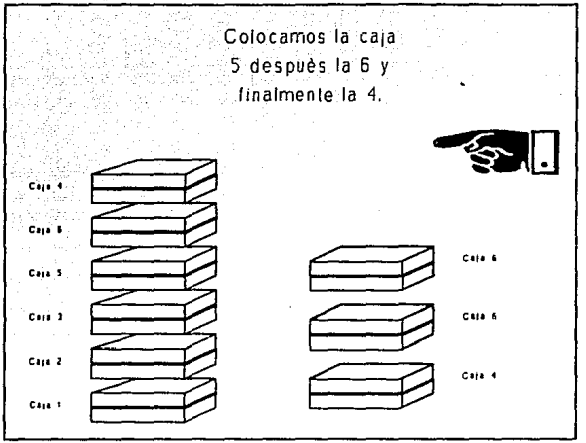
La ventaja especial para usar el árbol binario es que facilita la búsqueda. Aunque no es una estructura de acceso directo como el arreglo, suministra un acceso más rápido y más constante a los nodos individuales que una lista enlazada. Por tanto es particularmente útil en aquellas aplicaciones en las que debe minimizarse el tiempo de búsqueda o en la que los nodos no se procesan necesariamente en orden. En el árbol binario el tiempo de búsqueda esta en función del número de comparaciones que se necesitan para localizar un elemento.

III.1.1. PILA

La Pila ó Stack es una estructura de Datos Lineal, definida por el usuario (programador). Consideremos la ilustración de una pila de caja de zapatos, estas cajas fueron ordenadas una por una hasta quedar en forma de una pila de cajas.



Si quisiéramos obtener la caja número 5 tendríamos que sacar la caja número 6 y posteriormente la caja número 5, si quisiéramos ahora la caja 4 sacaríamos las cajas 6 y 5, si la número 3, la 6, 5 y 4 y así progresivamente. De igual forma, si pretendieramos darle otro orden a la pila de cajas, como por ejemplo 1 - 2 - 3 - 5 - 6 - 4 :



necesitamos quitar las cajas 6, 5 y 4 y después colocar primero la caja 5, posteriormente la caja 6 y finalmente la caja 4.

De esta manera podemos afirmar que la pila sigue cierta regla conocida como U.E.P.S. Ultimas Entradas Primeras Salidas ó lo que en inglés se conoce como Last In, First Out (LIFO) última en entrar primera en salir si pretendieramos obtenerla, ya que solo por un solo extremo (la cima) obtenemos la caja (elemento). Existen dos operaciones importantes en la estructura de datos Pila, agregar y retirar, se conocen como PUSH y POP respectivamente.

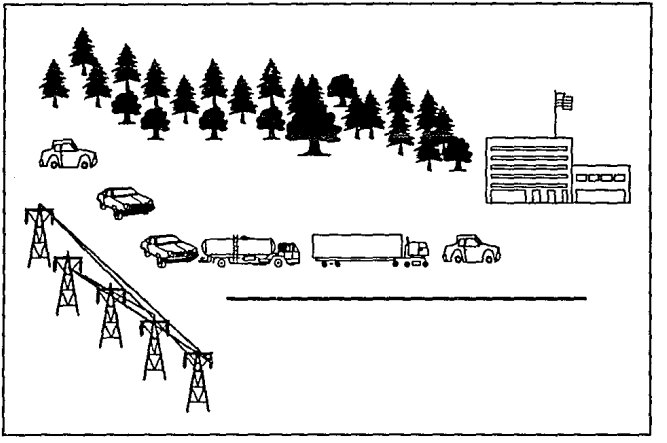
Definiremos entonces una pila como : Una Estructura de Datos Lineal en la que los componentes se agregan ó retiran solo por un extremo.

Accederemos a una pila sólo por la cabeza, no por la mitad o por el final. Reconocer esta distinción de la pila es importante: aunque la implementación de la pila puede hacerse con una estructura de acceso directo como el arreglo, la pila como entidad lógica no es accesible directamente.

Una pila es la Estructura de Datos apropiada cuando queremos que la información se guarde y luego recuperarla en forma inversa.

III.1.2. COLA

Una cola al igual que una pila es una Estructura de Datos Lineal definida por el Ingenio del programador. Pensemos ahora en una cola (fila) para salir a carretera (caseta de cobro) :



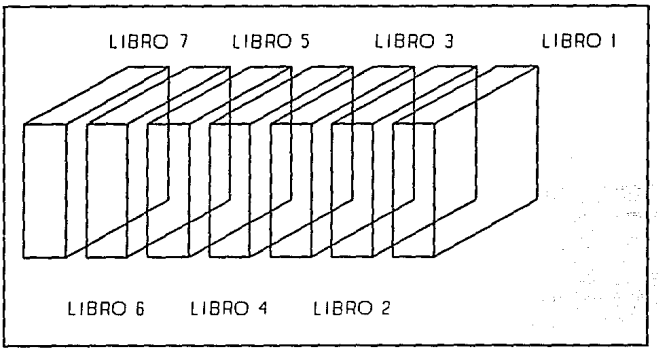
aquí el primer auto (elemento) en salir será el primero que llego, el segundo auto saldrá después del primero y el último en llegar al extremo de la cola sería entonces el último en salir. Notemos que aquí las operaciones se realizan por ambos extremos, se agrega por un extremo, el final de la cola y se retira por el otro, el principio de la cola.

De la forma en que se añaden los elementos a la cola, este método sigue la regla; primeras entradas - primeras salidas, P.E.P.S. ó en Inglés FIFO, First In First Out, primero en entrar primero en salir.

Definiremos entonces a una cola como : La Cola es una Estructura de Datos Lineal de la forma Primeras Entradas Primeras Salidas y en donde las operaciones se realizan por ambos extremos, en el extremo final se agrega y en el inicial se retira.

III.1.3. COLA DOBLE

La cola doble definida también por la habilidad del programador, es más dinámica que la cola simple. Pensemos ahora como muestra la figura, en una ordenación de libros :



y en donde las operaciones colocar (agregar) y quitar (retirar), pueden realizarse por ambos extremos, es decir podríamos pensar en retirar el libro 1, ó colocar uno más, y lo mismo sucedería al final de la cola, colocar un libro más, o retirar el último, si podemos percibir la Cola doble, puede comportarse como, ya sea una pila o como una cola.

Una Cola Doble es una Estructura de Datos Lineal en la cual las operaciones de retirar y agregar se realizan por ambos extremos, sin seguir necesariamente una regla como UEPS ó PEPS, pero en cambio es posible llevarlos a cabo ó combinarlos.

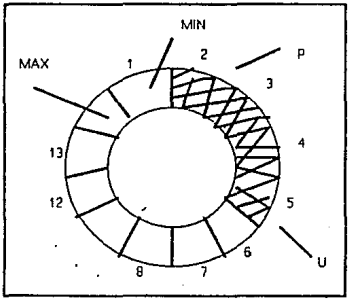
La operación de agregar puede entonces realizarse por cualquiera de los 2 extremos, lo mismo sucede cuando queramos retirar un elemento. De tal forma que serían 4 los algoritmos para la cola doble : agregar y retirar por " enfrente " y retirar por el " final " de la cola.

III.1.4. LISTA CIRCULAR

La Lista Circular es una Estructura de Datos Lineal, determinada por la destreza del programador, es un poco más elaborada que las anteriores estructuras vistas.

La lista circular tiene la característica de un orden, en el que a la última localidad le sigue la primera; esto permite que la cola que contiene la lista circular se desplace sobre la memoria, y tome las localidades disponibles por donde se inicio la cola.

Este tipo de lista conlleva a solucionar el problema del desplazamiento de las celdas sobre la memoria, como sucede en la estructura de datos Cola, al borrar un elemento y querer conservar fijo el frente en la primera celda tenemos que recorrer las celdas una posición, aparte de que la cola no puede seguir creciendo ya que tiene cierta localidad límite, en la lista circular esa localidad límite es la localidad disponible por donde se inicio la cola :



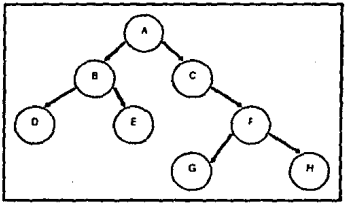
Una lista circular es entonces una Estructura de Datos Lineal que tiene la característica de que al último elemento ó nodo le sigue el primero, o que a la última localidad le sigue la primera, permitiendo el desplazamiento de la lista (cola circular); tomando las localidades disponibles donde se inicio la cola.

III.2. LISTAS NO LINEALES

III.2.1. ARBOLES BINARIOS

El Arbol binario tiene mucha importancia en las ciencias de la computación. Aunque no es una estructura de acceso directo como el arreglo, proporciona un acceso más rápido y más constante a las celdas individuales que una lista circular (enlazada). EL árbol binario es una estructura de datos compuesta NO lineal.

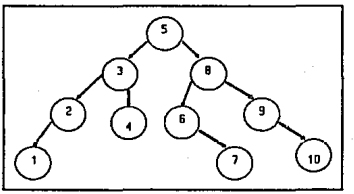
De hecho el árbol binario es una estructura en árbol que tolera que cada celda apunte a otras dos : la anterior y la posterior (la que le sigue), excepto la raíz, en donde solo estará la posterior. A diferencia de un nodo en una lista circular, un nodo en un árbol binario de búsqueda no apunta necesariamente a los nodos cuyos valores le preceden o le continúan próximamente; puede ser cualquiera pero que el nodo de la izquierda contenga un valor más pequeño del que le apunta y el de la derecha contenga un valor mayor.



En un árbol binario, el nodo posee uno, dos, o ningún subárboles, cuando tiene ninguno o 2 subárboles es llamado árbol estrictamente binario, si no es este el caso sería llamado de Knuth.

Cada árbol binario tiene un primer elemento la raíz del árbol, de este nodo parten todas las ramas hacia abajo. Dentro del vocabulario de los árboles binarios se manejan los siguientes términos que ayudan a comprender mejor esta estructura de datos.

A la raíz se le llama padre de los nodos que le siguen en este caso nodo-3 y nodo-8, estos nodos a su vez tienen hijos nodo-6 y nodo-9, que son nietos ahora del nodo-5; a su vez también los nodos 2, 6 y 9 tienen hijos 1,7 y 10 respectivamente, que son bisnietos del nodo raíz (nodo 5), sus abuelos son 3 y 8 y finalmente sus padres como ya se menciono son 2, 6 y 9 respectivamente.



En los árboles binarios existen niveles, el nivel de la raíz es 0, el de las ramas de la raíz es 1 y así sucesivamente hasta el nivel más bajo. El número máximo de cualquier nivel n es 2^n.

Por ejemplo si una lista contiene 1,000 nodos, se deberán hacer 1,000 comparaciones si se quisiera buscar alguno, si estos 1,000 nodos los colocamos en un árbol binario de búsqueda (y el árbol esta equilibrado) se necesitarán solo 11 comparaciones independientemente del nodo buscado.

II.2.2. RECORRIDO EN ARBOLES BINARIOS

Recorrer un árbol implica " visitar " todos y cada uno de los nodos, minimizando el tiempo de búsqueda. Existen varias formas de recorrer un árbol :

Top-Down. De arriba a abajo y de Izquierda a Derecha.

Bottom-Up. De abajo a arriba y de Izquierda a Derecha.

Preorder. Se visita el nodo.
Se recorre el subárbol izquierdo.

Inorder. Se recorre el subárbol izquierdo.
Se visita el nodo.
Se recorre el subárbol derecho.

Postorder. Se recorre el subárbol izquierdo.
Se recorre el subárbol derecho.
Se visita la raiz.

Ejemplo :

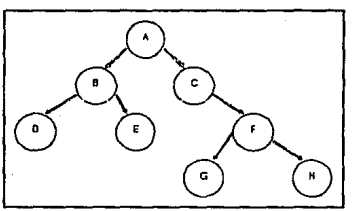
TOP - DOWN a b c d e f g h

BOTTOM - UP g h d e f b c a

PREORDER a b d e c f g h

INORDER d b e a c g f h

POSTORDER d e b g h f c a



Este tipo de recorridos deben de tratar de ajustarse a la aplicación que queramos; finalmente en cada recorrido se sigue un perfil de recuperación de la información y de hecho esta información puede ser recuperada antes de visitar sus subárboles, después de visitar alguno ó después de visitar los 2 subárboles.

Un árbol binario entonces nos facilitara la búsqueda. Es útil en aquellas aplicaciones en las que el tiempo de búsqueda es importante, recordemos que el tiempo de búsqueda es función del número de comparaciones para localizar un elemento.

La elección sobre la estructura de datos que elijamos dependerá fuertemente de los propósitos para los que se utilicen.

CAPITULO II

BASES DE DATOS

INTRODUCCION

Hoy en día, las organizaciones están requiriendo cada vez más de la determinación y la precisión de la información. Esta es un elemento clave dentro de cualquier empresa, por lo tanto el manejo y la administración de la información son parte importante en los Sistemas de Computo. Los Sistemas de Bases de Datos son en consecuencia parte angular en la correcta administración y explotación de la información. El presente capítulo comprende los modelos de datos de uso común en los Sistemas de Bases de Datos: Modelo entidad relación, Modelo relacional, Modelo Jerárquico y Modelo de Red; estos modelos ayudan a diseñar Bases de Datos y a extraer toda la información que puede ser de utilidad. Así como el uso de las Herramientas Case como opción en la implementación de metodologías de manera automatizada, en el desarrollo de un Sistema de aplicación.

En realidad la evolución de las Bases de Datos a través de la Historia ha permitido mejoras tanto en la utilización misma de la computadora como en la productividad y facilidad de uso de los programas. En los años cincuentas se significaron por dar a conocer el manejo de los archivos de Bases de Datos; en los años sesentas los usuarios podían hacer uso de Bases de Datos Jerárquicas; diez años más tarde ya se podía trabajar con Redes de Computadoras, pese a que las Bases de Datos relacionales aparecieron en los ochentas; y sin duda pronto hablaremos de Sistemas para manejo de Bases de Datos orientadas a objetos que podríamos considerar como el siguiente paso en los próximos años.

En el diseño de Bases de Datos los analistas determinan las entidades que se van a incluir en la Base de datos y después establecen cada relación entre las entidades. El modelo de Datos define como se muestran las relaciones entre las entidades de las Bases de Datos.

El Modelo relacional puede reducir al mínimo el mantenimiento de programas debido a la independencia de los datos. De hecho el usuario especifica la información que necesita y no como consultarla, esto aunado al hecho de que entonces la programación se eleva a la solución real de problemas.

Las Bases de Datos de Red (Reticular) ó Jerárquica son conceptualmente sencillas, pero cabe destacar que en un entorno de grandes Bases de Datos, pueden rápidamente transformarse en una complicada confusión de interrelaciones que pueden llegar a ser difícil manejar a medida que evoluciona la Base de Datos con el uso.

Y finalmente las Herramientas CASE que surgen de la necesidad de atenuar los problemas de desarrollo de Software como : Costos de proyectos que sobrepasan el presupuesto previsto así como el tiempo real de desarrollo de los Sistemas, que exceden a lo planeado.

CASE son las siglas de Computer Aided Software Engineering que significa Ingeniería de Programación apoyada en el uso de computadoras. Estas herramientas CASE tienen como objetivo automatizar la creación y el mantenimiento de Sistemas.

I. DEFINICION DE SISTEMAS DE BASES DE DATOS

Un Sistema de Bases de Datos consiste en un conjunto de datos relacionados entre sí por medio de una forma estructurada y organizada de manera lógica, así como un conjunto de programas que sirven para explotar de forma eficiente la información contenida en el Sistema. El conjunto relacionado de datos habitualmente se conoce como Bases de Datos. En el Sistema de Bases de Datos se debe asumir que el usuario debe tener una visión abstracta de la información, es decir no debe "enterarse" que pasa con su almacenamiento sino más bien optimizar la consulta eficiente del Sistema interpretando y utilizando adecuadamente la información.

II. MODELO ENTIDAD RELACION

II.1. DEFINICION

El Modelo Entidad Relación esta basado en la apreciación de una situación real, dicha situación la componen objetos básicos denominados entidades y también de relaciones entre estos objetos.

Una "entidad" es un objeto sobre el cual guardamos información y el cuál puede distinguirse de otros objetos, esta puede ser material ó abstracta, como por ejemplo : un doctor, un paciente, un sitio, un fragmento, etc ó por el otro una cuenta bancaria, un evento, o el mismo concepto abstracto.

Una entidad posee propiedades que deseamos registrar y guardar:

Ejemplo :

designemos a esta entidad como la entidad

Alumno :

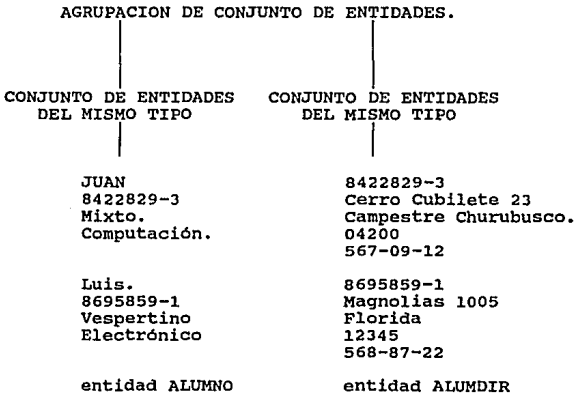
Nombre.
Número de cuenta.
Turno.
Carrera.

Cada campo es un " atributo " asociado a la entidad Alumno, tendríamos entonces un conjunto de entidades del mismo tipo Alumno. Si definiéramos otra entidad

Alumdir :

Número de cuenta.
Dirección.
CP.
Tel.

tendríamos otro conjunto de entidades del mismo tipo Alumdir; entonces una base de datos abarca una agrupación de conjuntos de entidades, cada una de las cuales a su vez contienen cualquier número de entidades del mismo tipo, es decir existe una agrupación de conjuntos de entidades y a su vez cada subconjunto de esta agrupación es un conjunto de entidades del mismo tipo, como se ilustra en la figura :



Por otro lado una relación es una asociación entre varias entidades, por ejemplo Juan tiene una relación con su número de cuenta, su turno y su carrera, es decir un conjunto de relaciones de un grupo de relaciones del mismo tipo, el grupo de relaciones son todas las relaciones que guardan la agrupación de conjuntos de entidades y cada conjunto de entidades del mismo tipo tiene un conjunto de relaciones del mismo tipo; así también hay una agrupación de relaciones.

II.2. RELACIONES DENTRO DE UN MODELO DE DATOS

Antes de mencionar las relaciones dentro de conjuntos de entidades (del mismo tipo) ó limitaciones de Mapeo, consideremos que cada conjunto de entidades del mismo tipo(Llamemosle de ahora en adelante simplemente entidad), debe tener una Llave que identifique en forma única cada suceso, como ya se menciono la entidad esta formada por uno ó más atributos (toda entidad debe ser identificada con una Llave); la Llave es un atributo que caracteriza de forma única una relación del conjunto de relaciones del mismo tipo; por ejemplo en las entidades que definimos anteriormente el Número de cuenta es único, basta con tenerlo para saber que solo una y solo una persona representa a ese Número de cuenta, la Llave puede estar formada por uno o varios atributos. Una Llave es NO-REDUNDANTE si no se puede eliminar ninguno de sus campos sin destruir su habilidad para identificar de forma única cada suceso; pueden existir diferentes conjuntos de atributos que identifiquen en forma única un suceso; a estas se les llama Llaves candidatas; la Llave primaria es la que se escogió para identificar a los sucesos y normalmente debe ser aquella que tenga el menor número de campos.

Una relación es entonces una conexión entre dos ó mas entidades.

Existen cuatro tipos de relación básicos :

A) 1 a 1 (uno a uno)

B) 1 a N (uno a muchos)

C) N a 1 (muchos a uno) y

D) M a N (muchos a muchos)

Consideremos las entidades Alumno (A) y Alumdir (B) :

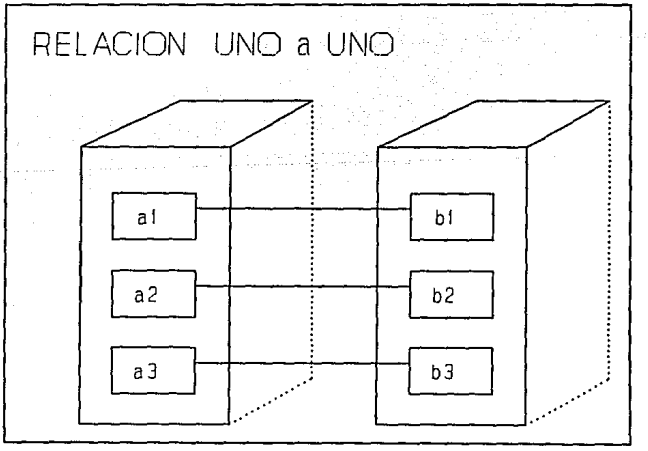
A) Una entidad A esta asociada únicamente con la entidad B, y la entidad B esta a su vez asociada solo con la entidad A.

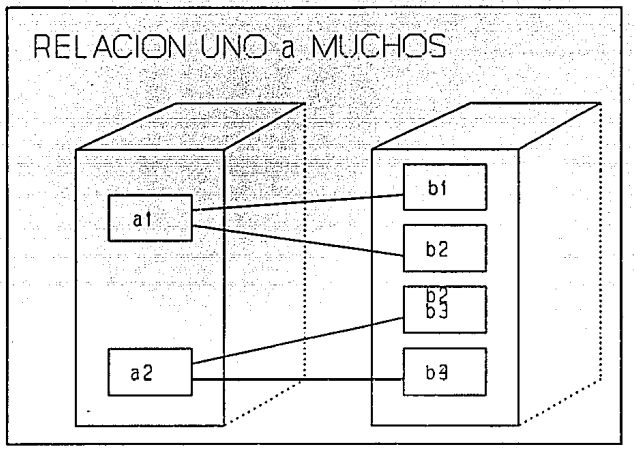
B) La entidad A esta relacionada con cualquier número de entidades en B, pero una entidad en B esta relacionada con cualquier número de entidades en A.

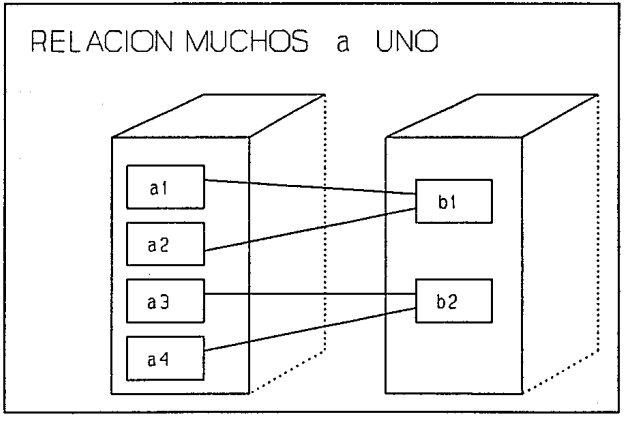
C) Una entidad en A esta relacionada únicamente con una entidad en B, pero una entidad en B esta relacionada con cualquier número de entidades en A.

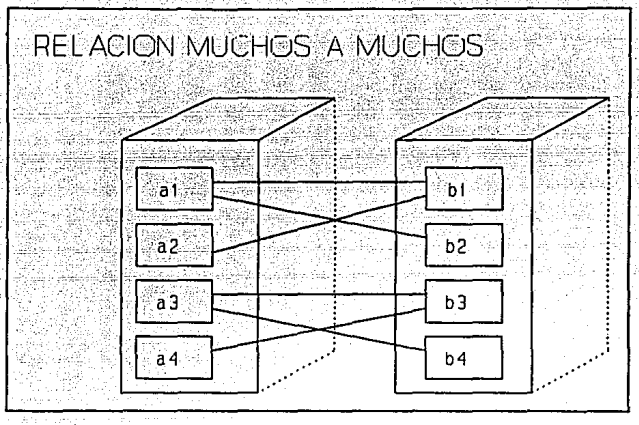
D) Una entidad en A esta relacionada con cualquier número de entidades en B, y una entidad en B esta relacionada con cualquier número de entidades en A.

Ejemplifiquémoslo de la siguiente forma :




RELACION UNO a MUCHOS
a1
a2
b1
b2
b3


RELACION MUCHOS a UNO
a1
a2
a3
a4
b1
b2



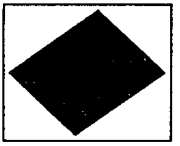
II.3. DIAGRAMA ENTIDAD RELACION

El diagrama entidad relación puede representarse gráficamente por los siguientes componentes :

Rectángulos : Que representa un Conjunto de entidades del mismo tipo (entidad).

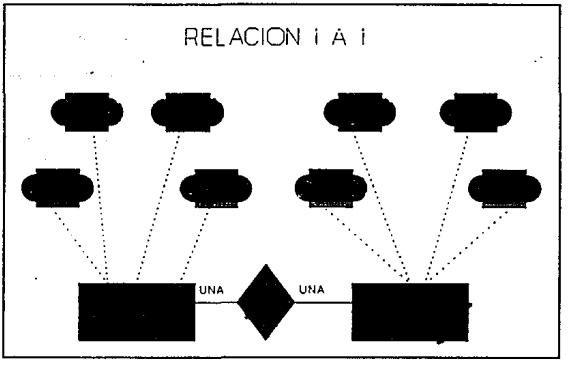
Elipses : Que representan los atributos (Campos).

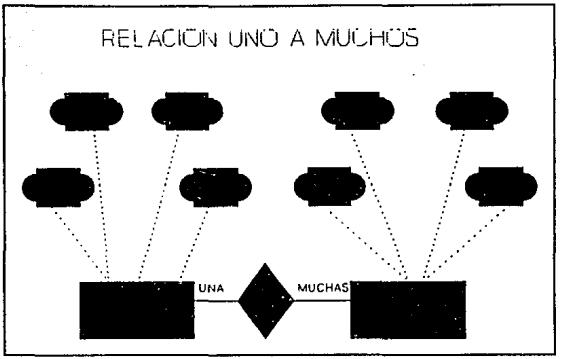
Rombos : Que representan conjuntos de relaciones (relaciones).

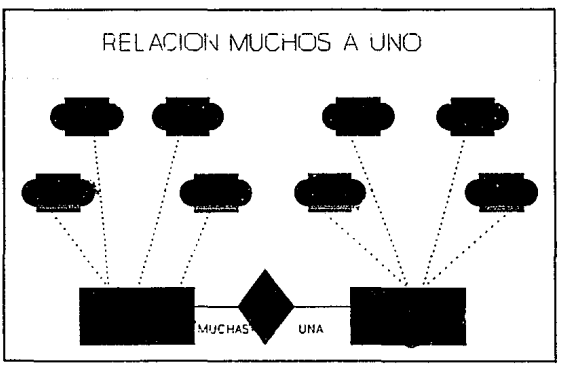


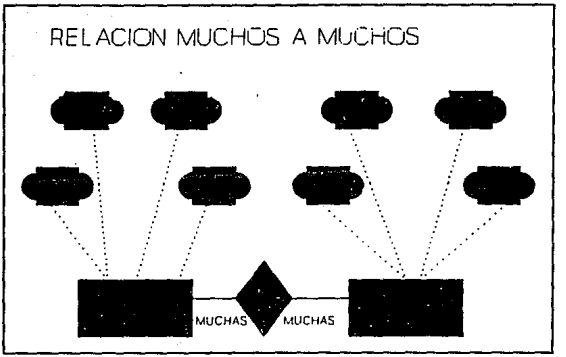
Líneas : Que conectan los componentes anteriores.

Consideremos las entidades definidas en los ejemplos anteriores e ilustremos el Diagrama entidad relación, cuatro veces es decir relación 1-1, 1-muchos, muchos-1, y muchos-muchos.




RELACION UNO A MUCHOS
UNA
MUCHAS


RELACION MUCHOS A UNO
MUCHAS
UNA



III. MODELO RELACIONAL

III.1. DEFINICION

A diferencia del modelo entidad relación, este modelo relacional consiste en un conjunto de tablas, el modelo en si esta basado en una relación, relación que se representa por medio de una tabla de dos dimensiones : los renglones que representan el conjunto de relaciones del mismo tipo y las columnas los atributos de la entidad (conjunto de entidades del mismo tipo), la tabla tiene un nombre único que es el nombre de la entidad.

III.2. ESTRUCTURA DE LAS BASES DE DATOS RELACIONAL

Como ya se mencionó una base de datos relacional consiste en un conjunto de tablas, mismas que se ejemplificarán con el ejemplo que se ha venido tratando en este capítulo, para ilustrar como quedaría :

NOMBRE	NUMERO DE CUENTA	TURNO	CARRERA
JUAN	842282 9-3	MIXTO	COMPUTACION
LUIS	869585 9-1	VESPER TINO	ELECTRONICO
PEDRO	832279 3-2	MATU TINO	CIVIL
CLAUDIA	843785 3-1	MIXTO	INDUSTRIAL
MONICA	843578 2-9	MIXTO	GEOLOGO
VERO NICA	822282 9-3	MATU TINO	GEOLOGO
. .	. .	. .	. .

NUMERO DE CUENTA	C.P.	TEL	DIRECCION
842282 9-3	04200	6670912	CERRO DEL CUBILETE 23 CAMPESTRE
869585 9-1	03217	6688722	MAGNOLIAS 1005 FLORIDA
832279 3-2	05321	7291312	CALLE 3 COYOACAN
843785 3-1	04213	6710834	FLORENCIA 76 ROMA
843578 2-9	04200	6222109	CERRO DEL CRESTON 1 CAMPESTRE
822282 9-3	06543	6542901	AV 12 PRADOS
. .	. .	. .	. .

IV. MODELO DE DATOS DE RED

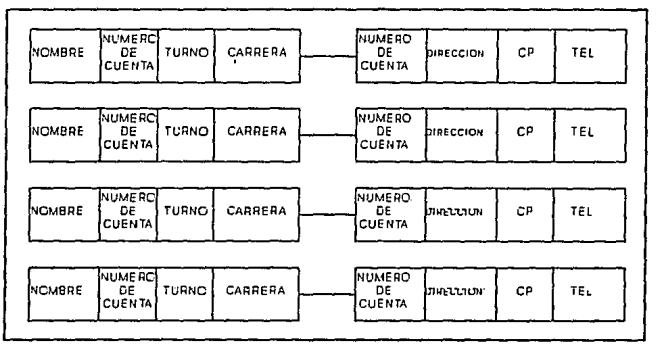
IV.1. DEFINICION

Este modelo consiste en una serie de registros conectados por medio de ligas (links). Una liga conecta exclusivamente a dos registros. Un registro es un conjunto, de atributos (campos); la información se representa por medio de conjuntos de registros y las relaciones entre los datos se representan por ligas.

IV. DIAGRAMA DE ESTRUCTURAS DE DATOS

El Diagrama de Estructuras de Datos es el esquema que representa el diseño de una base de datos reticular o de red, por medio de cuadros y líneas.

Utilizando el mismo ejemplo, (con 4 registros) pasaremos de un modelo entidad relación a un modelo de Red :



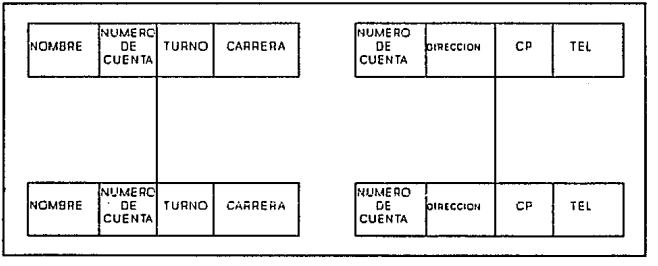
V. MODELO DE DATOS JERARQUICO

V.1. DEFINICION

Este modelo es similar al de Red, consiste también es un conjunto de registros que se conectan entre sí por medio de ligas, solo que este modelo al ser jerárquico esta formado por un conjunto de árboles de este tipo, llegando a formar un bosque, y esto implica que haya veces que el contenido específico de un registro pueda repetirse, caeríamos entonces en una inconsistencia de los datos y por supuesto existiría en el aspecto práctico, un desperdicio de memoria. Gráficamente este modelo tiene el aspecto de un árbol al reves, en donde la raíz se encuentra en la parte superior y los nodos representan las entidades.

V.2. DIAGRAMA DE ESTRUCTURA DE ARBOL

Este diagrama usa los símbolos : Cuadros y Líneas, que corresponden a registros y ligas respectivamente, este diagrama tiene la misma función que los ya vistos solo que esta organizado de forma diferente.



VI. HERRAMIENTAS CASE

La Ingeniería de Software es un "conjunto de técnicas que consideran al software como un producto que debe pasar por sus etapas de planeación, análisis y diseño, antes de su construcción, y una vez finalizado el producto, este se somete a pruebas que garanticen su calidad para proporcionarle el mantenimiento necesario durante la fase de producción ".

Ya que la función de las computadoras es facilitar cualquier trabajo de procesamiento de información, las herramientas CASE dan la posibilidad de utilizar las computadoras como auxiliares en la elaboración de los propios programas.

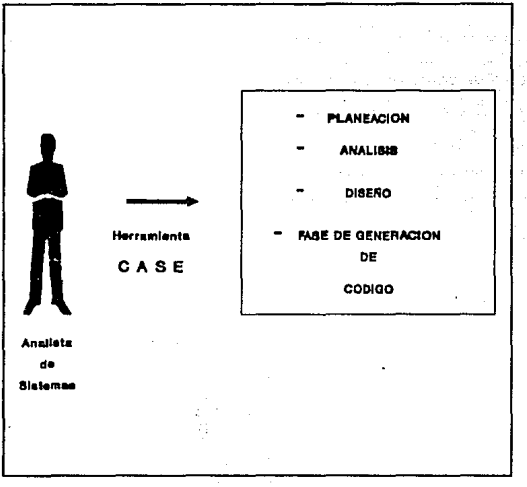
Las herramientas CASE invariablemente siguen una metodología y orden para el análisis y diseño. Muchas de las herramientas CASE incluyen la fase de generación de códigoen algún lenguaje de programación además de ayudar en la elaboración de la mayor parte de la documentación del sistema.

Normalmente se basan en un programa CASE * DICTIONARY para almacenar y estructurar la información de relevancia en las etapas de análisis y diseño. El CASE * DICTIONARY recibe la información a través de la interfase gráfica CASE * DESIGNER para de esta manera tener elementos para poder crear una base de datos que satisfaga las especificaciones dadas. En algunos casos las Herramientas CASE proporcionan una primera versión del sistema, en base a los requerimientos que le fueron proporcionados.

A pesar de que las herramientas CASE son muy poderosas, de ninguna manera pueden reemplazar el trabajo de los analistas de Sistemas, por el contrario, lo refuerzan haciéndolo más productivo al estandarizar el uso de metodologías para desarrollar sistemas, simplificando el trabajo al automatizar algunas funciones que facilitan el manejo de información, sobre todo en las etapas de análisis y diseño.

Las herramientas CASE, han sido el tipo de software que ha tenido mayores innovaciones en los últimos años. Las metodologías soportadas por herramientas CASE y las implementaciones que de estas se deriven, serán el entorno de trabajo de los Sistemas de aplicación en el futuro próximo.

Los productos CASE contemplan una o varias de las diferentes etapas del ciclo de vida de los Sistemas.



CASE obliga a los Analistas a utilizar consistentemente alguna metodología para el desarrollo de Sistemas; aparte de que le brindan una poderosa herramienta en el "ciclo de vida" del Sistema.

CAPITULO III

INTELIGENCIA ARTIFICIAL

INTRODUCCION

La Inteligencia Artificial aparece apenas en los años 50's, es realmente un área nueva pero muy moderna, en el sentido de que conduce a la automatización de tareas antes soñadas por el hombre y ahora quizás no tan lejos de ellas : Robots que realicen tareas en un mundo real por si solos, pilotos automáticos, un robot en un tumulto de gente que se confunda entre ellos, así como los vemos en las peliculas de ciencia ficción; o la participación de Sistemas Inteligentes en guerras como la del Golfo Pérsico, o la antesala para un conflicto bélico de mayores dimensiones.

De hecho muchas actividades requieren cierto grado de Inteligencia, un juego, un teorema o hasta el simple hecho de atravesar una calle, en el sentido de que deberíamos calcular (decidir) si iríamos despacio, rápido, detectar si viene algún automóvil, si voltearemos hacia todas las direcciones o solo a alguna, si nos guiaremos por el semáforo, si a pesar de que se encienda la luz roja verifiquemos si los automóviles se han detenido por completo etc; entonces si la computadora puede simular con éxito alguna de estas actividades se le debe entonces atribuir cierto grado de inteligencia. En el presente capítulo revisaremos estos planteamientos así como técnicas a seguir para la solución de problemas en la Inteligencia Artificial.

I. DEFINICION DE INTELIGENCIA ARTIFICIAL

Realmente no existe una definición universal acerca de que es la Inteligencia Artificial; sin embargo veamos algunas definiciones más conocidas:

" Inteligencia Artificial es el estudio de como lograr que las computadoras hagan cosas que, por el momento las personas hacen mejor "
Inteligencia Artificial.
Elaine Rich.
Colección Ciencias Informática.

" Inteligencia Artificial es el estudio de técnicas para resolver problemas exponencialmente difíciles en un tiempo polinomial explotando el conocimiento sobre el dominio del problema "

Inteligencia Artificial.
Elaine Rich.
Colección Ciencias Informática.

" Es una ciencia cuyo objetivo es lograr que una máquina haga todo lo que el hombre es capaz de hacer. Se trata pues, de analizar los comportamientos humanos en los campos de la comprensión, la percepción y la resolución de problemas, con el fin de poder reproducirlos a continuación por medio de una máquina "

Articulo : El Nacimiento de la Inteligencia Artificial.
Jacques Pitrat.
Revista : Mundo Científico No. 53

" Artificial Intelligence is the sciencia of making machines do things that would require intelligence if done by men "

Articulo : Principles of AI.
Minsky (1968)

" Un programa inteligente que muestra un comportamiento similar al de un ser humano que se enfrenta a un mismo problema. No es necesario que el programa resuelva, o intente resolver el problema de la misma forma que lo haría un ser humano "

Nota : El autor define la I.A. como un programa inteligente.
Utilización de C en Inteligencia Artificial.
Herbert Schildt
McGraw Hill

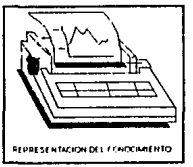
Nuestra definición será la siguiente :

Simulación del pensamiento humano a situaciones complejas que requieren una percepción y comprensión más aguda en su tratamiento; mediante el uso de las computadoras.

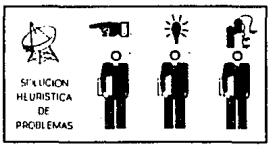
II. CAMPO DE LA INTELIGENCIA ARTIFICIAL

La Inteligencia Artificial abarca varias ramas, en donde cada una de ellas es un campo fértil así como sorprendente :

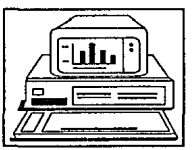
Representación del conocimiento.



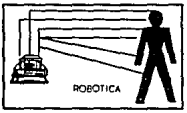
Solución heurística de problemas.



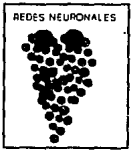
Sistemas Expertos.



Robótica.



Redes Neuronales.



REPRESENTACION DEL CONOCIMIENTO :

La representación del conocimiento implica como representar :

¿ los objetos hechos individuales ?
Es decir un solo punto en el espacio del problema.
¿ Cómo pueden combinarse las representaciones de los objetos individuales para formar una representación del estado del problema completo ?

Radica aquí el problema de la Representación del Conocimiento pese a que todavía no se tiene una solución completa, se han desarrollado técnicas que tienen un campo grande de aplicación.

Una se basa en la lógica de predicados que se discutirá en el punto III.1 Representación del Conocimiento; otra más indica que el conocimiento puede representarse como un conjunto de objetos cada uno de los cuales tiene una colección de atributos y un conjunto de relaciones con otros objetos, que discutiremos en el mismo punto.

SOLUCION HEURISTICA DE PROBLEMAS.

Muchas ocasiones existen problemas difíciles en donde no estamos garantizados a encontrar la mejor respuesta, pero si por lo menos una muy buena. Es decir esta respuesta no es la mejor pero muchas de las ocasiones pocas veces necesitamos la solución óptima; sino mas bien aquella que cumpla con nuestros requerimientos e inmediatamente se abandona la búsqueda, esto se observa dentro de lo común. Usualmente una buena aproximación nos servirá muy bien. Así una técnica heurística es aquella que ayuda a hallar soluciones de problemas (La palabra Heurística viene del griego heuriskein que significa "descubrir", lo cuál es también el origen de eureka, que se deriva de la exclamación de Arquímedes "lo encontré").

En la Inteligencia Artificial las soluciones heurísticas de problemas ayudan (algunas de ellas) a guiar un proceso de búsqueda sin sacrificar ninguna aspiración de completitud que el proceso pueda haber tenido previamente. Sin embargo otras soluciones pueden llevar a que pase inadvertido un camino excelente. Pero, en promedio, mejoran la calidad de los caminos que se exploran. De hecho utilizando buenas técnicas heurísticas, podemos esperar lograr soluciones óptimas a problemas difíciles en Inteligencia Artificial.

Una de la mejores descripciones de la importancia de las técnicas heurísticas en la resolución de problemas interesantes es How to solve it (Cómo hacerlo) [Poyla, 1957] Inteligencia Artificial. E.Rich. Ciencia Informática.

SISTEMAS EXPERTOS :

Resulta apasionante y sobre todo convincente el área de los Sistemas Expertos, son Sistemas que dan resultados prácticos a problemas complejos, o que al menos manejan una gran cantidad de información y el que la utiliza agrega su experiencia; en este caso el Experto en el área.

El Sistema Experto esta presente en muchas areas u oficios dentro del mundo real; ayudan a una mejor toma de decisiones al conjuntar el conocimiento de 1 o varios Expertos; cuando el Sistema se tiene instalado en un computador, basta copiarlo a otro computador para crear por así decirlo "un nuevo Experto", situación que en un humano necesitaría mucho tiempo para convertirse en un especialista en dicho campo, lo que hace difícil que puedan aparecer nuevos especialistas humanos.

Los Sistemas Expertos en teoría llegarían a la misma propuesta de solución que propuso el Experto en el área considerada. Un Sistema Experto esta disponible las 24 hrs aparte de que no decae su rendimiento, no se cansa, un humano merma su rendimiento después de mucho trabajar, finalmente si a usted no le simpatiza el Experto, o el Experto a usted, probablemente no le suministrará la información fidedigna; con el Sistema se evitaría estos problemas.

Un Sistema Experto imita el comportamiento humano; mas adelante en el Capítulo Sistemas Expertos definiremos formalmente el Sistema Experto, ahora solo nos compete señalarlo en este punto coma área de la Inteligencia Artificial.

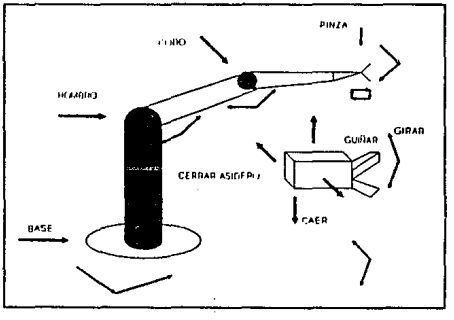
ROBOTICA :

Sin duda uno de los campos más dinámicos de la Inteligencia Artificial es la Robótica; cuando hablamos de un Robot imaginamos un mecanismo armado cuidadosamente y que se mueve por "si solo", aunque esto no es totalmente cierto; digamos que tiene algo de verdad al dejar solo a ese mecanismo y que realice lo programado; pero para que esto suceda este mecanismo debe tener una parte que interactúe con el Hadware del robot; que es el Software, es entonces la Inteligencia que existe detrás del mecanismo y justamente es esta la diferencia a otras formas de Automatización. Existen básicamente dos tipos de Robot, uno que incluye los robots fijos de montaje industrial, como por ejemplo aquellos que están presentes en el ensamblaje de automóviles, esta clase de Robots deben operar en medios altamente controlados finalmente solo fueron hechos para realizar alguna(s) labor(es) en especial. El segundo tipo son los Robots Autónomos, estos están diseñados para operar en un mundo real, donde no necesariamente habrá medios controlados aparte de innumerables situaciones que se presentarán en su dinámica de acción.

Aparte de algunos modelos muy simples, los robots autónomos como tales aún no existen. La razón es que la creación de robots auténticamente autónomos requiere la resolución de varios problemas de muy difícil solución dentro de la Inteligencia Artificial.

Examinemos un ejemplo de movimiento de un brazo que pudiera implementarse en un Robot para advertir lo complejo que puede resultar. Imaginemos el movimiento para tomar un vaso de agua, aparentemente pareciera fácil pero no es así debe existir una coordinación exacta de distintos músculos aparte de cierta velocidad, aceleración de nuestra mano, fuerza, posición, rotación, etc todo esto y más debería ser entonces analizado para llevar a cabo el movimiento. Recuerde que un niño necesita varios meses para llevar a cabo coordinadamente este movimiento.

Un brazo robótico común esta diseñado como un brazo humano. La mayoría de los brazos robóticos tiene seis ejes que le permiten una gran libertad de movimiento. Abajo en la Figura se muestra un dibujo de un brazo de seis ejes. Cada eje, al que se le denomina normalmente junta, funciona por su propio motor o, en el caso de grandes brazos, por un cilindro hidráulico.



Como se observa un brazo robótico de seis ejes contiene, en realidad, dos estructuras coordinadas. La base y las dos partes del brazo forman un sistema de coordenadas X,Y,Z y la pinza permite movimientos delicados dentro del sistema. Por tanto, encontrar cualquier punto específico en el espacio requiere el uso de cinco juntas en el brazo. El movimiento del sexto eje supone la rotación de la pinza, lo que afecta sólo a la orientación.

Si un robot autónomo ha de operar en un medio real incontrolado, necesitará la destreza que un robot industrial no requiere : por ejemplo, necesitará sensores que le permitan oír y ver, debe entender el lenguaje natural y que significa. Esto no es nada fácil. Aparte de que el robot debe ser capaz de resolver problemas, siendo esta la labor de programación más complicada. Esto es necesario para que el robot pueda adaptarse a distintas situaciones, ya que no se puede programar el robot de antemano para cualquier situación posible. El mayor obstáculo para la creación de un robot autónomo es que los programadores no han desarrollado aún, las técnicas de software necesarias, pero sin duda cada vez se aprende e investiga más para en un futuro contestar a esta y otras areas de la Inteligencia Artificial.

REDES NEURONALES.

En la década de los años 40 y a principios de los años 50 varios investigadores entre los que destacan McCulloch, W.S. y W. Pitts 1943 elaboraron modelos matemáticos de Neuronas de Redes Neuronales. En la década de los años 50 varios investigadores entre ellos Farle Clar combinaron los resultados obtenidos por los matemáticos, biólogos y los psicólogos y desarrollaron modelos de simulación de Neuronas de Redes Neuronales dando lugar a la forma actualmente más generalizada de trabajar con estos Sistemas de simulación mediante software en una computadora digital común.

Pronto se obtuvieron éxitos muy promisorios Frank Rosenblatt (1958) desarrolló el Perceptrón que fue la primera red neuronal artificial especificada con toda precisión orientada computacionalmente. Como era una máquina que podría aprender y demostrar comportamiento adaptativo complejo atrajo de inmediato la atención de los investigadores. Su procedimiento de convergencia de aprendizaje fue un avance definitivo sobre la teoría de Hebb. Asimismo Rosenblatt desechó el enfoque de teóricosanteriores que veían al cerebro como una computadora lógica. En vez de ello lo consideraron como un asociador clasificador cuya misión era la de asociar respuestas de clasificación a estímulos específicos.

En 1960 Rosenblatt publicó su libro "Principles Neurodynamics" el que presentó formalmente el Perceptrón como modelo para construir Redes Neuronales Artificiales.

Los perceptrones se aplicaron rápidamente a resolver problemas tales como la predicción climatológica, la interpretación de electrocardiogramas. Tal parecía que se había hallado la clave para comprender el funcionamiento cerebral, emulando las Redes Neuronales naturales mediante redes complejas de perceptrones.

Sin embargo pronto se comprobó que las redes con una capa de perceptrones eran incapaces de resolver problemas tan simples como la simulación de una compuerta lógica de tipo exclusivo y tras una investigación sobre las limitaciones de los perceptrones, Minsky,M. y S. Papper publicaron el libro Perceptrons, MIT Press, Cambrifge, Mass, 1972 donde se hacian patentes estas limitaciones. Como consecuencia los fondos para nuevas investigaciones reorientaron su objeto de estudio.

La neurona artificial.

La neurona artificial es una unida procesadora con cuatro elementos funcionales :

1.- El elemento receptor a donde llegan una o varias señales de entrada Xi que generalmente provienen de otras neuronas que son atenuadas o amplificadas cada una de ellas con arreglo a un factor de peso Wij que constituye la conectividad entre la neurona fuente de donde provienen la neurona de destino en cuestión.

2.- El elemento sumador que efectúa la suma algebraica ponderada de las señales de entrada ponderandolas de acuerdo con su peso.

3.- El elemento de función activadora que aplica una función no lineal de umbral, que frecuentemente es una función escalón o una curva logística a la salida del sumador para decidir si la neurona se activa disparando una salida o no.

4.- El elemento de salida que es el que produce la señal de acuerdo con el elemento anterior que constituye la salida de la neurona.

Este modelo neuronal es el utilizado en casi todas las Redes Neuronales artificiales variando únicamente el tipo de función activadora.

A continuación se presentaran los modelos mas simples de redes neuronales artificiales.

El Perceptrón

La red neuronal más simple construida con perceptrones tiene dos capas una capa receptora de entrada en la que la salida de cada neurona reproduce simplemente su entrada, una capa de salida formada por perceptrones totalmente conectados con la capa de entrada a través de líneas de comunicación con conductividades o pesos ajustables.

Así cada neurona de entrada esta conectada con cada neurona de salida a través de una línea de comunicación con conductividad o peso ajustable. La Red de aprendizaje del perceptrón ajusta estos pesos de manera que se obtenga con mayor probabilidad la salida deseable correspondiente a un cierto conjunto de entradas.

Redes de Hopfield

En 1982 John Hopfield introdujo un nuevo tipo de redes que posteriormente se llamaron redes de Hopfield.

La red de Hopfield elemental tiene una sola capa de neuronas pero todas éstas están conectadas entre si mediante arcos dobles ponderados que van en ambas direcciones. Cada peso puede ser positivo ó negativo.Esta topología doble de interconexión convierte a la red en una red retroalimentada o recursiva.

Cada una de las neuronas tiene una salida que puede estar en uno de dos estados X o Y. La salida es 1 si la suma ponderada de las salidas de las otras neuronas es mayor que un umbral especificado T, en caso contrario la salida es 0.

Toda la red esta entonces en un cierto estado a cada momento definido por un vector de estados cada uno de cuyos elementos corresponde a una de las n neuronas. Los estados de la red se pueden también representar como los vértices de un hipercubo de n dimensiones siendo n el número de neuronas. Una de las aplicaciones de la red de Hopfield es la de servir como memoria asociativa.

Una memoria asociativa es aquella en que una fracción de una señal es capaz de reproducir la totalidad de la señal. Por ejemplo con una fracción de una melodía el cerebro humano puede reconocer y reproducir la melodía completa. Las memorias asociativas son útiles para regenerar señales incompletas o deterioradas por ruido.

La inteligencia artificial finalmente intenta simular el pensamiento humano donde es posible representar su razonamiento lógico pero jamas reemplazara a este.

III. TECNICAS DE LA INTELIGENCIA ARTIFICIAL

III.1. REPRESENTACION DEL CONOCIMIENTO

III.1.1. REPRESENTACION DEL CONOCIMIENTO MEDIANTE LOGICA DE PREDICADOS

Cuando tratamos los problemas de la Inteligencia Artificial se necesita a la vez una gran cantidad de conocimiento y algunos mecanismos para manipular dicho conocimiento a fin de poder crear soluciones a dichos problemas. Existen algunos métodos muy generales de manipulación del conocimiento mediante la búsqueda, pero supongamos que representamos cada nodo como una lista de todos los fragmentos de la descripción de ese nodo (un nodo representa un punto en el espacio del problema), entonces que sucede en el proceso de búsqueda si estas descripciones ¿ son muy largas ? Se agotaría rápidamente la memoria RAM o desperdiciaríamos inadecuadamente Memoria Secundaria. Entonces aunque estos métodos son útiles y serán el esqueleto de muchos otros métodos, su potencia esta limitada precisamente a causa de su generalidad. Los Sistemas de Inteligencia Artificial han aprovechado diversas formas de representación del conocimiento (hechos).

* Hechos : Verdades en algún mundo relevante, esto es lo que queremos representar.

* Representación de los hechos, esto es lo que realmente podemos manipular.

Existe una representación de hechos muy usual : las frases en lenguaje natural, ya que aparte de la representación de hechos que se realice en el programa, también no debe dejarse a un lado la representación en lenguaje natural para facilitar la entrada y salida de información en el Sistema.

Ejemplo :

Usando lógica matemática.

Dodi es un perro.	Dodi tiene cola.
Perro(Dodi).	Tienecola(Dodi).

El formalismo lógico es atractivo puesto que sugiere inmediatamente una potente forma de deducir nueva información a partir de la antigua; por ejemplo si se que Dodi es un perro y Dodi tiene cola, deduzco que todos los perros tienen cola, podemos entonces afirmar que esta nueva sentencia es cierta demostrando que fue deducida de las dos sentencias anteriores.

Podemos representar fácilmente lo que sucede en el mundo real mediante proposiciones lógicas como usualmente se dice formulas bien formadas (fbf) mediante lógica de predicados. La lógica de predicados proporciona una forma de deducir nuevas declaraciones a partir de la viejas. Sirve aún como forma útil de representar y manipular algunas clases de conocimientos que podrían necesitar los Sistemas de la Inteligencia Artificial.

Un buen Sistema que represente el conocimiento estructurado y complejo en un dominio concreto debería poseer las siguientes propiedades :

Adecuación representacional : La capacidad de representar todas las clases de conocimiento que se necesitan en ese dominio.

Eficiencia Inferencial : La capacidad de manipular las estructuras representativas de forma que puedan derivarse nuevas estructuras correspondientes a conocimiento nuevo a partir de las anteriores.

Eficiencia adquisional : La capacidad de adquirir nueva información con facilidad. El caso más simple es el de una persona que inserta directamente el nuevo conocimiento a la base de datos. Idealmente, el programa mismo debería ser capaz de controlar la adquisición del conocimiento.

Para cumplir estos puntos se han desarrollado varias técnicas, básicamente podemos dividirlas en dos :

Métodos Declarativos ---> Lógica de Predicados

La mayor parte del conocimiento se representa como una colección estática de hechos junto con un pequeño conjunto de procedimientos generales para manipularlos.

Ventajas de la representación declarativa :

- Cada hecho solo necesita almacenarse una vez, sin importar el número de formas diferentes en que puede usarse.
- Es fácil añadir nuevos hechos al Sistema sin cambiar los otros hechos ni los procedimientos pequeños.

Otra técnica son los Métodos Procedurales, estos serán discutidos en el punto III.1.3. Representaciones Estructuradas del Conocimiento; pero es importante notar que muchas áreas necesitan ambas, de hecho se combinan bastante; en ese mismo punto regresaremos un poco a Lógica de predicados.

III.1.2. REPRESENTACION DEL CONOCIMIENTO USANDO OTRAS LOGICAS

Las técnicas de la lógica de predicados así como son útiles para solucionar problemas en muchos dominios en otros no proporcionan una adecuada manera de manejar y manipular la información importante. Por ejemplo :

" Pedro dice que los acereros de pittsburgh perderán, pero yo creo que ganaran "

¿ Cómo se puede entonces representar ambas creencias ?

" Hoy hace mucho frío "

¿ Cómo representaríamos los grados para medir correctamente el frío ?

Mucho del razonamiento de las personas esta basado en su conjunto de lo que se supone (creencias). Cada suposición la basa en evidencias. Entonces para permitir que programas manipulen conjuntos de creencias, debe construirse un algoritmo de razonamiento que pueda manejar el conjunto de información anterior.

Mucha información puede provenir de conocimientos inciertos y difusos. Existen varias técnicas para manipular estos problemas:

* Lógica NO monótona. Que tolera que una sentencia sea eliminada además de añadida de la base de datos, es decir que si afirmo algo, es porque tengo una falta de creencia en la otra.

* Razonamiento probabilístico. Que permite representar inferencias confusas pero posibles.

* Lógica difusa. Que facilita una forma de representar propiedades difusas ó contiguas de los objetos.

* Espacios de creencias. Que permite la representación de modelos anidados de conjuntos de creencias.

En este capítulo solo discutiremos las dos primeras, la tercera, no se ha explotado mucho en la Inteligencia Artificial; la cuarta depende mas bien de otros mecanismos para su representación.

III.1.2.1. RAZONAMIENTO NO MONOTONO

Muchas ocasiones un Sistema no llega a tener toda la información posible del área tratada. Cuando esto sucede, pueden hacerse algunas suposiciones sensatas mientras no se presente evidencia contradictoria. El construir suposiciones se llama Razonamiento por defecto dentro del razonamiento no Monótono. Como la elección más probable; por ejemplo :

Razonamiento por defecto : Definición 1
Si no se puede demostrar X, concluir Y

Si estuviéramos en lógica de predicados diríamos, ¿ Cómo sabríamos estrictamente que X no puede demostrarse ?, tendríamos entonces :

Razonamiento por defecto : Definición 3
Si X no puede demostrarse en una determinada cantidad de tiempo, entonces concluir Y

Notese que ahora el concluir Y depende de cuanto proceso computacional en términos de Informática puede hacerse en determinado tiempo; esto por su puesto ya no es tan estrictamente lógico como el primero.

Hemos perdido la capacidad que teníamos en lógica de predicados de verificar la corrección de una demostración propuesta, aún cuando no siempre pudiésemos garantizar el hallazgo de dicha demostración si esta existiese. La necesidad del razonamiento por defecto, que surge de una falta de información completa, nos fuerza a usar Sistemas cuya conducta no puede caracterizarse fácilmente de una manera formal.

Sin embargo suponiendo que tuviéramos la información completa sobre una situación dada, es poco probable que continuáramos por mucho tiempo, puesto que el mundo cambia constantemente. Esto significa que asertos que eran totalmente precisos tal vez más adelante ya no lo serán. Otra forma de resolver el problema sería eliminar las sentencias cuando ya no describen con precisión la solución y añadir aquellas que ahora se ajusten.

Finalmente un Sistema de Razonamiento no Monótono puede ser necesario por cualquiera de las siguientes razones :

- La presencia de información incompleta requiere razonamiento por defecto.
- Un Mundo cambiante debe describirse mediante una base de datos cambiante.
- La generación de una solución completa para un problema puede requerir suposiciones temporales sobre soluciones parciales.

III.1.2.2. RAZONAMIENTO ESTADISTICO Y PROBABILISTICO

Hasta ahora hemos visto que hay hechos que son ciertos, que no lo son, o que definitivamente no sabemos nada acerca de ellos. Pero también podemos considerar algún hecho que sea "probablemente cierto".

Hay tres clases de situaciones en las que podemos usar el Razonamiento probabilístico.

- El mundo relevante es realmente aleatorio. Por ejemplo, el número de boletos que entrarán en un torniquete ó el número de autos que pasaran por cierto crucero.

- El mundo relevante no es aleatorio dada la suficiente cantidad de datos, pero nuestro programa no siempre tendrá acceso a todos esos datos. Por ejemplo la probabilidad de éxito que un Televisor se vea en todos sus canales en cualquier parte de la república.

- El mundo parece aleatorio porque no lo hemos descrito en el nivel conveniente.

III.1.3. REPRESENTACIONES ESTRUCTURADAS DEL CONOCIMIENTO

En el inciso III.1. mencionamos que se trataba de representaciones declarativas; ahora definiremos el tipo de representación que trataremos : La Representación Procedural en donde la mayor parte de los conocimientos se representan como procedimientos para usarlos.

Ventajas de la Representación Procedural :

- Es fácil representar el conocimiento sobre como hacer cosas.

- Es fácil representar el conocimiento que no encaja bien en muchos esquemas declarativos simples.

- Es fácil representar conocimiento heurístico de cómo hacer las cosas eficientemente.

Una de las razones por que las Estructuras de Conocimiento son tan importantes, es que proporciona una forma de representar la información sobre modelos de cosas que ocurren usualmente. Es decir Esquemas :

Un esquema es una organización activa de reacciones o experiencias pasadas que deben suponerse que funcionan en cualquier respuesta orgánica bien adaptada.

En los índices anteriores se mencionaron algunos métodos para ilustrar el conocimiento usando formalismos lógicos. La característica principal (en especial la lógica de predicados) es que pueden combinarse con mecanismos de inferencia potentes. En este inciso se mencionan diversas estructuras de conocimiento.

El término Estructuras de Conocimiento indica unas veces una base de datos completa de información sobre un dominio concreto.

Una razón importante para que las estructuras de Conocimiento sean relevantes radica en el hecho de que gracias a ellas se puede representar la información sobre modelos de cosas que ocurren usualmente. Estas descripciones son llamadas esquemas (Un esquema es una organización activa de reacciones o experiencias pasadas que deben suponerse que funcionan en cualquier respuesta orgánica bien adaptada [Barlett, 1932, p 261]).

Diversos tipos de esquemas son útiles en la Inteligencia Artificial :

* Frames (Armazones).
* Scripts (Guiones).
* Estereotipos.
* Modelo de Reglas.

Hablemos primero de los Estereotipos y Modelo de Reglas, los dos primeros se incluirán en el siguiente inciso.

* Estereotipos. Se han usado como base para implementar modelos de usuarios individuales de un Sistema interactivo. Describir grupos de características que frecuentemente encontramos juntas en la gente.

* Modelos de Reglas. Usados para mejorar la capacidad de un Sistema Experto en lo que se refiere a la interfase con el usuario.

Tanto los Frames como los Scripts se incluirán también como representaciones declarativas pertenecientes a estructuras del conocimiento.

III.1.3.1. REPRESENTACIONES DECLARATIVAS

Existen básicamente 4 mecanismos declarativos que representan el conocimiento :

* Redes semánticas.
* Dependencia Conceptual.
* Frames (Armazones)
* Scrips (Guiones)

* Redes Semánticas.

Las Redes Semánticas son bastante generales para poder describir a la vez acontecimientos y objetos. En una red semántica, la información se representa como un conjunto de nodos conectados entre sí por medio de arcos etiquetados que significan relaciones entre los nodos. La potencia de una Red semántica radica en la capacidad de los programas para manipular dichas redes a fin de dar solución a los problemas. Las aplicaciones últimas de las Redes Semánticas han usado procedimientos de búsqueda más dirigidos a responder cuestiones particulares.

* Dependencia Conceptual.

La Dependencia Conceptual es una teoría de como podemos llegar a representar el significado de frases del lenguaje natural; implicando facilidad para sacar conclusiones de las frases. Existe independencia del lenguaje en que se encontraban las frases originalmente.

A diferencia de las Redes Semánticas en dónde solo proporcionan una estructura en la cual pueden colocarse los nodos que representan la información en cada nivel, la Dependencia Conceptual proporciona al mismo tiempo, una estructura y un conjunto específico de primitivas a partir de las cuales pueden construir representaciones de elementos concretos de información.

* Frames.

Dentro de la generalidad se observa que la gente no analiza nuevas situaciones empezando desde cero y al mismo tiempo construyendo nuevas estructuras de conocimiento para describir esas situaciones. Ahora se tiene una colección de estructuras que simbolizan sus experiencias anteriores con objetos, lugares y situaciones. De tal forma que para analizar una nueva experiencia, invocan las estructuras almacenadas y ahí generan nuevos detalles con la experiencia actual. El mecanismo común es el Frame (Armazón). Un Frame describe objetos como mesa, cajón, etc. Consiste en una colección de ranuras que describen el aspecto de los objetos. Cada ranura puede llenarse también por un valor por defecto.

* Scripts.

Al igual que en la representación de hechos individuales, los grupos de hechos pueden poseer estructuras útiles de propósito especial que exploten propiedades específicas de su dominio restringido. Tal estructura es el Script. Un Script es una estructura que describe una secuencia estereotipada de acontecimientos en un contexto particular.

III.2. SISTEMAS DE PRODUCCION

Un Sistema de producción es útil para estructurar programas de Inteligencia Artificial que faciliten la búsqueda ya que esta forma el núcleo de muchos procesos inteligentes.

Mencionemos algunas definiciones :

- Un Conjunto de reglas, cada una de las cuales consta de una parte izquierda (un modelo) que determine la aplicabilidad de la regla, y de una parte derecha que describa la acción que debería realizar si se aplicase la regla.

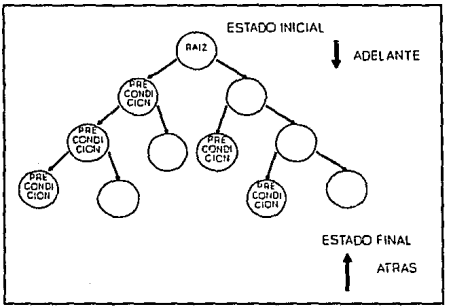
- Una o más bases de datos que contengan cualquier información apropiada para la tarea a realizar. Algunas partes de la base de datos debe estar estructurada del modo apropiado.

- Una estructura de control que especifica el orden en el que se compararán las reglas con la base de datos y una manera de resolver los conflictos que surgen cuando diversas reglas casan a la vez.

Por el hecho de estructurar el Sistema, esto nos permite añadir nuevas reglas sin alterar las otras estructuras del Sistema considerando que las agregamos a una estructura ó modulo. Esta es una gran ventaja de los Sistemas de producción.

III.3. TIPOS DE RAZONAMIENTO (HACIA ADELANTE - HACIA ATRAS)

Dentro de los Sistemas de producción el proceso de búsqueda que se puede realizar puede ser hacia adelante (desde los estados iniciales) y hacia atrás (desde los estados finales).



Cuando existe un razonamiento hacia adelante las precondiciones es decir la parte izquierda se compara con el estado inicial y las partes derechas se utilizan para generar nuevos nodos, ya que esta parte derecha es un resultado, es decir un nodo terminal al que se le pueden generar nuevos nodos.

En cambio para razonar hacia atrás es " al contrario " las partes derechas se comparan con el nodo actual y las partes izquierdas se utilizan para generar nuevos nodos que finalmente nos llevan al nodo estado-meta a alcanzar.

III.4. BUSQUEDA Y FUNCIONES HEURISTICAS

Hablamos ya de Solución Heurística de problemas en la Sección II. Campo de la Inteligencia Artificial; ahora lo revisaremos más estrictamente ya que para los Sistemas Expertos es muy útil, al estar muy relacionado el que la solución óptima, es para muchos expertos tal vez no la mejor pero si la que resuelve el problema, muchas ocasiones no queremos la mejor solución para que esto o aquello funcione, si no aquella que simplemente lo haga funcionar.

De hecho alguna evidencia muestra que la gente, al resolver problemas, no suele optimizar sino más bien satisfacer los requerimientos [Simon, 1981] Inteligencia Artificial pag.44. E.Rich.

Es decir buscar aquella solución que cumpla nuestros objetivos y tan pronto la tenemos abandonar la búsqueda.

Dado un problema, averiguar si ya se han resuelto problemas similares y como se solucionaron, para ver si tal vez nos sirvan estas soluciones de problemas, aunque no estemos seguros de que siempre nos llevará en la dirección correcta.

Existen 2 formas en las que la información heurística puede añadirse a un procedimiento de búsqueda basado en reglas :

* En las reglas mismas.
  Por ejemplo : En el ajedrez definimos un conjunto de movimientos legales, pero también un conjunto de movimientos sensatos.

* Como una función heurística que evalúe cada estado del problema y determine hasta que punto es aceptable.

El propósito entonces de una función heurística es guiar el proceso de búsqueda en la dirección más provechosa.

Una función heurística representa generalmente con números las medidas de deseabilidad. Las funciones heurísticas llevan a un proceso de búsqueda de la solución " exitosa ". La función heurística guía el proceso de búsqueda hacia la dirección más provechosa.

La Heurística (búsqueda) es el análisis de soluciones mediante búsqueda inteligente. Entonces con la " búsqueda inteligente " o heurística nos dirigimos hacia soluciones factibles y deseables en forma rápida.

IV. SISTEMAS EXPERTOS

INTRODUCCION

Los Sistemas Expertos como ya se menciono son parte de la Inteligencia Artificial, estos de ninguna manera pretenden substituir ni al trabajador común y mucho menos al profesional, si no por el contrario brindar un apoyo a las actividades que este realice. Los Sistemas Expertos, son almacenados en la computadora y posteriormente brindan la oportunidad de articular ese conocimiento a partir de una estructura de razonamiento inteligente, en donde se busca reproducir el esquema de razonamiento de los especialistas. Si se diseñan e implementan adecuadamente, los Sistemas Expertos proporcionan a los empleados empresariales complementos (apoyo) de gran poder para sus programas convencionales. Así mismo simulan el proceso de razonamiento de los Expertos en la solución de problemas específicos, ofreciendo explicaciones respecto a sus inferencias, manejando un dominio de complejidad real que normalmente requiere de una gran cantidad de conocimientos y experiencias. La importancia de construir Sistemas Expertos son varias; los programas pueden realizar tareas de manera más rápida y barata, la capacitación de un solo especialista significa una gran inversión y tiempo, y un Sistema Experto puede ser difundido dentro de toda organización, al mismo tiempo.

Se puede formalizar el conocimiento explícito que puede ser duplicado, enseñado, criticado, adecuado y aumentado; se reduce el tiempo de inoperabilidad debido a la ausencia de expertos, lo que se refiere a que generalmente los especialistas no están todo el tiempo junto a los procesos o problemas, y casi siempre los problemas surgen cuando los especialistas no se encuentran en la organización y es el momento en el que se toman las malas decisiones. Con la utilización de Sistemas Expertos se evitan esas malas decisiones que siempre vienen a resultar contraproducentes para la organización. Se puede llevar el conocimiento de un Sistema Experto hacia lugares cuyas condiciones sean peligrosas para el ser humano : de hecho todos los robots son manipulados a través de cierto software que a fin de cuentas es un Sistema Experto. Los Sistemas Expertos son didácticos es decir se puede con facilidad aprender de ellos y no es necesario que el Experto se la pasa capacitando ya que por lo general es gente que está ocupada y que difícilmente tendrá tiempo de capacitar a otras personas.

Ahora bien, es importante señalar que no todos los problemas nos los van a solucionar los Sistemas Expertos es necesario delimitar un dominio de problemas que pueden resolver. Finalmente los Sistemas Expertos brindan explicaciones lo que no hacen los Sistemas convencionales, un Sistema Experto como se está avocando hacia problemas más críticos tiene la necesidad de dar explicaciones.

IV. DEFINICION DE SISTEMAS EXPERTOS

Existen varias definiciones en relación a los Sistemas Expertos, todos ellos nos conducen a intuir la misma deducción acerca de lo que es un Sistema Experto, enunciemos algunas de ellas :

- Son Sistemas capaces de almacenar la experiencia y el saber hacer (Know how) de uno o de varios especialistas humanos. (Apuntes. Clase Inteligencia Artificial Dr. Felipe Lara).

- Una estructura de programación capaz de almacenar y utilizar algún tipo de conocimiento sobre una determinada área. (Informática Nafin Año 2, Noviembre 1991 21)

- Son Herramientas empresariales que ayudan a manejar la calidad y la disponibilidad del conocimiento que requieren los profesionales de la toma de decisiones en una amplia variedad de industrias (Periódico World Computer).

Definamoslo de la siguiente manera :

Un Sistema que intenta simular el pensamiento de uno ó más Expertos en una área determinada y que se fundamenta en una base de conocimientos donde se almacena y utiliza la información (conocimiento).

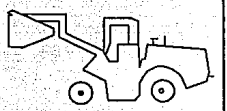
Entonces un Sistema Experto debe permitir básicamente 2 cosas:

1) Almacenar información como si fuese una base de datos, sobre el área considerada.

2) Tener la facultad de utilizar esa información para extraer resultados que no existían anteriormente dentro del Sistema, es decir inferir de esta información nueva información.

(inferir significa Deducir una cosa de otra, ejemplo :
por su aspecto infiero que no tiene la edad que dice)

CAPACIDAD DE ALMACENAR



CAPACIDAD DE UTILIZACION

SISTEMA EXPERTO

CAPACIDAD DE INFERENCIA

CAPACIDAD DE APRENDIZAJE

IV.2. INGENIERIA DEL CONOCIMIENTO

Cuando hablamos de Ingeniería del Conocimiento, nos referimos al hecho de aquel ó aquellos especialistas que se harán cargo de realizar entrevistas estructuradas a todos aquellos Expertos en el área considerada, hay que recordar que los Expertos son aquellas personas que están manejando los diferentes problemas de una organización; gente que afronta problemas y es capaz de resolverlos; tienen entre sus características, que reconocen y formulan el problema, lo resuelven rápidamente, explican su solución, son capaces de aprender de su experiencia, reestructuran el conocimiento y determinan la relevancia de las cosas; por eso es tan importante la consulta de la Ingeniería del conocimiento; el Ingeniero del Conocimiento vera la manera en que la información llegue a la base de datos, de hecho es codificada manualmente por este, como un programa tradicional.

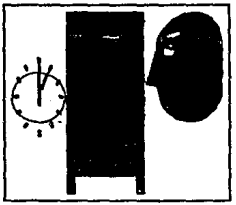
Hay ocasiones en la que los Expertos se enfrentan a problemas que les presentan una gran cantidad de variables y si el experto debe analizar sus valores y la correlación que tienen entre sí, para cuando llegue a una solución será demasiado tarde. El Experto, por lo general, tiene que tomar decisiones en cuestión de segundos, y para tomar una buena decisión tiene que trabajar con información incompleta y en ocasiones incierta; el Ingeniero del Conocimiento debe ser capaz de descubrir cuáles son los elementos más relevantes, cuáles las correlaciones más importantes entre las variables para que el Sistema Experto al igual que el Experto tome una decisión adecuada.

Ahora bien, muchos Expertos pueden colaborar en la creación de un Sistema Experto, de manera que el conocimiento almacenado pueda ser mayor.

IV.3. AREAS DE LOS SISTEMAS EXPERTOS

Los Sistemas Expertos estarán presentes en todas aquellas áreas en donde por lo menos exista un Experto en esa área. Los Sistemas Expertos fundamentalmente se orientan a problemas de interpretación en la que se busca, a partir de ciertos datos, buscar otros de un nivel conceptual mayor a los primeros. Se encuentran en :

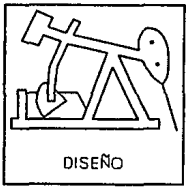
La predicción



El diagnóstico :

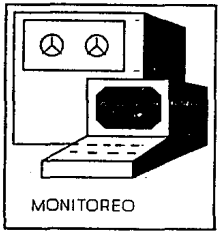
Clínico
Automotríz
Naval
Militar
etc

El diseño



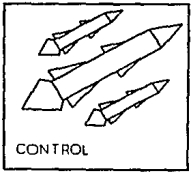
Planeación de la producción

Monitoreo



Enseñanza :

Control :



IV.4. ELEMENTOS BASICOS DE LOS SISTEMAS EXPERTOS

IV.4.1. BASE DE CONOCIMIENTOS

La base de conocimientos es una base de datos que posee una información y unas reglas especificas sobre una materia determinada. Hay dos términos a conocer :

OBJETO : La conclusión que es definida por sus reglas asociadas.

ATRIBUTO : Una cualidad especifica que con su regla, ayuda a definir el objeto.

Entonces considere a la base de conocimiento como una lista de objetos con sus reglas y atributos asociados.

En el sentido más simple (y para muchas aplicaciones), la regla que se aplica a un atributo determina si un objeto <<tiene>> o <<no tiene>> dicho atributo. Así entonces se puede definir un objeto usando una lista de atributos que el objeto posea o no.

La base de conocimientos no es otra cosa que la biblioteca de reglas que el Sistema Experto utiliza cuando toma una decisión u ofrece una solución al problema. De hecho existe un Sistema diseñado especificamente para manipular información de la forma particular que tiene la base de conocimientos. Las bases de conocimientos se describen tradicionalmente como grandes sistemas de instrucciones :

if then else
do while
case etc,

pero tal situación puede ser engañosa es posible que las bases de conocimientos no contengan reglas definitivas, sino muchos otros objetos :

Relaciones asociativas entre diferentes conceptos

Información estadística acerca de la posibilidad de ciertas soluciones.

Grandes bases de datos que se pueden comparar una con otra.

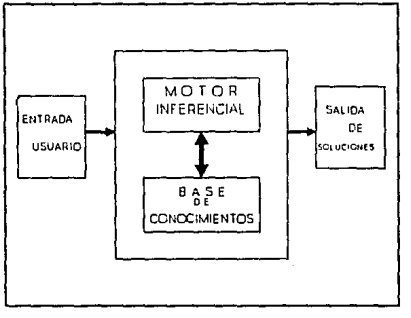
La manera en que la información llega a la bases de conocimientos depende de cada Sistema Experto. Por lo general es codificada manualmente por los Ingenieros del conocimiento, quienes realizan entrevistas estructuradas como ya se menciono.

El propósito fundamental de la base de conocimientos es proporcionar el Sistema Nervioso del Sistema Experto : las conexiones entre ideas, conceptos y probabilidades estadísticas que permiten a la parte razonada del Sistema realizar una precisa evaluación de un problema, activo o inversión.

IV.4.2. MOTOR DE INFERENCIA

La función del Motor de Inferencia es la de justamente accesar la Base de Conocimientos que acabamos de mencionar, de tal forma que ante preguntas del usuario el Sistema ofrezca respuestas aceptables, es decir, se ocupa de tomar un conjunto de reglas y realizar inferencias para encontrar una solución; consecuentemente el Sistema Experto debe tener capacidad para :

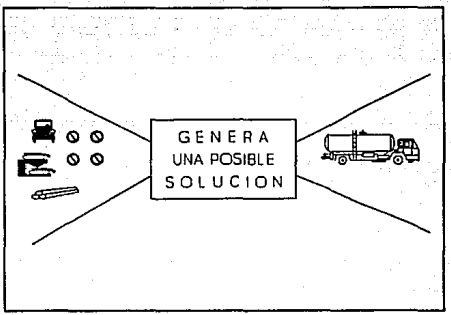
- Completar la información almacenada.
- Utilizar nueva información que necesite de la base de conocimientos para hacer inferencia.



Podemos generalizar un poco y considerar dos tipos de Sistemas Expertos :

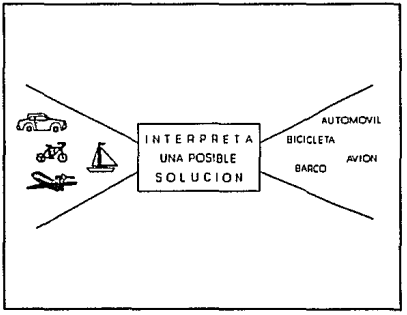
1) Sistemas Expertos Generativos :

Realizan tareas de tipo " Creativo " en su mínimo significado; se les intenta introducir los máximos niveles de inteligencia para que dependan lo menos posible de las directrices fundamentales de cada caso introducidas por el hombre. La información que utilizan, está compuesta por la que le introduce el usuario para cada problema y por la que ya tiene almacenada en la base de conocimientos.



2) Sistemas Expertos Interpretativos :

Su forma de trabajo es menos original que la de los de tipo generativo. La información que manejan proviene básicamente de la que el ordenador posee almacenada es decir la base de conocimientos. Son Sistemas de tipo operativo y consultivo.



Finalmente entonces el motor de inferencia es un algoritmo que establece la correspondencia entre hechos conocidos y premisas de reglas, con el objetivo de ejecutar reglas y deducir nuevos hechos.

IV.4.3. DISEÑO DE INTERFASES

Tradicionalmente, los diseños de interfases para los Sistemas Expertos han dependido de capacidades gráficas y de métodos poco convencionales para introducir datos en el Sistema. Las interfases gráficas pueden proporcionar información de muy diversas maneras : simples textos dentro de ventanas, menúes instantáneos, o con verdaderos objetos gráficos diseñados para representar cosas como información estadística o el flujo de la lógica.

Muchos de estas formas de representar la información al usuario han sido integradas en las aplicaciones convencionales, pero en lo que se refiere a Sistemas Expertos tienen un uso particular, por diversas razones : Ayuda a fluir el lenguaje natural, las lógicas tratadas anteriormente así como la interfase pura con el usuario.

El Sistema Experto expresa una idea, una solución o una explicación y sería ideal que hubiera una interfase gráfica más sofisticada que un mensaje, que por supuesto no deja de ser útil y práctico.

Como ejemplo de una interfase gráfica podemos mencionar el dibujo de un mapa hecho por un Sistema Experto que proporciona direcciones o análisis gráficos, o la expresión de relaciones financieras por medio de complejas gráficas organizadas dinámicamente para enfatizar un punto en especial; por supuesto debe comprenderse que existen aplicaciones que se prestan para una interfase gráfica hasta luciente podríamos decirlo, pero en otras no es así. Internamente, la interfase de usuario no limita las capacidades del Sistema Experto para "razonar"; solo mejora la capacidad del usuario y de la computadora para comunicarse entre sí y perfeccionar la compresión de las evaluaciones por parte de los usuarios.

En suma debe por supuesto intentar realizar una interfas adecuada e ilustrativa para la comunicación con el usuario, finalmente para el y por el esta hecho el Sistema.

IV.5. SHELL DEL SISTEMA EXPERTO

El Shell del Sistema Experto proporciona un estrato entre la interfas de usuario y el Sistema operativo de la computadora para manejar la entrada y salida de datos.

Manipula también la información proporcionada por el usuario, conjuntamente con la base de conocimientos, para llegar así a una determinada conclusión.

La estructura de la concha es muy similar a la de un intérprete o módulo frontal de un programa de bases de datos, pero más rica y más capaz de tomar los datos, aserciones y condiciones contenidos en la base de conocimientos y aplicarlos a los datos que se introducen.

La concha maneja también la interfase de usuario, ejecutando funciones que van desde la validación de valores numéricos introducidos en la pantalla, hasta el manejo del ratón y la representación de objetos gráficos. Con frecuencia, la concha se vende como un producto final, permitiendo que el codifique una base de conocimientos desde el principio, de la misma manera que un usuario comprará una base de datos para almacenar información proporcionada por su compañía.

Por otra parte, la base de conocimientos se pueden vender como productos de la misma manera que un usuario puede comprar datos, tales como El mundo salvaje, Geografía Universal etc. En este caso, el usuario aprovecha de inmediato la biblioteca de juicios que proporciona ya el Sistema.

CAPITULO IV

ELABORACION DEL SISTEMA EXPERTO

I. PLANTEAMIENTO DEL SISTEMA

El Soporte Técnico en las áreas de Informática se desarrolla de entre algunas de sus funciones más importantes, brindando Servicio Técnico (Software) a los usuarios de microcomputadoras, cuya población la constituyen usuarios ubicados en distintas instalaciones geográficas : dentro y fuera de la cd. de México, en su forma más amplia.

El número de usuarios es susceptible de aumentar lo mismo que el número de equipos; implicando como consecuencia un aumento de requerimientos del Servicio Técnico; lo cual nos conduce a pensar que se requieren cada vez más recursos (humanos, materiales, económicos, etc) para hacer frente al creciente servicio que presta el área.

Dada esta situación, es necesario en cualquier organización que preste este tipo de Servicio, verificar la calidad del mismo, ya que en la generalidad se observa que cuando se hace una petición de apoyo por parte de los usuarios, no existe formalidad al registrar, canalizar y evaluar el apoyo técnico, que implica una falta de información sobre :

* Servicios Pendientes.
* Tiempo de Respuesta.
* Actividades de cada Técnico.
* Número de Servicios Prestados.
* Tipo de Fallas más usuales.
* Servicios atendidos a cada área.

Que inciden directamente en la eficacia y eficiencia del servicio, así como carencia de información para la toma de decisiones en la estrategia de la empresa.

Se recomienda entonces un Sistema que proporcione información del tipo de Soporte a través de una propuesta hacia una posible

solución del servicio (Busque Soluciones), esto mediante un Sistema Experto, que al tener un alto potencial para resolver problemas dará respuestas más acertadas y oportunas al usuario.El Sistema debe por su puesto estar orientado al usuario, es decir a resolver problemas reales del usuario. Su estructura externa este basada en la forma que el usuario ve su problemática. La terminología funcional debe ser congruente con la que maneja el usuario, que satisfaga necesidades y que cumpla con los requerimientos; en lo cual la experiencia humana juega un papel importante; así como también genere estadísticas, capte y clasifique información.

El Sistema fue diseñado bajo las siguientes fases :

* Análisis.

  Con los Expertos Informáticos.
  Del seguimiento que lleva a cabo el área para solucionar el problema.

* Diseño.

  Sistema Experto : Base de Conocimientos y Motor de Inferencia.

  Definición de entidades, atributos y relaciones.

* Desarrollo.

  Sistema Experto en C++ Orientado a Objetos (Borland).

  Para generar estadísticas, información, control etc, en Clipper 5.01.

* Depuración.

  Para todo el Sistema en Conjunto.

El Sistema tiene las siguientes características :

- De uso Generalizado para toda Organización que requiera un Soporte Técnico.
- Proporciona un Servicio más oportuno a los usuarios de los Recursos Informáticos.
- Genere una Economía de Costos.

- Se optimizan recursos tanto materiales como humanos.

- Existe una reducción del tiempo de respuesta del Servicio (Oportunidad y Eficiencia).

- Hay una mayor productividad al existir una mejor Organización del Servicio.

- Un mejor control del Tipo de Soporte Técnico que presta el área, al poder cuantificar y evaluar lo realizado.

- El Sistema concentra y aplica los conocimientos de la gente Técnica, altamente especializada.

- Herramienta de apoyo a la capacitación tanto especializada al personal técnico como particular al usuario final.

- Apoyo a la normatividad y estandarización.

- Apoyo en la toma de decisiones para :

    - Equipo
    - Software
    - Recursos Humanos.

en base a la experiencia obtenida con el Servicio Técnico y sustentada con las estadísticas generadas por el Sistema.

II. ANALISIS DEL SISTEMA

En Nacional Financiera la gerencia de Soporte Técnico da Servicio (entre otras funciones) a la población de usuarios; para cada servicio existe una Ficha de Servicio (F.S.) que es llevada como control del servicio por el técnico que presta el Servicio; la Ficha de Servicio esta estructurada de la Siguiente forma :

DIRECCION DE SISTEMAS Y PROCEDIMIENTOS
SUBDIRECCION DE INFORMATICA
SOPORTE TECNICO
FICHA DE SERVICIO

FOLIO : _____ RECIBIDO POR : _____

FECHA- (DD/MM/AA) : _____

VISITA : SI( )
VIA TELEFONICA : SI( )
ASISTECIA DEL USUARIO SI( )

NOMBRE DEL USUARIO : _____

PUESTO : _____

AREA : _____ UBICACION : _____

TELEFONO : _____

FALLA REPORTADA : _____

FECHA DE LA VISITA : _____ SERVICIO HECHO POR : _____

SERVICIO REALIZADO :

_____
_____
_____

OBSERVACIONES :

_____
_____
_____

FIRMA DE CONFORMIDAD :

_____ FECHA DE FIRMA (DD/MM/AA) : _____
U S U A R I O

Con este formato de Ficha de Servicio se atiende al usuario, pero no existía forma de evaluar el servicio, las fichas ahí se tenían archivadas con los datos del servicio, así es difícil evaluar un área. Había una falta de información inmediata de lo ya mencionado :

- Fichas de Servicio pendientes.

Algunas fichas de servicio quedaban pendientes, resultando poco práctico revisarlas manualmente y ordenarlas por tipo de falla o del técnico asignado.

- Tiempo de Respuesta del Area.

No se conocía exactamente cuanto se tardaba el técnico en cada servicio y por lo tanto se ignoraba el tiempo de respuesta del área.

- Actividades de cada Técnico.

No se conocía exactamente el número de servicios prestados por cada técnico ni de que tipo.

- El número de servicios prestados.

Tampoco se tenía un total de los servicios prestados por el área, así como el número en cada una de las distintas Fallas.

- Que tantos servicios a cada Area se prestan.

No se conocía cuántos servicios se prestan a cada área, tal vez una área demandaba más que otra y esto es un punto importante pues ahí faltaba capacitación ó una revisión más seria del equipo y esto difícilmente era reflejado por las fichas de servicio almacenadas.

En resumen la fase de análisis mostró que cuando solicitaban servicio los usuarios, no existía forma práctica donde se registraran sus requerimientos para referirlas después a los técnicos, así como el tipo de servicio prestado; es decir si se le atendió por teléfono, se le visito ó simplemente fue en la misma instalación física; el tipo de "falla", si fue una Instalación, Reinstalación, Configuración, Reconfiguración, Asesoría u Otros; que técnico atendió; cuando llamo el usuario (la fecha de su llamada), la fecha de cuando se le visito, la fecha de cuando quedo atendido el problema; todo esto era finalmente archivado a través de las Fichas de Servicio y revisado manualmente, se necesitaba entonces un Sistema lo suficientemente poderoso para que resolviera los problemas reportados por los usuarios, es decir un Sistema Experto que conjuntando los conocimientos de los Expertos Informáticos involucrados en el área diera propuestas de soluciones a problemas prácticos de los usuarios y en caso de no encontrarse los Expertos (o estos mismos) consultar el Sistema para dar una pronta solución a la Falla reportada; aparte de generar información estadística de ayuda, Reportes, Catálogos, llenado de la Ficha de Servicio a través del computador etc.

III.DISEÑO

III.1. DISEÑO EN LA DEFINICION DE ENTIDADES ATRIBUTOS Y RELACIONES

Durante el diseño del Sistema hubo un proceso de definición de entidades, atributos y relaciones que son la base para un buen diseño.

* Se identificaron las entidades involucradas en el problema.
* Se establecieron las relaciones entre estas entidades.
* Definimos el comportamiento de las relaciones (Explicar porque existe la relación).
* Se normalizaron las entidades.
* Se ajusto el diseño conceptual con consideraciones de eficiencia (Existió redundancia controlada y mínima para optimizar la generación de reportes).
* Se repitieron los pasos anteriores hasta obtener resultados satisfactorios (Un proceso iterativo).

Se concluyo que existen 2 entidades muy importantes :

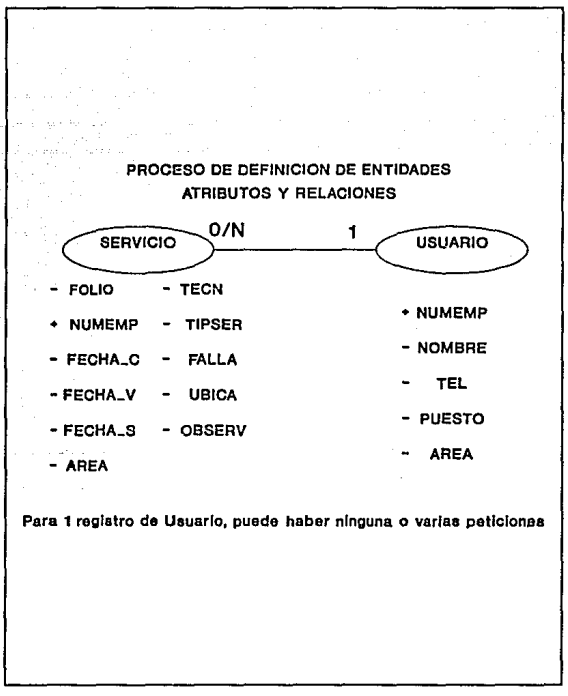
SERVICIO.DBF
USUARIO.DBF

SERVICIO.DBF

-	FECHA_C	Fecha de llamada del usuario.
-	FECHA_V	Fecha de cuando se visitaba al usuario.
-	FECHA_S	Fecha de cuando se concluyo el servicio.
-	FOLIO	Número de la Ficha de Servicio.
+	NUMEMP	Número de nómina del usuario.
-	AREA	Adscripción a la que pertenece el usuario.
-	TECN	El técnico que realiza el servicio.
-	TIPSER	El tipo de servicio que presta la gerencia Alpina,Externo y Teléfono.
-	FALLA	El tipo de falla (servicio) que se reporta Asesoría, Instalación, Reinstalación, Configuración, Reconfiguración y Otros.
-	UBICA	La ubicación del usuario (Dirección).
-	OBSERV	Observaciones del servicio u otras anotaciones.

USUARIO.DBF

+	NUMEMP	El número de nómina del usuario.
-	NOMBRE	Nombre del usuario.
-	TEL	Teléfono del usuario.
-	AREA	Adscripción a la que pertenece el usuario.
-	PUESTO	El puesto que ocupa el usuario.



Se definieron 5 bases de datos más, como catálogos auxiliares, para evitar confusión ó errores en la captura y manejo del Sistema; ya que existe una clave única para cada campo del atributo de la base de datos.

Hubo consideraciones de eficiencia; se ajusto el diseño conceptual con consideraciones de eficiencia. Existió redundancia controlada y mínima para optimizar la generación de reportes (AREA).

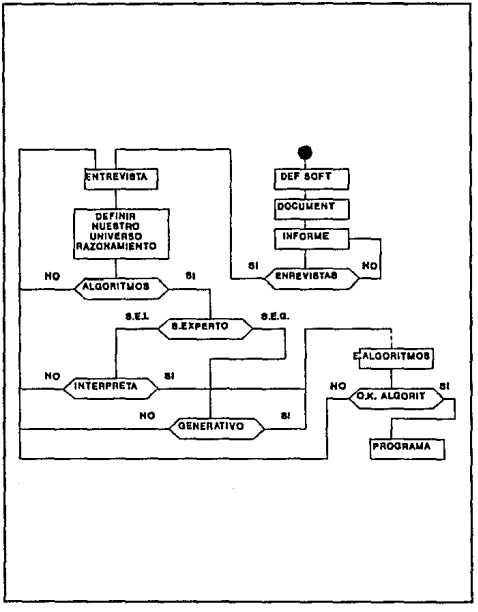
BASES DE DATOS AUXILIARES.

TIPO_S.DBF	TIPO DEL SERVICIO
1. TIPSER	Clave del tipo del servicio.
2. DESCRP	Descripción del tipo de servicio.
TIPO_F.DBF	TIPO DE FALLA
1. FALLA	Clave del tipo de falla.
2, DESCRP	Descripción de la falla.
TIP_T.DBF	TIPO DE TECNICO
1. TECN	Clave del tipo de técnico.
2. DESCRP	Descripción del técnico.
TIP_AR.DBF	TIPO DE AREA.
1. AREA	Clave del tipo del área.
2. DESCRP	Descripción del área.
TIP_PUE	TIPO DEL PUESTO
1. PUESTO	Clave del tipo del puesto.
2. DESCRP	Descripción del tipo del puesto.

Estas son las cinco bases de datos auxiliares que son lo catálogos para un mejor interno del Sistema. Finalmente esto se programo en clipper 5.01, al ser un lenguaje orientado al manejo de Bases de datos.

III.2. DISEÑO DEL SISTEMA EXPERTO

Durante el Diseño del Sistema Experto se elaboro una propuesta de seguimiento; abajo se muestra el Diagrama y la Explicación a cada módulo :



* DEFINICION DEL SOFTWARE : En base al tipo del software que más utiliza el usuario y que más consecuencias pueda tener es sus aplicaciones. Se eligió MS-DOS ver 5.0 al ser el Sistema Operativo usual en cualquier empresa; no existe usuario que no haya interactuado con MS-DOS alguna vez. Siendo DOS la plataforma de arranque de muchos paquetes debe cuidarse y optimizarse su uso. También se eligió Windows 3.1 como ambiente operativo opcional a DOS, muchos usuarios optan por Windows para su ambiente en su computador.

* ELABORACION DE UN DOCUMENTO PARA HACERSE LLEGAR A CADA EXPERTO. Se realizo un documento que se hizo llegar a cada Experto Informático, mucho antes de realizar las entrevistas estructuradas, esto con la finalidad de que cada Experto conociera antes; cual era el objetivo del Sistema, el planteamiento, el Software propuesto, así como el alcance. Y al llevar a cabo las entrevistas existiéra un antecedente de colaboración.

* INVESTIGACION EN LA ELABORACION DE CADA SOFTWARE PROPUESTO ASI COMO ELABORACION DE UN CONJUNTO DE FALLAS. Investigar a fondo cada software propuesto (MS-DOS y WINDOWS)elaborando un conjunto de Advertencias, Fallas, Errores y Mensajes enviados por la computadora al utilizar algún comando o aplicación de estos dos Ambientes. De tal forma que al tener un informe serio antes de las entrevistas con los Expertos, existieran preguntas más técnicas, más profundas y por ende una entrevista más productiva.

* ELABORAR ENTREVISTAS ESTRUCTURADAS. Se entrevistaron aproximadamente 15 Expertos en Informática, dejando al último a los más capacitados, de tal forma que al final de las Entrevistas tendríamos la Experiencia e información anterior.

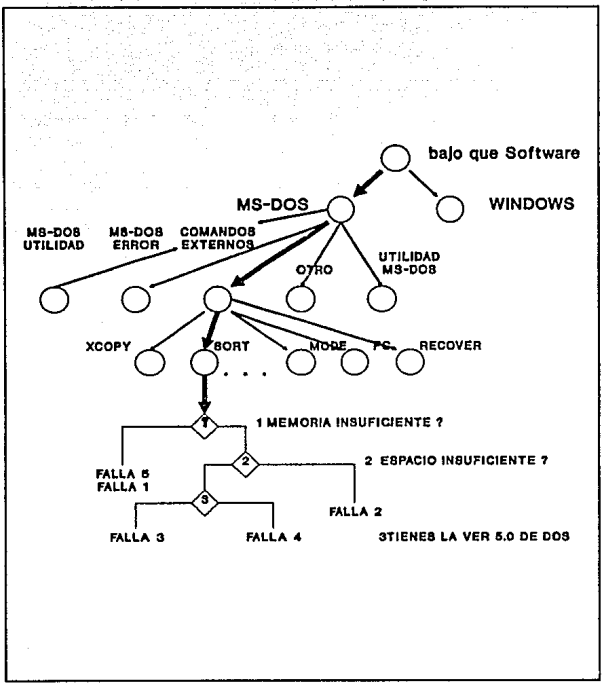
* DEFINIR NUESTRO UNIVERSO. Durante las últimas entrevistas ya se tenía un amplio conjunto de Errores, Fallas, Advertencias ó Mensajes enviados por el computador bajo MS-DOS ó Windows. Se percibió que en realidad pocas veces necesitamos la solución óptima, usualmente una buena aproximación nos servirá muy bien.

El Sistema Experto esta diseñado como método de encadenamiento hacia atrás (reglas de producción); es decir se comienza con una hipótesis (un objeto) y se pide información para confirmarlo o negarlo al mismo tiempo que esta nueva información elimina la mayor incertidumbre posible del Sistema (Como un programa que diagnóstica enfermedades infecciosas). Razona hacia atrás desde su meta para determinar la causa de la enfermedad del paciente.

Es decir en acuerdo con los Expertos, se dividió cada ambiente en pequeños conjuntos para en cada pregunta reducir nuestro universo y al final dar una propuesta de solución a la hipótesis inicial.

Por ejemplo :

Al principio se le pregunta al usuario bajo que ambiente se presento el problema MS-DOS ó WINDOWS; aquí al elegir alguno de estos, nuestro universo se redujo a la mitad, y así sucesivamente.



En el ejemplo anterior, inicialmente se le pregunta al usuario bajo que ambiente se presento la Falla, Advertencia, Mensaje o Error, una vez elegido el Software en este caso MS-DOS, se pregunta donde se presento :

- Dispositivo de MS-DOS
- Utilidad de MS-DOS
- Al utilizar un Comando Interno
* Al utilizar un comando Externo

- Otro, es decir preguntaremos características del computador y se hace inferencia a partir de las preguntas elaboradas.

* Al utilizar un Comando Externo :

Se sugieren 16 comandos externos (los usualmente más utilizados), y el usuario elige aquel que utilizo.

* SORT

Ahora ya sabemos que fue bajo MS-DOS, que fue un Comando externo y sabemos cual fue (SORT), deben ahora hacerse 3 cuestiones muy relacionadas con SORT :

Memoria Insuficiente.
Espacio Insuficiente.
Versión Incorrecta de DOS.

ya que SORT al consultar con los Expertos presenta mensajes relacionados con estas preguntas, por ejemplo :

SORT : Espacio insuficiente en el disco.

SORT : Memoria insuficiente.

SORT : Versión incorrecta de DOS.

Memoria Insuficiente.

Al usuario así le aparecen los mensajes; no necesita forzosamente sospechar de donde puede ser la Falla, en este caso el comando se lo advierte, pero muchas veces o se ignora la solución o al aplicar alguna no es la adecuada.

Aquí como en prácticamente todos los comandos muchos Mensajes son iguales, por ejemplo el último mensaje "Memoria Insuficiente" también se puede presentar idéntico con el comando REPLACE, BACKUP, CHKDSK, DISKCOPY y RESTORE (esto representa una gran ventaja para la programación); los mensajes de arriba se ve claramente que pertenecen exclusivamente a SORT; el Sistema finalmente llega a una propuesta de Falla y da una posible solución, así como la causa del Mensaje.

Todas las preguntas hechas desde un principio, llevan a subconjuntos usuales de presentación de mensajes, por ejemplo se pregunta primero por la Memoria Insuficiente ya que se SORT dentro de la generalidad se presenta en Memoria Insuficiente, después en Espacio Insuficiente y Finalmente en Versión incorrecta de DOS, no así en otros comandos donde se evaluaron las preguntas (si es que las hubo) para ponerlas en orden donde más se intuya por la Experiencia que existe la Falla; no tendría caso preguntar primero por la versión en MS-DOS con SORT si esto casi no llega a presentarse; es más común la Memoria Insuficiente.

Por ejemplo se concluyo que preguntas como :

- ¿ El Sistema se detuvo ?
- ¿ Sistema tarda en responder ?
- Memoria Insuficiente.
- Espacio Insuficiente.

Llevan por un lado a una reducción importantísima de nuestro universo y por el otro conducen adecuadamente a las hipótesis para los asertos.

Este Sistema es Interpretativo es decir el motor de inferencia está mezclado con los objetos y los atributos; esto significa que la base de conocimientos fue codificada manualmente por el Ingeniero del conocimiento, es decir es fija dentro del programa. Aún cuando esto merma un poco al Sistema, el Sistema cuenta con capacidades humanas (Expertos) muy importantes que primero se entendieron lo suficiente, para codificarlas en el programa y difícilmente en un tiempo prolongado habrá cambios a la base de conocimientos; se realizo un estudio largo y serio para realizar un trabajo conciso y de la manera más simple posible.

III.2.1. C++ ORIENTADO A OBJETOS PARA EL SISTEMA EXPERTO

El lenguaje C++ Orientado a objetos que se reviso para su utilización (al tener un conocimiento previo de C++), brinda excelentes posibilidades en el desarrollo del Sistema ya que se vio que se cumplían satisfactoriamente nuestros objetivos, al permitir aceptar : los Objetos, el Polimorfismo y la Herencia. Aparte de que C++ prácticamente no tiene "barreras" dentro de la programación.

Objetos.

La característica más importante de un lenguaje orientado a objetos es el objeto. Un objeto es simplemente una entidad lógica que contiene datos y un código que manipula esos datos. Dentro de un objeto, parte de ese código o datos pueden ser privados e inaccesibles fuera de él.

Polimorfismo.

El polimorfismo soportado en C++ Orientado a objetos permite usar un nombre para varios propósitos relacionados pero ligeramente diferentes. El fin del polimorfismo es permitir el uso de un nombre para especificar una clase de acción general. Se ejecuta una parte especifica de la clase general dependiendo del tipo de dato con el que está trabajando.

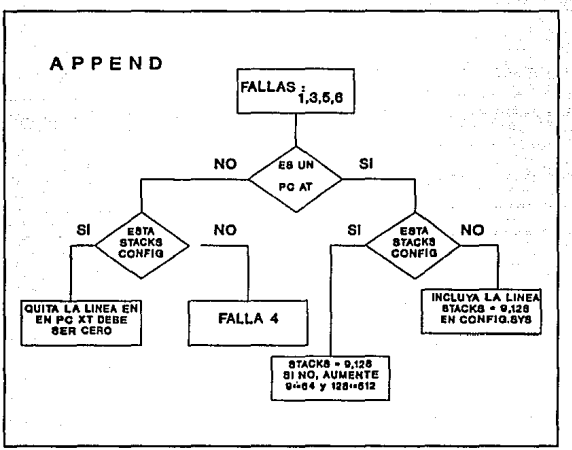
Herencia.

La herencia es el proceso por el cual un objeto puede adquirir propiedades de otro objeto. Es decir soporta el concepto de clasificación. Si lo considera, la mayoría del conocimiento se hace manejable por medio de clasificaciones jerárquicas. Por ejemplo un Mensaje en SORT es parte del Comando Sort que es parte de los Comandos Externos, estos a su vez pertenecen a Mensajes en DOS y DOS es parte de uno de los dos Softwars tratados en el Sistema. Realmente usando las clasificaciones, sólo se necesita definir las cualidades que hacen único a un objeto dentro de su clase. La herencia es el mecanismo que hace posible que un objeto sea un ejemplo específico dentro de una clase más general.

En el Sistema cada Comando es un objeto; cada mensaje es un atributo del objeto, algunas veces el atributo llega a usarse para más de un objeto, lo importante aquí es que a respuestas del usuario el objeto va teniendo más, o menos atributos, al final la respuesta a la propuesta de solución será el objeto final con sus atributos.

En C++ Orientado a objetos se definen clases, una clase es sintácticamente similar a una estructura y ahí se define el objeto mismo. La ventaja de las clases es que pueden verse como un árbol con sus nodos, habrá clases amigas, es decir que necesiten ser accesadas por otra clase. En el Sistema se declararon clases amigas aquellas que tenían un mismo atributo en común; cada clase tiene su parte privada y su parte publica, la parte privada fue diseñada únicamente para mensajes (atributos) exclusivos del comando u opción, es decir solamente ahí se presentaban; la parte publica fue para aquellos atributos que mínimamente se coincidían con otro comando u opción o aquellos que se referían a lo mismo (atributos similares).

Verifiquemos el siguiente ejemplo para el comando APPEND :



En el comando APPEND después de que el usuario advirtió que al utilizarlo tenía problemas; inicialmente como muestra el diagrama se proponen cuatro posibles y comunes fallas al utilizar el comando APPEND, posteriormente se le pregunta al usuario si se resolvió el problema; el usuario distinguirá si su mensaje es uno de los propuestos inicialmente y sabrá la solución, en caso de no ser así pulsará NO, para que se le cuestione sobre más situaciones como se indica en el diagrama; abajo se muestra el código fuente para este comando; aquellas fallas que no se encuentran aquí y se proponen como solución en APPEND, están llamadas a través del objeto al que pertenecen o con el mismo objeto que tiene APPEND aprovechando que se declararon como funciones amigas y por lo tanto pueden accederse entre ellas y por supuesto que coinciden en el tipo de falla o mensajes :

Inicialmente se definen la clase APPEND, a la que posteriormente se le asigna un para identificarla (recuerde que el objeto se compone de atributos, y cada propuesta finalmente es un objeto con atributos obtenidos a través de la inferencia realizada).

Posteriormente se codifica la base de conocimientos relacionada o mezclada con los atributos y objetos.

Después se codifican separadamente las sentencias if then else para Inferir sobre las respuestas del usuario, también se utilizaron secuencias do while, aquí no se ven ya están en subrutinas.

Definimos como fallas privadas a la Falla 1,5 y 6; (esto claro bajo una consulta exhaustiva con los Expertos) ya que solo se presentan (así como las planteamos) exclusivamente en APPEND por eso es que son privadas.

Al mismo tiempo que la clase append_priv se hace Amiga de las clases command_idem, command_extern y ms_dos; ya que en cada una de ellas existe por lo menos un mensaje que también puede relacionarse con APPEND, y que en este caso el objeto que corresponda a append pueda tener acceso directo a estas clases.

```
//     A   P   P   E   N   D

  class append_priv : public command_idem,
                      public command_extern, public ms_dos {
    char fcn1[80], fcn5[80], fcn6[80];
  public:
    void asign_fcn1(char *fpa_fcn1);
    char obt_fallan1(void);
    void asign_fcn5(char *fpa_fcn5);
    char obt_fallan5(void);
    void asign_fcn6(char *fpa_fcn6);
    char obt_fallan6(void);
 );
```

De hecho en el Diagrama se observa que la Falla 3 propuesta inicialmente pertenece a la clase comman_idem, la falla 4 pertenece a ms_dos y la falla 2 (referente a Stacks) pertenece a command_extern.

```
//   A P P E N D
 void append_priv :: asign_fcn1(char *fpa_fcn1='\0')
 {
  strcpy(fcn1,fpa_fcn1);
  obt_fallan1();
 }

 char append_priv :: obt_fallan1(void)
 {
  gotoxy(3,x++);
  cout << fcn1;
  gotoxy(6,x++);
  cprintf("CAUSA : No puede utilizar append en una unidad
                   asignada con asign");
  gotoxy(3,x++);
  cprintf("SOLUCION : Cancele la asignación de la unidad antes
                      de empezar el");
```

```
 gotoxy(14,x++);

 cprintf("comando append otra vez con esta unidad ");
 return 0;
}

void append_priv :: asign_fcn5(char *fpa_fcn5='\0')
{
 strcpy(fcn5,fpa_fcn5);
 obt_fallan5();
}

char append_priv :: obt_fallan5(void)
{
 gotoxy(3,x++);
 cout << fcn5;
 gotoxy(6,x++);
 cprintf("CAUSA : Ha establecido un archivo o un directorio
                  que no existe");
 gotoxy(3,x++);
 cprintf("SOLUCION : Introduzca el correcto con append");
 return 0;
}

 void append_priv :: asign_fcn6(char *fpa_fcn6='\0')
{
 strcpy(fcn6,fpa_fcn6);
 obt_fallan6();
}

char append_priv :: obt_fallan6(void)
{
 gotoxy(3,x++);
 cout << fcn6;
 gotoxy(6,x++);
 cprintf("CAUSA : No ha asignado una ruta de búsqueda para
                  los archivos");
 gotoxy(14,x++);
 cprintf("de datos");
 gotoxy(3,x++);
 cprintf("SOLUCION : Si desea hacerlo utilice el comando
                     append");
 return 0;
}
```

La forma de traer las fallas no incluidas en APPEND es a través del objeto asociado a cada una de estas o como ya se menciono con el mismo objeto de APPEND (que es este caso) al tener la posibilidad de acceder a las funciones amigas; por ejemplo :

A APPEND le fue asignado el objeto ob13, con el que accesamos las funciones :

```
ob13.asign_fcc3          ---> clase commad_idem
ob13.asign_fcmsdos15     ---> clase ms_dos
ob13-asign_fcl2          ---> clase command_extern
```

cada falla es un atributo del objeto, al final se le propone técnicamente un objeto final de solución al usuario que tiene atributos inferidos de varios o del mismo objeto.

```
//                          FUNCION APPEND()
void append(void)
{
  char b,s;
  int i;
  append_priv ob13;

  clrscr();
  ve_uno_llama_pre();
  x=3;
  ob13.asign_fcn1("1. FALLA : Conflicto Append/Asign");
  ob13.asign_fcc3("2. FALLA : La combinación de parámetros no
                              es válida");
  ob13.asign_fcn5("3. FALLA : La ruta de acceso o parámetro no
                              es válido");
  ob13.asign_fcn6("4. FALLA : No existe ninguna asignación");
  gotoxy(10,19);
  cprintf("Se soluciono el problema ?");
  x++; x++; x++;
  s = pregunta(b);
  if(s == 'S') {
    exit(0);
   } else {
      limpia();
      x=3;
      gotoxy(4,x++);
      cprintf("Pulsa [S/N] según lo adviertas");
      x++;
      gotoxy(3,x++);
      cprintf(" El Compuatador que tienes es un AT ");
      s = pregunta(b);
      if(s=='S') {
     x++;
     gotoxy(3,x++);
     cprintf(" Verifica si tienes la línea stacks en
                        config.sys ");
     s = pregunta(b);
      if(s=='S') {
       x++; x++;
       ob13.asign_fcl2("Tal vez los parámetros del comando
                              stacks no son válidos");
      } else {
        x++; x++;
        ob13.asign_fcl2("Incluya la línea stacks = 9,128 ");
       }
```

```
    ) else {
      x++;
      gotoxy(3,x++);
      cprintf(" Verifica si tienes la línea stacks en
                        config.sys ");
      s = pregunta(b);
        if(s=='S') {


        x++; x++;
        gotoxy(3,x++);
        cprintf(" Quita la línea stacks en XT debe ser 0 por
                                  default");
         } else {
         x++;
         gotoxy(3,x++);
         cprintf(" Al parecer los Stacks estan bien");
         x++;
         ob13.asign_fcmsdos15("1. FALLA : La ruta de acceso
                                          no es válida");
        }
   }
   }
  gotoxy(10,22);
  salir();
}
```

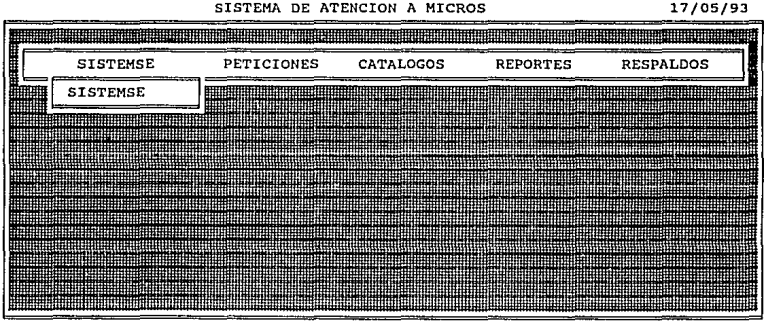
Dentro del Sistema también existen subrutinas para funcionar en tiempo de ejecución, cuando se pregunta acerca del Espacio Insuficiente en Disco se corre un programa para verificar el espacio que tiene disponible el disco duro, asegurandonos así en forma exacta si en efecto es o no de Espacio insuficiente el problema.

Como puede observarse es práctica la codificación en C++, al ofrecer grandes ventajas (herencia, polimorfismo, etc) el lenguaje; aparte de que crea un código muy pequeño y los accesos son muy rápidos, contribuyendo así a un Sistema con mucha información de pequeño código, con mucha inferencia (en algunos casos) y muy rápido.

IV. DEPURACION DEL SISTEMA

En esta última fase del Sistema se une el Sistema Experto (Desarrollado en C++ orientado a objetos) y la parte de estadísticas, reportes, consultas, etc (Desarrollado en Clipper), para de esta forma conformar el producto final, orientado primordialmente a el Sistema Experto.

El Sistema presenta la siguiente máscara :



SISTEMA EXPERTO

SISTEMSE : Al posicionarnos en este menú, nos llevara a entrar al Sistema Experto.

PETICION : Se refiere a la llamada Ficha de Servicio, donde se podrá dar una Alta, Cambio, Baja o Consulta de esta, que aparecerá en pantalla tal cual la lleva el técnico al servicio.

CATALOGOS : Se accesarán los catálogos de Usuario, Tipo de Falla, Tipo de Servicio, Tipo de Técnico y Area, en donde en cada uno se podrá dar una Alta, Baja, Cambio o Consulta.

REPORTES : Genera Reportes estadísticos como : Reporte de Servicios pendientes, de Tiempo de Respuesta, de Actividades de cada técnico, Total de Servicios prestados, Fallas atendidas y un Reporte general de servicios prestados a cada área así como el total.

RESPALDOS : Hace un respaldo de la información, ya sea a Disco Duro, o Disco Flexible.

Haciendo énfasis en que la parte más importante es el Modulo de SISTEMSE (Sistema Experto).

El como se unieron los Sistemas, fue a través del comando RUN de clipper que ejecuta un programa externo y regresa al ambiente original, (invoca y garantiza que solo el código ejecutable de Clipper o C++ estará en Memoria RAM, esto con el fin de evitar problemas posibles de Memoria Insuficiente).

El Sistema requiere ser instalado en Disco duro ocupando poco menos de 400 kbytes, que resulta poco al tener dos archivos ejecutables, así como siete bases de datos, siete indexaciones y algunos archivos requeridos.

El Sistema es Monousuario, no es para Red, y requiere las siguientes características de Software y Hardware :

Hardware	Software
AT-286	Sistema Operativo MS-DOS 5.0
MONITOR VGA	Tener la siguientes líneas declaradas en CONFIG.SYS FILES = 30 BUFFERS = 30 Tener la siguiente línea declarada en AUTOEXEC.BAT SET CLIPPER=/F25

CONCLUSIONES

El Sistema desarrollado proporciona soporte en la utilización de MS-DOS y WINDOWS al hacer accesible el conocimiento a una gran cantidad de usuarios y tener una herramienta útil en el empleo correcto y eficiente de estos ambientes operativos.

EL Sistema Experto concentra los conocimientos de mucha gente especializada en el campo, resultado de un trabajo de largas investigaciones y organización de la información que resultan ser muy valiosos al concentrar las soluciones que da la experiencia; y se ofrece al profesional una herramienta de apoyo en las labores que desempeña.

El usuario de preferencia debe tener minimamente nociones de estos Sistemas Operativos, ya que entre más capacitado sea será mucho mejor el dialogo técnico entre el usuario y el Sistema, conduciendo a un diagnóstico final mucho más exacto.

Siempre existirá algo más que investigar en el área en la que se desarrolló el Sistema Experto, por esto es importante desarrollar nuevos y más novedosos algoritmos generativos de soluciones (Base de conocimiento inteligente), de forma tal que al Sistema, el usuario pueda suministrarle nueva información.

GLOSARIO DE TERMINOS

FUNCION DE ACCESO : Es el código fuente que nos permitirá manipular nuestros elementos. La sintáxis y semántica del lenguaje especifica la función de acceso.

HABILIDAD DEL PROGRAMADOR : Se refiere a que tanat destreza o ingenio posee para el desarrollo de la programación.

USUARIO : Aquella persona que hace uso del Software y Hadware de aplicación.

OBJETO: Una entidad lógica que contiene atributos y sobre el cual guardamos información.

ASOCIACION : Relación entre un conjunto de entidades.

METODOLOGIA : Procedimiento bajo ciertas reglas a seguir.

FASE DE GENERACION DE CODIGO : Se refiere a obtener código fuente en alguna aplicación, sin la necesidad de programarla desde un principio, generado con menúes y después añadirlo o reforzarlo (Herramientas CASE).

INTELIGENCIA : Facultad de entender o de conocer; de razonar. Habilidad y Experiencia.

ARTIFICIAL : Hecho por el hombre, es decir no natural.

COMPRENSION : Entender un conjunto de cualidades que integran un concepto.

PERCEPCION : Percibir o apropiarse. Recepción de una idea a través de los sentidos.

MAQUINA : Se refiere a un computador donde interactua Hadware y Software.

NEURONA : Anat. Conjunto de la célula nerviosa adulta y sus prolongaciones, considerado como unidad histológica y fisiológica independiente.

CONOCIMIENTO : Efecto y acción de conocer, tener la noción de algo.

HECHO : Suceso ocurrido. Crear algo.

FALLA : No se refiere propiamente a una Falla como tal, se definió así para englobar el servicio prestado al usuario. Falla en la concepción mas limitada de la palabra.

BIBLIOGRAFIA.

MANUALES DE CLIPPER 5.0 :

- CLIPPER COMMANDS.
- CLIPPER FUNCTIONS.
- PROGRAMACION Y UTILIDADES.

TURBO C/C++
MANUAL DE REFERENCIA
HEBERT SCHILDT
Mc GRAW HILL

EL LENGUAJE DE PROGRAMACION C
BRIAN W. KERNIGHAN
DENNIS M. RITCHIE.
PRENTICE HALL

PROGRAMACION EN LENGUAJE C
HEBERT SCHILDT
Mc GRAW HILL

GUIA PARA USUARIOS EXPERTOS C
HERBERT SCHILDT
Mc GRAW HILL

UTILIZACION DE C EN INTELIGENCIA ARTIFICIAL
HERBERT SHILDT
Mc GRAW HILL

CLIPER 5.1 NANTUCKET
C++ ORIENTADO A OBJETOS BORLAND

MANUAL DEL USUARIO MS-DOS 5.0
MICROSOFT

MANUAL DEL USUARIO WINDOWS 3.1
MICROSOFT