UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ENEP ARAGON

DISEÑO DE UN SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO P.C. Y COMPATIBLES

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A :

ARTURO GARCIA CALOGIANI

MEXICO, D. F. 1989

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

DISEÑO DE UN SISTEMA EXPERTO PARA DIAGNÓSTICO DE FALLAS EN EQUIPO P.C. Y COMPATIBLES

OBJETIVO DE LA TESIS.

PARTE I
(SISTEMAS EXPERTOS)

PARTE II
(DISEÑO DEL SISTEMA EXPERTO)

PARTE III
(LISTADO DEL SISTEMA EXPERTO)

PARTE IV
(ANALISIS)

PARTE V
(CONCLUSIONES)

OBJETIVO DE LA TESIS

Dentro del campo de la Ingeniería de desarrollo de sistemas computacionales e informáticos, se han visto muchos avances tecnológicos en lo que se refiere a la simplificación de tareas cotidianas.

En nuestro país, se ha tratado de llevar ese margen de evolución, sin embargo nuestra tecnología como país subdesarrollado nos limita a revasar fronteras o simplemente competir con otros paises.

Hace cuatro años se dió a conocer en México el campo de la Inteligencia Artificial (IA) con una variedad de gamas que enfocan un nuevo avance en la Ingeniería en Computación -llamada tambien la 5a. Generación- como lo es la Robotica, Sistemas Expertos, Autómatas, etc.

La presente tesis se enfoca a una de esas ramas de la IA como lo es los Sistemas Expertos. Los Sistemas Expertos (SE) se utilizan para aplicar el saber de los especialistas a la resolución de problemas de diversas areas de la industria. Sus posibilidades son varias, abarcando todo campo en el que exista una base de experiencia reconocida y ofreciendo soluciones a problemas que han resultado complejos con la tecnología de programación convencional.

En el trabajo titulado: '*Diseño de un Sistema Experto para Diagnóstico de fallas en Equipo P. C. y compatibles*' presenta desde una introducción a la IA, que son los SE, así como una aplicacion practica de la vida real, el planteamiento, analisis y solución de fallas en equipo PC. ¿Porque un diagnóstico de fallas en PC? Uno de los problemas cotidianos actualmente son las maquinas de moda: las Computadoras Personales.

Estas máquinas por ser uno de los equipos más usados en las pequeñas y grandes industrias tienden a tener un porcentaje alto de fallas en sus componentes.

A pesar de que existen grandes conocedores en reparación de equipos (tecnicos), son muy pocos los que conocen a fondo las fallas más mínimas en lo que se refiere a PC (memorias, periféricos, tarjetas, circuitería, etc).

El presente trabajo contiene encuestas realizadas en grandes industrias donde han existido estas fallas, así como la recopilación de información de expertos en la materia para la solución de problemas.

Cabe mencionar que el desarrollo del SE tiene un nivel técnico sencillo que podra ser comprendido a un nivel de usuario, es decir, solamente podrá ser comprendido por aquellas personas que tengan conocimientos básicos del funcionamiento de la máquina, así como técnicos que deseen consultar el sistema en caso de que ellos no puedan localizar cierta falla.

La elaboración del SE está escrito y ejecutado en un lenguaje práctico (TURBO PROLOG), donde muestra un menú de posibles fallas y en la cual el usuario consultará dando las opciones que se piden (preguntas), y el mismo sistema se encargará de dar una solución al problema (respuestas), manejando ciertas reglas para la localización de el problema y así poder aportar una solucion.

PARTE I

SISTEMAS EXPERTOS

CAPITULO 1

I.I. QUE SON LOS SISTEMAS EXPERTOS.

I.1. INTRODUCCION.

Hace aproximadamente 20 años desde que Newel [1] empezó a explorar diversas alternativas de organización para la solución de problemas. Exploro ideas acerca de como se debería proceder con relación al diseño de sistemas que resolvieran problemas. A partir de esa fecha, muchas tecnicas se han desarrollado dentro del campo de la Inteligencia Artificial (IA); desde entonces se hizo una investigación y se construyeron muchos Sistemas Expertos. Los Sistemas Expertos (SE) son programas para resolver problemas que son sustancialmente generales, admitiendo en principio dificultad y requerimiento de experiencia. Los SE se han utilizado como un vehículo de investigación de IA bajo el razonamiento de que estos proporcionan un elemento poderoso para el desarrollo en la solución de problemas.

No fue sino hasta 1940 en que con la emergencia de las computadoras digitales modernas se renovo el interes, mas allá de lo teórico, en Inteligencia Artificial. Es en 1947 cuando aparecen los primeros automaticos en '*damas*'. No faltan precursores científicos que escribieron [2] en 1933: *Las ideas fundamentales de la Psicología Gelsalt y de Comportamiento justifican los intentos para construir y desarrollar maquinas de un nuevo tipo -maquinas que piensan-*.

El tiempo pasó y no fue sino hasta 1956 cuando se estableció el Dartmouth Summer Research Project on Artificial Intelligence bajo la responsabilidad de John McCarthy. Es el propio McCarthy quien acuna entonces el nombre de Inteligencia Artificial. [3]

En cuanto al presente, se destaca computadoras que son quizá más rápidas que las anteriores, dotadas de memorias grandes, y de sistemas operativos mas potentes, hablamos de una Cuarta Generación. Finalmente se habla de computadoras que aprendan de su experiencia y que conversaran libremente con los usuarios (ver fig. 1).

La primera predicción es como sabemos es correcta, para la segunda habrá que esperar a las computadoras de la Quinta Generación (Japón ya trabaja en ello), pero mientras tanto habrá que diseñar proyectos sobre IA.

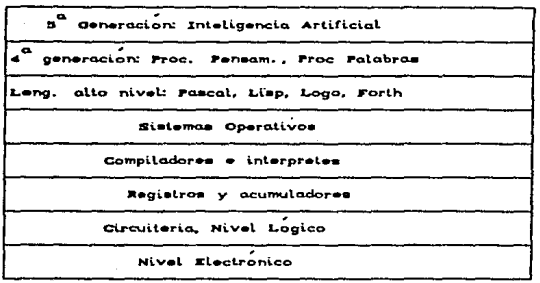


FIG. 1. – MUESTRA UNA JERARQUÍA HIPOTÉTICA EN LA CUAL SE RECORRE EL ESPECTRO QUE LLEVA DESDE EL NIVEL ELECTRÓNICA HASTA EL NIVEL DE LA INTELIGENCIA ARTIFICIAL PARA EL TRATAMIENTO DEL LENGUAJE NATURAL. NOTESE QUE EL NIVEL DE LA 4a. GENERACIÓN APARECEN ALGUNOS PRODUCTOS TÍPICOS DE LA IA RELATIVOS AL LENGUAJE NATURAL MODIFICADA.

Por ahora la IA se ha convertido en la rama más controvertida, interesante y de mayores persectivas en la ciencia de la computación. se habla de entre otras cosas, KIPS (Knowledge Interference Processing Systems), de LISP (List Programming) y de una Quinta Generación de computadoras, desarrollandose en Japón desde 1982. Esta nueva generación de maquinas implica, entre otras cosas, el diseño de computadoras especificamente para el proceso de sistemas de IA.

En la Fig. 2 se muestra un 'Arbol Familiar de la IA' y algunos temas de Investigación en las que se relaciona:

Arbol de la I. A.

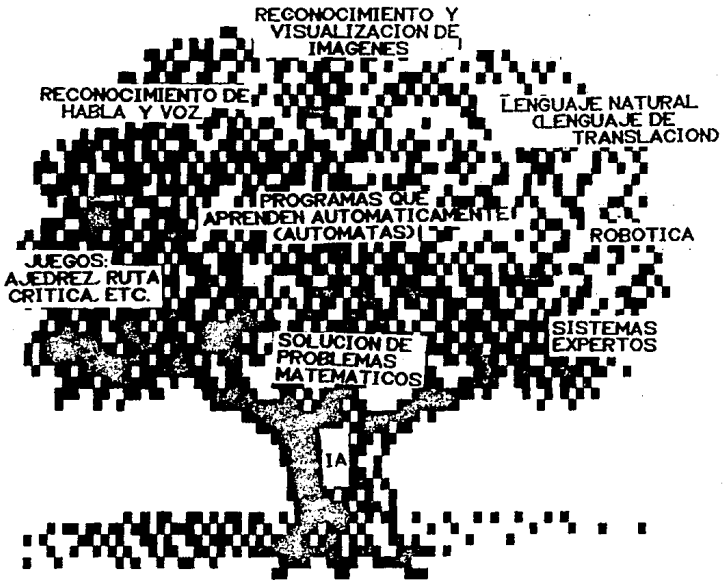


FIG. 2. RAMAS DE LA INTELIGENCIA ARTIFICIAL

Hay dos maneras de concebir la IA: desde el punto de vista teórico, esta disiplina se preocupa por comprender el modo de conseguir computadoras que perciban y entiendan a nivel humano, mientras que desde un enfoque de ingeniería su objetivo es desarrollar computadoras con facultades humanas, tal es el caso de la Robotica que imita la manipulación, la comprensión de Lenguaje Natural y el habla, la de comunicación y visión, mientras que los Sistemas Expertos tienen la facultad de resolver problemas (ver figura 3).

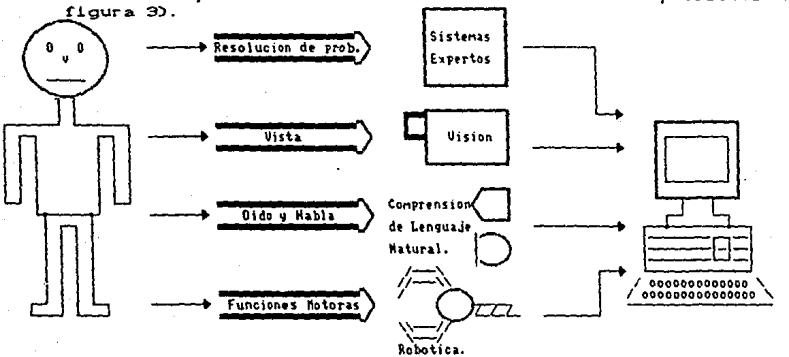


FIG. 3. La Inteligencia Artificial modelo de la conducta humana.

Con estas orientaciones se entrecruzan las técnicas de desarrollo de sistema informáticos para IA, los cuales se clasifican en varios grupos:

1.- Representación de los conocimientos que cubre los formalismos utilizados para descrubrir tanto los *conocimientos declarativos* como los de *procedimientos*; los primeros se encargan de los hechos, teorías e hipótesis, mientras que los de procedimiento describen como utilizar aquellos conocimientos (estrategias y tácticas).

2.- Es el proceso del conocimiento, es decir, los mecanismos que exploran el conocimiento y hacen deducciones razonadas para llegar a una conclusión.

3.- Comprenden las técnicas de aprendizaje, que producen nuevos conocimientos.

4.- Las estrategias de planificación para organizar la resolución del problema.

5.- Interfaz del usuario.

I.2. SISTEMAS EXPERTOS.

Actualmente y ya desde la década pasada han aparecido los denominados *'Sistemas basados en conocimiento'*, quiza mejor conocidos como 'Sistemas Expertos', aunque estos son un conjunto particular de aquellos.

I.3. DEFINICION

Un Sistema Experto se puede definir como 'Un sistema de programas que contienen una estructura de datos que son inteligentes ya que amplian el conocimiento para la solucion de problemas'.

Feingenbaum un pionero en SE, en 1982 establece lo siguiente [4]:

'Un Sistema Experto en un programa de computadora inteligente que utiliza conocimientos y procedimientos de inferencia para resolver problemas que son bastantes difíciles, requiriendo de una experiencia humana significativa para su solucion. En sí, un modelo de la experiencia de los mejores practicantes de su área se puede pensar como el conocimiento que se ejecuta a un nivel dado más la utilizacion de procedimientos inferenciales'.

El conocimiento de un SE consiste de hechos y heurísticas. Los hechos constituyen el cuerpo de la informacion que es compartida ampliamente, públicamente disponible y del cual existe una concordancia general entre los expertos en el campo. El buén nivel de desempeño del SE es una función fundamental del tamaño y la calidad de la base de conocimientos que esta tiene.

Un SE presenta las características siguientes:

a) Es un programa que actúa 'inteligentemente'

b) Utiliza conocimientos en un área específica del saber.

c) Trabaja en base a procedimientos de inferencia, para llegar así a conclusiones y resolver problemas.

d) Aprende a través de la experiencia.

e) Se comporta como un experto humano en el área correspondiente.

I.4. ESTRUCTURA BASICA DE UN SE.

Un SE consiste fundamentalmente en tres partes fundamentales [4]. Interfaz de usuario, Procedimientos de Inferencia y Base de conocimiento.

a) La *Interfaz de usuario*, permite al usuario conectarse e interactuar con el sistema para presentar el problema y enterarse de las conclusiones. Un aspecto importante, es que el SE a igual que el experto humano, justifican sus conclusiones. Una consideración importante para el diseñador del interfaz es repartir la iniciativa entre sistema y usuario. Cuando el sistema toma la iniciativa, dirige el diálogo y hace preguntas al usuario, quien al menos que lo solicite, no puede presentar información al sistema. Si se trata de un SE de diagnóstico (como lo es nuestro objetivo), este toma la iniciativa seleccionando una hipótesis e interroga al usuario hasta que tal hipótesis se confirma o falla. Esto es adecuado para usuarios inexpertos que carezcan de opinión propia pero tal vez es frustrante para usuarios expertos caapaces de aportar una hipótesis alternativa o cierta información adicional, ya que se sentirían menospreciados por el sistema.

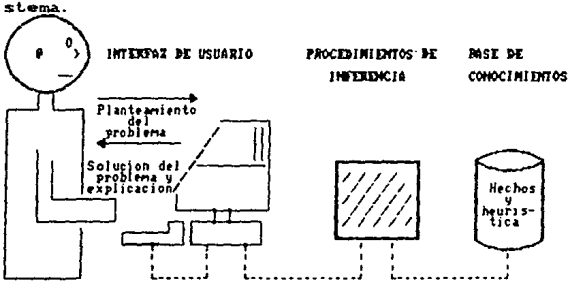


Fig. 4. Estructura de un Sistema Experto.

b) Un *Procedimiento de inferencia* (o estructura de control) para utilizar la base de conocimientos en la solución de problemas. Opera por deducciones y selecciona el conocimiento relevante para llegar a una conclusión. De ahí que el sistema pueda contestar a preguntas del usuario aún cuando la respuesta no este explícitamente almacenado en la base de conocimientos.

c) Una *base de conocimientos* (o fuente de conocimientos) de un dominio de hechos y heurísticas asociada con los problemas. Es la parte más importante, pues ya que contiene el conocimiento y las aptitudes de los expertos en la materia. Es por eso que a los SE se les concoce como *sistemas basados en conocimiento*. Es una gran ventaja que la base de conocimiento esté separada de la etapa de inferencia (fig. 4), pues ello permite añadir o cambiar conocimientos sin cambiar los procesos de desarrollo que se requieren en la programación.

Una vez que un SE ha sido desarrollado, necesitará de un humano experto para que colabore en la implantación de una base de conocimientos. Teniendo la capacidad más tarde de resolver problemas, ademas de que, se puede usar para ayudar a instruir a otros comparando con su propia experiencia.

Donald Michie [5] señala que: idealmente existen tres modos diferentes de usuarios para un SE, en contraste al modo simple de obstención de respuestas a problemas cuya características es común en computación. Los tres modos de usuario para un SE son:

1.- Obstención de respuestas a problemas (usuario como cliente).

2.- Perfeccionamiento o incremento del conocimiento del sistema (usuario como tutor).

3.- Aprovechamiento de la base de conocimientos para uso humano (usuario como alumno).

Los usuarios de un SE en el modo (2) son conocidos como '*El dominio de especialistas*'. Y no es posible construir un SE sin la ayuda de alguno de ellos.

Un SE actúa como un recipiente en que se deposita sistematicamente el conocimiento a través del tiempo, que es acumulado por muchos especialistas de diversas experiencias. En consecuencia este podra alcanzar una experiencia de consulta exagerada*, más aún, experiencia mayor que cualquiera de sus Tutores.

* No han existido ejemplos de SE cuya ejecución sobre pase la capacidad de un experto, sin embargo se trabaja en ello.

I.5. BASES DE CONOCIMIENTOS

La base más popular para representar el dominio de conocimientos que se requiere para un SE son las reglas de Producción. Estas pueden ser referidas como reglas de *situación-acción* o *IF-THEN*. De esta forma, con frecuencia una base de conocimientos está apoyada principalmente de reglas, las cuales son invocadas por un casamiento de patrones con las peculiaridades de las tareas circunstanciales que van apareciendo en la base de datos global.

las reglas en una base de conocimientos representa el dominio de hechos y la heurística reglas de un buen juicio de acciones a tomar cuando uno llega a situaciones específicas. La fuerza de un SE yace en el conocimiento específico del dominio del problema. Casi todos los sistemas existentes basados en reglas contienen un centenar de ellas y generalmente se obtienen de entrevistas con expertos por semanas o meses.

En cualquier SE, las reglas se conectan una y otra por ligas de asociación para formar redes de reglas. Una vez que han sido ensambladas tales redes, entonces tenemos una representación de un cuerpo sustancial de conocimientos (ver FIG. 5).

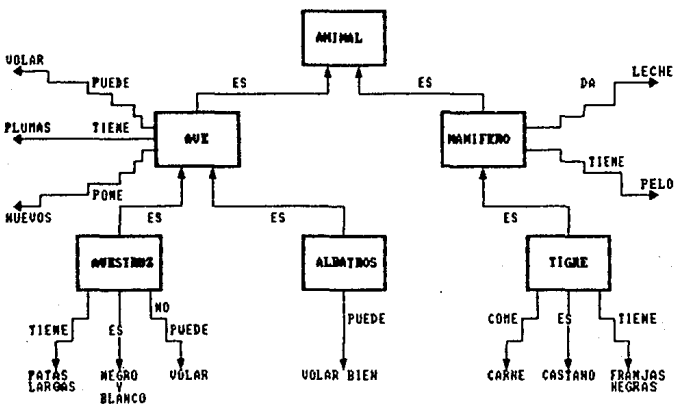


Fig. 5. Representacion del conocimiento, utilizando redes semanticas.

Un experto por lo general tienen muchas reglas de criterio o empíricas, por lo que el soporte suele ser incompleto para la evidencia que se dispone. en tales casos una tecnica es la de vincular valores numericos a cada regla para indicar el grado de certidumbre que se esta asociando. En la operacion de SE los valores de certidumbre se combinan unos a otros a traves de la red, considerando tambien, la certidumbre de los datos de entrada del problema, para llegar finalmente a un valor de certidumbre de la solucion final.

I.6. ARQUITECTURA DE SE.

En el articulo de Stefik [6], el estudio arquitectonico de SE esta caracterizado por el tipo de tareas de cada SE. El mismo Stefik menciona que: el examinar las caracteristicas que se atribuyen a las tareas genericas que realizan los SE, ayudaran a entender la dificultad que tiene el experto* en su razonamiento. Por ende, las dificultades proporcionaran una guia para mostrar las relevancias arquitectonicas de los SE.

Para que fuera más fructífero el análisis de las estructuras de los problemas y su complejidad, se considero una clasificación [7] con los siguientes elementos; dirección de búsqueda, control y transformación de espacios de búsqueda. Los casos se ilustran con ejemplos típicos de SE, que se analizan en términos de su funcion,propósito y problemas claves que tienen.

En la tabla 1 se muestran tipos de funciones y significados, para los diferentes sistemas que se presenten y se ubiquen adecuadamente en las tareas que se desempeñen. En la tabla 2 se muestra un contraste de la clasificación de SE en términos de la función que se desempeñe y una clasificación de la estructura de control que usan los mismos SE.

El esquema se forma de dos ideas básicas:

a) Encontrar formas para una búsqueda eficiente en el espacio.

b) Encontrar formas para transformar un espacio grande de búsqueda en pedazos pequeños manejables.

* En lo sucesivo cuando se menciona la palabra 'experto' se refiere al sistema o a un experto de carne y hueso.

FUNCION	SIGNIFICADO
INTERPRETACION	Interpretación para el análisis de datos para determinar su significado.
DIAGNOSTICO	Diagnóstico es un proceso de encontrar fallas en un sistema basado en la interpretación de datos con ruido potencial. O determinación de un edo. enfermo en un sistema vivo.
MONITOREO	Monitoreo significa continuamente interpretar señales, prendiendo alarmas cuando la intervención se requiera.
PREDICCION	Predicción significa pronosticar el curso del futuro que forma un modelo en base del presente y pasado.
PLANIFICACION	Un plan es un programa de acciones que puede ser llevado a cabo para alcanzar metas. Planeación significa crear planes.
DISEÑO	Diseño es la elaboración de especificaciones para crear objetos que satisfagan requerimientos particulares.

Tabla 1.

I.7. ELECCION DE DIRECCION DE SOLUCIONES.

El modelo de problemas y solución y sus métodos, organizan y controlan las etapas que se van tomando para la resolución de problemas. Un modelo muy poderoso y común, es el de encadenar reglas IF-THEN para formar alguna línea de razonamiento. Si el encadenamiento se inicia a partir de un conjunto de condiciones y se mueve hacia alguna conclusión, el método es llamado *encadenamiento hacia adelante*. Por otro lado, si la conclusión se conoce (meta a llevar a cabo), pero la trayectoria a la conclusión no es conocida, entonces trabajamos hacia atrás y el método se conoce como *encadenamiento hacia atrás*.

En realidad el conocimiento de una tarea es el que domina el curso de las etapas que se van tomando para la solución de problemas [7]. En algunos casos, cuando el conocimiento es bastante abstracto (dominio de un problema simbólico), la inferencia que se produce del modelo de abstracción a relaciones de más detalle, es llamado modelo de **manejo de inferencias**. Siempre y cuando uno se mueva de relaciones simbólicas más abstractas a relaciones menos abstractas, uno está generando expectaciones y el comportamiento del problema-solución es denominado **manejo de expectación**.

Con frecuencia [7], se tiene que la solución de los problemas se trabajan hacia arriba, es decir, de los detalles o de los datos del problema específico a niveles más altos de abstracción; en este caso las etapas en esta dirección se llaman **manejo de datos**. Si la etapa siguiente se elige en base de algún dato nuevo o en base a la última etapa tomada dentro del problema y solución, la actividad es llamada **manejo de eventos**.

Como se mencionó anteriormente, un SE consiste de tres componentes principales: un conjunto de reglas, una base de datos global y un interpretador de reglas. Las reglas actuan por medio de un patrón dentro de la base de datos global y depende de la dirección de búsqueda la cual se toma en ambos lados con las reglas de IF-THEN.

La aplicación de las reglas cambian el estado del sistema y de la base de datos, esto es, habilitando y deshabilitando algunas otras. El interprete de reglas usa una estrategia de control para encontrar las reglas disponibles y decidir que regla se va aplicar. Por lo regular estas estrategias son las conocidas como *Top-Down* (manejo de metas) y *Bottom Up* (manejo de datos). En el caso de una combinación, es necesario recurrir a un proceso de convergencia para poder unir las líneas 'opuestas' de razonamiento en algun punto intermedio y así poder ofrecer una solucion a los problemas.

1.7.1. ENCADENAMIENTO HACIA ADELANTE.

Cuando se tiene un conjunto de datos o de ideas básicas como punto de partida, el encadenamiento hacia adelante resulta ser una técnica natural para direccionar las soluciones de los problemas [7]. Esta metodología ha sido usada para SE en el área análisis de datos, diseño, diagnóstico y formación de conceptos. Como se ve, este tipo de encadenamiento entra en relación a nuestro objetivo.

1.7.2. ENCADENAMIENTO HACIA ATRAS.

Esta aproximación es aplicable cuando nuestro punto de partida es una meta o una hipotesis [7]. Planificacion es un buen ejemplo para este tipo de aproximación, debido a que la función del planificador es construir un plan para poder alcanzar las metas deseadas. Un planificador considerará sus metas sin el consumo de recursos excesivos o violación de restricciones. Si hay una meta en conflicto, el planificador tiene que establecer prioridades.

Existen algunos problemas claves con sistemas que se construyen en esta area.

1.- Los problemas de planificación son suficientemente grandes y complicados para que un planificador entienda inmediatamente todas las consecuencias de sus acciones.

2.- Si los detalles son abrumadores, el experto deberá enfocarse fundamentalmente a los puntos mas importantes.

3.- Con frecuencia el contexto del planificador no se conoce con exactitud, por lo que el planificador deberá operar con cierta incertidumbre.

4.- En problemas grandes y complejos existen fuertes interacciones entre los planes para las diferentes submetas. Un planificador deberá entender estas relaciones para salir adelante con las interacciones de las metas.

5.- Si el plan es llevado a cabo por múltiples ejecutantes, es necesario implementar un procedimiento de coordinación.

1.7.3. ENCADENAMIENTO HACIA ADELANTE Y HACIA ATRAS.

Cuando el espacio de búsqueda es grande, la técnica de doble búsqueda suele ser eficiente [7]. El método consiste en tomar un estado inicial y las metas o hipótesis siguiendo un proceso de convergencia para poder casar las soluciones en un punto intermedio.

El método que se usa es parecido al de relajación, Esta aproximación también es muy útil, cuando el espacio de búsqueda puede dividirse jerárquicamente. En tales casos, la búsqueda se combina apropiadamente en términos de *Top-Down* y *Bottom-Up*. Una búsqueda de tal naturaleza se aplica en forma particular a problemas complejos, incorporando además incertidumbre.

1.7.3.1 RAZONAMIENTO EN LA PRESENCIA DE INCERTIDUMBRE.

En muchos casos las soluciones de los problemas se conducen en presencia de incertidumbre en los datos o en el conocimiento. Para este tipo de problemas es posible utilizar técnicas numéricas, o también, las incertidumbres pueden ser manejadas con una aproximación de la forma de regreso de rastreo. El razonamiento en la presencia de incertidumbre sucede en ejemplos típicos de diagnóstico y análisis de datos.

I.8. VENTAJA DE LOS SISTEMAS EXPERTOS.

Una de las principales ventajas es que los SE captan la inteligencia y el juicio humano para utilizarlos en areas donde los recursos son escasos y por lo tanto muy apreciados. Esto es importante sobre todo en las grandes compañías de alta tecnología, donde es grande la demanda de expertos.

Se puede plantear el uso de la tecnología de SE si de algún modo se toma en cuenta el beneficio que aporta; sin embargo no es fácil ya que todavía no hay metodos establecidos para el desarrollo de nuevos sistemas. En algunas empresas con grandes tecnologías se han trabajado en formalizar y aplicar tales medidas. Como ventaja cabe mencionar que:

* Reduce los costos por necesitar menos personal capacitado.

* El acortamiento de tiempo de resolución de problemas.

* Disponibilidad de técnicas especializadas en un campo concreto

La tecnología de SE ofrece la oportunidad de desarrollar nuevos productos y servicios, incluyendo herramientas para construcción, y sistemas para aconsejar a usuarios.

CAPITULO 2

II. PROGRAMACION PARA SISTEMAS EXPERTOS

Los sistemas expertos pueden trabajar en computadora siempre y cuando sean de un lenguaje natural para sus aplicaciones que el usuario necesite. Para este capítulo se enfocará a un lenguaje en especial llamado *prolog*.

LENGUAJES DE ALTO Y BAJO NIVEL.

Existen diversos lenguajes de programación para diferentes aplicaciones y soluciones, esto se debe al tiempo de ejecución y a los resultados.

FORTRAN, BASIC, COBOL, C, y PASCAL son lenguajes de alto nivel de la cuarta generación. Estos lenguajes tienen herramientas para procesos numéricos, pero tienen una ineficiencia para ser usados como lenguajes de SE.

Otro de los lenguajes que se manejan al final de esta cuarta generación es DBASE III, en comparación con los otros lenguajes tiene una ventaja de comparación muy alta, ya que su propia base de datos lo hace un lenguaje de programación dinámico. Sin embargo no se puede comparar con los lenguajes de programación como lo es LISP y PROLOG.

II.2 LENGUAJES DE PROGRAMACION PARA SE.

Sistemas Expertos (SE) es el nombre dado a los programas que captan y usan el conocimiento de expertos para resolver difíciles problemas, aunque pueden construirse SE utilizando cualquier lenguaje es importante poder representar la expresión (casos reales por modelación).

Los SE incorporan control por programas, sin embargo la sentencia de estos lenguajes no contienen ninguna explicación explícita en cuanto a la secuencia de ejecución, lo que da a tales programas una apariencia de representación que los expertos tienen del problema, siendo por consiguiente fáciles de entender y mantener.

Muchos lenguajes de SE se basan en el concepto de relación para modelar casos reales aunque la tecnología sea diferente.

Una de las tareas más comprometidas para diseñar SE es la selección del lenguaje. se consideran a grandes rasgos, tres niveles de lenguajes.

PROLOG, LISP y FORTH son lenguajes naturales captables de procesos simbólicos. LISP se puede considerar como un lenguaje de *bajo nivel*, ya que su interprete para SE es muy pobre, LISP se apega mas a las reglas de busqueda dentro de la IA.

El lenguaje FORTH es un lenguaje de nivel *medio*, ya que tiene un lenguaje interpretativo de mas tecnica para el entendimiento de rutinas y de las reglas de inferencia, sin embargo su aplicacion es muy limitada ya que su estructura solo se apega a funciones de pianificacion y diseño (ver tabla 1).

El lenguaje que tiene un compilador más preciso en lo que se refiere a *prediccion, diagnostico e interpretacion* es PROLOG, PROgraming LOGic

Programar en Prolog, es una experiencia completamente diferente, ya que el usuario al programar debe de definir el dominio, la platica con la maquina, las reglas y las relaciones que pueden existir en con respecto a un cierto tema, y todo con una facilidad que el mismo compilador proporciona.

II.2.1 REQUERIMIENTOS PARA UN LENGUAJE DE SE.

Para construír un SE se requiere de un lenguaje certero que cumplan con lo siguiente:

1.- El lenguaje debe soportar la estructura del programa.

2.- Debe soportar procesos simbólicos, en otras palabras, se debe de apegar a las clausulas que se requieren para ciertas variables.

3.- Debe de manejar reglas de inferencias, hipótesis y relacion con las mismas reglas.

II.3. INTRODUCCION AL LENGUAJE PROLOG.

El lenguaje de programación Prolog soporta razonamientos simbólicos, y hace posible que la computadora funcione inteligentemente y entenderse con el humano, hace tambien el papel de un experto Médico.

La ventaja que tiene Turbo Prolog es que soluciona los problemas mas facilmente y mas eficientes que otros lenguajes. Una computadora que usa Turbo Prolog funciona 'inteligentemente' ya que es como si el usuario estuviera con un experto humano.

II.4. BREVE HISTORIA DE PROLOG

Durante los años 1970's Prolog logró su fama en Europa por sus aplicaciones en Inteligencia Artificial. En los Estados Unidos, el Lenguaje que se conocia para el manejo de IA era LISP (LIst Programing). LISP se consideraba como uno de los lenguajes de más poder dentro de las aplicaciones en el campo de la IA, pero se dificultaba para leer y entender, Prolog no era el caso.

Durante esa epoca, Ingenieros, Doctores y Físicos trataron de modificar a LISP para que fuera en lenguaje de aplicación para Sistemas Expertos, pero se dieron cuenta de que se dificultaba para el dominio del *'Lenguaje Natural'* y que se sus funciones eran limitadas.

Poco a poco las microcomputadoras fueron ganando terreno y popularidad, y poco despues comenzaron a funcionar las versiones de Prolog. Sin embargo su interprete de Prolog abarcaba gran parte de la memoria de las microcomputadoras de aquel tiempo, lo que origino crear primeramente un microprocesador con una memoria extensa y un tiempo de respuesta rapida.

El ambiente fué cambiando, y en 1981 en la **FIRST INTERNATIONAL CONFERENCE ON FIFTH GENERATION SYSTEMS** en Tokio Japón, los japoneses ganaron la competencia a los estadounidenses en el mercado de la computación formando una nueva tecnología. Crearon una nueva corriente tecnológica con un nuevo Hardware y Software para los años 90's, es decir comenzaron a trabajar en lo que es la **Quinta Generación**, en lenguajes de computadoras para nuevos sistemas de Prolog.

Como resultado, los japoneses iniciaron desarrollarse alrededor del mundo en lo que se refiere al campo de la IA y de los SE. Hoy en día las versiones de Prolog tienen un poder que excede fronteras. Prolog es uno de los mejores lenguajes de aplicación en lo que se refiere a Sistemas Expertos.

II.5. QUE ES TURBO PROLOG ?

El nombre de Prolog se deriva del nombre del *PROgraming in LOGic*. Este lenguaje fué desarrollado en 1972 por Alain Colmenrauer and P. Roussel en la Universidad de Marseilles en Francia [9].

Prolog es único en la habilidad de Inferencias (derivación del razonamiento), factores y conclusiones desde otros factores. Para enlistar los problemas en Prolog, se describe el problema lógico tal y como es en el medio ambiente su base de conocimiento se alimenta del dialogo que realiza con el usuario.

El primer compilador de Prolog fué desarrollado en la Universidad de Edinburgh como un lenguaje experimental de la Quinta Generación. Hoy en día Prolog fue adoptado por Japón para el desarrollo de proyectos de la Quinta Generación, e incluso su compilador fué desarrollado más potente por los mismos japoneses a tal grado que surgió un compilador especial llamado *Turbo Prolog*.

II.9.1 EL PODER DE TURBO PROLOG.

La base de datos de Prolog es una característica del lenguaje natural ideal para solucionar problemas, su estructura es tan bien que soluciona lo desconocido por el usuario.

Si el dato es conocido, Prolog lo trabaja como una referencia. El uso de la *meta* o del *hipótesis*, son usados por su lógica para el razonamiento y así lograr una solución a los problemas.

Las características de procedimientos y lenguajes orientados están contenidos en la tabla 2.

LENGUAJES DE PROCESO BASIC, COBOL, PASCAL	LENGUAJE NATURAL PROLOG, LISP
USA PREVIAMENTE LA DEFINICION DE PROCESOS PARA LA SOLUCION DE PROBLEMAS.	MANEJO DE HEURISTICA PARA LA SOLUCION DE PROBLEMAS
MAS EFICIENTE PARA PROCEDIMIENTOS NUMERICOS	MAS EFICIENTE PARA UN RAZONAMIENTO FORMAL
LOS SISTEMAS SON MANEJADOS POR PROGRAMADORES	LOS SISTEMAS SON MANEJADOS POR LA INGENIERIA DEL CONOCIMIENTO
USA ESTRUCTURAS DE PROGRAMACION	INTERACTUA Y DESARROLLA CICLOS

TABLA 2

II.8 VENTAJAS Y DESVENTAJAS.

Prolog tiene pocas desventajas y que no es posible continuar sin antes mencionarlas:

1. El órden de reglas y de hechos es importante para el significado.

2. Prolog no tiene verdaderamente procedimientos.

3. Todas las reglas residen el la memoria de la computadora. El número de reglas del SE pueden usuarse si la memoria de la computadora lo permite.

4. Cuando se usan otras versiones de prolog (incluso de Turbo Prolog) la metodología puede cambiar así como las reglas y la extensión de la memoria, pero se puede alternar virtualmente con un programa corto.

5. Cuando el usuario va aprendiendo más a programar en Turbo Prolog, se dará cuenta de otras desventajas que existen.

Las ventajas son las siguientes:

1. Soluciona problemas con gran facilidad

2. Su lenguaje es natural (entendible)

3. Simula a un experto

4. Sus Procedimientos de respuesta son rápidas.

II.9. FUTUROS SISTEMAS EXPERTOS (APLICACIONES).

Para poder construir SE capaces de resolver una gama total de problemas de un dominio. Tales sistemas necesitan integrar con capacidad razonadora una amplia variedad de conocimientos sobre diversas cuestiones. La mayoria de los SE utilizan bases de conocimientos empíricos, es decir, fundados en la experiencia. Esto tiene como ventaja la eficiencia en el cálculo, y con frecuencia es la única solucion cuando no se dispone de conocimientos de tipo casual (es decir, del efecto producido en un modelo basico del problema), como ocurre en un diagnóstico médico. Los SE de base empirica son satisfactorios cuando se poseen experiencias sobre una gama total de problemas detectados. Pero son inadecuados para problemas nuevos, ante los cuales suelen reaccionar abandonando y llamando al experto. Tambien se ven limitadas las bases de conocimientos si se requiere una detallada explicación del método de resolución de problemas, ya que estas solo pueden expresarse por asociaciones empíricas y no por referencias a otras causas.

Los expertos se valen del conocimiento empírico para resolver problemas por ser cognoscitivamente más facil de y mas rapido que el conocimiento casual (si-entonces), el cual razona desde los principios primarios. Sin embargo cuando los expertos se presentan con problemas nuevos estos no abandonan, sino que utilizan sus conocimientos de estructura, comportamiento y causa para llegar a una solucion. Por lo tanto los futuros SE combinaran ambos tipos de procedimientos.

Los SE se hacen más complejos, al paso que la tecnología posibilita prestar nuevos y mejores servicios a la industria. Esto salta a la vista si nos referimos a la mayor extension de programas desarrollados y a la densidad de transistores en un CHIP (ver figura 6), exponiendo asimismo algo sobre la mayor potencia y funcionalidad ofrecida a los usuarios.

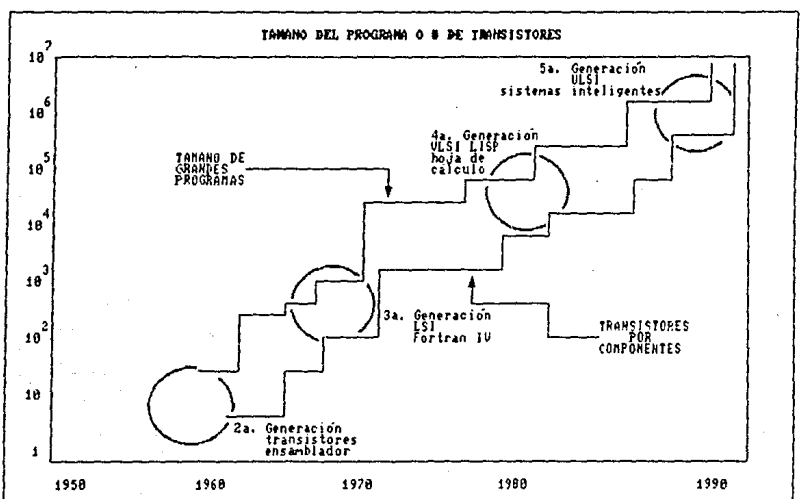


FIG. 6.- CRECIMIENTO DE LA COMPLEJIDAD FISICA Y LOGICA DE LAS COMPUTADORAS A LO LARGO DE LAS CUATRO DECADAS.

Los sistemas de la próxima generación, (ya en desarrollo), ofrecerán posibilidades todavía mayores, y por primera vez se podrá gestionar tal complejidad por medio de sistemas basados en inteligencia artificial.

De esta combinación de equipos y programas y de las diversas tecnologías que lo hacen posibles (ver fig. 7), de ahí nacerán los futuros sistemas inteligentes.

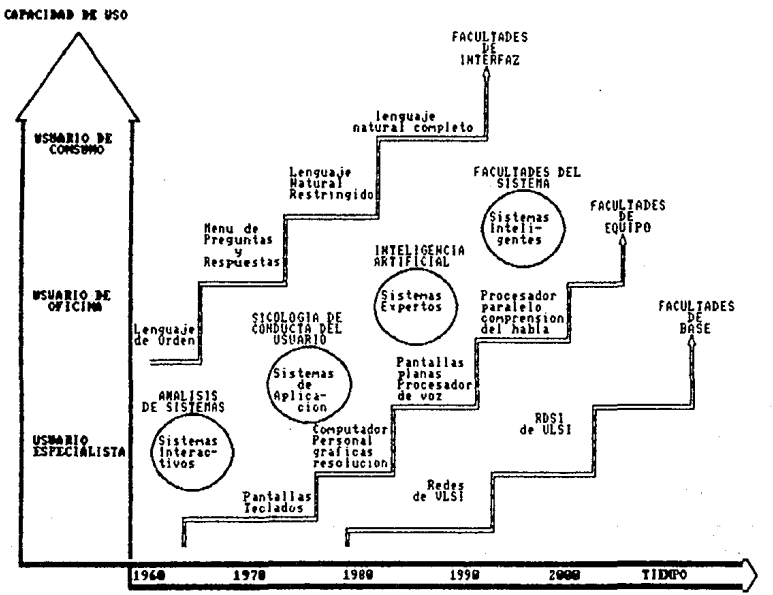


FIG. 7 Desarrollo de los Sistemas Inteligentes del proximo siglo.

CAPITULO 3

III. SISTEMAS EXPERTOS BASADOS EN TURBO PROLOG.

En los capítulos anteriores se habló del lenguaje natural, .stemas Expertos y de Turbo Prolog. En esta sección entenderemos ›s aspectos basicos de como se maneja el Turbo Prolog para el rsarrollo, creación, edición y ejecución de SE, así como sus ›mandos principales. No es conveniente expanderse ya que solo se ·etende dar a conocer el paquete aplicado al desarrollo del .stema y no como un curso de éste.

:I.1 ENTRANDO A TURBO PROLOG.

Turbo Prolog está contenido en 2 diskettes de 5¼ pulgadas, ı la cual el disco 1 contiene el Sistema del programa, mientras ıe el disco 2 son programas de aplicación y algunos ejemplos de ·ogramación basica a compleja de SE.

Para activar el programa Turbo Prolog, solo basta dar la ıstrucción siguiente en el disco duro de su PC:

```
>PROLOG
```

Si no está instalado en disco duro puede usar el disco 1 del .stema en la unidad A>. Una vez tecleada la instrucción arriba ›strada, Prolog desplegara su pantalla de presentación.

```
                    Turbo Prolog

              Copyright (c) 1986
            Borland International
          TURBO PROLOG version 1.1


                    Configuration
  Work          File          WORK.PRO
  PRO           directory     A:\
  OBJ           directory     A:\
  EXE           directory     A:\
  Turbo         directory     A:\

                Press the SPACE bar
```

FIG 3.1 ENTRADA AL SISTEMA DE TURBO PROLOG

III.2. PRESENTACION DE PANTALLAS.

Después de teclear la barra espaciadora, Turbo Prolog muestra cuatro ventanas, cada ventana tiene un uso específico que se definen a continuacion:

EDITOR.- Ventana en la cual se podrá editar el programa.

DIALOG.- Ventana que presenta las corrida de un programa, por lo regular es un diálogo.

MESSAGE.- Ventana que visualiza los mensajes de error del sistema Prolog.

TRACE.- Ventana que se usa para localizar problemas en un programa.

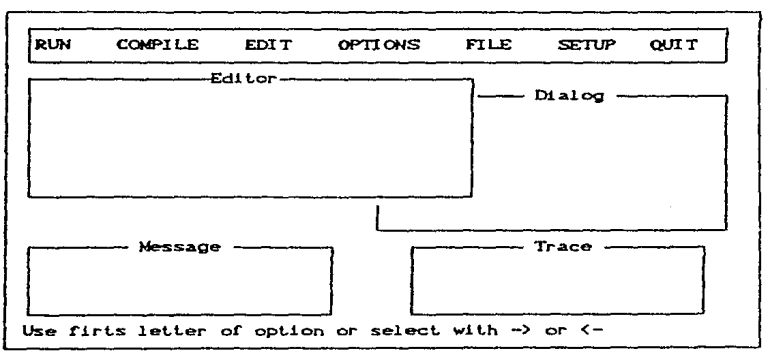


FIG 3.2 PANTALLA PRINCIPAL DE TURBO PROLOG.

III.3. LAS OPCIONES.

En la parte superior de la pantalla se aprecian siete opciones que se pueden elegír con las flechas del cursor -> o <-, o con la primera letra de la opción R, C, E, O, F, S, y Q. Cada opción tiene una función específica:

EDIT.- Crea un nuevo programa o edita los programas ya existentes.

Al seleccionar esta opción el *cursor* aparecerá en el cuadro de *edición*, esto quiere decir que ya podemos comenzar a elaborar un programa en prolog.

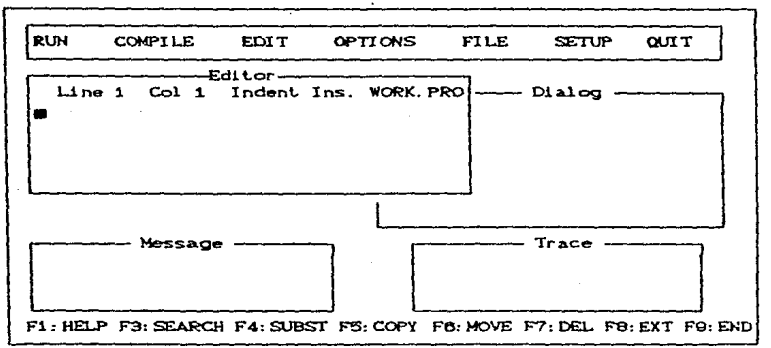


FIG 3.3 PANTALLA DE EDICION.

FILES.- Carga un programa desde el disco a la memoria o salva un programa en memoria en el disco flexible. Esta opción presenta otras utilerías como son *renombrar*, *copiar*, *borrar*, etc.

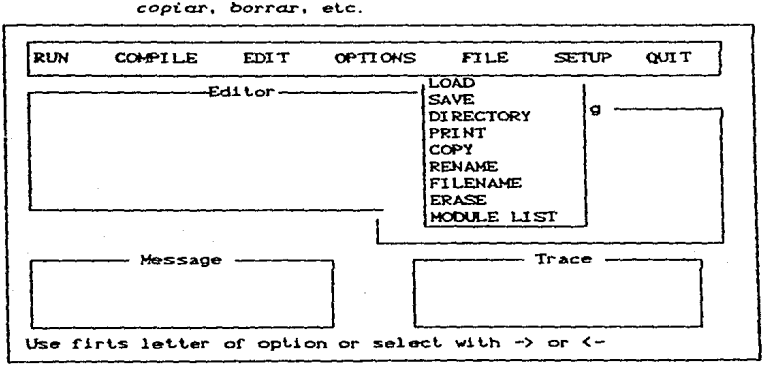


FIG. 3.4 OPCIONES DE FILES

COMPILE. - Compíla un programa en memoria para su ejecución.

RUN. - Compíla y corre un programa. Si el programa es corto, la *corrida* aparecerá en el cuadro de Dialog.

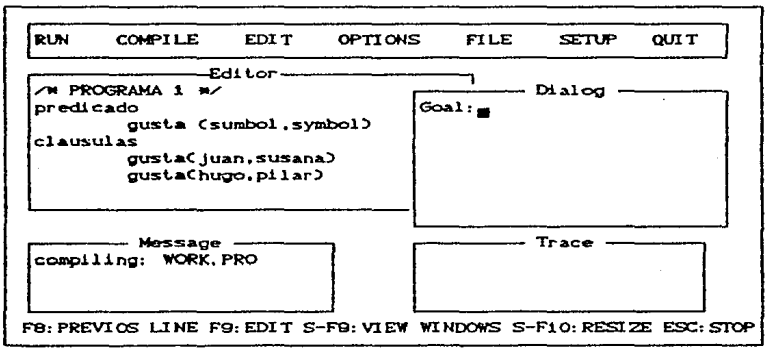


FIG. 5.5 PANTALLA DE EJECUCION DE UN PROGRAMA.

OPTIONS. - Selecciona el tipo de compilación para su uso.

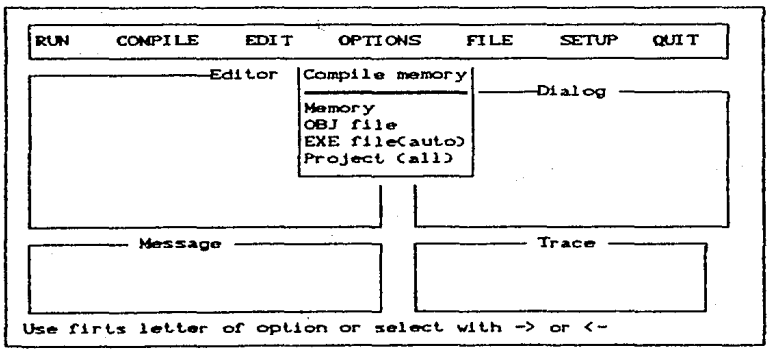


FIG 5.6 PANTALLA DE OPCIONES

SETUP. - Cambia los parámetros (Configuración).

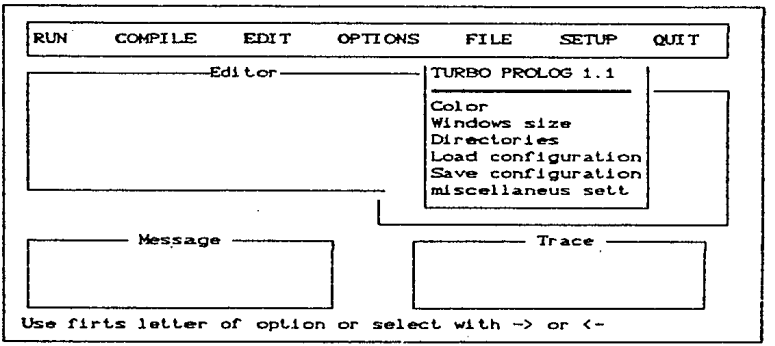


FIG. 3.7 PANTALLA DE LA OPCION -SETUP-

QUIT. - Sale de Turbo Prolog y regresa al Sistema Operativo. Si hubo una modificación en el programa, PROLOG preguntará si desea salvar el programa, en caso contrario regresará automáticamente al sistema operativo.

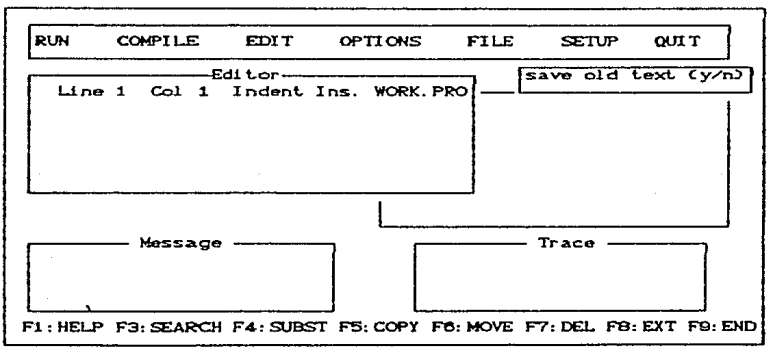


FIG. 3.8 OPCION -QUIT-

III.4. FACTORES, OBJETOS Y PREDICADOS.

Para la elaboración de un Sistema Experto en TURBO PROLOG es necesario conocer las reglas de programación, como son los *factores de expresión.*

III.4.1. FACTORES DE EXPRESION.

Turbo Prolog permite describir factores como una relación simbólica. Por ejemplo para un enunciado en español:

EL RADIO ESTA DESCOMPUESTO

En la siguiente expresión, Turbo Prolog lo maneja como:

ESTA(RADIO,DESCOMPUESTO).

Esta expresión en Turbo Prolog es llamada *una CLAUSULA.* A continuación se muestra una forma de relacionar palabras con sus factores.

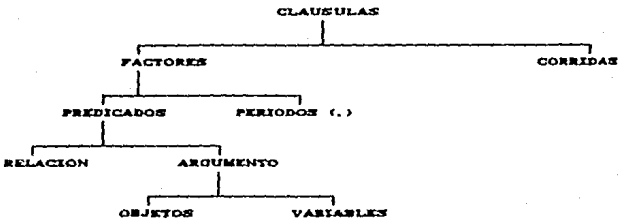
factor	relacion
tiene(juan,computadora)	tiene
es(perro,colie)	es
gusta(susana,chocolate)	gusta
está(beto,casado)	está

III.4.2 LOS OBJETOS.

A continuación veremos como Turbo Prolog maneja las expresiones de los *objetos* como clausulas para Prolog.

```
Español:  Juan es un empleado.
Prolog:   Empleado(juan).

Español:  beto esta casado con mary
Prolog:   casado_con(beto,mary)

Español:  el radio esta defectuoso
Prolog:   Diagnóstico(radio,defectuoso)
```

Como hemos observado existe una relación con las clausulas, los factores de expresión, los predicados, etc. Para que sea más claro se presenta a continuación una forma de entender las reglas:




```
Español:  Juan tiene 10 años de edad
Prolog:   edad(Juan,10).

Español:  beto gusta jugar.
Prolog:   gusta(beto,jugar).

Español:  beto gusta carros y computadoras.
Prolog:   gusta(beto,carros) y
          gusta(beto,computadoras).
```

No importa el órden de la expresión en Prolog, siempre es válida, ejemplo:

```
Español:  Juan gusta a mary.
Prolog:   gusta(juan,mary).
                  o
          gusta(mary,juan).
```

III.4.3. TIPOS DE DOMINIO.

En todas las expresiones existen diferentes formas de definir un enunciado (en este caso las expresiones), es decir si es de tipo *caracter, numérico, entero,* etc. En Turbo Prolog existen seis tipos de objetos básicos que a continuación se muestran:

char	caracter.
integer	número entero (desde -32,768 a 32,768)
real	número exponencial (desde $1e^{-307}$ a $1e^{308}$)
string	secuencia de un caracter (un comentario).
symbol	secuencia de caracter de letras, números, etc.
file	nombre simbólico de un archivo.

Ejemplo:

```
componente,status = symbol
```

Cabe mencionar que la palabra *componente* y *status* deben de ir separados por una coma (,) y al final el tipo de dominio que se define.

Un programa en Turbo Prolog debe de llevar una secuencia de la siguiente manera:

1.- Dominio (domain)

2.- Predicados (predicates)

3.- clausulas (clauses)

A continuación se muestra un simple programa de aplicación en donde muestra el uso de las expresiones en Turbo Prolog.

```
/* EJEMPLO PARA USO MEDICO */

Domains
        causa,indicación = symbol

Predicates
        síntoma(causa,indicación)

Clauses
        síntoma(fiebre_alta)
        síntoma(fiebre_baja)
        síntoma(frio,ronchas,acne_cuerpo)
        síntoma(resfrio,calentura)
        síntoma(garganta_congestionada,gripe)
```

En pantalla de Turbo Prolog aparece de la siguiente manera:

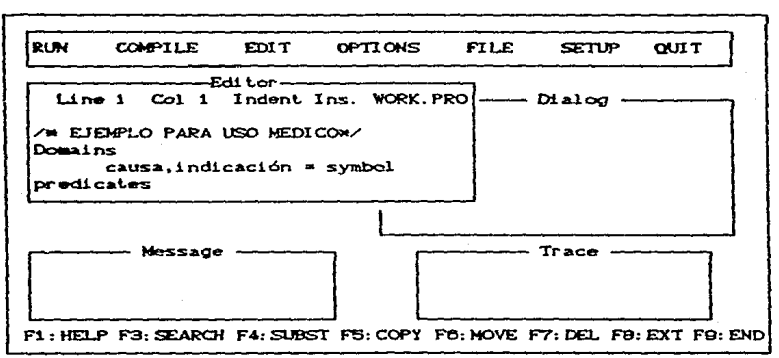


FIG 2.9 DESPLIEGUE DEL PROGRAMA EN TURBO PROLOG

Al ejecutar el programa en la parte de *dialog* aparece la palabra *GOAL* (comienza dialogo). Ejemplo:

```
goal:síntoma(resfrio,calentura)
true
goal:
```

Esto quiere decir que se pregunta si el paciente padece frío y/o calentura, la respuesta de Prolog es *cierta*, ya que lo tiene definido en sus clausulas.

```
goal:síntoma(resfrío,calentura)          el síntoma es frio y/o
                                         calentura
true                                     CIERTO

goal:
```

Si al preguntar "el paciente padece de vómito y calentura", Prolog responderá *False*. Esto quiere decir que tiene definido ese síntoma en sus clausulas. Ejemplo:

```
goal: síntoma(vómito,calentura)
false
goal:
```

III.4.4. REGLAS.

Las reglas en Turbo Prolog son las condiciones de una expresión, consideremos el ejemplo:

SI Juan está debil

Y tiene gripe

ENTOCES está resfriado.

Esta expresión en Turbo Prolog es de la siguiente manera:

```
hipótesis(resfriado)si
        síntoma(gripe) y
        síntoma(debil).
```

III.4.5. VARIABLES EN REGLAS.

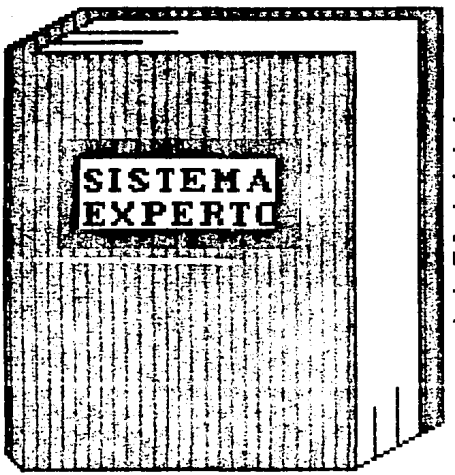
Podemos usar variables en reglas dentro de una expresión en Turbo Prolog, ejemplo:

```
hipótesis(resfriado):-
  síntoma(fiebre),
  síntoma(debil)
  síntoma(gripe),
  síntoma(dolor_cabeza).
```

Es decir, todos los síntomas como la *fiebre*, *debil*, *gripe* y *dolor de cabeza* son las variables que dan como solución la *hipótesis*.

Las variables en reglas son las expresiones que se utilizarán dentro del diseño del SE, ya que se necesitan muchas reglas con diferentes variables y viceversa; para que el programa tenga la facilidad de obtener una repuesta dentro de sus reglas de expresión.

PARTE II



DISEÑO DEL SISTEMA EXPERTO

IV.- APLICACION DEL SISTEMA EXPERTO AL DIAGNOSTICO DE FALLAS EN EQUIPO PC.

En esta sección se decribe la arquitectura básica de una computadora personal, en terminos intermedios, es decir no se trata de hacer un manual de usuario del MS-DOS, ni tampoco adentrar a detalle sobre circuiteria y diseño lógico, solo se trata de explicar como funciona las partes fundamentales del PC, como son los periféricos y como responden al usuario, cuales son sus fallas, etc. Solo así se puede conocer las fallas principales, para después poder dar mantenimiento al equipo (diagnóstico e hipótesis).

La computadora denominada PC (*Personal Computer*), está formada por tres partes físicamente separadas entre sí:

* La unidad central
* La pantalla
* El teclado

Las dos últimas sirven para que el usuario pueda comunicarse con el sistema. Al pulsar las teclas del teclado se pueden observar los caracteres que aparacen en la pantalla, como respuesta del propio PC. Ambos elementos realizan esta labor de comunicación con el mundo exterior (usuario) bajo el control de la unidad central, a la que van unidos mediante cables.

La computadora presenta externamente dos elementos de interés. En su parte frontal, las unidades de disketes (PC original) o de disketes y disco duro (PC-XT). Y en la parte posterior los diversos conectores a los que se enlazan los restantes elementos, así como la toma de corriente eléctrica.

El verdadero complejo (corazón) de la computadora se encuentra en el interior de la PC, denominada *unidad central de proceso (CPU)*. En la siguiente sección se explicará la arquitectura interna de la computadora personal.

IV.1.- ARQUITECTURA DE MAQUINAS.

Básicamente la computadora está compuesta de una serie de circuitos integrados (CI) conectados en una placa base denominada *Placa principal* ó *Tarjeta madre*. Esta placa base es la encargada de llevar el papel principal, ya que sobre ella destaca la CPU, la memoria, los buses, el manejo de los discos etc.

IV.1.1.- LA UNIDAD CENTRAL DE PROCESO (cpu).

A primera vista, destacan una serie de bloques y componentes. Mirando desde la parte frontal, se observa en la zona posterior izquierda una caja metálica con perforaciones concéntricas. Esta es la fuente de alimentación, cuya potencia parte de 65 watios en el modelo original y alcanza los 130 watios en el modelo XT. Al recibir la corriente alterna de la red, la transforma en contínua y se la suministra a los diversos componentes. A este efecto, la pantalla es la única excepción. El monitor monocromo tiene su propia fuente de alimentación, aunque reciba la corriente alterna a traves de la unidad central. En cuanto al monitor a color, lleva incorporada su propia toma de red.

En la zona frontal a la derecha puede observarse las unidades de diskete (PC original) o de diskete y disco duro (PC-XT). Ambas unidades de diskete son prácticamente idénticas y van conectadas mediante sendos cables a una tarjeta colocada de canto en la parte izquierda. A esta placa se le denomina *adaptador de diskete*. En cuanto al XT, la unidad de disco dura va sellada en el interior en una caja metálica. En este caso, la unidad de diskete se conecta en una tarjeta de disco duro a otra. Esta última placa se denomina *adaptador de disco duro*.

IV.1.2. LOS SLOTS.

En la parte izquierda de la unidad central pueden verse una serie de placas o tarjetas colocadas de canto. Estas placas van conectadas a la placa base mediante unos *zócalos* (llamados *slots de expansión*), situadas en la parte posterior. El PC normal presenta cinco *slots*, y el PC-XT ocho.

Todos los *slots* son indistingibles, en el sentido de que una tarjeta determinada pueda colocarse en cualquiers de los *slots* sin que ello afecte a su funcionamiento. En primer lugar, en el modelo XT los dos *slots* más cercanos a la parte central solo pueden colocarse placas de longitud corta. La razon es puramente física [1]. La colocación de la unidad de diskete hace que sea imposible insertar en ellos tarjetas largas. Por esta razón se habla a veces de *slots largos* (los seis restantes) y *slots cortos* (los dos mencionados). Este problema no existe en el PC original, donde todos los *slots* son largos.

IV.1.3. LAS CONEXIONES y PERIFERICOS.

Dentro del equipo básico de tarjetas, deben mencionarse ciertos detalles. El adaptador monocromo lleva incorporada una salida paralelo, de forma que si se desea conectar una impresora (con salida paralelo), no es necesaria ninguna tarjeta adicional. Por el contrario, el adaptador de gráficos/color no lleva dicha salida. En este caso, para poder conectar la misma impresora se necesita una tarjeta extra que posea esa salida en paralelo.

El modelo XT trae de fábrica una tarjeta adicional que no siempre se utiliza. Esta tarjeta posee una salida en serie y al ser de tamaño reducido va montada en el slot 8. Puede emplearse para comunicaciones con protocolo asíncrono o para conectar cualquier periférico que tenga una interface serie. Ejemplos típicos son los *modems*, *plotters*, *ciertas impresoras*, etc.

La PC puede tener conectada simultáneamente un monitor monocromo y un monitor a color. cada uno de ellos debe de ir cableado a su adaptador correspondiente, y se puede pasar de uno a otro mediante el comando MODE del DOS.

En la esquina del panel lateral izquierdo con el panel frontal puede verse una pieza redonda, colocada de canto. Se trata del *altavoz* del PC, que tiene una potencia máxima de 0.5 watios.

En cuanto al suelo de la unidad central, (ya mencionada), existe una placa base denominada *placa principal (tarjeta madre) ó tarjeta del sistema de terminología de I B M.*

IV.1.4. LA PLACA BASE.

De una forma u otra, todos los elementos de la computadora van conectados a la placa base. La pantalla, la impresora y cualquier otro dispositivo externo que se conecte a una tarjeta de expansión, estan conectandose a través de la placa base.

En realiadad, la placa base es la computadora. En ella están situadas, el microprocesador, la memoria, el reloj y demas componentes fundamentales. Lo demás, unidades de diskete, pantalla e impresora son periféricos. Son elementos imprescindibles para el conjunto, pero no para el núcleo del sistema.

IV.1.5. LA MEMORIA.

La memoria del sistema, monatada directamente sobre la placa base, la forman una serie de *chip* (circuitos integrados -CI-) situadas hacia la parte delantera a la izquierda, parcialmente ocultas por las tarjetas colocadas. En esta zona pueden verse hasta cuatro filas de componentes, etiquetadas a su izquierda con las palabras BANK 0, BANK 1, BANK 2 y BANK 3. Según la cantidad de memoria instalada en la computadora.

Cada fila supone 64 K de memoria RAM, por lo que si tiene ocupadas las cuatro, se tiene 256 K. Si se desea más memoria hay que recurrir a una tarjeta especial de ampliación, ya que la placa base el tope son 256 K.

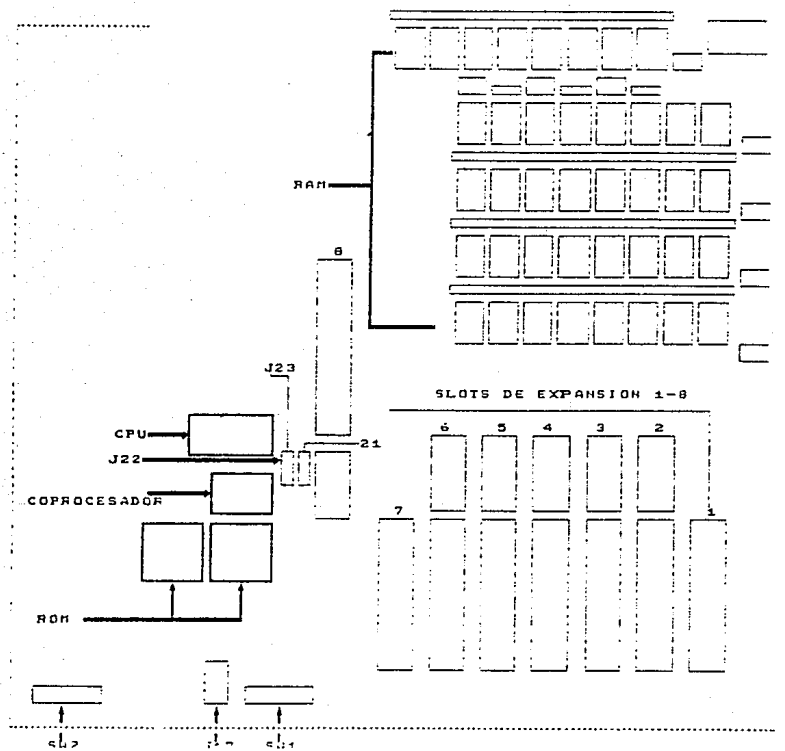
El papel de la memoria es fundamental, debido a que los programas siempre se ejecutan en ella. Aún cuando se tenga un programa almacenado en diskete, para poder ejecutarlo se debe cargar una copia suya en la memoria de la computadora. Lo que procesará entonces será la copia de la memoria, no la original que está grabada en el diskete.

IV.1.6. EL 8088.

Entre todos los componentes de la computadora, el más importante es un *chip* situada en la parte posterior hacia el centro, de forma alargada y tamaño superior a las demas, sobre su superficie puede leerse la cifra 8088.

Este componente es el *microprocesador* 8088, fabricado por la casa Intel, y es el verdadero cerebro del sistema. A menudo se le denomina *chip*, en lugar de microprocesador, esto es incorrecto, ya que todos los componentes son *chips* pero no son microprocesadores.

TARJETA PRINCIPAL O PLACA BASE



El 8088 es el responsable de la ejecución de cualquier programa que se procese en la computadora. Una vez cargada la copia en la memoria, se le dice al microprocesador -que más adelante se explica- cual es la dirección donde se encuentra la primer instrucción del programa. El 8088 transmite entonces una órden a la memoria para que le envíe el contenido de dicha dirección, o sea, la instrucción. Una vez recibida, la decodifica y la ejcuta. A continuación calcula la dirección de la siguiente instrucción del programa, la obtiene la memoria, la decodifica y la ejecuta. Este proceso se repite hasta la última instrucción del programa. es decir:

1. - Se carga en memoria una copia del programa a ejecutar.
2. - Se obtiene la dirección de la instrucción inicial.
3. - El 8088 le pide a la memoria el contenido de dicha dirección.
4. - La memoria se lo envía.
5. - El 8088 decodifica y ejecuta la instrucción.
6. - Si es la última, el proceso termina.
7. - Si no, el 8088 calcula la dirección de la siguiente instrucción y vuelve al paso 3.

De esta pequeña introducción se desprenden ya varias características del microprocesador. Por un lado, debe de llevar la cuenta de cuál es la dirección de memoria en la que se encuentra la siguiente instrucción a ejecutar. Por otro, debe ser capaz de decodificar la instrucción y ejecutar lo que se pida en ella. Y ademas, debe contar con algún tipo de memoria interna para almacenar la instrucción o el dato a procesar.

En cualquier caso, existe un diálogo casi constante entre el 8088 y la memoria. Para poder realizar la comunicación entre estos dos componentes y entre muchos otros, se utiliza el *bus*.

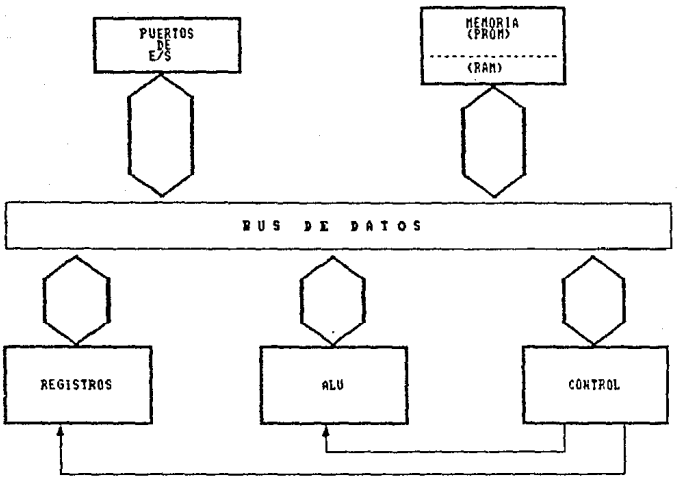
IV.1.7. EL BUS DEL PC.

Un bus puede verse como una serie de conexiones paralelas entre dos o más componentes, dentro del sistema de la computadora (ver. fig. 4.1).

El bus del PC está formado por un gran número de conexiones, puede dividirse conceptualmente en tres buses: de datos, de direcciones y de control (fig. 4.2). El 8088 envía las señales de sincronización y las órdenes a través de las conexiones que forman el *bus de control*, las direcciones de memoria o de otros tipos a las que se quiere acceder a través del *bus de direcciones*, y los valores que desea transferir el *bus de datos*.

El *bus de direcciones* del 8088 es un bus de 20 bits (20 conexiones), por lo que puede transportar direcciones hasta 220 dígitos binarios.

FIG.4.1 EL SISTEMA DE LA COMPUTADORA.



EN UN SISTEMA DE COMPUTADORA PC ESTANDAR, (LA SECCION DE CONTROL) INCLUYE LA LOGICA DE CONTROL Y LAS INSTRUCCIONES PARA DECODIFICAR Y EJECUTAR EL PROGRAMA ALMACENADO EN MEMORIA. LOS REGISTROS PROPORCIONAN LA SECCION DE CONTROL CON ALMACENAMIENTO TEMPORAL DE LA FORMA DE MEMORIA DE ACCESO ALEATORIO (RAM) Y SUS FUNCIONES ASOCIADAS. LA ALU REALIZA LAS OPERACIONES ARITMETICA LOGICAS BAJO LA SUPERVISION DE LA SECCION DE CONTROL. LOS PUERTOS DE E/S PROPORCIONAN ACCESO A LOS DISPOSITIVOS PERIFERICOS, TALES COMO EL TECLADO, MONITOR, IMPRESORA ETC.

Por su parte el *bus de datos* es algo más complicado. dentro del propio 8088 conectando su memoria interna (lo que se llama bus interno) tiene 16 bits (16 conexiones). Por lo tanto puede transportar de una sola vez valores de hasta 16 dígitos binarios. Sin embargo la conexión externa del 8088 con los *chips* de la memoria principal es de sólo 8 bits (8 conexiones). esto quiere decir que para enviar un valor de 16 bits desde el 8088 hasta la memoria, se necesitan en realidad dos envíos, uno con la primera mitad del número y el otro con la segunda mitad. Por el contrario, para transladar ese mismo valor de 16 bits de un punto a otro, en el interior del propio microprocesador, tan solo se necesita un envío.

Habitualmente se dice que un *microprocesador* es de tantos bits como su bus de datos indique. el 8088 suele considerarse como un chip de 16 bits, aunque en sentido estricto, es de 8/16 bits.

Existe un microprocesador de la misma familia, el 8086, que es de 16 bits, ya que su bus de datos, tanto interno como externo es de 16 conexiones. Como es lógico, el 8086 presenta una velocidad mayor que el 8088, ya que solo hace un envío en lugar de dos.

IV.1.8 EL COPROCESADOR ARITHETICO.

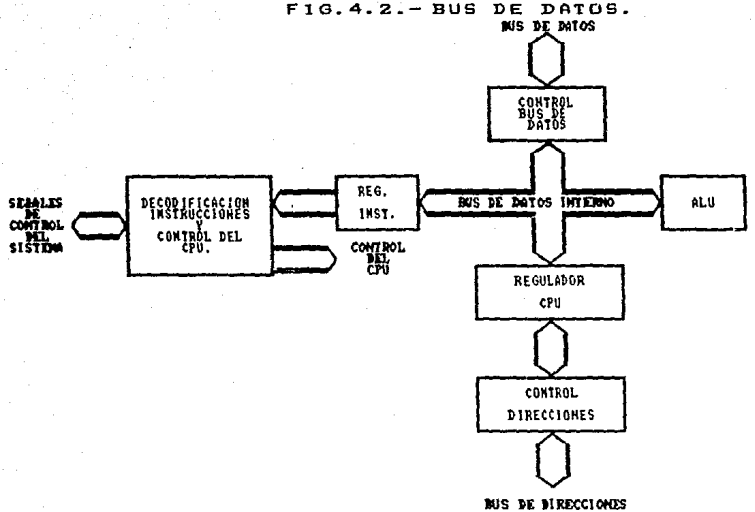
Otro componente, ya fuera del microprocesador es el chip 8087, denominado *coprocesador aritmético*. No se suministra desde el equipo básico, sino que debe se adquirirse por separado. En la placa base, a la derecha del 8088 puede verse un zócalo vacío del mismo tamaño, ahí es donde se monta el 8087. Como curiosidad, se dice que generalmente al solicitarlo, se recibe el propio 8087 junto con un nuevo 8088, de tal forma que al montarlo también hay que sustituir el 8088 original del equipo por el nuevo, sin embargo esto puede ser opcional.

El 8087 es un chip especialmente para realizar cálculos aritméticos a gran velocidad y con una notable precisión. Funciona de forma coordinada con el 8088, sustituyendolo a la hora de realizar dichos cálculos. Cuando se trabaja con determinados lenguajes ó paquetes de aplicación puede reducir el tiempo de ejecución. Sin embargo un programa normal no utilizará las ventajas del 8087 aunque esté presente.

IV.1.9. EL RELOJ.

Otro componente imprescindible es el chip del reloj 8284A, que se emplea para que el microprocesador pueda sincronizar todos sus procesos.

FIG.4.2.- BUS DE DATOS.



LA MAYOR PARTE DE LOS MICROPROCESADORES SE COMUNICAN MEDIANTE EL EMPLEO DE UN BUS DE DATOS. LA MAYORIA DE LOS BUSES SON BIDIRECCIONALES, CAPACES DE TRANSMITIR DATOS A, Y DESDE, LA CPU, ALMACENAMIENTO DE MEMORIA Y DISPOSITIVOS PERIFERICOS. EL BUS DE DATOS COMPRENDE 16 LINEAS DE DATOS BIDIRECCIONALES. EL BUS DE TEMPORIZACION PROPORCIONA LAS LAS SEÑALES DE RELOJ BASICA DEL SISTEMA ASI COMO LAS SEÑALES (STROBE) DE SELECCION DE DATOS Y DIRECIONES, QUE INDICAN CUANDO HAY DATOS VALIDOS EN EL BUS.

Supongamos que el microprocesador ha enviado a la memoria una petición de lectura. Antes de poder recoger del *bus de datos* el valor solicitado, el 8088 deberá esperar el tiempo necesario para que la memoria realice el acceso. Ese tiempo de espera se calcula en base al reloj. Si por ejemplo se sabe que el tiempo de acceso de la memoria es de 800 nanosegundos y el reloj oscila (hace un tic) cada 200 nanosegundos, el 8088 deberá esperar 4 ciclos de reloj (4 tics) antes de recoger el valor del *bus de datos*. de esta forma, la computadora se diseña para que opere en función del reloj.

El chip 8284A contiene un cristal de cuarzo que oscila, en el caso de la PC, a una velocidad de 14,31818 Megaherzt (Mhz). Esto quiere decir que realiza 14.318.180 tic por segundo. esta frecuencia se divide por tres para obtener la velocidad adecuada para el microprocesador, que es de 4,77 Mhz. Entonces, a efectos del 8088, el reloj de la PC hace casi 5 millones de tics por segundo. Visto de forma inversa, un tic del reloj dura 210 nanosegundos (un nanosegundo es la milmillonesima parte de un segundo).

La velocidad del reloj es la característica que elige el fabricante a la hora de diseñar el equipo.

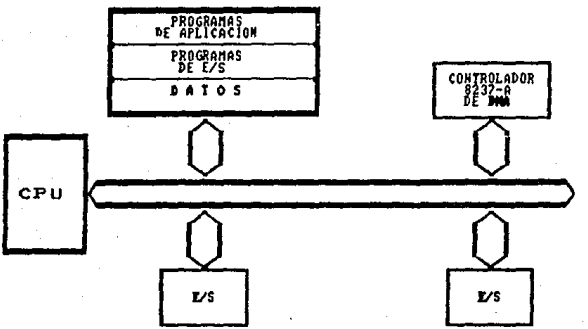
En el mercado existen diversos compatibles más rápidos que el PC original, debido entre otras razones a que se les ha colocado el reloj a una frecuencia superior a los 4,77 Mhz. Esta mejora puede aumentarse, como ya se ha visto, utilizando, como microprocesador un 8086, en lugar de un 8088.

IV.1.10. EL CONTROLADOR DE DMA.

El funcionamiento de la PC está fuertemente centralizado. El microprocesador controla todo el sistema y prácticamente todos los datos tienen que pasar a través de él. Existen casos tales como la carga de un programa en memoria para ser ejecutado. Hay que transferir una enorme cantidad de bytes desde el disco duro o diskette a la memoria RAM del PC.

Para este tipo de transferencis masivas de datos se utiliza un chip 8237A, conocido como *controlador de DMA* (direct memory access). Cuando se van a realizar uno de estas transferencias de datos, el 8088 cede el control del bus al 8237A y éste se encarga de pasar los datos directamente, por ejemplo del diskette a la memoria. El 8237A tiene cuatro canales distintos. Uno de ellos se utiliza internamente para el refresco de memoria. Los otros tres se emplean para la transferencia directa de los datos entre dispositivos de E/S y la memoria (o viceversa). Habitualmente, dos de ellos son utilizados por los controladores de disco duro y diskette, quedando el tercero libre para su uso por otros periféricos.

FIG.4.3 CONTROLADOR DEL DMA



IV.1.11. LA DIFERENCIA CON EL PC-AT.

Hoy en día, bajo el sistema operativo MS-DOS, el PC-AT es poco más que un PC muy acelerado. La promesa de que el sistema operativo OS/2 de la nueva familia de computadoras personales de IBM podrá correr en las máquinas equipadas con un 80286, parece abrir por fin importantes posibilidades para este modelo. Pero en la actualidad, puede decirse que en cierto modo, está infrautilizado.

En cualquier caso, se resume a continuación las principales características propias que presenta el modelo AT.

- El procesador del PC-AT no es un 8088, sino el 80286, también de la casa *intel*. Este chip es un verdadero 16 bits, es decir, que su *bus de datos*, tanto externo como interno es de 16 bits, su capacidad de direccionamiento alcanza hasta los 16 M.

- La frecuencia de trabajo del 80286 puede ser según el modelo del AT, de 6 u 8 MHz, frente a los 4,77 de los modelos básicos.

- La memoria con que viene equipada de fábrica en la placa base pasa a ser de 51 K (el doble de los modelos anteriores).

- El disco duro ha pasado a ser de 20 ó 30 M, según el modelo de AT, por 10 M del disco del XT.

- interiormente se ha ampliado la memoria ROM (64K) y el *bus* ha pasado de 62 a 98 líneas.

- También se ha colocado un segundo 8237A (el controlador de DMA). El nuevo coprocesador aritmético, aún más potente, se denomina, siguiendo la tradición, 80287.

Estos cambios y otros que no se han mencionado se traducen a una velocidad de proceso notablemente superior a la de los modelos basados en el 8088.

IV.1.12.- EL DIRECCIONAMIENTO.

la cantidad de memoria de una computadora es un dato fundamental para evaluar sus posibilidades. Como se sabe, los programas se almacenan habitualmente en discos o diskettes, pero unicamente se pueden ejecutar desde la memoria. Por ello para procesar cualquier programa, el primer paso es cargar una copia del mismo en la memoria del equipo.

La computadora tiene un límite máximo de memoria, impuesto por el microprocesador que le gobierna, en el caso del PC básico o del XT es un INTEL-8088.

La memoria de una computadora puede verse como un conjunto de cajones, estando formado cada cajón por un byte de información. El sistema trabajará con esa memoria, leyendo y cambiando el contenido de dichos cajones o bytes cuantas veces sea preciso. Por ello el microprocesador debe ser capaz de acceder individualmente a cada byte de la memoria. Para hacer posible es necesario otorgarle a cada uno un número que le sirva de identificación inequívoca. En otras palabras, es preciso que cada byte de la memoria tenga una dirección.

El 8088 posee un *bus de direcciones* de 20 bits, es decir, que pueda indicar una dirección mediante 20 señales en *on* u *off*. esto es equivalente a decir que cada dirección puede venir dada por un número de 20 dígitos binarios.

Por lo tanto el microprocesador del PC podrá direccionar tantos elementos de memoria como números posibles pueada formar con 20 dígitos binarios. La numeración de dichos elementos de memoria irá entonces desde el 0 (imaginemos veinte ceros) hasta el 11111111111111111111.

Trabajando en modo hexadecimal, las direcciones de la memoria oscilan desde 0 hasta FFFFF. Así pues, cualquier dirección de memoria puede expresarse mediante cinco dígitos hexadecimales.

Entonces, al sistema decimal, resulta que puede utilizarse 1,048,576 bytes de memoria, con direcciones oscilan desde 0 hasta 1,048,575. En conclusión la PC puede manejar algo más de un millón de bytes.

Dado que el número es bastante elevado, se suele expresar, en lugar de en bytes, en lo que se denomina *kilobytes*, o más abreviado *K* (o *Kb*) un K equivale a 1024 bytes. La computación rompe la regla de que un K no significa 1000, sino 1024.

El 8088 puede direccionar un máximo de 1024 K. Si este número es grande, existe otro aún mayor, puede emplearse una unidad superior de memoria que se denomina *megabytes* ó M. Un megabyte equivale a 1024 bytes o lo que es lo mismo a:

$$1024 \times 1024 = 1,048,576 \text{ bytes}$$

de nuevo aparece una unidad de medida algo extraña. En computación un mega no significa un millón, sino un poco más.

En cualquier caso, por fin se ha llegado a un número pequeño. la conclusión es que el 8088 puede direccionar hasta un mega de memoria, tal como se ve en la figura:

	DIRECCION		
	BINARIO	HEXA	DECIMAL
	00000000000000000000	00000	0000000
	00000000000000000001	00001	0000001
	: : : : : : : : : : : : : : : : : : : :	: : : : :	: : : : : : :
1,048,576	00000000010000000000	00400	0001024
=1024 k	00000000010000000001	00401	0001025
= 1 M	: : : : : : : : : : : : : : : : : : : :	: : : : :	: : : : : : :
	11111111111111111110	FFFFE	1048574
	11111111111111111111	FFFFF	1048575

IV.1.13 MEMORIA RAM Y ROM.

A grandes rasgos, pueden distingirse dos tipos de memoria en las computadoras personales, denominadas ROM (Read Only Memory) y RAM (Random Access Memory).

ROM (memoria de solo lectura) viene ya ocupada de fábrica con diversos programas. Como su propio nombre lo indica, puede ser leida cuantas veces se desee, pero su contenido no puede alterarse en ningún momento. Se dice de ella, que es una memoria no volátil, ya que la información que posee no se pierde al apagar la computadora.

La memoria RAM literalmente memoria de acceso aleatorio, pude leerse y modificarse un número limitado de veces, pero su contenido se pierde cada vez que se corta la corriente eléctrica. Se trata, pues, de una memoria volátil.

Es importante resaltar que la memoria ROM no puede alterarse por razones físicas. NO se trata de un mecanismo existente en el cual se impida escribir en determinadas direcciones (las que corresponde a elementos de la ROM), sino que simplemente el proceso electrónico que realiza la computadora para grabar en un elemento de memoria no funciona con el tipo de chip que se emplea en la memoria ROM.

Cuando se solicita la ejecución de un programa que está almacenado en diskette o disco duro, el sistema operativo de la computadora lo carga a la memoria RAM de forma que el microprocesador puede comenzar a procesar las instrucciones que lo componen.

Algo similar ocurre al ejecutar un programa de los que vienen ya pregrabados en la memoria ROM. Ahora ya no es necesario el paso previo de cargarlo en memoria, puesto que ya se haya en ella. Este programa podrá ser igualmente leido y escrito en cualquier posición de la memoria RAM pero, en cambio, nunca podrá alterarse a si mismo.

IV.1.14.- LOS PUERTOS.

Dentro de la computadora existen otro tipos de direccionamiento, que se denominan *puertos de entrada/salida.*

Como se sabe, la computadora está compuesto por una serie de componentes especializados que funcionan bajo el gobierno centralizado del microprocesador. Como ejemplo el teclado posee en su interior un chip denominado 8048, que es el responsable de identificar cada tecla que se pulse. De forma análoga, la unidad de diskettes incluye otro chip especial, que se encarga de gestionar todos los accesos a diskettes que le soliciten. Asimismo se pueden conectar diversas tarjetas de expansión, para diferentes usos. Ejemplos típicos son las diferentes placas de comunicación existentes en el mercado.

Cada dispositivo de E/S conectado a la computadora tienen asignados uno o varios números de puertos. Cuando el 8088 desea enviar un dato, coloca el *bus de direcciones* el número del puerto que quiere acceder, indica mediante el *bus de control* que esta vez la dirección expresada no es de memoria sino de un dispositivo de E/S y pone el valor concreto en el *bus de datos.*

Si lo que se desea es recibir un dato del dispositivo, el proceso es muy similar, se indica el número del puerto, se manda la señal de control, y una vez transcurrido el tiempo necesario, se recoge el dato deseado.

El PC posee hasta 65,536 puertos, si bién la mayor parte no están asignados a ningún dispositivo. Normalmente, cada componente utiliza varios puertos, o, como también se denomina a menudo, varias *direcciones de E/S.*

Aun cuando puede establecerse cierto paralelismo entre los puertos y la memoria, sus diferencias son notables. El contenido de una dirección concreta de memoria puede utilizarse en distintos momentos por diferentes programas. Por el contrario, el puerto número n siempre estará asignado a un determinado dispositivo.

IV.2. FUNCIONAMIENTO Y FALLAS EN MICROCOMPUTADORAS PC'S (HW/SW)

En esta sección, se verá lo que ocurre paso a paso al conectar la PC a la red eléctrica y pulsar el interruptor de encendido, así como las posibles fallas que puedan existir.

IV.2.1 EL PROCESO DE ARRANQUE.

Este proceso se denomina IPL ó *Initial Program Loading* (carga de programa inicial). Existe otro término como el *boot-strapping*.

En el PC existe dos formas de hacerlo. La primera consiste simplemente, en encender la computadora y se denomina *arranque en frío*. La segunda se efectúa con el equipo ya encendido, pulsando simultáneamente las teclas CTRL-ALT-DEL, y se denomina *arranque en caliente*.

En teoria ambos métodos conducen al mismo resultado: la inicialización completa del equipo. En la práctica sin embargo, hay ocasiones en las que se producen determinados errores de los que no es posible salir realizando el arranque en caliente. En estos casos es imprescindible apagar y volver a encender la computadora. Por lo demas, ambos procedimientos se pueden considerar válidos.

IV.2.2. LA ROM-BIOS.

La memoria ROM no puede alterarse en ningún momento y viene ya de fabrica con un contenido preficado, que suele denominarse ROM-BIOS (por ROM Basic Input Output System).

El programa contenido en la ROM-BIOS, atendiendo a su función, puede dividirse en dos partes fundamentales.

La primera de ellas tiene por finalidad el realizar un test para comprobar que el funcionamiento del hardware del PC es correcto. Esta parte es la primera en ejecutarse al encender el equipo y es, por ejemplo, durante el test cuando el modelo XT va mostrando en la esquina superior de la pantalla, sucesivos mensajes de XXX Kb OK, en incrementos de 16 K. Por el contrario, el modelo básico no muestra ningún mensaje similar, ya que su ROM-BIOS, anterior a la del XT, no lo tiene previsto.

La segunda parte del programa de la ROM-BIOS consiste en un juego de rutinas que proporcionan la interface entre el software y el hardware del PC. Dicho en otras palabras, estas rutinas consiguen que una órden dada por un programa para, por ejemplo, leer un caracter del teclado se lleve a cabo a través del hardware disponble físicamente. Naturalmente, estas rutinas, cada una con un proposito distinto, no se ejecutan al encender la computadora, sino que se limitan a estar es su puesto, disponible para cualquier momento en que se necesiten.

IV.2.3. EL ARRANQUE.

Al encender la computadora, el 8088 comienza a ejecutar el programa contenido en la ROM-BIOS, se realiza el chequeo interno, que suele denominarse con las siglas POST (Power On Selft Test), que significa *Auto chequeo Al Encender*, y se carga la tabla de interrupciones en la primera K de memoria RAM, con las direcciones de las rutinas de servicio básicas existentes.

Carga el registro de arranque (o boot-record) del diskette o disco duro. este proceso es de al siguiente forma:

1) Se intenta acceder al drive A (diskette).
   * si no hay en el un diskette, o si la puerta está abierta, se pasa al punto 2.
   * si hay un diskette, pero es defectuoso o no contiene el sistema operativo, se saca por pantalla el mensaje:

     NON-SYSTEM DISK OR DISK ERROR
     REPLACE AND STRIKE ANY KEY WHEN READY

     y al pulsar una tecla se vuelve al paso 1.
   * si hay un diskette y contiene el sistema operativo, se pasa al punto 4.
2) En caso de existir una unidad C (disco duro) se accede a ella.
   * si en dicha partición activa del disco duro no existe el sistema operativo, se pasa al punto 3.
   * si en dicha partición existe un sistema operativo se pasa al punto 4.

3) Se transfiere el control a la dirección F600:0000 dentro de la memoria ROM, donde hay un interprete de BASIC que viene de fabrica.

4) Este punto se ejecuta si ha encontrado un sistema operativo, ya sea en diskette o disco duro. En este caso, el sector 1 de la pista 0, cara 0, del medio que se trate, se encuentra el mencionado *boot-record* o registro de arranque. Este *boot-record*, es una pequeña porción de código que lee el disco duro o diskette, se carga en memoria y, finalmente, se transfiere el control.

El interprete de BASIC existente en memoria ROM se denomina habitualmente BASIC de disco, ya que el único periférico de almacenamiento de datos que soporta es un *magnetófono* de cintas especiales. Este interprete se identifica por el mensaje que aparece al cargarse:

```
IBM Personal computer BASIC
version CX.XX
```

donde C y X.XX indican la versión. El BASIC de disco constituye el núcleo de los otros dos interpretes de BASIC que se suministran en el diskette del sistema operativo. Por esta razón, en el IMB-PC es imprescindible la presencia del BASIC de disco apara el funcionamiento de los dos intérpretes del diskette. En ciertos compatibles, el interprete BASIC de la ROM no existe y por ello el intérprete del diskette es completo en sí mismo. Esta es la razón por la que los interpretes del IBM-PC pueden no funcionar en determinados compatibles.

En caso de cargarse este BASIC de disco, la única manera de salir de él es haciendo el IPL.

IV.2.4. EL DOS.

A partir de ahora la parte restante del proceso de arranque del sistema se realiza ya en memoria RAM.

El *boot-record* se encuentra en una porción del código que realiza la búsqueda, en el disco ó diskette de la unidad donde se ha hecho el IPL, de dos programas concretos, cuyos nombres son IBMBIO.COM e IBMDOS.COM, estos dos archivos se encuentran ocultos en discos y diskettes, razón por la cual no aparecen al ejecutar el comando DIR.

El primero de ellos, IBMBIO.COM, es un ejemplo de *programa residente*, ya que, una vez cargado por el registro de arranque, permanece en la memoria RAM hasta que se apague el equipo. Funciona de forma similar a la ROM-BIOS, es decir, que proporciona una serie de rutinas de servicio que deben poder ser llamadas en cualquier momento.

Otra función importante de este primere módulo consiste en localizar en la unidad de arranque un archivo llamado CONFIG.SYS, y procesarlo en caso de existir. Este archivo contiene básicamente dos tipos de órdenes, una encaminada a configurar ciertos parámetros del sistema (como el número de buffers de E/S que se mantendrá en memoria), y otra que identifica controladores de dispositivos que se desean tener disponibles.

En lugar de suministrar las rutinas adecuadas en una nueva versión de IBMBIO.COM, se proporcionan en forma de rutina controlador de dispositivo invocada desde un archivo CONFIG.SYS. Si se desea poder utilizar sus servicios, debe incluirse en el archivo CONFIG.SYS la siguiente línea:

```
DEVICE = ANSI.SYS
```

y, naturalmente debe de reinicializarse el sistema para que durante el arranque se incorpore el controlador deseado.

El segundo programa del sistema operativo IBMDOS.COM también tiene una característica de ser un programa residente. A igual que el anterior, suministra una serie de rutinas de servicio, solo que en este caso de un nivél más alto, para ser más claros, la diferencia que existe entre las rutinas ROM-BIOS, IBMBIO.COM y las del IBMDOS.COM podrían ser las existentes entre leer un sector concreto del diskette (petición de bajo nivel) o borrar un archivo del diskette (petición de nivel más alto).

Finalmente, el tercer programa, COMMAND.COM, habitualmente llamado *procesador de mandatos*, nos proporciona la interface directa del DOS con el usuario. Este programa es el responsable de hacer aparecer el *prompt* o indicador del DOS (A>, B>, C>) y de procesar lo que el usuario teclea en respuesta al discho prompt. Una vez pulsada la tecla <RET> ó (↵), el programa COMMAND.COM obtiene lo teclado y realiza una serie de comprobaciones.

Los archivos con extensión COM ó EXE son programas ejecutables, con ciertas diferencias entre sí. Los programas COM están en formato directamente cargable, con todas las direcciones ya en valor absoluto. Por ello, su carga es muy rápida y no necesitan de ningún preprocesamiento. En cambio, los programas EXE tienen aún direcciones en términos relativos, por lo que, al cargarlos se van convirtiendo las direcciones a los valores absolutos correspondientes. Como resultado, el proceso de carga es bastante lento en comparación al caso anterior.

La permanencia del programa COMMAND.COM en la memoria RAM es bastante especial. El programa se carga en memoria dividido en dos partes. Una de ellas se carga a continuación de los otros programas, IBMBIO e IBMDOS y permanece residente. La otra parte se carga en el otro extremo de la RAM y puede descargarse si se necesita toda la memoria para un determinado programa de usuario. En este caso, al terminar la ejecución del programa de usuario y devolver control a la parte residente del COMMAND.COM, se detectará que se ha descargado la otra parte y se procederá a cargarla de nuevo. Entonces, si se ha descargado parte del COMMAND.COM y al ir a recargarla no se encuentra el archivo en el disco o diskette por defecto, aparecerá el mensaje:

```
CANNOT FIND COMMAND.COM
INSERT DISKETTE AND PRESS ANY KEY
```

IV.2.4.1 El archivo AUTOEXEC.BAT

Una vez cargado en memoria los tres archivos que constituye el núcleo del DOS, el proceso de arranque del PC estará practicamente finalizado.

Lo que queda por realizar consiste en la búsqueda en la unidad desde la que ha arrancado de un archivo AUTOEXEC.BAT. Este es un archivo normal en el que se colocan los mandatos que se desean ejecutar al encender la PC, ejemplos son, la hora, fecha, adaptación del prompt, y ejecutar automáticamente cualquier archivo con extensión COM ó EXE.

El AUTOEXEC.BAT, como cualquier otro archivo de mandatos, es procesado por el COMMAND.COM y además como última labor del proceso de arranque.

CONCLUSION.

Conviene resumir, brevemente, la situación resulante al finalizar este laborioso proceso de arranque.

La computadora habrá presentado el mandato del DOS y estará esperando nuestra siguiente acción. En este momento, las herramientas del software que dispone el PC son las siguientes:

* El COMMAND.COM, responsable de leer nuestra órden y realizar lo necesario para poder ejecutar el comando deseado.

* Un juego de rutinas de servicio almacenadas en diversas localizaciones de la memoria, tanto en la ROM (dentro de ROM-BIOS) como en la RAM (dentro de IBMDOS ó IBMBIO.COM ó de los controladores de dispositivos cargados).

Este juego de herramientas permacenerá durante el periodo en que la computadora esté encendida, si bién, con la excepción obvia de la ROM-BIOS al sufrir diversos cambios y el COMMAND.COM al descargarse.

IV.2.4.2.- EL MAPA DE MEMORIA.

Una de las conclusiones que puede desprenderse de el proceso de arranque, es la necesidad de conocer como se estructura la memoria de PC y para que se utiliza cada parte, es decir, es necesario conocer el mapa de memoria del PC.

Como ya se explicó, el microprocesador 8088 puede direccionar hasta 1 M de memoria. dentro de este mega debe incluirse toda la memoria RAM y ROM que se vaya a emplear. Al momento de diseñar la PC, IBM decidió dedicar las primeras 640 K para la memoria RAM de usuario, y las restantes 384 K para usos internos de la computadora. Dentro de estas últimas se encuentra la memoria RAM utilizada para la pantalla de visualización de la memoria ROM suministrada*.

* Debe quedar claro en todo momento que no es lo mismo hablar de memoria direccionable que de memoria realmente instalada. El 8088 siempre es capaz de direccionar hasta 1 M de memoria, pero en la práctica, la PC puede tener diferentes cantidades de memoria.

Al hablar de la *memoria* de la PC (ejemplo 512 K), se esta haciendo referencia unicamente de la memoria RAM de usuario. En este sentido, una PC puede tener desde 64 hasta las 640 K mencionadas (ó más). El modelo original de IBM se puso a la venta con tan solo 64 K de memoria de usuario. Mas adelante se comenzó a suministrar los equipos con 256 K, que es el máximo que puede ponerse en la placa base del modelo original.

Por lo que respecta a las restantes 384 K, la cantidad realmente instalada también varía en unos equipos a otros. En primer lugar el tamaño de la memoria ROM ha ido creciendo con los nuevos modelos. De forma analógica, la memoria de visualización, va colocada físicamente en la tarjeta adaptadora, varía según se trate de un equipo con pantalla monocromo y color.

Como ya se ha mencionado, se destinan a la memoria RAM de usuario las primeras 640 K (de la dirección 0000:0000 a la 9000:FFFF). Sin embargo no todas estas 640 K (en caso de estar instaladas) están realmente libres para los programas de usuario.

La primera K de la *memoria* de (direcciones 0000:0000 a la 0000:03FF) se destina a la tabla de vectores de interrupciones. (Una interrupción quiere decir la suspensión de las operaciones normales o de la rutina de programación de un microprocesador).

A continuación se destina 1/2 K (512 bytes) a lo que se denomina área de parámetros (direcciones 0000:0400 a 0000:05FF). Los primeros 256 bytes los emplea la ROM-BIOS y los segundos el DOS. Se puede decir que esta es una zona de trabajo para el BIOS y el DOS.

Como ejemplos de información almacenada en esa zona puede citarse el estado del teclado (tipo Caps Lock ó Shift) ó la posición actual del cursor en la pantalla.

El IBMBIO.COM que suele encontrarse a partir de esta dirección y cuyo tamaño depende de la versión del DOS empleado. Inmediatamente del IBMBIO.COM se carga el IBMDOS.COM de tamaño igualmente dependiente de la versión del DOS. Finalmente se carga el COMMAND.COM, pero en este caso en dos partes y en diferentes direcciones. Una porción se carga en memoria a continuación del IBMDOS.COM; mientras que la otra se carga en las direcciones más altas de la memoria del usuario, justo al final de las K (256,512,etc) que el PC tenga instaladas.

En resumen, la memoria disponible en la práctica depende de una serie de factores:

- La memoria de usuario realmente instalada en nuestro PC (de 64 K a 640 K).
- La versión del DOS empleada (por el tamaño de sus componentes).
- Los posibles controladores de dispositivo de y/o programas residentes cargados durante el proceso de arranque.

La forma más rápida de saber la memoria disponible consiste en utilizar el commando CHKDSK del DOS. Dicho comando suministra por pantalla (fig. 4.4) dos bloques de información. Primero, y por espacio de varias líneas, nos informa el estado del disco duro ó diskette. A continuación, nos indica en dos líneas consecutivas, primero la cantidad total de memoria instalada y después la memoria libre actual. Ambas cantidades se suministran en bytes.

```
c>CHKDSK

 21309440 bytes en espacio total en disco
    38912 bytes en 2 archivo(s) oculto(s)
    30720 bytes en 12 directorio(s)
  9326592 bytes en 414 archivo(s) de usuario
 11913216 bytes disponibles en disco

   524288 bytes total en memoria
   478192 bytes libres en memoria
```

En cualquier caso, no es correcto pensar que una vez arrancado el sistema, la memoria disponible para el usuario permanecerá constante. Algunos de los progamas más populares que pueden ejecutarse en el PC tienen también la característica de permanecer residentes. Tras la ejecución de uno de estos programas, la memoria libre habrá disminuido una vez más, puede decirse, como regla casi general, que la memoria disponible, en caso de variar, disminuye.

Las siguientes 128 K, direcciones 0000:0000 a D000:FFFF, se reservan, en principio, para extensiones adicionales de la ROM que pueda instroducir IBM. Como ejemplo, al aparecer el modelo XT, la ROM adicional para el control del disco duro se ha colocado a partir de la dirección C800:0000.

Finalmente las últimas 128 K de las E000:0000 a la F000:FFFF, son las originalmente reservadas para la memoria ROM. El PC básico trae 40 K de memoria ROM, situadas a partir de la dirección F800:0000 de las 40 K, las 32 primeras (F800:0000 a FD00:0FFF) están ocupadas por el interprete de BASIC (el denonidado BASIC de disco); mientras que las últimas 8 (FE00:0000 a F000:FFFF) las utiliza la ROM-BIOS.

El XT trae esas mismas 40 K, más otras 2 K para la BIOS del disco duro, que se coloca, como ya se ha dicho, en las 128 K anteriores.

La figura 4.4 representa en forma esquemática el mapa de memoria del PC.

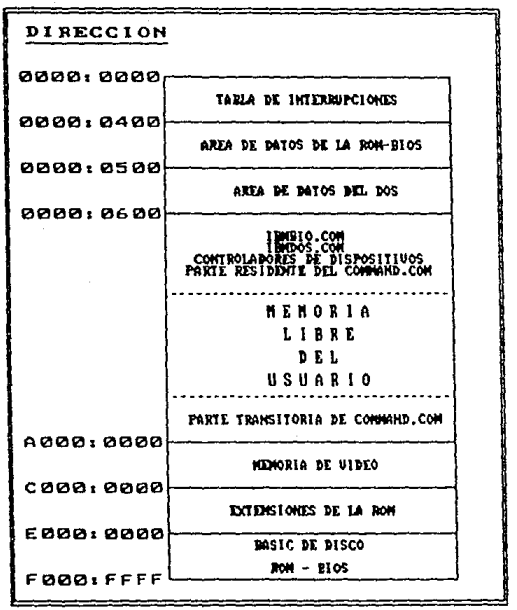


FIG. 4.4

NOTA: LA MEMORIA LIBRE DEL USUARIO SE EXTENDERA EN CADA CASO HASTA UN LIMITE DIFERENTE SEGUN LA MEMORIA INSTALADA FISICAMENTE. DE ESTA FORMA: SI EL PC POSEE HASTA LA POSICION 3000 FFFF, SI POSEE 512 K, HASTA LA POSICION 7000 FFFF, Y SI POSEE LAS 640 K HASTA LA POSICION 9000 FFFF (QUE ES LA SITUACION EXPRESADA) Y COMO CONSECUENCIA ADICIONAL, LA PARTE TRANSITORIA DEL PROCESADOR DE MANDATOS COMMAND.COM SE COLOCARA EN LA ZONA SUPERIOR DE DICHA MEMORIA LIBRE DE USUARIO, EN LA POSICION ADECUADA EN CADA CASO.

IV.2.5. EL ALMACENAMIENTO EXTERNO (DISKETTES/DISCOS).

Como es sabido, la memoria del PC es de dos tipos: RAM y ROM, la primera trae un contenido fijo de fábrica y no se puede alterar en ningún momento. En cuanto a la segunda, su contenido se pierde al apagar la computadora.

Los programas y datos de los usuarios deben poder ser almacenados en algún sitio seguro por tiempo indefinido, de forma que pueda utilizarse cuantas veces se requiera. Es obvio que la memoria no es el sitio adecuado.

Para solventar este problema es para lo que se han desarrollado diversos periféricos de la unidad central, que suelen denominarse *unidades de almacenamiento externo*. Los dos básicos en el caso del PC son los diskettes de cinco y cuarto pulgadas (5,25'') los nuevos de tres y cinco pulgadas (3.5'') y los discos duros.

IV.2.5.1.- EL DISCO FISICO.

En el caso de los diskettes, nos encontramos con tres divisiones posibles: pistas, caras y sectores.

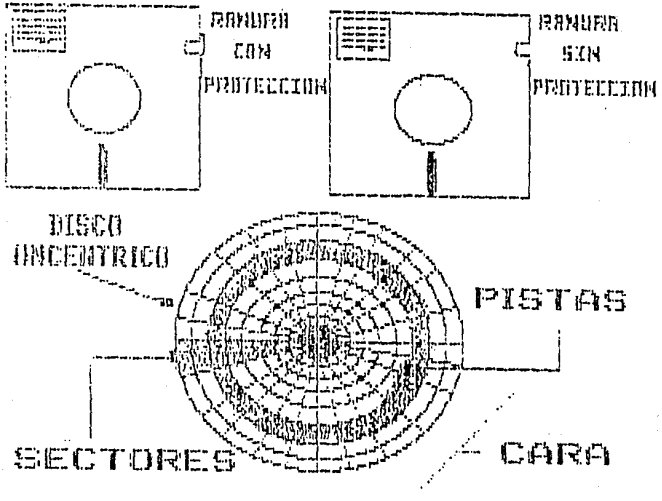
Cada diskette consta de un número fijo de pistas concéntricas. En concreto, los diskettes normales del modelo básico y del XT tienen 40 pistas (numeradas del 0 al 39) mientras que los diskettes de alta capacidad del AT presentan el doble, es decir, 80 pistas (numeradas del 0 al 79). Todo diskette tiene dos caras (0 y 1).

finalmente cada pista puede dividirse en un número variable de porciones o sectores. Los diskettes del PC y XT pueden tener 8 o 9 sectores por pista, mientras que los de alta capacidad del AT tienen 15, el número de sectores se puede seleccionar por software. En cualquier caso, los sectores presentan una diferencia de numeración, que a menudo conduce a errores. Sea cual sea el número de sectores por pista, siempre empiezan a numerarse desde el 1 y no desde el 0 como ocurre en las pistas o caras.

Los tipos de diskettes de 5,25'' disponibles en el mercado puede resumirse en el cuadro siguiente:

TIPO	UNA CARA DENS. SIMPLE	UNA CARA DENS. DOBLE	DOS CARAS DENS. SIMPLE	DOS CARAS DENS. DOBLE	DOS CAR ALTA C.
ABREVIADO	1S/1D	1S/2D	2S/1D	2S/2D	2S/HD
NUM. PISTA	40	40	40	40	80
NUM. CARAS	1	1	2	2	2

ESTRUCTURA DEL DISKETTE



El PC original puede estar equipado con unidades de una sola cara (es decir, que solo sabe leer una cara del diskette), si es muy antiguo; ó de dos caras que es lo usual. En ambos casos puede utilizarse cualquier tipo de diskettes, excepto los de alta capacidad; pero, si el drive es de una cara, sea como sea el diskette, únicamente podra accederse a su cara 0.

En cuanto al AT, normalmente trae incorporada una primera unidad de dos caras y alta capacidad, y una segunda unidad también de dos caras pero de capacidad normal.

La correspondencia natural entre formato físico y número de sectores a elegir parece sencilla: el el caso de diskette de densidad simple (1S 2SD), deberían elegirse 8 sectores por pista; y en el caso de diskette de doble densidad (1D 2D) deberían elegirse 9 sectores por pista. ¿Porque?, la respuesta es que se presentan con dos problemas. En primer lugar durante el proceso de formateo es bastante probable que el DOS detecte errores y deseche sectores enteros al tratar de utilizar un diskette de baja densidad como si fuera de doble densidad. Y en segundo lugar, haya o no haya ocurrido eso durante el formateo, tiene muchas probabilidades de que más tarde ó más temprano el diskette de errores de lectura/escritura, con el riesgo de perder archivos con información valiosa.

Por todo ello, la norma a seguir es bién clara: si el diskette es de densidad sencilla, habra que formatearlo a 8 sectores por pista; mientras que si es de doble densidad, sera de 9 sectores por pista. Siguiendo esta indicación, se puede complementarse con el cuadro siguiente:

TIPO	UNA CARA DENS. SIMPLE	UNA CARA DENS. DOBLE	DOS CARAS DENS. SIMPLE	DOS CARAS DENS. DOBLE	DOS CAR ALTA C.
ABREVIADO	1S/1D	1S/2D	2S/1D	2S/2D	2S/HD
NUM. PISTA	40	40	40	40	80
NUM. CARAS	1	1	2	2	2
SECT/PIST	8	9	8	9	15
TOTAL SEC	320	360	640	720	2400

Como es facil de deducir, el número total de sectores se obtiene sin más que multiplicar el número de pistas por el de caras y por el de sectores/pistas.

Sin embargo, aún no sabemos la capacidad real de cada diskette, para ello es necesario saber cual es el tamaño de un sector. Este parametro tambien viene fijado por el software, y en concreto, el DOS utiliza un tamaño universal de sectores de 512 bytes (1/2 K).

La capacidad de un diskette es la indicada en la tabla siguiente:

TIPO	UNA CARA DENS. SIMPLE	UNA CARA DENS. DOBLE	DOS CARAS DENS. SIMPLE	DOS CARAS DENS. DOBLE	DOS CAR ALTA C.
TOT. SECTR	320	360	640	720	2400
CAPACIDAD	160K	180K	320K	360K	1200

Para los diskette de 3.50'', los valores aumentan el doble de lo arriba mostrado.

IV.2.5.2. EL DISCO DURO FISICO.

Los discos duros tambien pueden divivirse en pistas, caras y sectores; con la única diferencia de que suelen poseer mas de una superficie y, por lo tanto, mas de dos caras. Pueden considerarse como un conjunto de diskette fijos de gran capacidad, colocados de forma permanente unos encima de otros.

Como las diversas superficies que lo forman estan dispuestas una encima de otra, las diferentes pistas caen también una encima de la otra. Debido a este hecho, surge el concepto de cilindro, que tradicionalmente es el conjunto de todas las pistas de igual numero pertenecientes a las distintas caras. Así por ejemplo, el cilindro 3 de un disco duro de dos superficies, estara formado por las pistas número 3 de la superficie 1 cara 0, de la superficie 1 cara 1, de la superficie 2 cara 0 y de la superficie 2 cara 1. Por supuesto, esta división en cilindros es puramente ficticia y no responde a ningún hecho físico. De cualquier forma, al hablar de discos duros también se utiliza a menudo la palabra cilindro con el mismo significado de pista individual.

En el caso del PC, existen tres formatos físicos fundamentales de discos duros: el disco de 10 M del XT y los de 20 M ó 30 M del AT. Los parámetros básicos de los modelos, son mostrados en la figura siguiente:

TIPO	10 M (XT)	20 M (AT)
NUM. CILINDROS	306	615
NUM. CARAS	4	4
SECTORES/PISTA	17	17
TOTAL SECTORES	20808	41820
CAPACIDAD (K)	10404	20910
CAPACIDAD (M)	10,16	20,41

IV.2.5.3. LOS DISCOS BAJO MS-DOS.

El concepto fundamental para localizar un determinado sector de un disco, el DOS no emplea los tres parámetros que conocemos; número de pista, cara y sector. Por el contrario, para el DOS un disco sólo tiene sectores, pero no pistas ni caras.

En el caso de un diskete de dos caras, el DOS comienza numerando los sectores de la pista 0 cara 0 hasta la pista 39 cara 1. Esto es valido para los formatos de diskette y disco duro.

A pesar de todo esto, el sector es una unidad física fundamental de los discos, el DOS utiliza una unidad lógica de capacidad superior, que se denomina cluster.

Un cluster es igual a uno ó más sectores físicos, dependiendo del medio y del formato de que se trate; y su importancia proviene de esta unidad básica para el DOS de todas las operaciones de entrada y salida. con los dispositivos de almacenamiento externo.

Al crear un archivo, independientemente de su tamaño lógico, el espacio se reserva para él en múltiplos de cluster. Si se borra el archivo, el cluster completo vuelve a incrementar el espacio libre.

De forma analógica, las operaciones de lectura o escritura a disco se realizan en unidades de cluster. Esto no quiere decir que no se pueda leer ó escribir en un solo sector. Si se desea leer un único sector, las operaciones físicas se realizará sobre un cluster entero (el que contiene el sector deseado), pero el sistema se encargará de devolver únicamente el sector concreto.

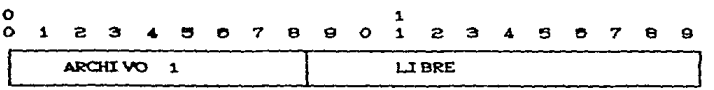
La equivalencia entre sectores y cluster, en los medios y formatos que no se ocupan, se resume a continuación:

TIPO	1 cluster igual a
DISKETTE 1S/1D	1 SECTOR
DISKETTE 1S/2D	1 SECTOR
DISKETTE 2S/1D	2 SECTORES
DISKETTE 2S/2D	2 SECTORES
DISKETTE 2S/HD	1 SECTOR
DISCO DURO 10M	HASTA 8 SECTORES
DISCO DURO 20M	HASTA 4 SECTORES

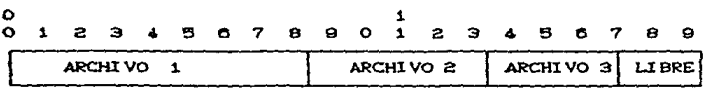
IV.2.5.4. ORGANIZACION DEL DISCO.

El espacio requerido para crear un archivo se asigna en forma de sectores contiguos. Esto trae como consecuencia el que, al borrar y crear nuevos ficheros, puedan irse dejando huecos inutilizados en el disco.

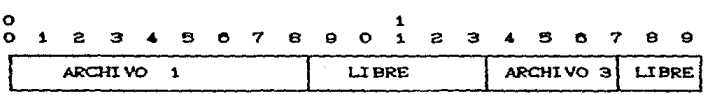
Ejemplo. al crear un dispositivo con una capacidad total de 20 sectores (ciertamente, no muy potente). inicialmente, se crea un archivo de 9 sectores. ocupando los sectores números 0 al 8. tal como se ilustra:



Al crear otro de 5, que ocupe los sectores 9 al 13 y un archivo más de 4, que ocupe los sectores 14 al 17, quedaría así:



Ahora, al borrar el archivo de 5 sectores, quedarán ocupados los sectores del 0 al 8 y del 14 al 17. Por el contrario, quedarán libres del 9 al 13 y del 18 al 19. Hay un total de dos bloques de 5 y 2 sectores respectivamente.



En esta situación no es posible crear un nuevo archivo de 6 sectores, ya que no hay ningún sitio de 6 sectores libres. Por lo tanto, aunque hay más espacio libre del necesario (hay 7 sectores), al estar fragmentado no es factible utilizarlo como se desearía.

Por lo anterior, el enfoque elegido para el manejo de almacenamiento externo en el caso del PC es notablemente diferente. En pocas palabras, dicho enfoque se resume en dos principios básicos:

- Al crear un archivo, se le asigna únicamente el espacio que necesite en ese momento, no el espacio total que se va a llegar a ocupar. Como consecuencia, para crear un archivo no es necesario calcular el espacio máximo que se va a necesitar.

- Al añadir datos aun archivo ya existente, se le va asignando nuevo espacio según sea necesario. Dicho espacio adicional no tiene porqué ser contiguo, sino que se va incrementando de donde haya.

Para realizar esta gestión, el DOS se apoya fundamentalmente en las dos áreas: la *File Allocations Table (tabla de asignación de estados) FAT*, y el directorio.

IV.2.5.5. EL DIRECTORIO.

El directorio se utiliza para llevar cuenta de los archivos que existen en el disco o diskette. Por cada uno de ellos, se almacena en el directorio una serie de información, que se denomina una *entrada*. Así, hay una entrada en el directorio por cada archivo existente. Cada una de las entradas tienen una longitud fija de 32 bytes. Por lo tanto, el número máximo de archivos que pueden contener un determinado tipo de disco duro o de diskette vendrá dado por el tamaño del directorio expresado en bytes y dividido entre 32. dicho tamaño se fija durante el formateo y depende del medio empleado, según puede verse a continuación:

TIPO	TAMAÑO DIRECTORIO	NUMERO ENTRADAS
DISKETTE 1S/1D	4 SECTOR	64
DISKETTE 1S/2D	4 SECTOR	64
DISKETTE 2S/1D	7 SECTORES	112
DISKETTE 2S/2D	7 SECTORES	112
DISKETTE 2S/HD	14 SECTOR	224
DISCO DURO 10M	HASTA 32 SECTORES	512
DISCO DURO 20M	HASTA 32 SECTORES	512

En general, la información almacenada en cada entrada del directorio es la siguiente:

bytes 1 al 8: nombre del archivo ajustado a la izquierda.

Si una entrada del directorio aún no ha sido utilizada, lo más habitual es que su contenido sea todo ceros binarios.

bytes 9 al 11: Extensión del nombre del archivo.

bytes 12: Atributos del archivo.

Este byte tiene una enorme importancia, ya que con su valor se indican las características del archivo. Las distintas posibilidades se indican en la figura que se muestra a continuación:

VALOR HEXADECIMAL	SIGNIFICADO
01	DE SOLO LECTURA
02	OCULTO
04	DEL SISTEMA
08	NOMBRE DE VOLUMEN
10	SUBDIRECTORIO
20	ARCHIVO NORMAL
40	NO USADO
80	NO USADO

En la terminología de IBM, se denomina *volumen* a todo diskette o disco duro. Durante el proceso de formateo se le puede asignar un nombre a cada volúmen. Dicho nombre se denomina *identificador de volumen* o VOLID. El VOLID, en caso de darlo, se almacena en una entrada especial del directorio raiz cuyo byte de atributo tiene el valor de 08. El VOLID puede tener hasta 11 caracteres, ya que para guardarlo se emplean los 11 primeros bytes de entrada (los correspondientes al nombre y la extensión del archivo). Como es lógico, hay un solo VOLID por disco (siempre en el directorio raiz).

Para saber si un disco tiene VOLID, basta hacer un dir. Al comienzo de la información de salida se verá una línea con un texto así:

VOLUMEN EN UNIDAD X ES xxxxxxxxxx

en caso de no existir VOLID:

VOLUMEN EN UNIDAD X SIN ETIQUETA

bytes 13 al 22: Reservado para uso futuro.

bytes 23 al 24: Hora de creación/actualización del archivo.

Esta es la hora en que se grabó por última vez el archivo, se almacena en un formato bastante especial, esquematizado como sigue:

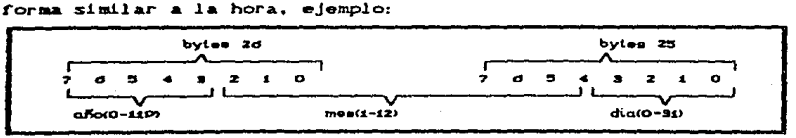
Por supuesto, todos los números se almacenan en binario. La hora ocupa los cinco bits de mayor peso del segundo byte (se ha representado los dos bytes en órden inverso, tal como se trata el PC). Puesto que en cinco bits se pueden guardar hasta 31, la hora, que oscila entre 0 y 23 cabe perfectamente.

Tomando como base lo anterior se puede calcular la hora de creación/actualización de un archivo COMMAND.COM como sigue:

```
DATO DE LA ENTRADA DEL DIRECTORIO:          00 60
INVIRTIENDO EL ORDEN:                       60 00
TRADUCIDO A BINARIO:                         0110 0000 0000 0000
POR LO TANTO:
      -HORAS (5 PRIMEROS BITS):             01100 BINARIO =
                                                  = 12 DECIMAL
      -MINUTOS (6 BITS SIGUIENTES):         000000 BINARIO =
                                                  = 0 DECIMAL
      -SEGUNDOS (5 ULTIMOS BITS):           00000 BINARIO =
                                                  = 0 DECIMAL

      HORA DE CREACION DEL ARCHIVO: 12:00p
```

bytes 25 al 26: Fecha de creación/actualización del archivo.

La fecha de la última grabación del archivo se almacena de forma similar a la hora, ejemplo:



En este caso, el año ocupa los siete bits de mayor peso del segundo byte. En esos siete bytes puede guardarse hasta el número 127, pero en realidad el año oscila tan sólo entre 0 y 119, cuando esta información se procesa, se le suma al valor encontrado 1980, de forma que el año real oscilará entre 1980 y 2099, en cuanto al mes, se almacena en cuatro bits, el de menor peso del segundo byte y los tres de mayor peso del primero. En cuatro bits cabe hasta el número 15, más de lo necesario. El día ocupa los cinco bits de menor peso del primer byte. En ellos caben hasta el número 31.

Del mismo ejemplo del archivo COMMAND.COM, se puede detectar su fecha de creación/Actualización, la figura siguiente muestra como se procesa:

```
DATO DE LA ENTRADA DEL DIRECTORIO:          32 09
INVIRTIENDO EL ORDEN:                       09 32
TRADUCIDO A BINARIO:                        0000 1001 0011 0010
POR LO TANTO:
      - AÑO   (7 PRIMEROS BITS):            0000100 BINARIO =
                                                  = 4  DECIMAL
      - MES   (4 BITS SIGUIENTES):          1001    BINARIO =
                                                  = 9 DECIMAL
      - DIA   (5 ULTIMOS BITS):             10010 BINARIO =
                                                  = 18 DECIMAL

   FECHA DE CREACION DEL ARCHIVO: 18 DE SEPTIEMBRE DE 1984
```

bytes 27 al 28: Primer cluster del archivo.

Esta información es de una enorme importancia, ya que no se indica en que cluster del disco empieza el archivo al que hace referencia la entrada. Siempre que se procesa un archivo, el funcionamiento del DOS es el mismo.

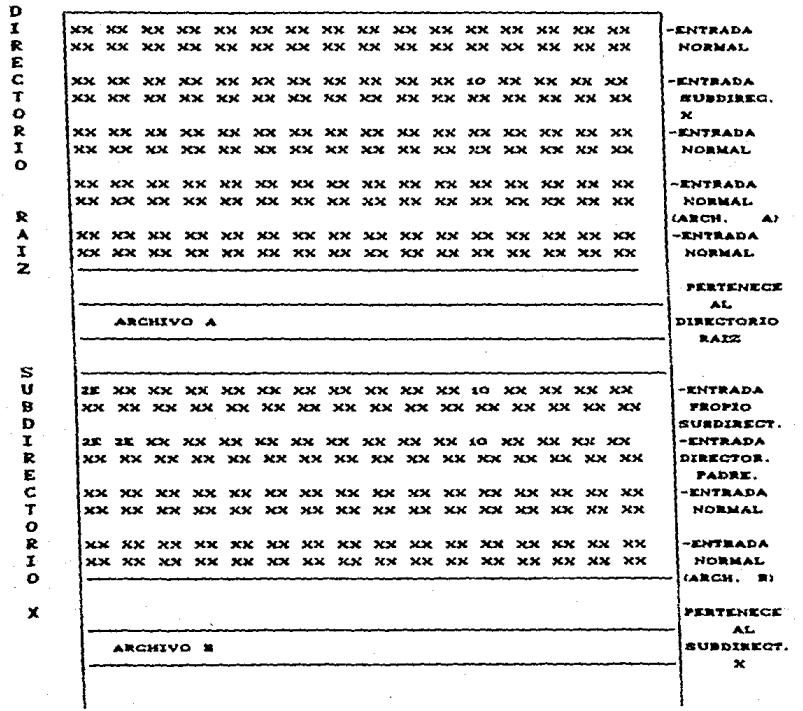
bytes 29 al 32: Tamaño del archivo.

Este tamaño se expresa en bytes, y es el valor que aparece al ejecutar un comando DIR.

IV.2.5.6. LOS SUBDIRECTORIOS.

En el DOS al crear un subdirectorio lo que se crea realmente es un archivo más que va a contener un nuevo directorio. A partir de ahí, si se crean archivos en el directorio *raiz*, la información correspondiente a dichos archivos se guarda en el directorio original del disco; pero si se crean en un subdirectorio, esa misma información se guarda en el archivo que se creó al dar de alta el subdirectorio.

Pueden crearse tantos subdirectorios como se desee (pero siempre con en un solo directorio raiz). Por otro lado el número de archivos que se pueden almacenar en un subdirectorio es limitado. Esto se debe al hecho de que el directorio de subdirectorio es un archivo y, por lo tanto, su tamaño sólo viene limitado por la capacidad del disco. La figura siguiente esquematiza lo explicado.



Al crear un subdirectorio, se crea un archivo normal y corriente, cuya entrada en el directorio raiz contiene el nombre del subdirectorio y con el byte de atributo con valor de 10H.

En todos los subdirectorios aparecen al hacer un DIR como archivos, de nombres '.' y '..' (uno y dos puntos), respectivamente.

Para comprenderse mejor lo anterior, se despliegua el siguiente esquema:

CLUSTER DE COMIENZO	NOMBRE ARCHIVO	ENTRADAS DIRECTORIOS		
		NOMBRE	ATRIBUTO (HEX)	PRIMER CLUSTER
-	DIRECTORIO RAIZ	ARCH-A	20	8
		ARCH-B	20	34
		ARCH-C	20	12
		SUBDIR.	10	29
		ARCH-D	20	25
8	ARCH-A			
12	ARCH-C			
17	ARCH-E			
25	ARCH-D			
29	SUBDIRECT SUBDI	.	10	29
		..	10	0
		ARCH-E	20	17
		ARCH-F	20	41
34	ARCH-B			
41	ARCH-F			

Este esquema puede prolongarse hacia abajo, ya que pueden crearse subdirectorios, hasta el nivel que se desee. En cualquier caso, el proceso es siempre el mismo. En el archivo que corresponde al subdirectorio padre se dá de alta una entrada con atributo 10H, y se crea el archivo que actuará se subdirectorio hijo. Y este último se dan de alta dos primeras entradas, con referencias a sí mismo y a su directorio padre.

IV.2.5.7. LA TABLA DE ASIGNACION DE ARCHIVOS.

Como ya se ha dicho, después del registro de arranque y antes del directorio, se encuentra la otra área fundamental para el manejo de discos, que se denomina FAT (*File Allocation Table*).

Así como el directorio se emplea para almacenar ciertas características de los archivos existentes en el disco duro o diskette, la FAT se utiliza para llevar la cuenta de qué *cluster* están ocupados por cada uno de estos archivos.

También en cada caso se emplea el concepto de *entrada*. Sin embargo, hay ahora una diferencia muy importante. En la FAT existe una entrada, no por cada archivo, sino por cada cluster del *área de datos* del disco (entendiendose por área de datos, todo el disco menos lo ocupado por el registro de arranque, el directorio y la propia FAT).

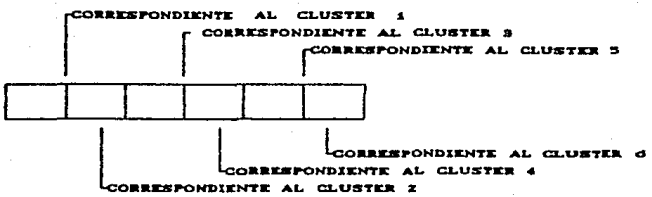
Cada una de estas entradas tiene una longitud fija de 12 bits (es decir un byte y medio), a excepción del disco duro del AT, en el que son de 16 bits (2 bytes).

A igual que el directorio, el tamaño del FAT se fija durante el proceso de formateo y depende del medio empleado. La tabla siguiente muestra los valores habituales. Sin embargo, debe resaltarse que, a diferencia del directorio, en el caso de la FAT, en cada disco se guardan dos copias idénticas, una de tras de otra.

T I P O	TAMAÑO FAT CADA COPIA (SECTORES)	TAMAÑO ENTRADA (BITS)	NUMERO ENTRADAS POSIBLES
DISKETTE 1S/1D	1	12	341
DISKETTE 1S/2D	2	12	682
DISKETTE 2S/1D	1	12	341
DISKETTE 2S/2D	2	12	682
DISKETTE 2S/HD	7	12	2389
DISCO DURO 10 M	HASTA 9	12	2730
DISCO DURO 20 M	HASTA 41	16	10496

Como ya se ha dicho, cada entrada de la FAT corresponde a un cluster del área de datos del disco. Puede decirse entonces que la FAT es un mapa de dicha área de datos del disco.

CLUSTER 1	CLUSTER 2	CLUSTER 3
CLUSTER 4	CLUSTER 5	CLUSTER 6



Como puede observarse, las entradas de la FAT se corresponden con los cluster del área de datos del disco, no solo por su contenido, sino por su posición. Es decir, la primera entrada de la FAT corresponde al primer cluster de dicha área, la segunda entrada al segundo cluster y así sucesivamente.

Como se sabe, en la PC los archivos se almacenan en sectores (ó con más propiedad, en clusters) que no tienen que estar contiguos. El problema que puede resolver la FAT entonces es saber que clusters contienen los datos de un archivo dado y en que órden han de leerse.

Como puede entenderse, la FAT es la zona más crítica de cualquier disco. Cuando se añaden o se actualizan datos. Su contenido va variando con mucha frecuencia. Por esta razón, el DOS suelen cargarla en la memoria al acceder por primera vez el disco duro o diskette, y trata de mantenerla allí el mayor tiempo posible, sin necesidad de irla regrabando constantemente en el disco.

Esto quiere decir que si se van añadiendo nuevos *clusters* a un archivo, los sectores con los datos se escriben en el disco, pero su encadenamiento sólo se altera en la copia de la FAT que está en memoria. Al cerrar el archivo es cuando normalmente se regraba la FAT en el disco y se actualiza el tamaño del archivo en la entrada del directorio. Como consecuencia, si hay un corte de corriente durante el proceso de grabación, puede ocurrir que los datos estén grabados correctamente en el área de correspondiente, pero que su encadenamiento de la FAT y su tamaño del directorio no estén convenientemente actualizados. Este es uno de los mayores problemas que presenta el DOS.

IV.2.5.8. EL MANEJO DE LA FAT.

En esta parte, se explicará como se pueden calcular los números de sectores del disco donde están los datos de un archivo, a partir de la información contenida en las entradas de la FAT.

Como ya se ha dicho, ha excepción del disco duro del AT, en todos los demas medios cada entrada de la FAT tiene 12 bits de longitud (1,5 bytes). Como de costumbre, este byte y medio se almacena en órden inverso, de forma que si una entrada tiene el valor de 85AH, en la FAT aparecerá como AB5H. Como además no se trata de un número entero de bytes, lo más común es considerar las entradas de la FAT de dos en dos. Así los valores 7E 04 BC equivalen a las entradas 47EH y BC0H.

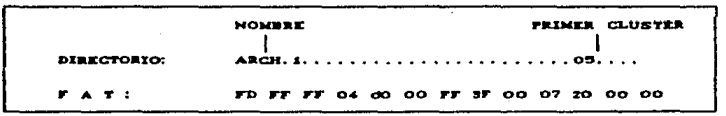
Hay dos problemas a resolver. El primero consiste en localizar la entrada de la FAT correspondiente al *cluster* número n. El proceso a seguir es el siguiente:

- Multiplicar el número de cluster por 1.5.
- Quedarse con la parte entera del resultado, sumarle una unidad y obtener el byte que ocupa ese número de órden dentro de la FAT y el siguiente.
- Intercambiar entre sí los dos bytes obtenidos para obtener su verdadero valor, ya que el DOS los almacena en órden inverso.
- Expresando los dos bytes antes mencionado en hexadecimal (es decir, como cuatro dígitos), si el número de cluster era impar, la entrada de la FAT la forman los tres primeros dígitos. Por el contrario, si el número de cluster era par, la entrada la constituyen los tres últimos dígitos hexadecimales.

En cuanto al segundo problema, consiste en convertir el número de cluster n en número de sector (para el DOS no hay número de pista cara y sector, sino solo de éste último). Para esto:

- Restar dos unidades al número de cluster (ya que la numeración de los cluster de datos empieza en dos y no en cero).
- Multiplicar el resultado por el número de sectores por cluster.
- Sumarle el número de sectores ocupados por el registro de arranque, el directorio y la FAT.

Como el manejo de la FAT, supongamos un diskette de 2S/2D con la entrada del directorio y op ción de la FAT indicadas en la figura siguiente:



El primer cluster del archivo es el número 5. Para convertirlo en número de sectores, se le resta dos unidades:

(5 - 2) = 3

se multiplica por dos, ya que en los diskette 2S/2D un cluster es igual a dos sectores:

(3 x 2) = 6

y finalmente se le suma 12, que es el número de sectores ocupados por el registro de arranque, el directorio y la FAT

(6 + 12) = 18

Así pues, el primer bloque de datos está en los sectores número 18 y 19 del diskette (recuerdese que, que en este caso, un cluster contiene dos sectores).

A continuación hay que buscar la entrada número 5 de la FAT. Se comienza multiplicando el número por 1.5, ya que cada entrada ocupa un byte y medio

(5 x 1.5) = 7.5

se retiene la parte entera (7) y se le suma una unidad (8). Se toman los bytes octavo y noveno de la FAT, 3F y 00, que dados la vuelta, pasan a ser 00 y 3F. Como el número de cluster era impar, el valor de la entrada son los tres primeros dígitos, es decir, 003H, por lo tanto, el siguiente cluster del archivo es el número 3.

A continuación se resume lo anterior:

```
CLUSTER NUMERO 5 ⟶ 5 - 2  = 3
                   3 X 2  = 6
                   6 + 12 = 18 ⟶SECTORES 18 Y 19
ENTRADA NUMERO 5 ⟶ 5 X 1.5 = 7.5
                   7 + 1  = 8
                 BYTES 8 Y 9 : 3F 00 ⟶ 00 3F
                CLUSTER IMPAR ⟶ NUEVO CLUSTER =003
```

Repitiendo los cálculos anteriores sucesivamente, se irán obteniendo los resultados siguientes:

```
CLUSTER NUMERO 3 ---> 3 - 2  = 1
                      1 X 2  = 2
                      2 + 12 = 14 --->SECTORES 14 Y 15
ENTRADA NUMERO 3 ---> 3 X 1.5 = 4.5
                      4 + 1 = 5
                    BYTES 5 Y 6 : 00 00 ---> 00 00
                   CLUSTER IMPAR --->NUEVO CLUSTER = 006
CLUSTER NUMERO 6 ---> 6 - 2  = 4
                      4 X 2  = 8
                      8 + 12 = 20---> SECTORES 20 Y 21
ENTRADA NUMERO 6 ---> 6 X 1.5 = 9
                      9 + 1 = 10
                    BYTES 10 y 11: 07 20 ---> 20 07
                   CLUSTER IMPAR --->NUEVO CLUSTER = 007
CLUSTER NUMERO 7 ---> 7 - 2  = 5
                      5 X 2  = 10
                      10+ 12 = 22---> SECTORES 22 Y 23
ENTRADA NUMERO 7 ---> 7 X 1.5 = 10.5
                      10 + 1 = 11
                    BYTES 11 y 12: 20 00 ---> 00 20
                   CLUSTER IMPAR --->NUEVO CLUSTER = 002
CLUSTER NUMERO 2 ---> 2 - 2  = 0
                      0 X 2  = 0
                      0 + 12 = 12---> SECTORES 12 Y 13
ENTRADA NUMERO 2 ---> 2 X 1.5 = 3
                      3 + 1 = 4
                    BYTES 4 Y 5 : 04 00 ---> 00 04
                   CLUSTER IMPAR --->NUEVO CLUSTER = 004
CLUSTER NUMERO 4 ---> 4 - 2  = 2
                      2 X 2  = 4
                      4 + 12 = 16---> SECTORES 16 Y 17
ENTRADA NUMERO 4 ---> 4 X 1.5 = 6
                      6 + 1 = 7
                    BYTES 7 Y 8 : FF 3F ---> 3F FF
                                        (FIN DE ARCHIVO)
```

Así pues, el archivo 1 (ARCH.1) ocupa los clusters 5,3,6,7,2 y 4; por este mismo órden

Otro detalle de importancia lo constituye la marca que indica el fin de archivo. En principio, el valor de una entrada cualquiera de la FAT puede oscilar entre 0 y FFFH (ó entre 0 y 4095 en decimal). Sin embargo hay ciertos valores, como el propio FFFH, que tiene un significado especial. La figura siguiente recoge los valores según sus significados:

VALOR DE LA ENTRADA F A T HEXADECIMAL	SIGNIFICADO
000	CLUSTER LIBRE
ENTRE 002 Y FEF	CLUSTER EN USO
ENTRE FF0 Y FF7	CLUSTER INUTILIZABLE
ENTRE FF8 Y FFF	MARCA FIN DE ARCHIVO

Entonces, el disco duro o diskette puede tener un máximo de 4096 cluster. En el mejor de los casos, el del XT, un cluster es igual a 8 sectores, es decir, a 4096 bytes. Por lo tanto, la capacidad máxima de un disco manejado por el DOS con un FAT de 12 bits, será de 4096 x 4096 = 16777216 bytes (16 MD. Como consecuencia, el disco duro del AT, que tiene 20 *megas*, no puede manejarse de esta manera.

Debido a este problema, la FAT del disco duro del AT es diferente a la de los demás formatos de disco duro o diskette. En el caso del AT, cada entrada de la FAT ocupa 16 bits (es decir, 2 bytes), en lugar de los 12 habituales.

El significado de los diversos valores posibles de cada entrada es análoga al caso de la FAT de 12 bits, tal como se puede ver en la tabla siguiente:

VALOR DE LA ENTRADA F A T HEXADECIMAL	SIGNIFICADO
0000	CLUSTER LIBRE
ENTRE 0002 Y FFEF	CLUSTER EN USO
ENTRE FFf0 Y FFF7	CLUSTER INUTILIZABLE
ENTRE FFF8 Y FFFF	MARCA FIN DE ARCHIVO

En cuanto al manejo de la tabla, se realiza de forma similar a la tabla de 12 bits. Para encontrar la entrada corresponde al cluster número n:

- Multiplicar el número de clusters por 2
- Sumarle una cantidad al resultado y obtener el byte que ocupa ese número de órden dentro de la FAT y el siguiente.
- intercambiar entre sí los dos bytes obtenidos para obtener su verdadero valor.

Y para convertir un número de cluster (como siempre, la numeración de los cluster de datos empiezan en dos, y no en cero).

- Multiplicar el resultado por el número de sectores por cluster.
- Sumarle el número de sectores ocupados por el registro de arranque, el directorio y la FAT.

IV.2.5.9. EL AREA DE DATOS.

Tras el registro de arranque, las dos copias de la FAT y del directorio, por fin comienza el área de datos del disco duro o diskette, que ya se extiende hasta el final del mismo. En la tabla siguiente pueden verse los tamaños de dicha zona de datos.

TIPO	NUMERO TOTAL DE SECTORES DISPONIBLES	SECTORES OCUPADOS POR R.A./FAT/ DIRECTORIO	SECTORES AREA DE DATOS
DISKETTE 1S/1D	320	7	313
DISKETTE 1S/2D	360	9	351
DISKETTE 2S/1D	640	10	630
DISKETTE 2S/2D	720	12	708
DISKETTE 2S/HD	2400	29	2371
DISCO DURO 10M	20723	49	20640
DISCO DURO 20M	41755	115	41620

Las tres áreas de que se ha hablado hasta ahora (registro de arranque, FAT y directorio) están presentes en todo disco duro ó diskette formateado bajo DOS. El resto del espacio, lo que se ha denominado área de datos, queda disponible para usarlo según sea necesario, y por lo tanto no tiene un contenido estándar. Sin embargo, en todos aquellos discos desde los que se vaya a cargar el sistema operativo (los formateados con la opción /S), hay una parte del área de datos que sí es estándar.

Al formatear un disco con la opción /S, se graba en él los archivos IBMBIO.COM IBMDOS.COM y COMMAND.COM. Es imprescindible que los dos primeros, ocupen las dos primeras entradas del directorio raíz del disco duro o diskette y que su contenido se halle en los primeros clusters del área de datos y en órden consecutivo. Solo que si se cumplen estas condiciones se podrá cargar el sistema operativo desde ese disco.

Por lo tanto si se formatea un disco sin la opción /S y se graba en algún archivo, ya no servirá de nada copiar posteriormente los dos archivos del sistema operativo. Los primeros clusters y la primera entrada del directorio estarán ocupados por el archivo grabado inicialmente con lo que no se podrá cargar el DOS.

Para evitar este problema, puede emplearse, al formatear, la opción /B. Con esta opción no se copian los archivos IBMBIO.COM IBMDOS.COM, pero se reserva el espacio y las entradas del directorio necesarios para poderlos incorporar luego. LLegado en momento, no basta con realizar un copy de esos dos archivos, ya que así no se asegura que fueran a parar al sitio adecuado. Para transferirlo existe un comando del DOS denominado SYS. este comando copia ambos archivos al disco y los coloca en la forma adecuada para que pueda cargarse el DOS (no hay que olvidar, el archivo COMMAND.COM).

IV.2.6. LA PANTALLA

Para algunos, la pantalla (display) es la parte más importante de la computadora, y en cierta forma lo es. La pantalla está formada por dos componentes *hardware* diferentes. En primer lugar está el monitor (pantalla), que se une mediante un cable a la unidad central. Si habrimos la computadora, se observa que ese cable procedente de la pantalla se conecta a una tarjeta de expansión. A esta placa se le suele denominar *adaptador de pantalla*.

Con el tiempo, han ido apareciendo en el mercado modelos diferentes de tarjetas adaptadoras. Las diferencias entre ellos suelen radicar si son capaces de representar imágenes en colores, si permiten el empleo de gráficos y en la resolución que poseen.

La tabla siguiente despliega los diferentes miembros de la familia:

NOMBRE	COLORES	GRAFICOS
MONITOR MONOCROMO	SI	NO
MONITOR COLOR BASICO	SI	SI
MONITOR COLOR ESTANDAR	SI	SI
MONITOR COLOR AMPLIADO	SI	SI
MONITOR COLOR PROFESIONAL	SI	SI

En cuanto a los adaptadores pueden encontrarse los indicados a continuación:

NOMBRE	ABREVIATURA	COLORES	GRAFICOS
ADAPTADOR MONOCROMO	-	NO	NO
ADAPTADOR COLOR ESTANDAR	CGA	SI	SI
ADAPTADOR COLOR AMPLIADO	EGA	SI	SI
ADAPTADOR COLOR PROFES.	PGC	SI	SI

Finalmente la correspondencia entre adaptadores y monitores se esquematiza en la siguiente tabla:

ADAPTADOR	MONITOR	OTROS MONITORES POSIBLES
MONOCROMO	MONOCROMO	—
CGA	COLOR BASICO	—
CGA	COLOR ESTANDAR	—
EGA	COLOR AMPLIADO	MONOCROMO BASICO STAND.
PGC	COLOR PROFES.	—

Estas tablas presentan dos detalles interesantes. En primer lugar, tanto el monitor de color básico como el de color estándar utilizan el adaptador CGA. El monitor de color básico no es sino una versión más barata y, logicamente de peor calidad del monitor de color estándar. En segundo lugar se observa que el adaptador de color ampliado EGA es una especie de adaptador universal, ademas de gran calidad ya que puede emplearse con todos los monitores existentes, a excepción del profesional.

La pareja CGA/monitor de color básico es funcionalmente identica a la pareja CGA/monitor de color estándar, por lo que cualquier cosa que se diga para esta última valdrá igualmente para la primera.

En cuanto a la pareja EGA/monitor de color ampliado, supone una mejora considerable, aunque su difusión es menor debido a su aparición muy posterior.

Finalmente el PGC y el monitor de color profesional, vienen de un nivel superior de prestaciones que da IBM, pero debido a su elevado precio están muy poco extendidos y su estudio excede en el estudio de éste.

IV.2.6.1. LOS MODOS DE VIDEO.

Como introducción al manejo de pantalla, debe decirse que el PC trabaja básicamente en dos modalidades de visualización: el *modo texto* y *el modo gráfico*. En modo texto, lo único que puede verse en la pantalla son los distintos caracteres del juego ASCII del PC. Por el contrario el modo gráfico pueden aparecer tantos dichos caracteres como cualquier disposición de gráfica.

Es importante no confundir ambos conceptos de color y de gráficos. El adaptador monocromo sólo puede funcionar en modo texto y ademas no permite el despliegue de color alguno. Los restantes adaptadores pueden trabajar en modo texto o en modo gráfico y ademas en cualquiera de ellos pueden presentar diferentes colores.

El modo de funcionamiento de la salida por pantalla se selecciona escogiendo lo que se denomina un *modo de vídeo*. Restringuiendonos a los adaptadores elegidos para la PC, se presentan los 8 modos de video fundamental, expresadas a continuación:

MODO	TIPO	RESOLUCION	ADAPTADOR	COLOR
0	TEXTO	40 X 25 C	CGA	NO
1	TEXTO	40 X 25 C	CGA	SI
2	TEXTO	80 X 25 C	CGA	NO
3	TEXTO	80 X 25 C	CGA	SI
4	GRAFICO	320 X 200P	CGA	SI
5	GRAFICO	320 X 200P	CGA	NO
6	GRAFICO	640 X 200P	CGA	SI (B/N)
7	TEXTO	80 X 25 C	CGA	NO

Con la tarjeta EGA además de los 8 modos de video, exiten otros cuatro de mejores prestaciones que llevan los números 13 al 16, ver tabla:

COLOR	NUMERO
NEGRO	0
AZUL	1
VERDE	2
CYAN (VERDE + AZUL)	3
ROJO	4
MAGENTA (ROJO + AZUL)	5
AMARILLO (ROJO + VERDE)	6
BLANCO (ROJO + VERDE + AZUL)	7
GRIS	8
AZUL BRILLANTE	9
VERDE BRILLANTE	10
CYAN BRILLANTE	11
ROJO BRILLANTE	12
MAGENTA BRILLANTE	13
AMARILLO BRILLANTE	14
BLANCO BRILLANTE	15

Pasemos a detalle al estudio de los ocho modos básicos.

1.- Modos 0 y 1

Estos dos modos pueden denominarse modo texto de 40 x 25 con *adaptador de gráficos/color*. Solamente pueden visualizarse los caracteres del código ASCII, en 25 líneas de 40 caracteres cada una.

La diferencia entre el modo 0 y 1, está en que el 0 se sustituyen los diferentes colores por tonalidad de grises. Sin embargo esto no quiere decir que este modo sea aplicable al monitor monocromo. Los dos sólo pueden emplearse con el adaptador de gráficos/color, sólo que en el 1 aparecen verdaderamente colores y en el 0 tonalidad grises.

2.- Modos 2 y 3

La denominación para esta pareja de modos en video es la de modo texto de 80 x 25 con *adaptador de gráficos en color*. Unicamente pueden representarse los caracteres del juego ASCII en 25 líneas de 80 caracteres cada una.

A igual que en los modos 0 y 1 pueden elegirse el color de cada caracter y el fondo de la misma por separado. De igual forma, para el fondo de cada caracter pueden elegirse entre los ocho primeros colores, mientras que para el caracter en sí pueden elegirse entre los 16 colores.

3.- Modo 4 y 5

En este caso la denominación más corriente es el *modo gráfico de media resolución*. En este caso puede considerarse la pantalla dividida en puntos, 320 horizontales por 200 verticales. Cada punto (ó PIXEL) es seleccionable individualmente.

4.- Modo 6

Suele denominarse *modo gráfico de alta resolución*. Ahora la pantalla se divide en 640 puntos horizontales por los mismos 200 verticales. cada punto sigue siendo seleccionable individualmente. Este modo suele presentarse, de forma un tanto cierta, como de dos colores. De hecho, cada punto puede presentar uno de dos colores, pero estos colores no son selecionables, sino que por definición son el blanco y el negro. Así pues, en la práctica es un modo en blanco y negro.

En términos de los dos últimos anteriores, diriamos que en este caso hay una sola paleta con únicamente dos colores, y ademas sin color de fondo seleccionable.

5.- Modo 7

Este es el *modo texto con adaptador monocromo*. es el unico modo utilizable con dicho adaptador y no puede ser seleccionado con los adaptadores de gráficos/color, a excepción del EGA.

IV.2.6.2. LA MEMORIA DE VIDEO.

Internamente el contenido de la pantalla se almacena en una zona de la memoria PC, de una forma tanto especial, la circuiteria del adaptador y del monitor se encarga de leer continuamente el contenido de dicha zona de memoria y de reflejarlo en la pantalla. Los programas cuando desean producir una salida por la pantalla, a través de diversos caminos al final siempre terminan por alterar el contenido de la zona mencionada.

En pocas palabras, la forma de conseguir que aparezca 'A' en determinado sitio de la pantalla, consiste en colocar cierto valor en una posición de la memoria dedicada al video. Una vez hecho esto, la siguiente vez que el adaptador lea esa posición encontrará el valor almacenado y provocará que aparezca en el monitor y caracter deseado.

Con este esquema de funcionamiento, suele decirse que la pantalla esta *bit mapped*, es decir, que cada punto de la pantalla tiene un reflejo en uno o más elementos de la memoria del ordenador.

La memoria de visualización no se encuentra físicamente (chips) en la placa base, sino en la propia tarjeta adaptadora. De esta forma, el adaptador monocromo trae en la placa unicamente las 4 K que necesita, mientras que el CGA lleva 16 K y el EGA 64 K. Para el microprocesador el lugar donde se encuentren las distintas porciones de la memoria utilizable le es completamente indiferente.

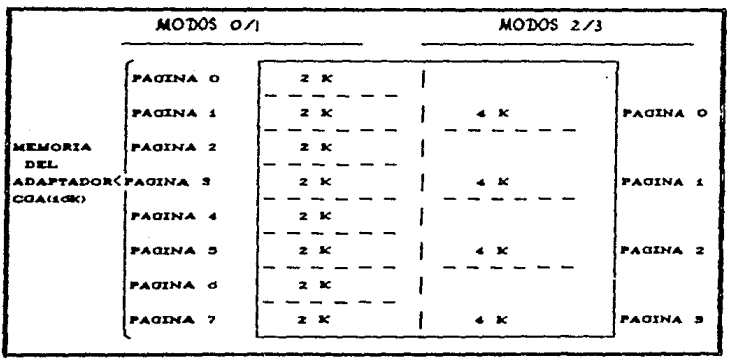
IV.2.6.3. MEMORIA DE VISUALIZACION CON ADAPTADOR MONOCROMO.

El adaptador monocromo puede representar 25 líneas de 80 caracteres cada una. Hay 25 x 80 = 2000 caracteres. cada caracter puede ser uno cualquiera del código ASCII, por lo que su identificación requiere un byte. Ademas se emplea otro byte por cada caracter, para almacenar sus características especiales (brillante, subrayado, etc.). Por lo tanto para representar 2000 caracteres se necesitan 4000 bytes, (casi 4 K).

IV.2.6.4. MEMORIA DE VISUALIZACION CON EL CGA MODO DE TEXTO.

Para el estudio del adaptador de gráficos/color, CGA, debe distinguirse entre el modo texto y el modo gráfico.

En cualquiera de los posibles modos texto (0 al 3), el manejo de la memoria de visualización es muy similar al de adaptador monocromo. Como en éste, se utilizan 2 bytes consecutivos por cada caracter de la pantalla. Por lo tanto en los modos 0 y 1 se estarán empleando 40 x 25 x 2 = 2000 bytes (algo menos de 2 K); mientras que en los modos 2 ó 3 se utilizan 80 x 24 x 2 = 4000 bytes (menos de 4 K -lo mismo que el adaptador monocromo-). Sin embargo, el CGA tiene 16 K de memoria, por lo tanto, en el primer caso nos sobran 14 K y el segundo 12 K. (Ver tabla).



En la práctica, se puede ordenar al adaptador que búsque el contenido a reflejar en pantalla en cualquiera de esas porciones, que suelen denominarse *páginas*. Numerando dichas páginas del 0 al 7 en el caso de los modos 0 ó 1, y del 0 al 3 en el caso del los modos 2 ó 3, aparecen entonces el concepto de *página activa*, que no es sino el número de la página con cuyo valor contenido se está construyendo actualmente a la imagen reflejada en la pantalla.

IV.2.6.5. MEMORIA DE VISUALIZACION CON EL CGA MODOS GRAFICOS.

En el modo gráfico el empleo de la memoria de visualización cambia notablemente. Ya no existen caracteres, sino meramente puntos, que pueden aparecer de un color u otro. En este caso cada *pixel* corresponde en memoria a 1 ó 2 bits

En los modos 4 ó 5, la resolución es de 320 x 200 *pixels* y cada uno de ellos puede tomar un color entre cuatro diferentes. se emplean dos bits para cada *pixel*, de forma que su valor conjunto nos indíque el color elegido tal como se indica a continuación:

	PRIMER BIT	SEGUNDO BIT
COLOR NUMERO 0	0	0
COLOR NUMERO 1	0	1
COLOR NUMERO 2	1	0
COLOR NUMERO 3	1	1

Como ya se ha dicho en la práctica, el color real corresponde a cada valor depende de la paleta que se haya seleccionado.

Puesto que se emplean dos bits para cada *pixel*, se necesitan 320 x 200 x 2 = 128000 bits = 16 bytes. Así pues, se están utilizando las casi 16 K del CGA.

En cuanto al modo 6, presenta una resolución de 640 x 200 *pixels* con tan sólo dos colores posibles. Basta entonces un bit para cada pixel (si está a cero es el primer color y si está a 1 es el otro). por lo tanto se necesitan 640 x 200 = 16000 (de nuevo las casi 16 K del adaptador).

RESUMEN

En estos tres modos gráficos, la correspondencia entre las direcciones de la memoria de visualización y los *pixels* de la pantalla no es del todo directa. Cabría pensar que el primero (modo 6) ó los dos primeros (modo 4 y 5) bits corresponden al primer pixel, el siguiente o los dos siguientes al segundo pixel y así sucesivamente. Esto no es del todo cierto. En realidad si se numeran las filas de la 0 a la 199 (son 200 en ambas resoluciones), se almacena en primer lugar los bits correspondientes a los pixels de las filas pares y después los correspondientes a las filas impares.

IV.2.7. EL TECLADO.

Tras los discos y la pantalla, es el turno del periférico fundamental de la computadora: el teclado. Si la pantalla se considera un dispositivo de salida (el sistema suministra información a través de ella), el teclado aparece como un periférico de entrada.

El teclado original del PC, tanto del modelo básico como el del XT, contiene 83 teclas, por su parte el teclado inicialmente suministrado con el PC-AT cuenta con unas tecla más y exhibe un diseño diferente (102 teclas). Cada tecla tiene un caracter del código ASCII del PC.

Sin embargo se pueden encontrarse con facilidades diversas excepciones a esta primera regla. Las teclas de función no parecen tener asignados caracteres concretos. Ademas el caracter representado varía según si algunas de las teclas SHIFT ha sido pulsada simultáneamente. De forma analógica, el caracter que aparece al pulsar la tecla 'L' no es el mismo en el teclado del formato USA (':') que en un teclado adaptado al español (que es la 'ñ'.

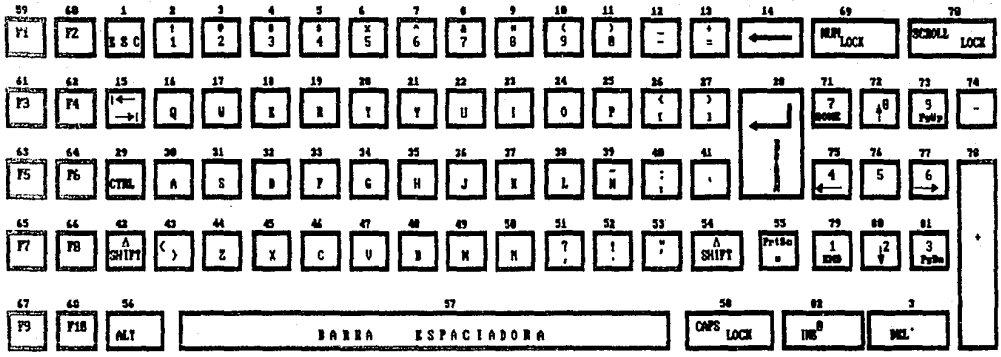
La relación tecla-caracter no es tan rígida como puede suponerse en un principio. En realidad es preferible desdoblarla en dos relaciones distintas, insertando en el medio el concepto de *código de exploración ó scan code.*

IV.2.7.1. CODIGOS DE EXPLORACION.

A cada tecla física se le asigna un número, del 1 al 38, que se denomina *código de exploración* (scan-code). De esta forma, el scan-code de la tecla ESC es el 1, mientras que el de la tecla NUM-LOCK es el 69. en la figura siguiente pueden verse los códigos de exploración de todas la teclas del teclado estandar.

Cada vez que el usuario pulsa una tecla, el chip controlador del teclado almacena en su buffer interno el scan-code correspondiente y envía una interrupción al 8088. Cuando al microprocesador le da servicio, se le envía el scan-code almacenado. Si no lo hace, por estar dedicadoa otros trabajos, y el usuario continúa tecleando hasta llenar dicho buffer interno, sonará un pitido indicando el problema.

En la práctica, los códigos de exploración no se generan solamente al pulsar una tecla. Cuando se suelta la tecla, también envía un nuevo scan-code, que en este caso es el resultado de sumarle 128 al enviado anteriormente. De esta forma, al pulsar la tecla 'R' se envía un 19 y al soltarla se envía un 147 (128 + 19).


F1
F2
ESC
!
1
"
2
·
3
$
4
%
5
&
6
/
7
(
8
)
9
=
0
?
¿
BACK SPACE
NUM LOCK
SCROLL LOCK
F3
F4
TAB
Q
W
E
R
T
Y
U
I
O
P
^
`
[
*
+
7
HOME
8
9
PgUp
-
F5
F6
CTRL
A
S
D
F
G
H
J
K
L
Ñ
{
}
4
5
6
F7
F8
SHIFT
<
>
Z
X
C
V
B
N
M
;
,
:
.
_
-
SHIFT
PrtSc
*
1
END
2
3
PgDn
+
F9
F10
ALT
BARRA ESPACIADORA
CAPS LOCK
INS
0
DEL
.

Ademas si una tecla se mantiene pulsada durante más de medio segundo, comienza a enviarse de forma repetida el código de exploración correspondiente.

Estos conceptos pueden comprenderse facilmente observando el ejemplo de la figura siguiente:

ACCION	CODIGOS ENVIADOS (DECIMAL)
SE PULSA LA TECLA J Y SE SUELTA	36 164
SE PULSA LA TECLA . Y SE SUELTA	52 180
SE PULSA LA TECLA < Y SE MANTIENE PULSADA DURANTE UN SEGUNDO Y SE SUELTA	45 45 45 45 45 45 45 100 80 157
SE PULSA LA TECLA F1 Y SE SUELTA	

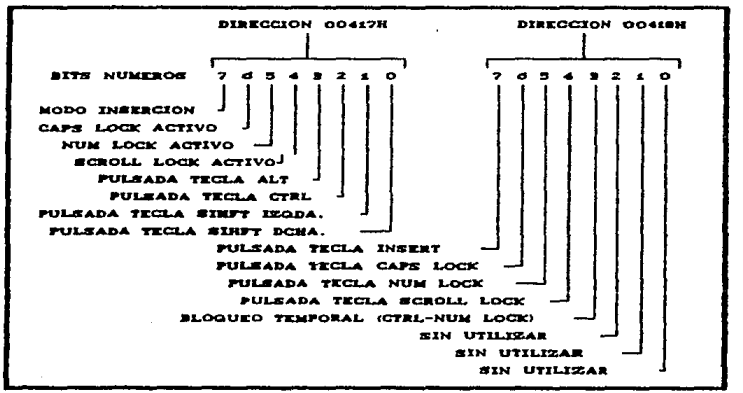
El paso que queda ahora , la traducción del código de exploración recibido al caracter ASCII correspondiente, realizada por la ROM-BIOS, es la parte realmente complicada.

IV.2.7.2. TECLAS DE ESTADO.

Suelen considerarse hasta 7 teclas de estado: CAPS-LOCK, NUM-LOCK, SCROLL-LOCK, INS, CTRL, ALT, y SHIFT. Las cuatro primeras son teclas tipo *interruptor*, es decir, que en un momento dado están dadas o quitadas. Al encender la computadora, las cuatro están desactivadas., si se pulsa cualquiera de ellas, automáticamente pasa a estar activada. Una nueva pulsación conduce a desactivada, y así sucesivamente. Lamentablemente, en el teclado del modelo básico no existe ninguna indicación del estado actual de cada una de ellas. Por el contrario, en los dos teclados del modelo AT (tanto el original como el ampliado), existen sendos indicadores luminosos para todas ellas, a excepción de la tecla INS.

En cuanto a las otras tres teclas de estado, por si solas no producen ningún efecto, sino que se emplean en conjunción con las teclas normales. de esta forma se considerará distinta la pulsación simple de la tecla A, de la pulsación simultanea de la tecla CTRL y A (suele expresarse CTRL + A). Ademas ambas serán diferentes a la pulsación de ALT y A (CTRL+A). y todas ellas serán distintas a la pulsación de SHIFT y A (SHIFT+A). Mediante este mecanismo se le pueden asignar varios caracteres a una misma tecla, que se generarán en cada caso según las teclas de estado empleadas simultáneamente. Se consigue con ello que el teclado de 83 teclas sea capaz de producir un número muy superior de caracteres.

La modalidad del teclado, en lo que se refiere a las teclas de estado, se almacena en todo momento dentro del área de trabajo de la ROM-BIOS, en la zona baja de memoria. Concretamente, los dos bytes situados a partir de la dirección absoluta 00417H contiene la información indicada en la figura mostrada a continuación:



La alteración de los valores existentes en un momento dado equivaldrá en algunos casos a cambiar el estado del teclado. Si se está escribiendo en minúsculas y en un determinado momento se pone a uno el bit número 6 de la posición 417H a partir de este momento se pasará a estar escribiendo en mayúsculas. Exactamente igual que si se hubiera pulsado la tecla CAPS LOCK.

Cabe mencionar, como curiosidad, que observando el valor de los dos bytes mencionados puede saberse si la tecla SHIFT pulsada en un momento dado es la situada a la derecha o situada a la izquierda del teclado. También puede comprobarse si se ha pulsado la combinación CTRL+NUM LOCK, que produce una pausa en cualquier salida a pantalla (a igual que CTRL+S).

IV.2.7.3. LOS CARACTERES.

Los caracteres ASCII se pueden simplificar dividiendo las 83 teclas en varios grupos:

1.-Teclas alfanuméricas.

El grupo más numerosos lo componen las teclas que forman el llamado *teclado de máquina de escribir*, es decir, las teclas de la A a la Z y las situadas en la fila superior que producen los números del 0 al 9. En este caso, la rutina de la ROM-BIOS devuelve los valores: el scan-code de la tecla pulsada en AH (primera mitad del registro AX), y el caracter ASCII al que corresponde en AL (segunda mitad del registro AX). Para hallar este último, se tiene en cuenta la situación de las teclas de estado, fundamentalmente la tecla CAPS LOCK y las dos teclas SHIFT.

- Si la tecla pulsada corresponde a una letra (A-Z) y la tecla CAPS LOCK está activa, se toma el carácter correspondiente en mayúsculas. Si no está activa, se toma en minúsculas.

- Si se ha pulsado simultaneamente la tecla alfabética y una de las teclas SHIFT, el caracter devuelto se invierte con respecto al caso anterior. Si la tecla CAPS LOCK está activa, se toma el caracter correspondiente en minúsculas, mientras que si no está activa, se toma en mayúsculas. Se dice entonces que la tecla SHIFT revierte temporalmente el estado de la tecla CAPS LOCK.

- Si la tecla pulsada pertenece a la fila superior, el estado de las teclas CAPS LOCK no tiene efecto alguno. Si no se ha pulsado simultaneamente la tecla SHIFT el caracter resultante será el indicado en la parte inferior de la tecla (0-9 y símbolos - y =)

2.- Teclas de movimiento del cursor (zona numérica)

Este grupo lo forman las once teclas de color blanco en la zona derecha del teclado, que se asemejan al teclado de una calculadora, en este caso, los valores devueltos por la rutina de servicio de la ROM-BIOS también depende de la situación de las teclas de estado, fundamentalmente la tecla NUM LOCK y las dos teclas SHIFT.

● Si la tecla NUM LOCK está activa, se devolverá el scan-code correspondiente en AH y el caracter dibujado en la parte superior de la tecla (es decir 0-9 y el caracter '.'). Si no está activa, se devolverá el mismo scan-code en AH y un valor 00 en AL. este es el primer caso de las teclas que no producen caracteres ASCII.

3.- Teclas de función F1 a F10

Este es el tercer grupo que está formado por las 10 teclas de color obscuro en la zona izquierda del teclado, al pulsar cualquiera de ellas, la rutina de la ROM-BIOS devuelve un valor 00 en AL y el código de exploración en AH. Por lo tanto, Tampoco producen caracteres ASCII.

4.- Teclas restantes.

Las teclas restantes (color obscuro), tienen finalidades muy diversas, entre ellas están CAPS LOCK, NUM LOCK, SCROLL LOCK, CTRL, ALT, Y SHIFT (esta última duplicada).

Existen además otras siete teclas, que si producen caracteres, empezando por la derecha, las teclas + y - no tienen ninguna característica especial y su funcionamiento es análogo a las teclas equivalentes de la zona de máquina de escribir.

La tecla '^' es algo curiosa. Pulsada sin shift proporciona dicho caracter, mientras que por SHIFT devuelve un valor 00 en AL y su scan-code en AH. Pero en este último caso, ademas, su efecto consiste en imprimir el contenido actual de la pantalla (print screen) (shift+prtsc).

La tecla especial [esc] representa un caracter no representable en pantalla, lo que hace (en la práctica) es que al pulsarla tiende a efectuar una salida. Esto se debe a que devuelve en AL y el código de exploración en AH.

De todo lo expresado, existen ciertas combinaciónes de teclas que también pueden emplearse para producir códigos extendidos, es decir, scan-code superiores a 83, y estos se muestran en la siguiente tabla:

COMBINACION	CODIGO EN AH (hex)
Shift + F1	84
Shift + F2	85
Shift + F3	86
Shift + F4	87
Shift + F5	88
Shift + F6	89
Shift + F7	90
Shift + F8	91
Shift + F9	92
Shift + F10	93
Ctrl + F1	94
Ctrl + F2	95
Ctrl + F3	96
Ctrl + F4	97
Ctrl + F5	98
Ctrl + F6	99
Ctrl + F7	100
Ctrl + F8	101
Ctrl + F9	102
Ctrl + F10	103
Alt + F1	104
Alt + F2	105
Alt + F3	106
Alt + F4	107
Alt + F5	108
Alt + F6	109
Alt + F7	110
Alt + F8	111
Alt + F9	112
Alt + F10	113
Ctrl + PrtSc	114
Ctrl + <-	115
Ctrl + ->	116
Ctrl + End	117
Ctrl + PgDn	118
Ctrl + Home	119
Alt + 1	120
Alt + 2	121
Alt + 3	122
Alt + 4	123
Alt + 5	124
Alt + 6	125
Alt + 7	126
Alt + 8	127
Alt + 9	128
Alt + 0	129
Alt + -	130
Alt + =	131
Ctrl + PgUp	132

Al producir códigos extendidos, cualquiera de estas combinaciones de teclado puede ser identificada por programas.

otra característica importante del teclado del PC reside en el hecho de que nos brinda la posibilidad de generar cualquiera de los 256 caracteres que componen el código ASCII. Para ello basta con pulsar la tecla ALT, y sin dejarla de pulsarla, la tecla el código número en decimal correspondiente al caracter deseado en el teclado de calculadora de la zona derecha. Este mecanismo funciona en todo momento, independientemente de si la tecla NUM-LOCK está ó no activa.

Si se teclea un número mayor de 255 (los códigos de los caracteres ASCII van desde 0-255), el sistema se encarga de dividirlo por 256 y emplearse el resto que resulte. De esta forma, teclear el código 300 es equivalente a teclear el 44 (300 = 256 + 44). En ambos casos se obtendrá el caracter ','.

A pesar de ello, en éste metodo no producirá caracter alguno cuando el código tecleado corresponda a algún caracter especial de control.

IV.3. MANTENIMIENTO DEL EQUIPO PC.

En la sección anterior se explicó en forma técnica como funciona internamente la computadora, así como también sus posibles fallas. El paso siguiente es dar mantenimiento al equipo.

Entiendase como *mantenimiento*, al 'mantener' en buén estado todos los componentes posibles del PC para su buén funcionamiento, esto a traves de los conocimientos adquiridos.

Pero eso no es todo, habrá quien logre detectar un error en la computadora, y habrá quien no. De esto se deriva hacer una serie de cuestionarios dividiendolos en partes fundamentales de la PC.

Como se mostró en la sección anterior, sabemos que la computadora se divide en los siguientes componentes :

- Al inicializar.
- Almacenamiento.
- Pantalla (monitor)
- Teclado
- Impresora

Podemos observar que el tema de *IMPRESORA* no se tocó en la sección anterior, ya que la impresora es independiente del sistema de la PC, sin embargo, es un tema paralelo al de los demás componentes, por tal motivo no hay que dejar en el aire este tipo de problemas (aunque no se tenga gran conocimiento sobre el tema), no obstante a traves de la recopilación de información se sabrá las posibles fallas que se pueda tener acerca de la impresora.

Los cuestionarios, serán divididos en los temas arriba mencionados. Para realizar esto, se debe tener conocimiento de que es lo que abarca cada tema para poder realizar las encuestas.

las encuestas nos ayudarán a saber más sobre posibles fallas y posibles respuestas, dichos cuestionarios serán evaluados por gente del medio informático de diferentes empresas en donde existan ó manejen la PC.

Lo siguiente será analizar los problemas que se puedan surgir ó derivar de los temas principales, la siguiente tabla muestra los problemas por tema:

1. PROBLEMAS AL INICIALIZAR	ERROR DEL SISTEMA DURANTE EL INICIO. NO ENCIENDE EL DRIVE AL INICIALIZAR. NO SUENA LOS BEEPS, SUENAN BEEPS CONTINUOS. NO ENCIENDE EL LED DEL DRIVE AL INICIALIZAR. DESPLIEGA MENSAJES COMO: NON-SYSTEM DISK, DISK ERROR, DISK BOOT FAILURE, DRIVE NO READY, BAD MISSING COMMAND INTERPRETER. ERROR EN EL ARCHIVO AUTOEXEC.BAT ERRORES DE PARIDAD. PROBLEMAS DE REINICIO. PROBLEMAS DE SISTEMA OPERATIVO.
2. PROBLEMAS DE CORRIDA.	PROBLEMAS DE DISCO DURO. PROBLEMAS DE DISCO FLEXIBLE. SOBRECARGAMIENTO. NO ACCESA A LA INFORMACION. NO LEE O NO ESCRIBE APROPIADAMENTE. LEE PERO NO ESCRIBE. SE PIERDE LA INFORMACION CONTINUAMENTE. TIENE SONIDOS EXTRAÑOS. EL ARCHIVO CHKDSK GENERA MENSAJE DE ERROR. PROBLEMAS CON PROGRAMAS ESPECIFICOS. ERROR EN EL ARCHIVO CONFIG.SYS AUTOEXEC.BAT COMMAND.COM PROBLEMAS DEL SISTEMA OPERATIVO.

tabla 4.1 Posibles fallas

3. PROBLEMAS DE PANTALLA	NO DESPLIEGA INFORMACION. NO DESPLIEGA ENTRADA/SALIDA DE DATOS. EL MONITOR SE SOBRECARGA DESINCRONIZACION VERTICAL DESINCRONIZACION HORIZONTAL DESPLIEGA CARACTERES EXTRAÑOS. LOS COLORES NO SON LOS CORRECTOS. PROBLEMAS CON EL ADAPTADOR DE GRAFICOS.
4. PROBLEMAS DE TECLADO.	EL TECLADO NO FUNCIONA (NO HACE NADA). DESPLIEGA CARACTERES QUE NO SE TECLEAN. NO FUNCIONA UNA O MAS TECLAS. APARECEN DOS O MAS CARACTERES AL PULSAR SOLO UNA TECLA. OBJETOS AJENOS AL TECLADO. DERRAME DE LIQUIDOS EN EL TECLADO.
5. PROBLEMAS DE IMPRESORA	IMPRESORA MUERTA. CONFIGURACION DE LA IMPRESORA. FALLAS DE ON-LINE. IMPRIME BASURA. OCASIONALMENTE IMPRIME BASURA. SE AUTOINTERRUMPE AL IMPRIMIR. LA CABEZA DE IMPRESION NO FUNCIONA CORRECTAMENTE. SE IMPRIME CARACTERES DIFERENTES. NO IMPRIME CIERTOS CARACTERES AL IMPRIMIR. EL PAPEL NO CORRE EN LA IMPRESORA.

tabla 4.1 Posibles fallas (continuación).

Como se puede observar, estas son los posibles problemas que pueda tener la PC (de hecho puede haber más desglosando todos los temas y subtemas), sin embargo ahí no termina todo, nos podemos ayudar de los cuestionarios para encontrar un problema que no se haya mencionado en la tabla.

Los cuestionarios, a igual que la tabla se dividen en 5 temas a tratar, basandose en la problemas de la tabla, se pueden ya formular las preguntas que se harán para cada tema.

Las respuestas de los cuestionarios nos ayudarán para el siguiente capítulo (Diagnóstico e Hipótesis), así como para la selección de preguntas y respuestas que se contemplará en el menú principal del sistema de diagnóstico, así, el usuario podrá consultar el problema con respecto al tema correspondiente.

Los cuestionarios serán repartidos en cinco empresas en donde se tenga computadoras *PC* y *compatibles*, así como personal informático para la evaluación de las preguntas que se formulen. Dichas empresas son:

a) **Universidad Nacional Autónoma de México.**
Centro de Cómputo -ENEP Aragón-.

b) **Petroleos Mexicanos.**
Unidad de Informática y Difusión.

c) **Instituto Mexicano del Petroleo.**
Gerencia de Tecnología Informática.

d) **Centro de computación y sistemas S. C.**
Arcos de Belem 10 planta baja -Ed. A-

e) **ICM de mexico.**
Berlin 9-bis col. Juarez piso 3.

Cada empresa le corresponden cinco cuestionarios, es decir se repartirán un total de 25 cuestionarios.

A continuación se mostrarán los cinco cuestionarios a evaluarse, cabe recordar que las preguntas serán por tema, ademas de las cuestiones que se han analizado con anterioridad.

EMPRESA: _____
TIPO DE
EQUIPO PC: _____

CUESTIONARIO NO. 1
PROBLEMAS DE INICIO O ARRANQUE

1.- SU COMPUTADORA PERSONAL TIENE FALLAS DE INICIO O DE ARRANQUE ?

☐	☐	☐	☐	☐	☐
0%	20 %	40%	60%	80%	100%

2.- ERROR DEL SISTEMA (BEEPS) DURANTE EL INICIO. Marque con una X.

( ) funciona bién ( ) suenan beeps continuos ( ) suenan beeps largos ( ) beeps cortos
( ) display en blanco -sin beeps- ( ) no suena el beep

3.- AL INICIALIZAR, ENCIENDE EL LED DEL DRIVE ? Marque con una X.

( ) si
( ) no
( ) a veces no funciona
( ) funciona bién
( ) trabaja bién pero no inicializa
( ) hace ruidos extraños.

4.- DESPLIEGA LOS SIGUIENTE MENSAJES DE ERROR ? Marque con una X.

( ) Funciona correctamente.
( ) no despliega nada ( ) NON-SYSTEM DISK
( ) DISK BOOT FAILURE ( ) DRIVE NOT READY
( ) BAD MISSING COMMAND INTERPRETER
( ) otro mencione: _____

5.- TIENE PROBLEMAS DE SISTEMA OPERATIVO ? Marque con una X.

( ) si ( ) no ( ) problemas de versión
( ) algún archivo específico del DOS
( ) otro mencione: _____

6.- CONOCE ALGUNA CAUSA POR LA CUAL NO SE INICIALICE CORRECTAMENTE EL SISTEMA. EXPLIQUE: _____
_____
_____

EMPRESA: _____

TIPO DE EQUIPO PC: _____

CUESTIONARIO NO. 2

PROBLEMAS DE CORRIDA

1.- CON QUE FRECUENCIA TIENE PROBLEMAS DE CORRIDA ?

☐	☐	☐	☐	☐	☐
0%	20 %	40%	60%	80%	100%

2.- TIENE PROBLEMAS DE CORRIDA EN:

( ) disco duro
( ) disco flexible
( ) ruidos extraños
( ) bloqueo de memoria
( ) No tiene problema
( ) otro, mencione: _____

3.- TIENE PROBLEMAS DE EJECUCION EN DISCO DURO ?_____
CUALES: _____
_____

4.- TIENE PROBLEMAS DE EJECUCION EN DISCO FLEXIBLE ?_____
CUALES: _____
_____

5.- DESPLIEGA ALGUNOS MENSAJES DE ERROR ?_____
CUALES: _____
_____

6.- TIENE PROBLEMAS DE EJECUCION EN ALGUNOS PROGRAMAS ESPECIFICOS?
MENCIONE: _____
_____

7.- CONOCE ALGUNA OTRA CAUSA POR LA CUAL UN CUNOS) PROGRAMAS NO CORRAN CORRECTAMENTE ? EXPLIQUE: _____
_____
_____
_____

EMPRESA: _____

TIPO DE EQUIPO PC: _____

CUESTIONARIO NO. 3
PROBLEMAS DE PANTALLA

1.- CON QUE FRECUENCIA TIENE PROBLEMAS DE DESPLIEGUE ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

2.- EL MONITOR SE SOBRECARGA ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo, conoce la causa?: _____

_____

3.- EL MONITOR SE DESINCRONIZA ?

( ) Horizontal
( ) Vertical
( ) Funciona bien
( ) otro mencione

4.- DESPLIEGA BASURA (CARACTERES EXTRAÑOS) ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo conoce, la causa?: _____

_____

5.- LOS COLORES SON MALOS ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo conoce, la causa?: _____

_____

6.- CONOCE ALGUNA OTRA CAUSA POR LA CUAL SU MONITOR NO FUNCIONA BIEN ? EXPLIQUE: _____

_____

_____

EMPRESA: _____

TIPO DE EQUIPO PC: _____

CUESTIONARIO NO. 4
PROBLEMAS DE TECLADO

1.- CON QUE FRECUENCIA TIENE PROBLEMAS DE TECLADO ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

2.- DESPLIEGA CARACTERES QUE UD. NO TECLEA ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo, conoce la causa?: _____

_____

3.- FUNCIONA CORRECTAMENTE ALGUNAS TECLAS ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo, conoce la causa?: _____

_____

4.- DESPLIEGA 2 O MAS CARACTERES CUANDO UD. SOLO TECLEA UNA SOLA TECLA ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo, conoce la causa?: _____

_____

5.- CONOCE ALGUNA OTRA CAUSA POR LA CUAL NO FUNCIONE CORRECTAMENTE EL TECLADO ? EXPLIQUE ? _____

_____

_____

EMPRESA: _____

TIPO DE EQUIPO PC: _____

CUESTIONARIO NO. 5
PROBLEMAS DE IMPRESORA

1.- CON QUE FRECUENCIA TIENE PROBLEMAS DE IMPRESORA ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

2.- OCASIONALMENTE IMPRIME ERRORES ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo, conoce la causa?: _____

_____

3.- FUNCIONA BIEN LA CABEZA DE IMPRESION ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo, conoce la causa?: _____

_____

4.- IMPRIME CARACTERES DIFERENTES ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo, conoce la causa?: _____

_____

5.- EL PAPEL SE DESPLAZA CORRECTAMENTE EN LA IMPRESORA ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo, conoce la causa?: _____

_____

6.- LA IMPRESORA SE SOBRECARGA ?

☐ 0% ☐ 20 % ☐ 40% ☐ 60% ☐ 80% ☐ 100%

en caso afirmativo, conoce la causa?: _____

_____

7.- CONOCE ALGUNA OTRA CAUSA POR LA CUAL NO FUNCIONE CORRECTAMENTE LA IMPRESORA ? EXPLIQUE: _____

_____

Una vez realizada la encuesta, se presenta a continuación las estadísticas (en porcentaje) de las respuestas dadas por las empresas.

CUESTIONARIO NO. 1
PROBLEMAS DE INICIO O ARRANQUE

Pregunta No.	Porcentaje falla	Tipos de Problemas
1	6 %	•no tiene el archivo COMMAND.COM ó no está transferido el SYS.
2	2 %	•no identificado.
3	1 %	•no funciona el LED
4	5 %	•DOS.
5	1 %	•DOS.
6	15 %	•mal instalado el DOS ó empieza a fallar la unidad del disco.

Se puede observar que la pregunta:

3.- AL INICIALIZAR, ENCIENDE EL LED DEL DRIVE ? Marque c/una X.
( ) si
( ) no
( ) a veces no funciona
( ) funciona bién
( ) trabaja bién pero no inicializa
( ) hace ruidos extraños.

tuvo un mínimo porcentaje (1 %), es decir que solo existen pocas fallas al inicializar. En cuanto a la pregunta 6, la mayoría coincide en que el DOS está mal instalado ó que el disco duro ó unidad de diskette comienza a fallar.

CUESTIONARIO NO. 2
PROBLEMAS DE CORRIDA

Pregunta No.	Porcentaje falla	Tipos de problemas
1	40 %	•no identificado.
2	60 %	•disco duro.
3	35 %	•lectura/escritura.
4	60 %	•lectura/escritura. •mal uso de los diskettes.
5	20 %	•mensajes como: NOT READY IN DRIVE X Abort, Retry, Ignore. •errores de memoria.
6	15 %	•mal compilados o mal copiados.
7	40 %	•mal compilados •diskettes con errores de fábrica. •caen en sectores dañados. •programas que funcionan unicamente con CAPS-LOCK activado. •fallas de E/S en los drive o disco duro. •interferencia de softwares residentes

Los problemas de corrida presentan mayor demanda, en lo que se refiere a problemas de diferente tipo. Se observa que la pregunta (6) tiene el mínimo problema.

Las preguntas (2) y (4) tiene el nivel más alto de problemas de ejecución en los diskettes y discos duros. La mayoria coincide en que el problema puede ser el disco duro y en el caso de los diskette, que tiene errores de fábrica ó simplemente se dañan con el mal uso ó por el tiempo.

CUESTIONARIO NO. 3
PROBLEMAS DE PANTALLA

Pregunta No.	Porcentaje falla	Tipos de problemas
1	10 %	•adaptador de gráficos
2	28 %	•mal uso de los controles del monitor. •falta de ventilación
3	30 %	•no tiene el adaptador que le corresponde. •mal ajustados los controles de Hor/Ver
4	2 %	•no identificado.
5	15 %	•mal ajustamiento de los controles de video. •pueden existir programas residentes que afecten los colores.
6	5 %	•adaptador de video defectuoso. •monitor defectuoso.

Las pantallas presentan problemas ya comunes como un adaptador de video que no le corresponde a la computadora ó simplemente que tienen defectos de fabrica (y que no son garantizados).

Por lo regular ningun monitor presenta caracteres extraños como podemos ver en la respuesta (4). Sin embargo la pregunta (3) presenta el mayor problema en los monitores y esto se debe a que simplemente tienen otro tipo de adaptador para video. o que los controles del monitor son los primeros en fallar.

Muchas veces también es por falta de ventilación los monitores son sencibles al las sobrecargas y al polvo.

CUESTIONARIO NO. 4
PROBLEMAS DE TECLADO.

Pregunta No.	Porcentaje falla	Tipos de problemas
1	45 %	•falsos contactos. •mal uso.
2	80 %	•el teclado está configurado al formato USA ú otro
3	40 %	•maltrato de los resortes. •mal colocación de las teclas •puede tener objetos extraños entre las teclas. •las teclas a veces tienen polvo.
4	15 %	•teclado no compatible
5	20 %	•derrame de liquidos •objetos extraños.

Los Teclados son la parte del Hardware que más trabajan y que son los menos tienen problemas. Sin embargo he aquí algo interesante. Todo teclado tiene un formato diferente aún hablando de la misma marca de computadora. Lo que sucede (y pocos saben) es es que toda computadora tiene el formato de USA. es decir que el teclado es adaptado al formato ingles, la letra 'ñ' es la tecla ';' y ':'. Los teclados que tienen la tecla 'ñ' pero no imprime el caracter que le corresponde es que su teclado no está configurado al español. Existen programas que pueden alterar el teclado a diferentes formatos. (italiano, aleman, español, etc.).

De hecho existen computadoras hechas en México (tal es el caso de Unisys, Printaform y Elektra) que en su sistema de arranque llevan incluido el formato español, también difiere mucho la ayuda de las versiones del DOS.

Otro problema que puede persistir en el teclado son los derrames de liquido (frecuentemente café o refresco). También los golpes fuerte que se le dá al pulsar cada tecla.

CUESTIONARIO NO, 5
PROBLEMAS DE IMPRESORA

Pregunta No.	Porcentaje falla	Tipos de problemas
1	35 %	•daños de la cabeza de impresión.
2	15 %	•configuración erronea de la impresora.
3	70 %	•uso frecuente y forzado de las cabezas de impresora
4	50 %	•configuración •erronea tanto del software como de la impresora.
5	5 %	•mal uso del carrete de la impresora •no se usa correctamente el encendido de ON-LINE
6	15 %	•uso continuo de impresiónes.
7	60 %	•mal configuración •sofware no adaptados •cabezas de impresora dañados. •partes ajenas a la impresora como papel que queda atorado. •no se le dá mantenimiento continuamente.

Las Impresoras son equipos sumamente complicado y de los que más se dañan (a igual que los graficadores). Como se menciona arriba, se debe al mal uso de los usuarios al desplazar el carrete como si fuera máquina de escribir. La acumulación de las orillas del papel continuo en la impresora afecta con el tiempo. La configuración muchas veces no son compatibles con la PC, es por eso que muchas veces recomiendan ver primero el manual.

IV.3.1. CONCLUSION.

Al detenerse a observar con detenimiento, se llega a la conclusión de que se tiene más problemas en teclado y que la parte de la PC que menos problema tiene es el de inicio ó arranque. Es decir, para el diseño del sistema de diagnóstico abarcará más a profundidad los temas que más fallas presenta en la vida cotidiana.

Los problemas presentaron los siguientes índices de falla:

Problemas:

Al inicializar ——————————→ 10 %

De pantalla ——————————→ 29 %

De impresora ——————————→ 71.4 %

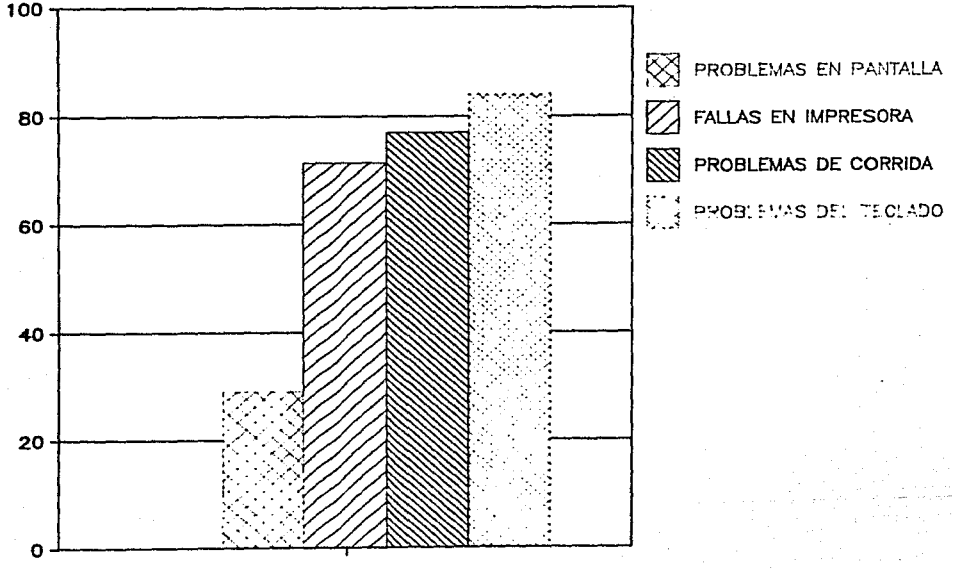
De corrida ——————————→ 77 %

De teclado ——————————→ 84 %

Una gráfica nos muestra como son los problemas menos presentados y los problemas más presentados en los equipos PC.

Ahora el tema siguiente, es hacer un diagnóstico de las posibles fallas y las hipótesis de las posibles respuestas, basandose en los cuestionarios mostrados en el presenta capítulo.

FALLAS TIPICAS EN LOS COMPONENTES DE PC'S



IV.4. DIAGNOSTICO O HIPOTESIS.

Cuando en un consultorio se le pregunta a un médico sobre un tema específico (lease malestar), éste a través de preguntas llega a un conclusión ó conclusiones. Así como el médico maneja un cuestionario, así también lo maneja Turbo Prolog a modo de preguntas para obtener una o varias respuestas.

Un sistema que trabaja en Turbo Prolog, maneja este tipo de cuestiones, es decir:

- Lleva un cuestionario.
- Evalua las respuestas.
- Llega a una conclusión.
- Si el problema es relacionado con almacenamiento de archivos, lo toma en cuenta, aún analizando otro problema.
- Las conclusiones pueden ser variables para todos los casos, es decir, si analizamos los problemas de corrida, también, analiza los problemas en disco. (Esto se debe al manejo de heurística).

Para esto, es necesario alimentar el sistema de diagnóstico a través de preguntas, para llegar a una conclusión (hipótesis).

Este capítulo presenta los diagnósticos y las hipótesis, de los diferentes temas analizados en el capítulo anterior.

IV.4.1 PROBLEMAS DE INICIO CON DIAGNOSTICO E HIPOTESIS:

DIAGNOSTICO	HIPOTESIS
●Error durante el inicio.	→Verificar el DOS.
●No suena el Beep.	→Problema interno checar la fuente de poder.
●Beeps cortos/no enciende el led del drive.	→Puede ser uno o más defectos de la memoria. →Verificar que los chips estén bién conectados. →Circuitos en mal estado.
●Suenan Beeps continuos.	→Problema interno, checar fuente de poder y los cables.
●Drive muerto, no enciende la luz del LED	→Drive defectuoso. →LED dañado.
●No enciende LED cuando el motor está funcionando.	→Drive defectuoso. →LED dañado. →Cheque visualmente el drive. →limpie las cabezas del drive. →Verifique cables internos ó del controlador. →Intercambie el drive.
●Despliega los mensjes: non-system disk. disk error. disk boot failure.	→El disco no está insertado correctamente. →No contiene el DOS el disco. →No está formateado el disco.
●Despliega el mensaje drive not ready	→No está formateado el disco. →Verificar las puertas del drive que estén cerradas. →Remplace el disco.
●despliega el mensaje: Bad missing command interpreter	→EL archivo COMMAND.COM no →se encuetra en el disco. →Copie el archivo en el disco.
●Despliega mensajes del tipo: 101 0 131 o 1xx	→El sistema de teclados de los switches esté dañado. →Puede haber un falso contacto.
●201 0 xxxx201 checar la paridad x o 20x o xx20x	→Checar los switches del sistema. →Defectos de la memoria. →Verificar que los chips estén bién conectados en la tarjeta principal.

DIAGNOSTICO	HIPOTESIS
•301 o xx301 teclado no funcionando	→El teclado puede estár mal conectado ó dañado.
•mensaje 601 o 6xx	→Deben estar conectados internamente los cables de las unidades del drive. →Pueden estar descincroniza-dos.
•mensaje 4xx	→Checar los cables del monitor. →Verificar la intensidad del monitor.
•mensaje 9xx	→Impresora defectuosa. →Checar los adaptadores y los cables.
•mensaje 13xx	→Adaptador de juegos está mal conectado.
•14xx o problemas de impresión	→Problema interno checar la fuente de poder y los cables.
•mensaje 1701 o 17xx	→Tiene un defecto el disco duro ó está mal conectado. →Checar fuente de poder y los cables.
•mensaje 1801 o 18xx	→Unidad de expansión puede estar defectuosa. →puede existir un cable en mal estado. →Inicializar desde un disco flexible y verificar de nuevo la unidad de disco duro.
•Suena Beep corto y monitor en blanco.	→Puede estar desconectado el cable del monitor. →monitor dañado.
•Cuando inicializa, no enciende el drive (LED), no hay nada en pantalla.	→Checar el cable principal. →Puede ser un poder bajo. →checar que estén bién conectados los cables. →Asegurarse que haya corriente. →Verificar Switch de ON/OFF →Cambiar el Switch.

DIAGNOSTICO	HIPOTESIS
●Cuando inicializa enciende el drive pero no hay nada en pantalla.	→Problamente sean los conectores de la fuente de poder.
●Trabaja bién pero no inicializa	→Checar los circuitos del microprocesador. y el circuito ROM. →Asegurar que el disco contiene el COMMAND.COM. →Verificar que tenga un defecto el disco duro, trate de leer el directorio y ejecute un programa desde el disco duro. →Transferir el sistema: SYS C: →Versión incorrecta del DOS. →Verifique el AUTOEXEC.BAT →Revice el archivo CONFIG.SYS →Usar otro sistema operativo. →inicializar desde un disco flexible. →Verificar los comandos de DEVICE que hagan que no funcione bién el Hardware y el software.

IV.4.2. PROBLEMAS DE CORRIDA (DIAGNOSTICO E HIPOTESIS).

DIAGNOSTICO	HIPOTESIS
•Existe Ventilación hacia la computadora	→La computadora debe tener de 60-80° F
•No funciona el encendido de la computadora.	→Verificar los cables.
•No reponde el programa de inicio con las istrucciones dadas.	→Tal vez funcione con CAPS-LOCK activado.
•No funciona bién al encender la computadora.	→Verificar buén estado de los cables (clavija). →Deje descansar unos 15 minutos a la computadora →inicializar desde el disco flexible y verificar que estén bién los archvios del DOS. →Verificar la memoria. →Verificar teclado.
•Mensajes de paridad ó localidad no encontradas.	→Verificar Chips de memoria →Verificar interruptores. →Verificar hardware.
•Ruidos extraños al encender la computadora.	→Fallas en un chip de memoria →Hardware dañado. →Verificar interruptores. →Puede ser un programa en particular. →Problemas de tarjeta principal. →El hardware o drives.
•Operación erratica y fallas generales.	→Verificar si lo hace un mismo problema. →Verificar cuando se hace la misma cosa. →Checar los programas residentes. →Hacer una nueva copia del programa.
•Problemas en disco duro. (disco duro muerto).	→Mal estado de los cables ó del controlador del disco duro. →El error puede ser de un programa específico. →Mover todos los programas residentes y empezar sin el autoexec.bat →Algún software que interfiere con la lectura.

DIAGNOSTICO	HIPOTESIS
●El archivo CHKDSK genera error	→Puede ser por diseño ó incompatibilidad. →Versión vieja u otra versión del dos. →Clusters perdidos (ver capítulo IV.1).
●El disco duro pierde información	→Revise que funcione bién con otros programas. →Revise el disco duro. →Movimientos físicos del la computadora. →disco duro rayado. →No 'aterriza' correctamente el dico (PARK).
●Problemas de disco flexible.	→Disco no colocado correctamente en el drive. →Daño físico del drive. →Verificar programas residentes. →Checar el autoexec.bat. →No esta bién formatedo el disco. →Defectos de fábrica. →Trabaje con otra marca de disco. →Dañada la cabeza de la unidad de diskette. →Limpiar las cabezas. →Drive mal alineado.
●Despliega el mensaje de Abort, Retry, Ignore.	→Disco no formateado. →Disco mal insertado. →Trabaje con otra marca de disco. →Dañada la cabeza de la unidad de diskette. →Limpiar las cabezas. →Drive mal alineado.
●Sobrecargamientos.	→Asegurar que no este defectuoso un elemento del sistema. →Falsos contactos. →Ventilación.

IV.4.3. PROBLEMAS DE PANTALLA (DIAGNOSTICO E HIPOTESIS).

DIAGNOSTICO	HIPOTESIS
•No despliega información	→Asegurar que el monitor esté encendido. →Mover los controles de contraste y brillantes. →Verificar el buén estado de los cables.
•No despliega E/S de datos	→La fuente de poder puede estár defectuosa. →Probar con otro monitor.
•El monitor se sobrecarga	→Asegurarse que el monitor tenga ventilación. →Monitor defectuoso.
•Descincronización vertical	→Ajustar controles del monitor. →Revisar los controles. →Tal vez no sea el tipo de resolución que necesite la máquina. →Verificar adaptador de video.
•Descincronización horizontal	→Ajustar los controles del monitor →Tal vez no sea el tipo de resolución que necesite la máquina. →Verificar adaptador de video.
•Despliega basura.	→Desconectar el teclado y revisar con otra computadora, si el teclado está bién, entonces probar el monitor. Si esto es correcto, es que la parte central de la memoria no está captando bién la información del teclado.
•Los colores son malos.	→Ajustar los controles de color. →Revisar adaptador de tarjetas.

IV.4.4. PROBLAMAS DE TECLADO (DIAGNOSTICO E HIPOTESIS)

DIAGNOSTICO	HIPOTESIS
•El teclado no hace nada	→Asegurarse que el teclado esté bién conectado. →Verificar que la pantalla despliege cualquier mensaje cuando se teclea otra cosa. →Checar los cables.
•Despliega otros caracteres	→Asegurarse que las teclas estén en buena posición. →Probar con otro teclado. →Teclado en malas condiciones. →Usar los archivos KEYxxx para adaptar el teclado al español. →Configuración al formato USA.
•No funciona una ó más teclas.	→Quitar las teclas y limpiar la base de estas. →Revisar los resortes del teclado. →Verificar que estén bién conectadas las teclas.
•Aparecen dos ó más caracteres al pulsar solo una tecla.	→Reinicializar con CTRL-ALT-DEL. →Si persiste es que el teclado no corresponde ó no es compatible con el sistema de la computadora.
•Tiene objetos ajenos al teclado.	→Verificar que el teclado sea de la configuración de la computadora. →Limpiar la base del teclado. →Si se derramó cafe, quite las teclas y limpie con un limpiador especial. →Utilice aire para remover la basura.
•Derrame de líquidos en el teclado.	→Remover la potencia inmediatamente. →Enjuagar con una franela →Usar aire presurizado. →No usar el teclado durante 48 hrs.

IV.4.5.- PROBLEMAS DE IMPRESORA (DIAGNOSTICO E HIPOTESIS)

DIAGNOSTICO	HIPOTESIS
•Impresora muerta	→Remplazar la fuente de poder de la impresora →Revisar si llega el voltaje adecuado.
•Configuración de la impresora •Fallas al ponerse en ON-LINE	→Ver manual. →Cable defectuoso. →Adaptador defectuoso. →impresora defectuosa.
•Imprime basura.	→Pruebe activando la impresora con CTRL-P →use DIR para verificar que imprima adecuadamente →Desactive la impresora con CTRL-P. →Si esto es correcto, Verificar si el programa a imprimir tiene configurado la salida con respecto a la impresora. →Verificar interruptores.
• Ocasionalmente imprime errores	→La impresora no está ajustada correctamente a los cables.
•Se autointerrumpe cuando imprime	→Verificar la salida del programa. →Checar el carrete del papel →Apage y encienda de nuevo la computadora.
•La cabeza de impresión no corre correctamente.	→Checar la fuente de potencia →Verificar si no existen objetos extraños que obstruyan con la impresión. →Verificar la cabeza de impresión.
•La cabeza corre correctamente pero no imprime.	→la impresora no está instalada correctamente. →Ajustar los interruptores de control. →Verificar la calidad de la cinta.
•La impresora se sobrecarga.	→La impresora se sobrecarga necesita ventilación y descanso.

DIAGNOSTICO	HIPOTESIS
●Imprime caracteres diferentes	→La impresora necesita ajustamiento. →Verificar si el programa a imprimir tiene configurado la salida con respecto a la impresora. →Verificar interruptores.
●El papel no corre correctamente en la impresora.	→Verificar la salida del programa. →Checar el carrete del papel →Apage y encienda de nuevo la computadora. →Checar la fuente de potencia →Verificar si no existen objetos extraños que obstruyan con la impresión.
●No imprime ciertos caracteres al imprimir.	→Verificar la cabeza de impresión. →Checar la configuración de la impresora. →Revisar si es de matriz. →Verificar que esté bién alineada.

CAPITULO V

V.- LOGICA DE PROGRAMACION

El experto humano usa una gran variedad de técnicas para la solución de problemas, ya sea del tipo analógico, numérico de razonamiento, etc. El humano se adapta a un cierto tipo de problemas, lo estudia por todos los ángulos posibles. Comparando alternativas, metodos y técnicas múltiples para la solución de sus problemas.

En TURBO PROLOG hay que manejar ese tipo de técnica puesto que es necesario el entendimiento de el usuario (exterior) y máquina (interior), es decir lograr ese entendimiento en la pantalla (pregunta-respuesta), para esto el sistema necesita de un razonamiento para llegar a su *meta*.

Imaginemos que visitamos al Médico, el malestar es fiebre y ojos inchados ademas de tos. El Doctor hace un *test* y obtiene un resultado (proceso numerico), compara el síntoma con el *test* y el resultado es un '*conocimiento médico*' (razonamiento), e hipótesis.

El síntoma no será a veces exacto como el de otros pacientes, pero si similar, de ahí, una desición (conclusión) para dar un medicamento.

Si el paciente no reestablece, entonces el Doctor toma otro estudio (*test*) y comienza otra hipótesis para otra solución, de lo anterior lo tomará en cuenta (intuición) para otros casos.

La mente del experto humano es tan 'variable' que puede trazar otros caminos para llegar a una solución. Cuando la mente del Doctor trabaja comienza un diálogo nuevo con su paciente; todo lo que percibe el doctor lo razona (base de conocimiento).

De ahi comienza una segunda estructura en su *base de conocimientos*, crea nuevas preguntas. Si el paciente no respondiese correctamente, el doctor toma otra estrategia haciendo un segundo diagnóstico. Entonces el diagnóstico sería múltiple.

En general, el experto humano (Doctor) no usa un proceso lineal para solucionar el problema. La trayectoria para la solución del problema es en ciclos, es decir, dialoga, obtiene una hipótesis y toma una determinación.

Esto no es garantía de una solución exacta, y el experto humano a voluntad usa (normalmente) heurísticas para obtener una solución como sea posible [1].

El Sistema Experto tiene que estar construido, de una manera tal que todas sus funciones sean efectivas. Un SE nunca debe de tener limitaciones, para esto se necesitan herramientas de un experto humano para poderse ayudar en las reglas y en las bases de conocimientos.

El diseño de un Sistema Experto es a un nivel de abstracción. El experto debe de abarcar todos los problemas del mundo real, se debe de contruir un modelo en miniatura de el problema con representación de simbolos. En conclusión, el problema básico en el diseño de un Sistema Experto es la base de conocimiento y la forma de toma de decisiones.

V.1 DESARROLLO DEL SISTEMA DE DIAGNOSTICO BASADO EN TURBO PROLOG.

V.1.1. DEFINICION DE LA META.

El primer paso es definir la especificación de la meta en el sistema. Para el diseño del *Diagnóstico de fallas en PC y compatibles* nuestro dominio es el problema, ya que la técnica es limitada para toda una definición de problemas en el uso de computadoras personales.

Esto puede facilitarse con un metodo que se puede manejar con el Turbo Prolog llamado *BACKWARD o vuelta atrás.*

Con éste método la búsqueda de una solución *(meta)* ya no es tan compleja. Sin embargo para su manejo se necesitan de otros métodos como el manejo de *ARIDAD, REGLAS, VARIABLES, CORTE* y *FALLO.*

V.1.2. OBJETOS Y RELACIONES.

La mayoria de los lenguajes de programación que existen para el manejo de las microcomputadoras -BASIC, PASCAL, etc.- han sido *procedurales.* Tales lenguajes permiten al programador decirle a la computadora lo que tiene que hacer, paso a paso, procedimiento a procedimiento, hasta alcanzar una conclusión o ejecutar una función [2].

El Prolog no es procedural, es *declarativo*. El lenguaje declarativo ahorra mucho trabajo de programación mientras que un lenguaje procedural exige que se introduzca el recipiente y los ingredientes, un lenguaje declarativo solo pide los ingredientes y el objetivo.

En forma más sencilla, el Prolog trata con *objetos* y las *relaciones* entre ellos. Incluso se puede oir hablar de él como un lenguaje orientado al objeto. Un objeto no necesariamente debe de ser algo tangible, puede ser cualquier cosa que pueda representar simbólicamente en una computadora. He aquí algunos ejemplos de objetos:

```
ibm_pc
apple_macintosh
teclado
impresora
```

Aquí hay algunas relaciones que son útiles cuando se consideran estos objetos:

```
periférico
computadora
```

Como se puede observar ninguna de estas palabras están escritas en mayúsculas. No es una falta, se *asume* que comienza con una letra minúscula. El comenzar con una palabra en mayúscula es que se trata de una variable.

V.1.3. -SINTAXIS.

La *sintaxis* de un lenguaje de programación es el conjunto de reglas que gobiernan que las palabras estén bien escritas, la posición en que deben de escribirse estas palabras, los lugares donde van las puntuaciones, etc.

Existen varios tipos de lenguajes en Prolog [2]. Las implementaciones del lenguaje de diferentes compañías de software son a veces muy diferentes. Aun así, muchas de las construcciones del lenguaje funcionarán igual aunque vengan vestidas de diferentes formas.

El Turbo Prolog contiene la mayoría de las características y sintaxis del Prolog desrito en el libro *Programming in Turbo Prolog* de W. F. Clocksin y C. S. Mellish (New York, New York, 1984). El texto de Clocksin y Mellish utiliza un estilo particular de escritura en Prolog, con un conjunto de puntuaciones y ortografía. Dicho estilo es tan útil que se ha convertido en un estándar.

V.1.4.- HECHOS.

La primera forma de combinar un objeto y una relación es usarlas para definir un hecho. La sintaxis en Turbo Prolog es de la siguiente manera:

relación (objeto)

Se observa que el objeto está dentro del paréntesis, y las relacion le precede. Este 'objeto' tiene esta 'relación'. Cuando se escribe de esta forma, la relación se conoce como el *predicado* y el objeto como el *argumento*. Ejemplo:

periférico (teclado)
computadora (ibm_pc)

La traducción al castellano es necesariamente vaga, porque la lógica de los hechos del Prolog no especifica si 'teclado era un periférico', 'teclado es un periférico' o 'teclado parecia un periférico'.

Para Prolog necesitará tener en cuenta lo que son las relaciones. El programa tendrá sentido solo si son consistentes a lo largo de un programa con el significado de una relación dada. Por ejemplo, si se quiere indicar el hecho de que Teclado es un Periferico, se puede utilizar la relación siguiente.

es_un_periferico (teclado)

Los dos signos de subrayado indican a la computadora y compilador que todo esto es una sola palabra para una relación.

V.1.5. DIVISIONES DEL PROGRAMA.

la mayoria de los programas en Turbo Prolog están organizados en cuatro secciones principales.

- Cláusulas (Clauses)
- Predicados (Predicates)
- Dominios (Domains)
- Objetivo (Goal)

Dentro del programa se escribe en <u>ingles</u>, ya que son palabras reservadas.

V.1.5.1. Clausulas.

Los hechos que se constituyen con los objetivos y las relaciones se listan en la sección de *Clauses*.

V.1.5.2 Predicados.

Los *predicates* son las relaciones. El término 'predicado' viene de la lógica formal, la cual es uno de los fundamentos iniciales del Prolog.

V.1.5.3 Dominios.

El Turbo Prolog necesita un nivel más de explicación antes de que un programa esté completo. Se necesitan decirle los argumentos que van a usar los *predicates*.

V.1.5.4 Objetivo.

Esta es la sección que le dice al Turbo Prolog lo que ha de encontrar o lo que desea que la computadora haga con la información que se le ha suministrado en las otras tres secciones.

El Turbo Prolog puede también trabajar de esa forma, pero suministra una sección objetivo (*goal*) para permitirle ejecutar los programas en forma no interactiva. Este puede trabajar en la búsqueda de la solución deseada.

V.1.6. VARIABLES.

Existe otra forma de hacer preguntas, si se conoce la relación pero no los objetos, puede usar *variables*. Cualquier palabra objeto que comienza con una letra mayúscula se considera una variable. Después de las mayúsculas, puede tener cualquier número de letras (en mayúsculas o minúsculas), junto con dígitos y caracteres de subrayado. La forma más rápida rapida de lo que pueda hacer una variable es hacer una pregunta al programa. Ejemplo:

```
periferico (X)
X = teclado
X = impresora
2 solutions
```

V.1.7. ARIDAD: HECHOS CON MULTIPLES ARGUMENTOS.

La *aridad* de un predicado es el número de argumentos que tiene. La mayoria de los programas en Prolog contienen predicados con una aridad mucho mayor, por ejemplo, puede desear utilizar el predicado. Ejemplo:

computadora (ibm_pc)

Puede utilizarse un predicado con una aridad bastante grande para describir la computadora, por ejemplo:

computadora (imb_pc,_512k,_2_floppy,_DOS_2,_2serial,_1paralelo, _Televideo,_Printer_enteta)

La mayoria de las entradas son letras minúsculas, las letras mayúsculas se tratarán como variables, a menos que vayan predecidas de un caracter de subrayado. Deben evitarse los números o precederlos también de un caracter de subrayado.

V.1.8. VARIABLE ANONIMA O BLANCA.

Otro tipo de variable que resulta práctico en algunas situaciones es la *anónima*. LLamada también 'blanca' y se escribe solo como un caracter de subrayado. La varaible anónima se utiliza en los mismos lugares que las variables estándares, pero núnca son definidas con un valor particular. Ejemplo:

periférico (teclado,impresora)

la computadora responderá:

true

utilizando la variable anonima:

periférico (teclado,_)

La respuesta será 'true' teclado es un periférico e impresora también lo es, pero debido a que se utilizo una variable blanca, no se le dice a que tipo de periférico se refiere.

Si utilizamos el objetivo

periférico (_,_)

Prolog entenderá que existen periféricos

V.1.9. REGLAS.

Los hechos no son los únicos en una sección de cláusulas, se pueden también introducir *reglas* en la base de datos. Una regla típica dice que algo es verdad (un objetivo se cumple) si alguna otra cosa es verdad.

Ejemplo, para entender todo lo anterior, se aplicará un ejemplo sencillo de diagnóstico en el editor del Turbo Prolog. Ver programa siguiente:

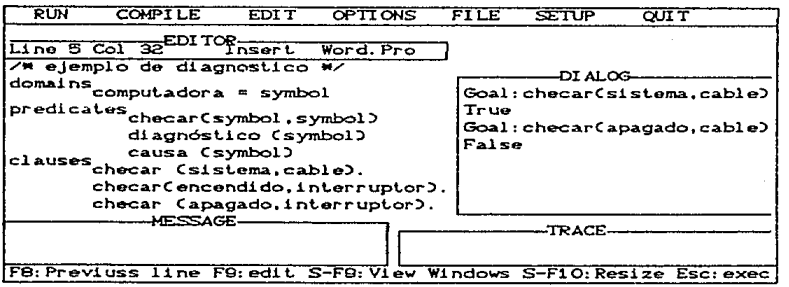


FIG. 5.1 PANTALLA DE EJEMPLO DE DIAGNOSTICO

Notese que en el dialogo se pregunta si *checar(sistema,cable) es cierto*, Prolog responde *TRUE* (si si es verdad); esto se debe a que en las cláusulas está definido. Si preguntamos que *checar(apagado,cable) es cierto*, la respuesta será *FALSE* (falso), ya que apagado corresponde a interruptor.

Continuando con el significado de las *reglas*, se amplia el programa añadiendo las siguientes reglas:

```
diagnóstico (cable) if checar (sistema,cable).
causa (Que) if checar (apagado,Que).
```

También se puede aplicar utilizando el símbolo :- que significa *if*, ejemplo:

```
diagnóstico (cable) :- checar (sistema,cable).
causa (Que) :- checar (apagado,Que).
```

Se hace notar que en las reglas arriba mostradas existe una variable *Que* (comienza con mayúscula). El programa completo en Turbo Prolog quedaría de la siguiente manera:

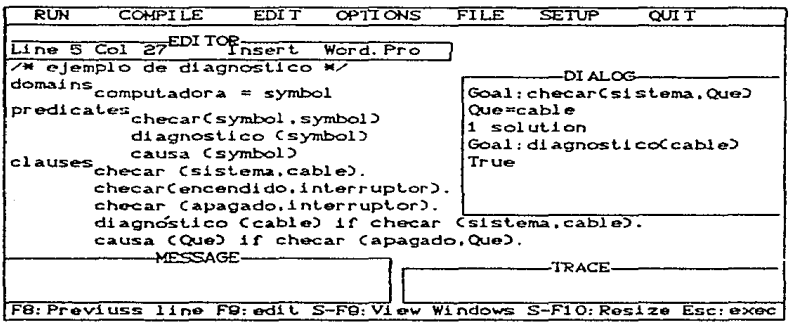


FIG. 5.2 PROGRAMA DE DIAGNOSTICO CON USO DE REGLAS.

Si se teclea:

Diagnóstico(cable)

Se obtendrá la respuesta *True*. El Turbo Prolog intentará satisfacer (o identificar) el objetivo 'diagnóstico (cable)'. De arriba a abajo, buscará en las cláusulas, buscando un predicado que coincida con el predicado 'diagnóstico'. Encuentra dicho predicado cerca del final de la lista. Luego comprueba si dicho predicado tiene la misma aridad -una-. Si es así, continúa el proceso. El Turbo Prolog encuentra la palabra 'if' y concluye que esto es una regla, no un hecho.

V.1.10. VUELTA ATRAS.

En este punto, el Turbo Prolog vuelve hacia la primera cláusula y comienza a buscar hacia atrás en la lista. Esto es *vuelta atrás* y es una característica del Turbo Prolog.

La vuelta atrás puede facilmente crecer en complejidad conforme a las reglas y los objetivos sean más complejos [2].

V.1.10.1 Control de la vuelta atrás.

La vuelta atrás es un elemento del Turbo Prolog. Cuando a un programa se le ha pedido que satisfaga un objetivo, busca de arriba a abajo y de izquierda a derecha a través de las *cláusulas* hasta definir el objetivo. Si llega al final, el programa volverá atrás hasta encontrar otra regla por la que pueda seguir buscando.

Esto no es una manera correcta de buscar información, ya que algunos programas gastan mucho tiempo buscando información que no se necesita para encontrar el material que se ha pedido. Estos programas se necesitan para 'cortar' espacios de búsqueda.

Al diseñar el sistema de diagnóstico, hay que hacerlo de una manera eficiente en la búsqueda. Se debe saber como organizar las clausulas para una mayor eficiencia. Para ello se debe de insertar unas cuantas órdenes especiales en el programa. Estas pueden dirigir la lógica y ahorrar más tiempo de cálculo.

V.1.10.2.- Corte (!).

Esta órden llamado *corte* (!) se escribe en el programa como un signo de exclamación. Corte se comporta como un predicado incorporado sin argumentos. Dentro del sistema se usarán bastantes; es útil para reducir el arbol de búsqueda a un tamaño manejable.

La órden corte (!) impide la vuelta atrás, actuando como una valvula en un sentido. La búsqueda puede llegar por las cláusulas más a la izquierda pero no puede volver a tratarla de nuevo viniendo de una derecha. Cuando se encuentra una !, se liberan las variables vinculadas a la búsqueda de un objetivo o subobjetivo particular de sus valores actuales. Esto significa que no puede trabajar dentro de una regla simple o incluso entre reglas. Si posteriores cláusulas buscadas por el objetivo hacen que falle, Prolog no puede volver al punto de ! dentro de la búsqueda del objetivo y examina otros valores para esas variables -incluso aunque esos otros valores puedan permitir que el objetivo se cumpla-.

V.1.10.3. Fallo (Fail).

Otra órden incorporada para controlar la vuelta atrás, *fail*, es u predicado sin argumentos que prácticamente falla y fuerza la vuelta atrás. En cierto modo, es opuesto al *corte* el cual elimina la vuelta atrás.

V.1.10.4. Traza. (Trace)

Para entender paso a paso como funciona la lógica de un programa, se puede utilizar la ejecución del programa. Si añadimos la palabra *trace* al programa de diagnóstico antes de la línea *domains* (ver fig. 5.3); esto le dirá al compilador que después de compilar la lógica del programa se ejecutará en el modo Trace. Ejemplo, se pregunta si:

checar(sistema,Que)

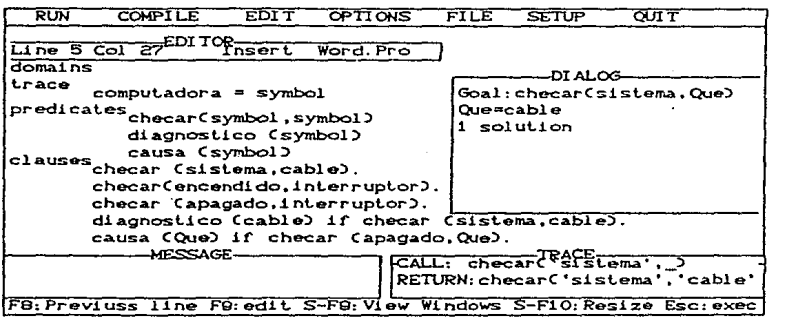


FIG. 5.2 USO DE LA INSTRUCCION TRACE.

El primer paso, Prolog busca un predicado 'checar' con cualquier varaiable. Si al pulsar la tecla de función F10 el cursor se moverá al primer predicado que identifica la sentencia CALL, la cual se encuentra en la línea de la regla:

checar (sistema,cable).

En la pantalla Trace regresa (RETURN) la expresión de la regla

RETURN: checar('sistema','cable')

y por consecuencia la respuesta será *cable*, y Turbo Prolog lo visualiza en el Dialog.

Ahora bién, si utilizamos otro ejemplo como preguntar:

checar(apagado,Que)

los pasos serán similares, en la ventana Trace se visualizará:

CALL: checar(apagado,_)

buscará en las clausulas la expresión: *checar(sistema,cable)*

en la pantalla Trace reintentará (REDO) localizar en la siguiente expresión:

REDO: checar('apagado',_)

y en la cláusula continúa en la siguiente expresión:

checar (encendido,interruptor)

Trace visualiza: *REDO: checar('apagado',_)*; como no cumple con la expresión pasa a la siguiente cláusula:

checar (apagado,interruptor)

Como esta es la expresión que se búsca, Trace lo visualiza en su ventana:

RETURN: ('apagado','interruptor')

por consiguiente la respuesta a la variable 'blanca' es *interruptor*, y en la ventana de diálogo lo despliega como única solución (ver fig. 5.5).

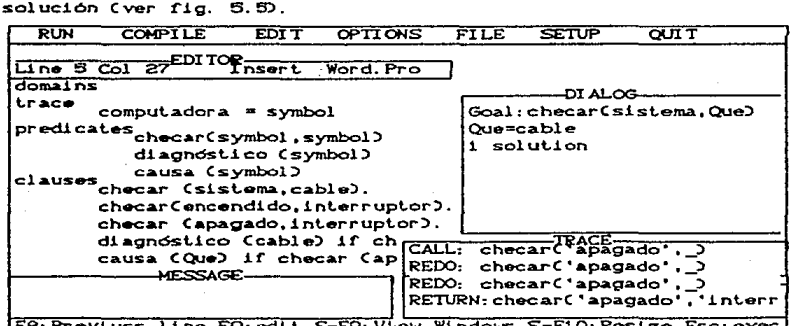


FIG. 5.4. VISUALIZANDO LA VENTANA TRACE

Como se observa en los ejemplos anteriores, Prolog ha respondido a una solución. Para el diseño de un sistema de diagnóstico, la respuesta a una solución sería bastante pobre.

El diseño del Sistema debe de dar más de una solución posible y no limitarse a responder. Para esto es necesario definir bién y concretamente las preguntas. Si se utiliza el mismo ejemplo y haciendo una buena pregunta para que el sistema dé más de una solución podemos lograr coincidir los argumentos del programa. Ejemplo, preguntemos por:

```
checar (Que, interruptor)
```

Turbo Prolog trabajará de la siguiente manera:

LINEA DE LOCALIZACION EN EL PROGRAMA	MENSAJE DESPLEGADO EN VENTANA T R A C E
	CALL: checar(_,'interruptor')
checar(sistema,cable)	*REDO: checar(_,'interruptor')*
checar(encendido,interruptor)	*RETURN: #checar('encendido','interruptor')*
checar(encendido,interruptor)	*REDO: checar(_,'interruptor')*
checar(apagado,interruptor)	*RETURN: checar('apagado','interruptor')*

Finalmenta al pulsar la última tecla F10, la búsqueda de vuelta atrás ha llegado hasta la 'etapa final'. En la ventana Trace muestra en el '*' que localizó una solución pero su búsqueda no queda ahi, ya que tiene que recorrer las demas clausulas, si no encuentra más reglas, entonces es cuando regresa (vuelta atrás) y finalmente despliega:

```
RETURN: #checar('encendido','interruptor')
RETURN:  checar('apagado','interruptor')
```

En la ventana Trace Prolog ha encontrado más de un hecho que coincide con el objetivo temporal y son devueltos con dicho hecho a la pregunta inicial (ver fig. 5.6)

```
goal: checar(Que,interruptor)
Que: encendido
Que: apagado
2 solutions
Goal:
```

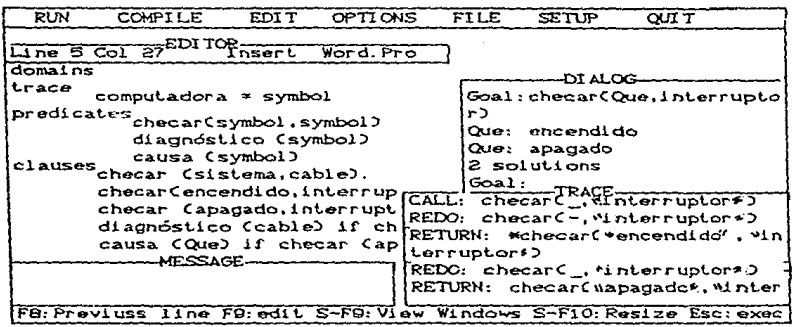


FIG. 5.5. DIAGNOSTICO CON MAS DE UNA SOLUCION

Sin embargo, esta no es la única forma de obtener una traza. Existen otras opciones, como se muestra en la tabla siguiente:

TABLA 5. DIRECTIVOS DE TRAZA, OPERADORES Y PREDICADOS.

trace	Activa la traza del programas
trace p1,p2,...	Hace que se realice la traza de predicados particulares.
trace (on)	
trace (off)	Este es un predicado, no un directivo. Activa o desactiva la traza. puede utilizarse si el directivo "trace" o "shorttrace" está al comienzo del programa, entonces ese es el tipo de traza que activará y desactivará.
shortrace	Activa la traza en el programa entero, pero utiliza la optimización del compilador y por tanto da menos información de traza en algunas circunstancias.
shorttrace p1,p2...	Activa shortrace para predicados particulares.
ALT-T	Esta combinación de tecla puede utilizarse para conmutar la traza. Pulsando ALT-T, puede activar y desactivar la traza mientras va trazando paso a paso un programa.

TABLA 5.1. MENSAJES DURANTE LA TRAZA.

CALL	Aparece cada vez que llama un predicado. Muestra el predicado y los valores actuales de sus argumentos.
RETURN	Aparece cuando ha sido satisfecha una cláusula y la lógica del programa ha vuelto al predicado anterior. Un asterisco después de RETURN significa que hay otra cláusula para la que podría identificarse los argumentos de entrada, en este punto necesita ser examinado y que ha sido marcado para una futura vuelta atrás.
FAIL	Aparece cuando ha fallado una cláusula, la cláusula no puede ser identificada o el Turbo Prolog no puede realizar una tarea.
REDO	Aparece cuando el programa está volviendo hacia Atras. Muestra el nombre del predicado rellamado y el valor actual de sus argumentos.

No todos los predicados son iguales a los ojos del modo traza. Por ejemplo los CALL y RETURN en un predicado write (que veremos más adelante) no son notificados en la ventana Trace porque tiene un número indeterminado de argumentos. Existen otros predicados que son casos excepcionales para la traza y se muestran a continuación:

TABLA 5.2 OPERADORES Y PREDICADOS QUE RECIBEN TRATAMIENTO DE TRAZA ESPECIAL

OPERADORES DE COMPARACION		PREDICADOS
OPERADOR	SIGNIFICADO	asserta
=	igual que	assertz
<>	Diferente	bound
><	Diferente	findall
>	Mayor qe	free
>=	Mayor o igual que	not
<	Menor	readterm
<=	Menor o igual que	retract
		write
		writef

V.2. FORMAS DE TRABAJO PARA EL DISEÑO DEL SISTEMA DE DIAGNOSTICO.

Debido a que el Turbo Prolog tiene un sistema de ventanas incorporado (y el editor), no ha existido la necesidad de hacer que el programa ejecute operaciones de E/S, como las que requeriría cualquier otro lenguaje de programación. Las cuatro ventanas -Editor, Trace, Message y Dialog- han estado muy ocupadas presentando mensajes y resultados. Para esto se puede construir ventanas propias y leer y escribir a través de ellas.

V.2.1. ESCRITURA.

El primero de los predicados de escritura, es el de uso general *write*. El predicado WRITE se cumple cuando escribe los argumentos que contiene en la pantalla en la posición actual del cursor. Si se le agrega un slash al revés convierte a esto en un caracter de control. A continuación se muestra los diferentes caracteres de control:

TABLA 5.3 CARACTERES DE CONTROL.

\	indica que el siguiente caracter es un caracter de control.
\n	Caracter de control para una nueva linea.
\t	caracter de control para un tab.
\b	caracter de control para un espacio atras.
\\	la forma de escribir un slash al reves en la pagina

V.2.2.- LECTURA.

Turbo Prolog también puede leer información desde el teclado o desde un archivo en disco. Puede leer cualquier cosa desde un carácter hasta una línea entera de caracteres. No puede leer objetos o listas compuestas. La tabla 5.4. muestra los tipos de lectura estándares.

TABLA 5.4. PREDICADOS ESTANDARES DE LECTURA.

readint (Variable Entera) integer -(O)

Lee un entero. La variable debe estar libre y el valor vinculado a ella debe estar dentro del dominio estandar 'int'. Lee desde el dispositivo de entrada actual hasta que se pulse RETURN.

readreal (Variable Real) real -(O)

Lee un número real. La variable debe de estar libre y el valor vinculado a ella debe de estar dentro del dominio estandar 'real'. Lee desde el dispositivo de entrada actual hasta que se pulse RETURN

readchar (Variable Caracter) char-(O)

Lee un carácter. La variable debe estar libre y el valor vinculado a ella debe de estar dentro del dominio estandar 'char'. Lee un carácter desde la posición actual del cursor. A diferencia de 'readint' y 'readreal', 'readchar' no espera que se pulse RETURN.

readln (Cadena) String-(O)

Lee una cadena. La variable debe de estar libre y el valor vinculado a ella debe estar dentro del dominio estándar 'string'. Lee desde el dispositivo de entrada actual hasta que encuentra un carácter ASCII de vuelta de carro.

inkey (car-O)

Lee un carácter desde el dispositivo de entrada estándar. Si no puede leer tal carácter, 'inkey' falla.

V.2.3.- NL

Este predicado incorporado -llamado *nueva línea*-, es similar al predicado 'write' sin argumentos. Todo lo que hace es mover el cursor a la línea siguiente.

V.2.4. VENTANAS Y POSICIONES DEL CURSOR.

Las *ventanas* -porciones de las pantallas que actúan a su vez como pantalla- se han convertido en una parte estándar en Turbo Prolog que depende fuertemente de sus cuatro ventanas, incorporadas para hacer el desarrollo de los programas más fácil y más estructurado. Turbo Prolog también contiene predicados que le permitirán construir sus propias ventanas para entrada y salida de programas.

En la siguiente tabla se observa la sintáxis para realizar diferentes tipos de ventanas.

TABLA 5.5. PREDICADOS DE VENTANAS ESTANDARES

makewindow (NumVentana,AtribPant,AtribCuadro,Cabecera,Fila, Col,Altura,Anchura)(int,int,int,string,int,int,int,int)(I, I,I,I,I,I,I)

CREA UNA VENTANA VACIA.	Todas las variables deben estar vinculadas.
NumVentana.	Identifica el número de ventana. Cada ventana tiene un número entero diferente.
AtribPant.	Determina el estilo de presentación de la ventana. Las ventanas 5.5a y 5.5b. muestran los valores de los atributos.
AtribCuadro.	Determina la existencia y estilo de presentación para el cuadro alrededor de la ventana. Ver tablas 5.5a y 5.5b.
Cabecera	Cadena escrita al principio de la ventana.
Fila.	Número entero de fila, las filas se encuentran desde el principio de la pantalla.
Col.	Número entero de columna, las columnas se encuentran desde la izquierda de la pantalla.
Altura.	Número entero de filas de la ventana.
Anchura.	Número entero de columna de la ventana.

NOTA: El predicado 'graphics' puede usarse para cambiar el número de columna de la pantalla. Los modos CGA y EGA son hasta 640 X 200 y 640 X 350 respectivamente.

vindow_attr(Atrib) Integer-(I)

Cambiará los atributos de la ventana activa al valor entero de la variable vinculada. Permite cambiar el estilo de presentación o color de la ventana abierta sin volver a tratar con el predicado 'makewindow'.

removevindov

Quita la ventana activa de la pantalla. No tiene argumentos.

Shiftvindov (NumVentana) integer-(I)(O)

Si la variable está libre, la vinculará al número de ventana activa. Si la variable está vinculada, hara la ventana 'NumVentana' la nueva ventana activa.

clearvindov

Borra todos los caracteres de la ventana activa y los remplaza con espacios en blanco de atributo de fondo.

TABLA .5.5 PREDICADOS DE VENTANAS ESTANDARES (CONTINUACION).

cursor(Fila,Columna) Integer,Integer-(I,I) (O,O)

Requiere de ambas variables estén libres o vinculadas. Si ambas están libres, las vinculrará a los enteros que representan la fila y columna de la posición del cursor dentro de la ventana activa o de la pantalla.

vindov_str (Cadena,Pantalla) string-(I)(O)

Si la variable está acotada, escribirá el valor de 'cadena de pantalla' en la ventana activa. Comienza en la posición actual del cursor Trunca cada línea que sea demasiado larga y trunca las líneas que pasarán del final de la pantalla.

En las tablas siguientes de da una lista de los valores que se pueden seleccionar para hacer una ventana.. Para escoger un atributo de video monocromo, hay que encontrar el valor de la principal decisión y ajustarla con valores.

TABLA 5.5A.

DECISIONES PRINCIPALES:

ESTILO	VALOR	ESCRITURA (CARACTERES)	FONDO
En blanco	0	negro	negro
Normal	7	blanco	Negro
Video Inverso	112	negro	blanco

DECISIONES MENORES:

1.- Subrayar caracteres en el color de escritura: añadir 1.
2.- mostrar la parte blanca de la representación en alta intensidad: añadir 8.
3.- Parpadear caracteres: añadir 128.

En la tabla 5b muestra los valores similares para un video en color.

TABLA 5.5B

COLOR	ESCRITURA	FONDO
POR COLORES:		
Negro	0	0
Gris	8	na
Azul	1	16
Azul claro	9	na
Verde	2	32
Verde claro	10	na
Celeste	3	48
Celeste claro	11	na
Rojo	4	64
Rosa	12	na
Magenta	5	80
Magenta claro	13	na
Marrón	6	96
Amarillo	14	na
Blanco	7	112
Blanco claro	15	na
POR VALORES:		
Negro	0	0
Azul	1	16
Verde	2	32
Celeste	3	48
Rojo	4	64
Magenta	5	80
Marrón	6	96
Blanco	7	112
Gris	8	na
Azul claro	9	na
Verde claro	10	na
Celeste claro	11	na
Rosa	12	na
Magenta claro	13	na
Amarillo	14	na
Blanco claro	15	na

* La designación 'na' significa que el color que se trate no está disponible para el fondo 'blaco claro', significa blanco de mayor luminosidad.

V.2.5. DIRECTIVOS DEL COMPILADOR.

Cuando se intenta compilar un programa, el Turbo Prolog comprueba primero, si el código del programa sigue a las reglas sintácticas del lenguaje. Luego examina los argumentos por si se han empleado mal los dominios. El Turbo Prolog es un compilador de Prolog (1a. versión), pero sus sistemas de menus y ventanas permiten desarrollar interactivamente los programas. Cada vez que el Turbo Prolog encuentra un error sintácticos o de dominios, señalará dicho error con el cursor (en la ventana del editor), dará un mensaje explicando el problema y se posicionará en el modo editor para resolver el problema. Después de cada corrección, solo se tiene que pulsar la tecla F10 para que continue con la compilación.

Para el diseño del *Sistema Experto para el diagnóstico de fallas en equipo PC*, es necesario utilizar los distintos directivos del compilador. Estos son instrucciones (listados en la tabla 5.6) que se añaden en el archivo fuente del programa para decirle al compilador cómo manejar la compilación. Algunos directivos permiten compilaciones complejas, de archivos múltiples, y otro ayudan a conocer lo que está haciendo el programa. De hecho no son predicados, son instrucciones al sistema de Turbo Prolog.

TABLA 5.6. DIRECTIVOS DEL COMPILADOR.

check_cmpio	Comprueba si algún predicado utiliza patrones de flujo compuesto.
check_determ	Comprueba si algún predicado tiene cláusulas no deterministas.
Code = nnnnn	Da el tamaño máximo de la memoria al código. La cuenta está en el párrafo. Cada párrafo es igual a 16 bytes. El tamaño por defecto es 1k por párrafo (en dicho caso nnnnn es igual a 1024).
include 'nombrearchivo'	Permite escribir programas que se extienden sobre varios archivos. El archivo denominado será incluido en la compilación y puede a su vez contener un directivo 'include'

TABLA 5.6 DIRECTIVOS DEL COMPILADOR (CONTINUACION)

diagnostics	Imprime datos de diagnóstico del compilador.
nobreak	Normalmente el Turbo Prolog comprueba periodicamente si ha pulsado alguna tecla. Esto permite utilizar CTRL-BREAK para salir del programa que pueden haber caido en bucles. Este directivo elimina tal comprobación.
nowarnings	Evita que el turbo Prolog mande mensajes de aviso.
shorttrace	Ver tabla 5
shorttrace p1,p2,...	Ver tabla 5.
trace	Ver tabla 5.
trace p1,p2,...	Ver tabla 5.
trail = nnn	Especifica el tamaño de la cola (en bytes). La cola es el área utilizada para registrar la vinculación y desvinculación de variables referenciadas. El tamaño por defecto es cero (0).

V. 3. DISEÑO DEL SISTEMA

Como se estableció en el Capítulo 1, los Sistemas Expertos son programas que imitan el comportamiento de un experto humano. Usa la información que suministra el usuario para devolver una opinión sobre una cierta materia. De este modo el SE le pide que responda preguntas hasta que pueda identificar un objeto que se adecúa a sus respuestas. Para comprender lo que hará el SE, se presenta un diálogo entre un experto en computación y alguien que busca un consejo.

Experto	Falla sus drives ?
Usuario	Si.
Experto	No graba información ?
Usuario	No.
Experto	No lee correctamente los datos ?
Usuario	Si
Experto	Despliega el mensaje de: **WRITE PROTECT IN DRIVE X ABORT, RETRY OR IGNORE ?**
Usuario	Si
Experto	Despliegue la etiqueta de protección que se encuentra en la parte superior derecha de su diskette.

El objetivo de el Sistema Experto en diagnóstico de fallas en equipo PC, es ser capaz de reproducir este diálogo. Más generalmente, un Sistema Experto intentará dar consejos al usuario sobre el dominio en el que es experto.

V.3.1. LA BASE DE CONOCIMIENTOS.

La base de conocimientos es una base de datos que toma información y reglas específicas sobre una cierta materia. He aquí dos términos que debería conocer para esta discusión:

- *Objeto.* Conclusión que se define por sus reglas asociadas.
- *Atributo.* Una cualidad específica que, junto con su regla, ayuda a definir el objeto.

En consecuencia, puede pensar en la base de conocimientos como en una lista de objetos con sus reglas y atributos asociados.

Para el diseño del SE, puede definir un objeto como una lista de atributos que el objeto posee o no. De este modo, para la mayoría de los propósitos, la regla que se aplica a un atributo establece que el objeto 'falla' o 'no falla' el atributo. Por ejemplo, el SE que identifica varios tipos de fallas debe de tener una base de conocimientos como se muestra:

DRIVE	*falla*	en la escritura
	no falla	en la lectura
	falla	con un mismo disco
	no falla	con otro disco
MONITOR	*falla*	en la visualización de caracteres
	no falla	al encender
	falla	con el mismo programa
	no falla	con otro programa
TECLADO	*falla*	al pulsar una tecla
	no falla	en las demas teclas
	falla	al pulsar una tecla y visualice otra
	no falla	con otro teclado

En la base de conocimientos, se puede simplificar: puede usar solo una regla -falla- y usar la forma negativa del atributo se debe establecerse una relación de "no tiene". De este modo, la regla se convirte simplemente en fallo, y la base conocimiento simplificada aparece asi:

DRIVE	tiene falla en la lectura y con un mismo disco
MONITOR	falla en la visualización de caracteres con un programa en si
TECLADO	tiene una falla al pulsar una tecla y se visualiza otra en la pantalla

Esto simplifíca en gran medida la base de conocimientos. Para el desarrollo del sistema considerese que la base de conocimientos consiste en solo objetos y atributos.

V.3.2. EL MECANISMO DE INFERENCIA.

El mecanismo de inferencia es la parte del sistema experto que intenta usar la información que le da para encontrar un objeto que case. Hay dos categorías claras de mecanismos de inferencia: determinísticos y probabilísticos [2]. Para comprender la diferencia, imagine que hay dos expertos, uno en química y otro en sociología. El químico puede decir que con *certeza* que si el átomo en cuestión tiene dos electrones, entonces es un atomo de helio. No hay duda acerca de la identidad del átomo porque el número de electrones determina el tipo de elemento. Sin embargo si se pregunta al sociologo cuál es la mejor forma de prevenir los abandonos en la universidad, el sociologo dará una respuesta que esta calificada como ser sólo probable o como que tiene una cierta probabilidad de éxito. En consecuencia la respuesta es verosímil, pero *incierta*.

la mayoría de las disciplinas no son deterministicas, sino probalisticas en cierta medida. Sin embargo para muchas de éstas, la incertidumbre no es estadisticamente importante. El diseño del SE tratará solo de la forma determinística porque su lógica es más clara.

De estas dos categorías claras de certidumbre e incertidumbre, hay tres formas de básicas de contruir el mecanismo de inferencia: encadenamiento hacia adelante, encadenamiento hacia atras y valor de la regla (encadenamiento hacia adelante y hacia atras). La diferencia de los métodos se refiere a cómo intenta el mecanismo alcanzar su objetivo.

V.3.2.1. ELECCION DE UN METODO.

El método de encadenamiento hacia adelante hace más fácil el proceso de derivar la mayor cantidad de información de base de la base de conocimientos porque construye un arbol. Los sistemas de encadenamiento hacia adelante encuentran comunmente todos los objetos posibles que encajan con los atributos. La ventaja de un SE que utiliza un encadenamiento hacia atrás es que requiere solo la información suficiente para encontrar un objeto. De este modo, debido a que los SE de encadenamiento hacia atrás son guiados por el objetivo (goal), solo permiten que entren en el sistema de información revelante, mientras que los sistemas expertos de encadenamiento hacia adelante puede tener que descartar información extraña.

Un sistema de encadenamiento hacia atrás es útil cuando se quiere encontrar un objeto -incluso aunque otros objetos también satisfagan los atributos-. Sin embargo si todos los casos son iguales , el enfoque de encadenamiento hacia atrás es un poco más fácil de implementar y produce un sistema experto eficaz. Por estas razones, este capítulo desarrollará un Sistema Experto con encadenamiento hacia atrás (decisión en estructura de arbol), y con una base de datos dinámica, (fig. 5.6).

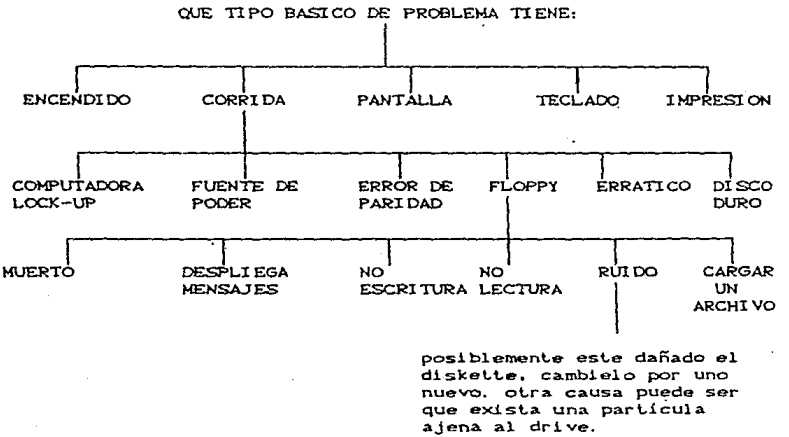


FIG 5.6 ENCADENAMIENTO HACIA ATRAS -MODO ESTRUCTURA DE ARBOL-.

V.3.2.2. OBTENCION DE FACTORES.

Antes de escribir el código para un sistema experto, se define, en términos prácticos , lo que se necesita hacer para crear un mecanismo de inferencia. Esta sección consiste que la base de conocimientos son las hipótesis y los síntomas. En cada caso la meta (goal) es definido como una hipótesis, y es la causa.

En la tabla 5.7 se muestra una lista de ciertos factores para arreglar una computadora.

TABLA 5.7 LISTADO DE FACTORES

HIPOTESIS	SINTOMA
La unidad de disco, no funciona correctamente.	La luz de la unidad de disco no enciende. Existe ruido durante la lectura/escritura. Despliega con frecuencia mensajes de I/O error.
El disco flexible esta físicamente defectuoso	El disco no contiene archivos del DOS. La luz del drive queda encendida sin encontrar un archivo. El disco es ruidoso dentro del drive. El disco tiene problemas de lectura/escritura. Despliega mensajes de sectores dañados.
El protector de lectura se encuentra en la ranura del diskette.	Puede leer pero no escribir. Despliega el mensaje: WRITE PROTECT ERROR READING IN DRIVE N? Abort,Retry,Ignore? El programa FORMAT no formatea el disco.
La unidad de disco esta fuera de alineamiento.	Tiene problema de lectura/escritura. despliega sectores dañados.
El disco esta insertado incorrectamente	No lee ni escribe. El disco genera ruido. Despliega el mensaje de: NOT READY READING IN DRIVE X. Abort,Retry,Ignore?
La puerta de la unidad de drive no está cerrada.	No lee ni escribe. El disco genera ruido. Despliega el mensaje de: NOT READY READING IN DRIVE X. Abort,Retry,Ignore? No formatea el disco.

V.3.3. INGENIERIA DEL CONOCIMIENTO.

El sistema experto a desarrollar en este capítulo comienza su búsqueda con la primera entrada en la base de conocimientos y procede simplemente en órden secuencial a lo largo de la lista de objetos. Mientras que este proceso está bién para pequeñas cantidades de objetos, podría causar problemas cuando se usan grandes bases de conocimientos. Por ejemplo, si hubiera mil objetos, y el que se describe es el número 999 y tiene muchos atributos comunes, entonces a este tipo de sistemas expertos le podría llevar mucho tiempo encontrar la solución. Intentar resolver este tipo de problemas, junto con otros conduce al campo de la ingeniera del conocimiento.

La ingeniería de conocimiento es la disiplina que trata de la forma en que se organizan, construyen y verifican las bases de conocimientos. Aunque no es el objetivo de este trabajo presentar una discusión completa de este tema, es importante estar atento a las dificultades que se puede encontrar cuando se cosntruya la base de conocimiento.

V.3.4. DIAGRAMA DE HECHOS.

El siguiente diagrama muestra la forma de relacionar los síntomas con respecto a las hipótesis. En la parte superior del encabezado muestra las definiciones de las hipótesis, mientras que a la izquierda se muestra los síntomas. (fig. 5.8)

Los síntomas marcados muestran una conclusión (hipótesis) que se describen en la parte superior. Esta tabla nos ayuda a encontrar más de una solución.

TABLA 3.8 DIAGRAMA DE HECHOS.

SINTOMAS	HIPOTESIS					
	DRIVE FUERA DE ALINEAMIENTO	DISCO DEFECTUOSO	FALLAS EN LA CABEZA DEL DRIVE	PROTECCION DE ESCRITURA	PROGRAMAS REDISENTES QUE CAUSEN PROBLEMAS	DISCO MAL INSERTADO.
NO INICIALIZA.	XXXX	XXXX	XXXX			XXXX
EL DRIVE NO ENCIENDE LA LUZ.			XXXX			
NO TIENE SONIDO EL DRIVE.			XXXX			
EL DRIVE HACE DEMASIADO RUIDO.		XXXX	XXXX	XXXX	XXXX	XXXX
PROBLEMAS DE LECTURA.	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
PROBLEMAS DE ESCRITURA.	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
DIFICULTAD PARA INSERTAR EL DISCO.	XXXX	XXXX				
MENSAJES DE ERROR DE PISTAS DAÑADOS.	XXXX	XXXX	XXXX			
EL DISCO NO TRABAJA DENTRO DEL DRIVE.		XXXX				
MENSAJE DE PROTECCION DE ESCRITURA.				XXXX		
EL PROGRAMA 'FORMAT' NO FORMATEA EL DISCO.	XXXX	XXXX	XXXX			XXXX
DESPLIEGA MENSAJE DE NO LECTURA.			XXXX			XXXX
EL DISCO TRABAJA EN OTRO DRIVE.	XXXX					
LA PUERTA DEL DRIVE ESTA ABIERTA.						XXXX

V.3.5. LA FORMA DE OPERAR.

La base de conocimientos consistirá sólo en objetos y atributos. para comprender lo que se debe ser capaz de hacer un mecanísmo de inferencia, se usa la pequeña base de conocimientos que se muestra aquí:

objetos	*atributos*
1	A,B,C
2	A,B,Y
3	B,X
4	A,B,D

En el nivel más primitivo, el mecanismo de inferencia comienza suponiendo que el objeto 1 es el objetivo y trata de confirmar esto preguntando si el objetivoo tiene los atributos del objeto 1. Si los tiene, entonces el mecanísmo de inferencia manifiesta que el objeto 1 es la respuesta. Si no, entonces el mecanismo de inferencia procede con el objeto 2 e inquiere sobre los atributos del objeto 2. este proceso se repite hasta que se encuentra el objeto apropiado, o bién no hay más objetos. Si son estos los pasos que se toma el mecanismo de inferencia, y si el objetivo es el objeto 4 , tendrá lugar el siguiente dialogo:

```
experto   ¿tiene A?
usuario   sí
experto   ¿tiene B?
usuario   sí
experto   ¿tiene C?
usuario   no                  rechazar 1
experto   ¿tiene A?           redundante
usuario   sí
experto   ¿tiene B?           redundante
usuario   sí
experto   ¿tiene Y?
usuario   no                  rechazar 2
experto   ¿tiene B?           redundante
usuario   sí
experto   ¿tiene X?           innecesario
usuario   no                  rechazar 3
experto   ¿tiene A?           redundante
usuario   sí
experto   ¿tiene B?           redundante
usuario   sí
experto   ¿tiene D?           encontrado
usuario   sí
experto   es el objeto 4
```

El sistema Exhibe dos defectos. Primero, pregunta sobre el mismo atributo varias veces. Segundo, cuando vuelve al objeto 3, hace una pregunta innecesaria. El sistema debe saber por sus preguntas previas que el objeto en cuestión tiene el atributo B, y deberá rechazar el objeto 3 porque no tiene el atributo B.

Aunque es posible que podría haber sido útil saber sobre el atributo X, en término medio, el *mecanísmo* de inferencia será más eficiente si pasa por alto todos los objetos que no se ajusten al estado actual. De este modo, lo que se desea es un *mecanísmo* de inferencia que, cuando se da la misma base de conocimiento con el objeto 4 como objetivo, produzca este dialogo:

```
experto   ¿tiene A?
usuario   sí
experto   ¿tiene B?
usuario   sí
experto   ¿tiene C?
usuario   no              rechazar 1
experto   ¿tiene Y?
usuario   no              rechazar 2
                          rechazar 3
experto   ¿tiene D?       encontrado
usuario   sí
experto   es el objeto 4
```

Con este tipo de mecanísmos de inferencia como objetivo, se concluye las especificaciones que debe conformar el mecanísmo de inferencia del ejemplo:

1.- El sistema experto no pregunta sobre el mismo atributo dos veces.

2.- El sistema experto rechazaría inmediatamente y pasaría por alto cualquier objeto que no tenga los atributos necesarios conocidos.

3.- Si se le ordena, el sistema experto debería ser capaz de manifestar por qué está siguiendo una línea de razonamiento.

La tercera especificación se añade no solo como una forma de verificar que el sistema experto está operando correctamente, sino como un *método* de educar al usuario [2].

V.4. LOGICA DEL SISTEMA EN TURBO PROLOG.

En esta sección comenzaremos a decribir las partes principales del diseño del *sistema de diagnóstico* utilizando todo lo que se ha descrito en los capítulos anteriores. Para esto, comenzaremos con la estructuración lógica del programa.

V.4.1. ESTRUCTURACION DE LA BASE DE CONOCIMIENTOS.

El primer paso para crear un sistema experto es definir la estructura de la base de conocimientos. La base de conocimiento es como sigue:

```
code=3072
database
dbase (symbol,char)
```

donde **char** será definida como una lista de símbolos. Como se vé, el mecanismo de inferencia debe llevar la cuenta de todos los atributos que pertenecen al objetivo y aquellos que no pertenecen. Por consiguiente, la porción de declaración de base de datos completa es:

```
code=3072
database
dbase (symbol,char)

include "a:menupro"

predicates
run
titulo
diagnostico(symbol)
diag(symbol)
checar(symbol,char)
check(symbol)
clear_facts
respuesta(char)
repeat
run_once
causa(symbol)
```

en esta sección se declara los predicados donde **checar** checa la lista de símbolos de **check**.

Nótese que se incluye un programa llamado **menu.pro**, este es un programa que solamente da presentación al sistema (color, ventanas, etc..).

El paso siguiente es hacer la meta y un cuadro de presentación para ser ejecutado. Una vez que el sistema experto ha hecho el cuadro, la instrucción siguiente dice que comience con la rutina maestra denominada **run**.

```
goal
makewindow(2,7,7,-,0,0,25,80),
run.
```

V.4.2. CARGA DE LA BASE DE CONOCIMIENTOS.

Antes de que pueda desarrollar el mecanismo de inferencia, se debe de crear rutinas que le permitan situar la información en la base de datos. De hecho se puede escribir por separado (como **menu.pro**), pero la mayoría de los sistemas expertos según el autor del libro de *Programación Avanzada, HERBERT SCHILDT, Mc. Graw Hill 1988.*, no es la forma de hacer un programa en Prolog. Como se muestra a continuación, en la parte de CLAUSES se ejecutan los objetos y atributos, es decir, run lee el nombre de un objeto y limpia los factores de la base de datos, dicho atributo busca una relación a ese objeto. Este proceso se repite hasta que se teclea n de 'no'. Si el usuario teclea y como respuesta, entonces se repite el procedimiento. He aquí la rutina maestra.

```
run :-
    clearwindow
    run_once,
    clear_facts,
    write(-\n Desea otra consulta (y-n)?-),
    readchar(Reply),
    write(Reply),nl,
    Reply=-y-
    run.
```

La rutina siguiente, es la rutina de ejecución del título principal, nótese que diagnóstico lleva consigo una 'variable blanca'. Esta variable es la que se utilizara para la búsqueda del objetivo, observe el símbolo de corte para la búsqueda de el objetivo.

```
run_once :-
    titulo,nl,diagnostico(_),!,clear_facts.
```

Si diagnostico(_) no encuentra el objetivo, el corte origina que despliegue un mensaje de la siguiente manera:

```
run_once :-
     write('\n Lo que desea determinar.'),nl,
     write('no esta en mi base de conocimientos \n.'),clear_facts.
```

La rutina run_once hace que despliegue el título principal del sistema, y este se localiza en la rutina con el mismo nombre titulo.

```
     titulo :-
     clearwindow,
     write('     IBM_PC\XT  SISTEMA EXPERTO PARA DIAGNOSTICO DE
FALLAS DE EQUIPO '),nl,nl.
```

La rutina siguiente repite los títulos de los menús.

```
     repeat.
     repeat :- repeat.
```

Una vez que despliegue en pantalla el título principal regresará a la rutina que lo llamó para continuar con la siguiente instrucción nl, seguido de la nueva línea, comenzará el diagnóstico diagnóstico(_), y una vez resuelto limpiará las variables y los factores que se usó en el programa con la rutina clear_facts.

```
clear_facts :-
    retract(dbase(_,_)),fail.

clear_facts :- nl.
```

Limpia los factores (variables blancas) de la base de datos, donde fail indica ir a la siguiente clausula.

V.4.3. CONSTRUCCION DEL MECANISMO DE INFERENCIA.

Como se observó anteriormente, la rutina run_once desplegó el título y se solicitó una nueva línea. A continuación, esta rutina pide se haga un diagnóstico de cualquier cosa diagnóstico(_). Como el predicado del mecanismo de inferencia es diagnóstico, ésta se irá a la primera rutina de diagnóstico que encuentre en el programa. La rutina es la siguiente:

```
diagnóstico(ac_power) :-                                    ....(1)
     checar(system,'1'),                                    ....(2)
     checar(startup,'2'),                                   ....(3)
     not(check(power)),                                     ....(4)
     causa(ac_power).
```

Ahora diagnostico(_) es denominada diagnóstico(ac_power) ya que encontró la primera rutina. Después, el predicado **checar(system,-1.)**, solicita se cheque el mecanismo de inferencia y manda a llamar la rutina que a continuación se muestra:

```
checar(Z,X) :-
  dbase(Z,Y),X=Y,!.
```

donde Z y X tienen el valor de system y -1- respectivamente, en la base de conocimientos **dbase** está denominada como **dbase(system,_)**, ya que la variable Y no tiene valor 'variable blanca'. La clausula siguiente hace que X tenga el valor de Y, es decir X=Y, donde X será de nuevo una 'variable blanca' Como Z tiene valor y X no lo tiene, Prolog pide se llame a una rutina que cumpla con las variables.

Dicha rutina se muestra en el siguiente punto.

V.4.4. RUTINA PRINCIPAL DEL SISTEMA.

Aqui podemos cargar la base de conocimientos del sistema, y entonces estará listo para atacar el mecanismo de inferencia. El mecanismo de inferencia es la fuerza conductora del sistema experto. A continuación se despliega el menú principal del sistema de diagnóstico usando la variables denominada de Z y X.

```
checar(system,X) :-                                          ....(4)
     not(dbase(system,_)),
     repeat,
     titulo,
     write(' QUE TIPO BASICO DE PROBLEMA TIENE ? '),nl,
     write('  1)Problema al inicializar '),nl,
     write('  2)Problema de corrida '),nl,
     write('  3)Problemas de pantalla '),nl,
     write('  4)Problemas de teclado '),nl,
     write('  5)Problemas de impresora '),nl,
     write(' SELECCIONE : '),
     respuesta(Reply),                                       ....(5)
     char_int(Reply,Z),                                      ....(6)
     Z<54,Z>48,!,                                            ....(7)
     asserta(dbase(system,Reply)),                           ....(8)
     X=Reply.                                                ....(9)
```

Esta es la rutina principal del Sistema Experto, donde X será la variable que almacene la opción deseada por el usuario. Las expresiones (5), (6), y (7) son usadas para que el usuario no pulse una tecla erronea, es decir, la expresión (5) indica llame a una rutina denominada respuesta(Reply).

```
respuesta(Reply) :-
        readchar(Reply),
        write(Reply),nl.
```

esta rutina se utilizará en todas las expresiones para obtener una respuesta. readchar(Reply) leerá el valor de la opción solicitada por el usuario (sin esperar se pulse la tecla <return>) y lo escribirá en la expresión write(Reply). Una vez hecho esto, regresará a la parte de la rutina donde fue llamada, es decir a la expresión (5).

En la expresión (6) el valor de Reply sera convertida en un valor entero equivalente de la variable vinculada Reply, es decir, el valor entero del código decimal ASCII -ver Apéndice A-, funciona con los dominios de char_int, y es almacenada en la variable Z, entonces Z y Reply tendrán el valor de 1 y 49 respectivamente. La expresión (7) es una comparación, si el código ASCII, al pulsar la opción 1 el valor de el caracter según el código ASCII) es 49, entonces, al comparar decimos:

```
49 < 54, 49 > 48
```

como esto se cumple, le dice a asserta (acierto) que lo almacene en la base de datos que está en la memoria RAM, y el valor se igualará a X, es decir:

```
asserta(dbase(startup,'1')),
X = '1'.
```

si la expresión (7) no se cumple, se hace un corte para que regrese a ejecutar otra vez la rutina principal, para limpiar la variable X de la base de conocimientos con la expresión:

```
not(dbase(system,_)).
```

Si el usuario tiene el problema de la opción

1) Problemas al inicializar.

la rutina de diagnostico(ac_power) checará la rutina de startup como sigue:

```
checar(startup,X) :-                                        ....(2)
      not(dbase(startup,_)),
      repeat,
      titulo,
      write('Que tipo de problema de inicio tiene ?'),nl,
      write('  1)Error del sistema durante el inicio.'),nl,
      write('  2)Cuando inicializa, no enciende el drive (LED)
                no visualiza nada en pantalla
      write('  3)Cuando inicializa, enciende el drive y no
                se visualiza nada en pantalla '),nl,
      write('  4)Trabaja bien pero no inicializa '),nl,
      write('SELECCIONE :'),
      respuesta(Reply),
      Char_int(Reply,Z),
      Z<53,Z>48,!,
      asserta(dbase(startup,Reply)),
      X=Reply.
```

Las etapas de respuestas son iguales y ya fueron explicadas anteriormente.

Ahora bién, si el usuario selecciona la opción 2 de la rutina (2) de startup, la base de conocimientos del Sistema Experto queda de la siguiente manera:

EXPERTO:	QUE TIPO BASICO DE PROBLEMA TIENE ?
USUARIO:	Problema al inicializar
EXPERTO:	QUE TIPO DE PROBLEMA DE INICIO TIENE ?
USUARIO:	Cuando inicializa, no enciende el drive (LED) no visualiza nada en pantalla

Una vez cargada la base de conocimientos del sistema experto, el sistema estará listo para utilizar el mecanismo de inferencia.

V.4.5. MANEJO Y LOGICA DEL MECANISMO DE INFERENCIA.

El predicado del mecanismo de inferencia que interactua con el usuario para encontrar el objeto objetivo es diagnostico. Recuerdese la rutina (1):

```
\* comienza el sistema de diagnostico *\

diagnostico(ac_power) :-                                    ....(1)
      checar(system,'1'),                                   ....(1)
      checar(startup,'2'),                                  ....(2)
      not(check(power)),                                    ....(3)
      causa(ac_power).                                      ....(4)
```

El sistema experto maneja una serie de hipótesis basandose en su base de conocimientos, para esto se utiliza a **check** como una 'pre-conclusión'. Es decir da una conclusión incierta y a la vez pregunta si persiste el problema, en caso afirmativo, se utiliza a **causa** como la conclusión final.

La expresión (;) indica que el usuario haga un chequeo visual de los componentes físicos -en este caso los cables-. He aquí la rutina del chequeo:

```
check(power) :-                                          ...([])
        titulo,
        write('Checar el cable principal, puede estar mal
                colocado'),nl,
        write('Esta bien colocado el cable principal (y\n) ?),
        respuesta(Reply),
        asserta(dbase(power,Reply)),Reply='y'.
```

El sistema dió una conclusión 'pobre' pero pregunta si el problema persiste, y dicha respuesta es almacenada en la variable **Reply**. La instrucción siguiente **asserta** añade un hecho a la base de datos de la rutina **power**.

Si la respuesta es n de 'no' regresa a la rutina de **diagnostico**, la cual se encargará de dar una conclusión. Aquí utilizará una búsqueda, en las rutinas especiales de búsqueda.

V.4.6. LA TECNICA DE BUSQUEDA.

Las rutinas de búsqueda de soluciones del Turbo Prolog estan denominadas como:

```
check(X) :-
        dbase (X,'y'),!.

check(X) :-
        dbase (X,'n'),!,fail.
```

Cada problema a consultar tiene su rutina específica. Lo que hace el programa es localizar la rutina exacta.

Estos predicados ayudan a restringir la vuelta atrás del Turbo Prolog. Ademas el corte (!) utilizada en la expresion:

```
check(X) :-
        dbase (X,'y'),!.
```

se usa libremente para impedir la vuelta atrás a varios puntos. Como ya se explicó anteriormente, algunas veces la parte más dura de programar en Turbo Prolog es limitar la búsqueda exhaustiva.

En este caso, si la variable X fue contestada como n de 'no', comienza la búsqueda de la rutina que le dará una solución.

La manera de definir esta cláusula es que X,-y- solo devuelve el valor verdadero si X,-y- devuelve verdadero, y la única forma de que ocurra eso es que la cláusula tenga éxito. Esto solo puede ocurrir si todos los atributos que se almacenan en la base de datos temporal encajan con aquellos del objeto en consideración. X tiene el valor de power.

Entonces la cláusula esta definida por:

```
check(power) :-
        dbase(power,-n-),!,fail.
```

Al generar el corte (!), el programa regresa a la rutina (╓) de diagnóstico y manda a llamar usando la expresión (4), la causa. Como la respuesta fué negativa, entonces se cumple la expresión (5) not(check(power)). Entonces la causa es causa(ac_power). he aquí esta rutina:

```
causa(ac_power):-                                    ...(╓)
    write(-la causa es probablemente un bajo poder.-),nl,
    write(-Checar para estar seguro de que el cable esta-),nl,
    write(-bien conectado al sistema de la computadora.-),nl,
    write(-Asegurese de que exista corriente en la salida),nl,
    write(-para la impresora -).
```

Una vez llamada la rutina, despliega el contenido de esta en la pantalla. Entonces el dialogo queda de la siguiente manera:

EXPERTO:	QUE TIPO BASICO DE PROBLEMA TIENE ?
USUARIO:	Problema al inicializar
EXPERTO:	QUE TIPO DE PROBLEMA DE INICIO TIENE ?
USUARIO:	Cuando inicializa, no enciende la luz del drive (LED)
EXPERTO:	Checar el cable principal puede estar mal colocado. Esta bien colocado el cable principal (y\n)?
USUARIO:	n.
EXPERTO:	La causa probablemente sea un poder bajo. Checar para estar seguro de que el cable esta bien conectado al sistema de la computadora. Asegurese de que exista corriente en la salida para la impresora.

Ahora bién, si la respuesta en la rutina C[] de check(power) hubiese sido y de 'si', esto quiere decir textualmente que:

EXPERTO: Checar el cable principal puede estar mal colocado.
Esta bien colocado el cable principal (y\n)?
USUARIO: y.

el sistema experto advierte que el problema persiste y es entonces que necesita de una búsqueda para la localización del objetc objetivo. De nuevo regresa a las rutinas de búsqueda y selecciona la rutina de respuesta 'y'.

```
check(X) :-
      dbase (X,-y-),!.
```

esto quiere decir que:

```
check(power ) :-
      dbase (power,-y-),!.
```

Como la expresión anterior se cumple para -y-, entonces simplemente hace un corte y realiza lo siguiente:

```
checar(Z,Y) :-
    dbase(Z,Y)X-Y,!.
```
regresa a la rutina y compara las variables de la base de datos.

```
checar(system,X) :-
  not(dbase(system,_)),
```
compara las variables con el de la rutina principal C[]. la 'variable blanca' tiene el valor de '1'.

```
checar(startup,X) :-
  not(dbase(startup,_)
```
compara las variables con el de la rutina de la rutina C[] la 'variable blanca' tiene el valor de '2'.

```
Check(X) :-
      dbase(X,-y-),!.
```
Como lo anterior se cumple genera la vuelta atrás, y busca una rutina que cumpla con los valores de '1' y '2'.

```
diagnóstico(ac_power ) :-
      checar(system,-1-),
      checar(startup,-2-),
      not(check(power)), <------
      causa(ac_power).
```
La búsqueda encuentra a C[], ya que los valores cumplen. pero la expresión dice que 'chequeo del poder -no-' entonces la causa es la fuente de poder. Esto quiere decir que no se cumple, ya que debe de ser acertado el chequeo. La búsqueda debe de continuar.

```
diagnostico(switch) :-
     checar(system,'1'),
     checar(startup,'2'),
     check(power),          <------
     causa(switch).
```

La búsqueda logra encontrar los mismos valores de '1' y '2' en esta rutina como se cumple esta expresión con -y- entonces la causa es el switch. Es ahí donde termina la búsqueda.

V.4.7. CONCLUSION DE LA BUSQUEDA DEL OBJETO OBJETIVO.

Una vez efectuada la búsqueda del objeto, la variable X llega a la rutina adecuada. En este caso la búsqueda es **causa(switch)**.

```
diagnóstico(switch) :-                    ...(5)
     checar(system,'1'),
     checar(startup,'2'),
     check(power),                        ...(10)
     causa(switch).
```

La rutina (5) tiene las mismas expresiones que la rutina (4), a excepción de **check(power)**. La expresión (9) funciona cuando la respuesta de la rutina (4) fué no, mientras que (10) corresponde a la respuesta si. Entonces la rutina (5) indica al programa ir a la última rutina del sistema denominada **causa(switch)**. Esta rutina es como sigue:

```
causa(switch):-
    write('La causa probablemente es el switch de ON\OFF'),nl,
    write('Checar el sistema interno del switch de ON\OFF '),nl,
    write('de no ser asi, cambie el switch.'),nl,
```

Como se puede observar, **causa** es la conclusión final que dá el sistema experto. Para esto se despliega una serie de causas posibles por las que la fuente de poder no funcionara correctamente.

Pues bién, el sistema experto funcionó en el exterior de la siguiente manera:

EXPERTO:	QUE TIPO BASICO DE PROBLEMA TIENE ?
USUARIO:	Problema al inicializar
EXPERTO:	QUE TIPO DE PROBLEMA DE INICIO TIENE ?
USUARIO:	Cuando inicializa, no enciende la luz del drive (LED)
EXPERTO:	Checar el cable principal puede estar mal colocado. Esta bién colocado el cable principal (y\n)?
USUARIO:	s.
EXPERTO:	La causa probablemente es el switch de ON\OFF Checar el sistema interno del switch de On\OFF de no ser asi, cambie el switch.

En conclusión se observa que el sistema experto contiene soluciones múltiples, ya que el mecanismo de inferencia permite identificar un cierto tipo de problema, pero cuando este problema no es buena solución, es cuando se procede a dar una serie de posibles soluciones.

En conclusión, el programa esta compuesto por:

checar	Pregunta al usuario cual es el tipo básico de el problema.
check	Hace una hipótesis de los que posiblemente pueda ser la causa del problema. Si dicho problema persiste. Se prepara para dar una solución mas exacta. Utilizando la técnica de búsqueda.
diagnóstico	Hace un diagnóstico según checar. Esta es la parte principal del mecanismo de inferencia.
diag	A igual que diagnóstico checa. La diferencia radica en que diag maneja mas aridad.
causa	Es la parte final donde se dan las conclusiones de cierta falla.

Una vez explicado las partes fundamentales del programa y como funcionan cada elemento de estas, se procede a listar el programa uniendo los componentes ya mencionados en las secciones adecuadas.

El siguiente listado no es el programa final, ya que solo se mostrará la manera de como está estructurada con respecto a un solo problema: el de fallas en la fuente de poder.

En el capítulo siguiente se mostrará el programa completo en la que se detectará todas las posibles causas que existan tanto internamente como externamente; es decir, el programa será capáz de detectar problemas de software y de hardware de la computadora, basandose en las cuestiones que se hicieron durante el *capítulo cuatro*.

```
/* SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC
      Y COMPATIBLES "\

      code=3072
      database
      dbase (symbol,char)

      include "a:menu.pro"

      predicates
      run
      titulo
      diagnóstico(symbol)
      diag(symbol)
      checar(symbol,char)
      check(symbol)
      clear_facts
      respuesta(char)
      repeat
      run_once
      causa(symbol)

\* D E F I N I C I O N   D E   L A   M E T A. *\

      goal
      makewindow(2,7,7,"",0,0,25,80),
      run.

clauses

\* R U T I N A   M A E S T R A   D E L   P R O G R A M A. *\

  run :-
      clearwindow
      run_once,
      clear_facts,
      write("\n Desea otra consulta (y-n)?"),
      readchar(Reply),
      write(Reply),nl,
      Reply="y"
      run.

 run_once :-
      titulo,nl,diagnóstico(_),!,clear_facts.
```

```
\* RUTINA QUE SE ACTIVA CUANDO NO
   ENCUENTRA EL OBJETO OBJETIVO. *\

 run_once :-
      write(-\n Lo que desea determinar no se -),nl,
      write(-encuentra en mi base de conocimientos \n-),clear_facts.

 titulo :-
      clearwindow,
      write(-    IBM_PC\XT SISTEMA EXPERTO PARA DIAGNOSTICO DE
FALLAS DE EQUIPO -),nl,nl.

      repeat.
      repeat :- repeat.

\* LIMPIA LAS VARIABLES DE LA BASE
     DE DATOS *\

   clear_facts :-
      retract(dbase(_,_)),fail.

   clear_facts :- nl.

\* DIAGNOSTICO INICIAL *\

      diagnóstico(ac_power) :-
            checar(system,-1-),
            checar(startup,-2-),
            not(check(power)),
            causa(ac_power).

      diagnóstico(ac_power) :-
            checar(system,-1-),
            checar(startup,-2-),
            check(power),
            causa(switch).
\*
 MANEJO DEL MECANISMO DE INFERENCIA
              (VUELTA ATRAS)
*\

      check(X) :-
            dbase (X,-y-),!.

      check(X) :-
            dbase (X,-n-),!,fail.
```

```
\* PRESOLUCION DEL MECANISMO DE INFERENCIA *\

check(power) :-
        titulo,
        write(-Checar el cable principal, puede estar mal
                colocado-),nl,
        write(-Esta bién colocado el cable principal (y\n) ?),
        respuesta(Reply),
        asserta(dbase(power,Reply)),Reply=-y-.

\* MENU PRINCIPAL DEL PROGRAMA *\

checar(Z,X) :-
        dbase(Z,Y),X=Y,!.

checar(system,X) :-
        not(dbase(system,_)),
        repeat,
        titulo,
        write(- QUE TIPO BASICO DE PROBLEMA TIENE ? -),nl,
        write(-  1)Problema al inicializar -),nl,
        write(-  2)Problema de corrida -),nl,
        write(-  3)Problemas de pantalla -),nl,
        write(-  4)Problemas de teclado -),nl,
        write(-  5)Problemas de impresora -),nl,
        write(- SELECCIONE :-),
        respuesta(Reply),
        char_int(Reply,Z),
        Z<54,Z>48,!,
        asserta(dbase(system,Reply)),
        X=Reply.

checar(startup,X) :-
        not(dbase(startup,_)),
        repeat,
        titulo,
        write(- QUE TIPO DE PROBLEMA DE INICIO TIENE ? -),nl,
        write(-  1)Error del sistema durante el inicio -),nl,
        write(-  2)Cuando inicializa no enciende el drive-),nl,
        write(-     (LED), no visualiza nada en pantalla-),nl,
        write(-  3)Cuando inicializa enciende el drive y no-),nl,
        write(-     se visualiza nada en pantalla-),nl,
        write(-  4)Trabaja bién pero no inicializa -),nl,
        write(- SELECCIONE :-),
        respuesta(Reply),
        char_int(Reply,Z),
        Z<53,Z>48,!,
        asserta(dbase(system,Reply)),
        X=Reply.
```

```
\*
RUTINA PARA EL MANEJO DE RESPUESTAS
*\

    respuesta(Reply) :-
        readchar(Reply),
        write(Reply),nl.

\* RUTINAS DE CONCLUSIONES *\

    causa(ac_power):-
        write('la causa es probablemente un bajo poder.'),nl,
        write('Checar para estar seguro de que el cable esta'),nl,
        write('bién conectado al sistema de la computadora.'),nl,
        write('Asegurese de que exista corriente en la salida'),nl,
        write('para la impresora ').

    causa(switch):-
        write('la causa probablemente es el switch de ON\OFF '),nl,
        write('Checar el sistema interno del switch de ON\OFF '),nl,
        write('de no ser así cambie el switch.'),nl,
```

V.5. APLICACIONES.

La utilidad de un Sistema Experto se basa principalmente en la eficacia y conveniencia. A diferencia de un experto humano que tiene que comer, descansar, tomarse vacaciones y otras cosas, el Sistema Experto está accesible para usarse las veinticuatro horas de día, todos los días del año. Ademas, se puede crear muchos Sistemas Expertos, mientras que el número de expertos humanos puede ser limitado [2], lo cual hace virtualmente imposible en muchas situaciones tener un experto disponible cuando se necesita.

El conocimiento de un sistema experto puede copiarse y almacenarse fácilmente siendo excepcional la pérdida permanente del conocimiento del experto.

El Sistema Experto desarrollado en este capítulo comienza en el principio en la base de conocimientos y trabaja a traves de ella secuencialmente. Sin heurísticas, es lo mejor que se puede esperar que haga un mecanismo de inferencia. Sin embargo se puede controlar la organización de la información, lo que implica que puede haber ordenaciones mejores y peores.

El programa servirá de gran utilidad, para el experto humano, ya que muchas veces, el conoce solamente Cy a su modo) las fallas más generales. Tal vez el experto humano tenga una especialidad en cierta área. El programa de diagnóstico, abarca las dos áreas de software y hardware.

Su base de conocimientos está basado en los cuestionarios realizados en el capítulo cuatro. El autor *Hebert S.* '*programación Avanzada*' *1988 (pag. 89)* dice en uno de sus parrafos:

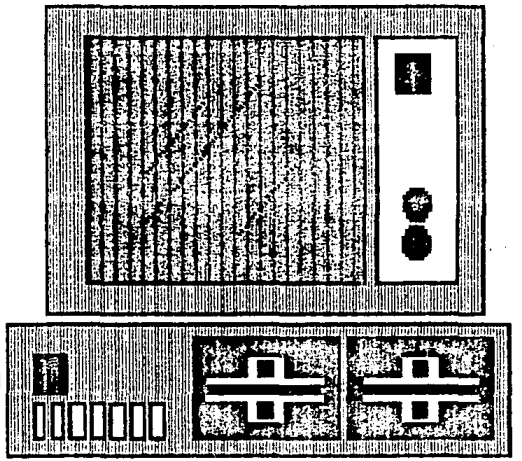
[Otro punto problemático es que la mayoría de los expertos humanos no saben lo que saben; los humanos no pueden hacer un volcado de memoria de la misma forma que una computadora. En consecuncia puede ser difícil extraer toda la información necesaria. Ademas algunos expertos estarán simplemente menos inclinados a perder tiempo todo lo que saben sobre la materia.]

En otro enfoque, tiene la autocomprobación del sistema por consistencia y ver que toda la información de la base de conocimiento concuerda consigo misma.

Aunque este programa no encontrará todos los problemas (como el virus), encontrará algunos. Sin embargo dependiendo de cómo esté implementada la autocomprobación, en algunas bases de datos grandes, se puede alcanzar a ser un programa perfecto, haciendo de este enfoque, un sistema perfecto.

De este modo, según crece el uso de los expertos, la verificación de la base de conocimientos será una de las más importantes en el área de la investigación informática.

P A R T E III



LISTADO DEL SISTEMA EXPERTO

VI. LISTADO DEL PROGRAMA.

Ahora entramos al listado del sistema experto. Como se podrá observar (en las siguientes hojas), el listado tiene como características lo siguiente:

- El sistema está contenido en un solo programa, es decir no contiene subrutinas.
- Es de 1630 líneas.(llena el límite del editor de Turbo Prolog 1900).
- Maneja ventanas para el diagnóstico y la conclusión.
- Se necesitan 640 K (ó más) para su ejecución.
- Puede editarse en cualquier editor de Borland® ó ASCCI.
- Se necesita el Turbo Prolog para su ejecución (en memoria y para compilarse a archivo ejecutable.
- El listado es de 29 páginas.

A continuación se muestra el listado con las características mencionadas.

```
/*    SISTEMA EXPERTO PARA DIAGNOSTICO FALLAS EN EQUIPO

          PC Y COMPATIBLES
*/

code=3000
database
dbase(symbol,char)

/*include "menu.pro"*/

predicates
run
titulo
diagnostico(symbol)
diag(symbol)
checar(symbol,char)
check(symbol)
clear_facts
respuesta(char)
repeat
run_once
causa(symbol)

goal
makewindow(9,14,112,"DIAG_PC",0,0,25,80),
run.

clauses

run :-
  clearwindow,
  run_once,
  clear_facts,
  makewindow(1,111,113,"opci"n",20,16,5,d4),
  write("\n\tDesea otra consulta (s/n)? "),
  readchar(Reply),
  write(Reply),nl,
  Reply='s',
  removewindow,
  removewindow,
  run.

run_once :-
  titulo,nl,diagnostico(_),!,clear_facts.

run_once :-
  makewindow(1,116,112,"mensaje",5,02,9,74),
  write("\n\t\tLo que desea determinar no se encuentra"),nl,
  write("\n\t\ten mi base de conocimientos. \n"),clear_facts.

titulo :-
  clearwindow,
  write("\tIBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO"),nl
  write("\t                              PC Y COMPATIBLE                    "),nl,nl

repeat.
repeat :- repeat.

clear_facts:-
```

```
  retract(dbase(_,_)),fail.

clear_facts :- nl.

/* COMIENZA DIAGNOSTICO */

  diagnostico(ac_power):-
    checar(sistema,'1'),
    checar(inicio,'2'),
    not(check(power)),
    causa(ac_power).
  diagnostico(switch):-
    checar(sistema,'1'),
    checar(inicio,'2'),
    check(power),
    causa(switch).
  diagnostico(no_poder) :-
    checar(sistema,'1'),
    checar(inicio,'3'),
    causa(no_poder).
  diagnostico(fuente_poder) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    checar(beep,'1'),
    causa(fuente_poder).
  diagnostico(fuente_poder) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    checar(beep,'3'),
    causa(fuente_poder).
  diagnostico(sistema_tarjeta) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    checar(beep,'4'),
    causa(sistema_tarjeta).
  diagnostico(despliegue) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    checar(beep,'5'),
    causa(despliegue).
  diagnostico(despliegue) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    checar(beep,'6'),
    causa(despliegue).
  diagnostico(fuente_poder) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    checar(beep,'7'),
    causa(fuente_poder).
  diagnostico(disk_drive) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    checar(beep,'8'),
    causa(disk_drive).
  diagnostico(dummy) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    checar(beep,'2'),
    check(un_beep),
```

```
    diag(floppy_disk_boot).
  diagnostico(memoria) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    not(check(memoria)),
    causa(memoria).
  diagnostico(sistema_tarjeta) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    check(memoria),
    checar(errorcodigo,'b'),
    causa(sistema_tarjeta2).
  diagnostico(memoria) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    check(memoria),
    checar(errorcodigo,'c'),
    causa(memoria).
  diagnostico(teclado) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    check(memoria),
    checar(errorcodigo,'d'),
    causa(teclado).
  diagnostico(floppy) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    check(memoria),
    checar(errorcodigo,'e'),
    causa(disk_drive).
  diagnostico(mono_monitor) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    check(memoria),
    checar(errorcodigo,'f'),
    causa(mono_monitor).
  diagnostico(color_monitor) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    check(memoria),
    checar(errorcodigo,'g'),
    causa(color_monitor).
  diagnostico(impresor) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    check(memoria),
    checar(errorcodigo,'h'),
    causa(impresor).
  diagnostico(adaptador_juegos) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    check(memoria),
    checar(errorcodigo,'i'),
    causa(adaptador_juegos).
  diagnostico(impresor) :-
    checar(sistema,'1'),
    checar(inicio,'1'),
    check(memoria),
    checar(errorcodigo,'j'),
    causa(fuente_poder).
```

```
diagnostico(disco_duro) :-
   checar(sistema,'1'),
   checar(inicio,'1'),
   check(memoria),
   checar(errorcodigo,'k'),
   causa(disco_duro).
diagnostico(expansion_unit) :-
   checar(sistema,'1'),
   checar(inicio,'1'),
   check(memoria),
   checar(errorcodigo,'l'),
   causa(expansion_unit).
diagnostico(fuente_poder) :-
   checar(sistema,'1'),
   checar(inicio,'1'),
   check(memoria),
   checar(errorcodigo,'m'),
   causa(fuente_poder).
diagnostico(disco_duro) :-
   checar(sistema,'1'),
   checar(inicio,'d'),
   check(disco_duro),
   check(floppy_disk_boot),
   check(dos),
   not(check(duro)),
   causa(disco_duro).
diagnostico(dos) :-
   checar(sistema,'1'),
   checar(inicio,'d'),
   check(disco_duro),
   check(floppy_disk_boot),
   not(check(dos)),
   causa(dos).
diagnostico(disco_duro) :-
   checar(sistema,'1'),
   checar(inicio,'d'),
   check(disco_duro),
   check(floppy_disk_boot),
   check(dos),
   check(duro),
   not(check(boot_lee_duro)),
   causa(boot_lee_duro).
diagnostico(disco_duro) :-
   checar(sistema,'1'),
   checar(inicio,'d'),
   check(disco_duro),
   check(floppy_disk_boot),
   check(dos),
   check(duro),
   check(boot_lee_duro),
   check(boot_escritura_duro),
   causa(mal_dos).
diagnostico(disco_duro) :-
   checar(sistema,'1'),
   checar(inicio,'d'),
   check(disco_duro),
   check(floppy_disk_boot),
   check(dos),
   check(duro),
   check(boot_lee_duro),
```

```
    not(check(boot_escritura_duro)),
    check(autoexec),
    causa(autoexec).
  diagnostico(disco_duro) :-
    checar(sistema,'1'),
    checar(inicio,'4'),
    check(disco_duro),
    check(floppy_disk_boot),
    check(dos),
    check(duro),
    check(boot_lee_duro),
    not(check(boot_escritura_duro)),
    not(check(autoexec)),
    check(config),
    causa(config).
  diagnostico(disco_duro) :-
    checar(sistema,'1'),
    checar(inicio,'4'),
    check(disco_duro),
    check(floppy_disk_boot),
    check(dos),
    check(duro),
    check(boot_lee_duro),
    not(check(boot_escritura_duro)),
    not(check(autoexec)),
    not(check(config)),
    causa(desconocido).
  diagnostico(floppy_disk) :-
    checar(sistema,'1'),
    checar(inicio,'4'),
    not(check(disco_duro)),
    diag(floppy_disk_boot).
  diagnostico(floppy_disk) :-
    checar(sistema,'1'),
    checar(inicio,'4'),
    check(disco_duro),
    not(check(floppy_disk_boot)),
    diag(floppy_disk_boot).
  diagnostico(configura) :-
    checar(sistema,'2'),
    not(check(configura)),
    causa(configura).
  diagnostico(ocupado) :-
    checar(sistema,'2'),
    checar(run,'1'),
    check(configura),
    check(ocupado),
    causa(ocupado).
  diagnostico(clavija) :-
    checar(sistema,'2'),
    checar(run,'1'),
    check(configura),
    not(check(ocupado)),
    not(check(clavija)),
    causa(clavija).
  diagnostico(software) :-
    checar(sistema,'2'),
    checar(run,'1'),
    check(configura),
    not(check(ocupado)),
```

```
    check(clavija),
    not(check(reboot)),
    causa(hardware).
diagnostico(software) :-
    checar(sistema,'2'),
    checar(run,'1'),
    check(configura),
    not(check(ocupado)),
    check(clavija),
    check(reboot),
    check(repeat),
    causa(software).
diagnostico(statica) :-
    checar(sistema,'2'),
    checar(run,'1'),
    check(configura),
    not(check(ocupado)),
    check(clavija),
    check(reboot),
    not(check(repeat)),
    causa(statica).
diagnostico(caer_power) :-
    checar(sistema,'2'),
    checar(run,'2'),
    check(configura),
    causa(caer_power).
diagnostico(parity_error) :-
    checar(sistema,'2'),
    checar(run,'3'),
    check(configura),
    causa(parity_error).
diagnostico(caesistema) :-
    checar(sistema,'2'),
    checar(run,'4'),
    check(configura),
    causa(caesistema).
diagnostico(software) :-
    checar(sistema,'2'),
    checar(run,'5'),
    check(configura),
    check(consistent),
    causa(erratico_software).
diagnostico(hardware) :-
    checar(sistema,'2'),
    checar(run,'5'),
    check(configura),
    not(check(consistent)),
    causa(caer_power).
diagnostico(programa) :-
    checar(sistema,'2'),
    checar(run,'6'),
    check(configura),
    check(run_programa),
    causa(run_programa_dos).
diagnostico(programa) :-
    checar(sistema,'2'),
    checar(run,'6'),
    check(configura),
    not(check(run_programa)),
    causa(run_programa_aplicacion).
```

```
diagnostico(disco_duro) :-
   checar(sistema,'2'),
   checar(run,'7'),
   checar(run_disco_duro,'1'),
   check(expansion),
   causa(expansion).
diagnostico(disco_duro) :-
   checar(sistema,'2'),
   checar(run,'7'),
   checar(run_disco_duro,'1'),
   not(check(expansion)),
   causa(run_disco_duro).
diagnostico(disco_duro) :-
   checar(sistema,'2'),
   checar(run,'7'),
   checar(run_disco_duro,'1'),
   check(expansion),
   causa(run_expansion).
diagnostico(disco_duro) :-
   checar(sistema,'2'),
   checar(run,'7'),
   checar(run_disco_duro,'2'),
   check(run_disk_software),
   causa(run_disk_software).
diagnostico(disco_duro) :-
   checar(sistema,'2'),
   checar(run,'7'),
   checar(run_disco_duro,'2'),
   not(check(run_disk_software)),
   not(check(run_resident)),
   causa(run_resident).
diagnostico(disco_duro) :-
   checar(sistema,'2'),
   checar(run,'7'),
   checar(run_disco_duro,'2'),
   not(check(run_disk_software)),
   check(run_resident),
   causa(run_durodisk_escritura).
diagnostico(disco_duro) :-
   checar(sistema,'2'),
   checar(run,'7'),
   checar(run_disco_duro,'3'),
   check(run_disk_software),
   causa(run_disk_software).
diagnostico(disco_duro) :-
   checar(sistema,'2'),
   checar(run,'7'),
   checar(run_disco_duro,'3'),
   not(check(run_disk_software)),
   not(check(run_resident)),
   causa(run_resident).
diagnostico(disco_duro) :-
   checar(sistema,'2'),
   checar(run,'7'),
   checar(run_disco_duro,'3'),
   not(check(run_disk_software)),
   check(run_resident),
   causa(run_durodisk_escritura).
diagnostico(disco_duro) :-
```

```
    checar(sistema,'2'),
    checar(run,'7'),
    checar(run_disco_duro,'4'),
    causa(run_durodisk_chkdsk).
  diagnostico(disco_duro) :-
    checar(sistema,'2'),
    checar(run,'7'),
    checar(run_disco_duro,'5'),
    causa(run_durodisk_ruido).
  diagnostico(disco_duro) :-
    checar(sistema,'2'),
    checar(run,'7'),
    checar(run_disco_duro,'6'),
    check(fisico_movimiento),
    causa(fisico_movimiento).
  diagnostico(disco_duro) :-
    checar(sistema,'2'),
    checar(run,'7'),
    checar(run_disco_duro,'6'),
    not(check(fisico_movimiento)),
    causa(run_duro).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_disco_duro,'1'),
    causa(run_floppy_power).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_disco_duro,'2'),
    causa(run_floppy_no_lee).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_disco_duro,'3'),
    not(check(run_floppy_fisico)),
    causa(run_floppy_fisico).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'3'),
    check(run_floppy_fisico),
    not(check(run_resident)),
    causa(run_resident).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'3'),
    check(run_floppy_fisico),
    check(run_resident),
    check(run_dosdiscos),
    check(second_disk),
    causa(mal_floppy).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'3'),
    check(run_floppy_fisico),
    check(run_resident),
    check(run_dosdiscos),
```

```
    not(check(second_disk)),
    check(drive),
    causa(sucio).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'3'),
    check(run_floppy_fisico),
    check(run_resident),
    check(run_dosdiscos),
    not(check(second_disk)),
    not(check(drive)),
    causa(mal_floppy_drive).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'3'),
    check(run_floppy_fisico),
    check(run_resident),
    not(check(run_dosdiscos)),
    check(drive),
    causa(sucio).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'3'),
    check(run_floppy_fisico),
    check(run_resident),
    not(check(run_dosdiscos)),
    not(check(drive)),
    check(run_disk_software),
    causa(run_disk_software).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'3'),
    check(run_floppy_fisico),
    check(run_resident),
    not(check(run_dosdiscos)),
    not(check(drive)),
    not(check(run_disk_software)),
    check(run_un_floppy),
    causa(mal_floppy2).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'3'),
    check(run_floppy_fisico),
    check(run_resident),
    not(check(run_dosdiscos)),
    not(check(drive)),
    not(check(run_disk_software)),
    not(check(run_un_floppy)),
    causa(mal_floppy_drive).
  diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'4'),
    not(check(run_floppy_fisico)),
    causa(run_floppy_fisico).
```

```
diagnostico(floppy_disk) :-
  checar(sistema,'2'),
  checar(run,'8'),
  checar(run_floppy_disk,'d'),
  check(run_floppy_fisico),
  not(check(escritura_protecc)),
  causa(escritura_protecc).
diagnostico(floppy_disk) :-
  checar(sistema,'2'),
  checar(run,'8'),
  checar(run_floppy_disk,'d'),
  check(run_floppy_fisico),
  check(escritura_protecc),
  not(check(run_resident)),
  causa(run_resident).
diagnostico(floppy_disk) :-
  checar(sistema,'2'),
  checar(run,'8'),
  checar(run_floppy_disk,'d'),
  check(run_floppy_fisico),
  check(escritura_protecc),
  check(run_resident),
  check(run_dosdiscos),
  causa(mal_floppy).
diagnostico(floppy_disk) :-
  checar(sistema,'2'),
  checar(run,'8'),
  checar(run_floppy_disk,'d'),
  check(run_floppy_fisico),
  check(escritura_protecc),
  check(run_resident),
  check(run_dosdiscos),
  not(check(second_disk)),
  check(drive),
  causa(sucio).
diagnostico(floppy_disk) :-
  checar(sistema,'2'),
  checar(run,'8'),
  checar(run_floppy_disk,'d'),
  check(run_floppy_fisico),
  check(escritura_protecc),
  check(run_resident),
  check(run_dosdiscos),
  not(check(second_disk)),
  not(check(drive)),
  causa(mal_floppy_drive).
diagnostico(floppy_disk) :-
  checar(sistema,'2'),
  checar(run,'8'),
  checar(run_floppy_disk,'d'),
  check(run_floppy_fisico),
  check(escritura_protecc),
  check(run_resident),
  not(check(run_dosdiscos)),
  check(drive),
  causa(sucio).
diagnostico(floppy_disk) :-
  checar(sistema,'2'),
  checar(run,'8'),
  checar(run_floppy_disk,'d'),
```

```
    check(run_floppy_fisico),
    check(escritura_protecc),
    check(run_resident),
    not(check(run_dosdiscos)),
    not(check(drive)),
    check(run_disk_software),
    causa(run_disk_software).
diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'4'),
    check(run_floppy_fisico),
    check(escritura_protecc),
    check(run_resident),
    not(check(run_dosdiscos)),
    not(check(drive)),
    not(check(run_disk_software)),
    check(run_un_floppy),
    causa(mal_floppy2).
diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'4'),
    check(run_floppy_fisico),
    check(escritura_protecc),
    check(run_resident),
    not(check(run_dosdiscos)),
    not(check(drive)),
    not(check(run_disk_software)),
    not(check(run_un_floppy)),
    causa(mal_floppy_drive).
diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'5'),
    not(check(run_floppy_fisico)),
    causa(run_floppy_fisico).
diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'5'),
    check(run_floppy_fisico),
    causa(floppy_ruido).
diagnostico(floppy_disk) :-
    checar(sistema,'2'),
    checar(run,'8'),
    checar(run_floppy_disk,'6'),
    causa(floppy_cargar).
diagnostico(sistema_recalentamiento) :-
    checar(sistema,'2'),
    checar(run,'9'),
    causa(run_recalentamiento).

diagnostico(no_despliegue) :-
    checar(sistema,'3'),
    checar(despliegue,'1'),
    causa(no_despliegue).
diagnostico(despliegue_desvanece) :-
    checar(sistema,'3'),
    checar(despliegue,'2'),
```

```
    causa(despliegue_desvanece).
diagnostico(despliegue_recalentamiento) :-
    checar(sistema,'3'),
    checar(despliegue,'3'),
    causa(despliegue_recalentamiento).
diagnostico(despliegue_vertical) :-
    checar(sistema,'3'),
    checar(despliegue,'4'),
    causa(despliegue_vertical).
diagnostico(despliegue_horizontal) :-
    checar(sistema,'3'),
    checar(despliegue,'5'),
    causa(despliegue_horizontal).
diagnostico(despliegue_basura) :-
    checar(sistema,'3'),
    checar(despliegue,'6'),
    causa(despliegue_basura).
diagnostico(despliegue_color) :-
    checar(sistema,'3'),
    checar(despliegue,'7'),
    causa(despliegue_color).
diagnostico(teclado_muerto) :-
    checar(sistema,'4'),
    checar(teclado,'1'),
    causa(teclado_muerto).
diagnostico(teclado_basura) :-
    checar(sistema,'4'),
    checar(teclado,'2'),
    causa(teclado_basura).
diagnostico(teclado_tecla) :-
    checar(sistema,'4'),
    checar(teclado,'3'),
    causa(teclado_tecla).
diagnostico(teclado_doble) :-
    checar(sistema,'4'),
    checar(teclado,'4'),
    causa(teclado_doble).
diagnostico(teclado_ajeno) :-
    checar(sistema,'4'),
    checar(teclado,'5'),
    causa(teclado_ajeno).
diagnostico(teclado_derrame) :-
    checar(sistema,'4'),
    checar(teclado,'6'),
    causa(teclado_derrame).
diagnostico(impresor_muerto) :-
    checar(sistema,'5'),
    checar(impresor,'a'),
    causa(impresor_muerto).
diagnostico(impresor_notest) :-
    checar(sistema,'5'),
    checar(impresor,'b'),
    causa(impresor_notest).
diagnostico(impresor_noline) :-
    checar(sistema,'5'),
    checar(impresor,'c'),
    causa(impresor_noline).
diagnostico(impresor_viejo) :-
    checar(sistema,'5'),
    checar(impresor,'d'),
```

```
    not(check(impresor_programa)),
    causa(impresor_viejo).
  diagnostico(impresor_viejo) :-
    checar(sistema,'5'),
    checar(impresor,'d'),
    check(impresor_programa),
    causa(impresor_programa).
  diagnostico(impresor_erratico) :-
    checar(sistema,'5'),
    checar(impresor,'e'),
    causa(impresor_erratico).
  diagnostico(impresor_stops) :-
    checar(sistema,'5'),
    checar(impresor,'f'),
    causa(impresor_stops).
  diagnostico(impresor_carro) :-
    checar(sistema,'5'),
    checar(impresor,'g'),
    causa(impresor_carro).
  diagnostico(impresor_noprint) :-
    checar(sistema,'5'),
    checar(impresor,'h'),
    causa(impresor_noprint).
  diagnostico(impresor_recalentar) :-
    checar(sistema,'5'),
    checar(impresor,'i'),
    causa(impresor_recalentar).
  diagnostico(impresor_desigual) :-
    checar(sistema,'5'),
    checar(impresor,'j'),
    causa(impresor_desigual).
  diagnostico(impresor_perdido) :-
    checar(sistema,'5'),
    checar(impresor,'k'),
    causa(impresor_perdido).
  diagnostico(impresor_bloqueo) :-
    checar(sistema,'5'),
    checar(impresor,'l'),
    causa(impresor_bloqueo).

/* REDONDEA EL DIAGNOSTICO */

  diag(floppy_disk_boot) :-
    checar(floppy_boot,'1'),
    causa(drive_power).
  diag(floppy_disk_boot) :-
    checar(floppy_boot,'2'),
    check(fisico_disk),nl,
    causa(fisico_disk).
  diag(floppy_disk_boot) :-
    checar(floppy_boot,'2'),
    not(check(fisico_disk)),nl,
    check(drive),
    causa(sucio).
  diag(floppy_disk_boot) :-
    checar(floppy_boot,'2'),
    not(check(fisico_disk)),nl,
    not(check(drive)),
    check(dosdiscos),
    causa(dosdiscos).
```

```
  diag(floppy_disk_boot) :-
    checar(floppy_boot,'2'),
    not(check(fisico_disk)),nl,
    not(check(drive)),
    not(check(dosdiscos)),
    causa(floppy_drive).
  diag(floppy_disk_boot) :-
    checar(floppy_boot,'3'),
    causa(format).
  diag(floppy_disk_boot) :-
    checar(floppy_boot,'4'),
    causa(puerta).
  diag(floppy_disk_boot) :-
    checar(floppy_boot,'5'),
    causa(command).

/* TECNICA DE BUSQUEDA SEGUN LAS RESPUESTAS */

    check(X) :-
      dbase(X,'s'),!.
    check(X) :-
      dbase(X,'n'),!,fail.

/* COMIENZA EL CHEQUEO */

    check(power) :-
      titulo,
      write("\n\tChecar el cable principal, puede estar mal colocado."),nl,
      write("\n\tEsta bin colocado el cable principal (s/n) ? "),
      respuesta(Reply),
      asserta(dbase(power,Reply)),Reply='s'.
    check(dosdiscos) :-
      titulo,
      write("\n\tSu proxima tarea, es para aislar el problema entre el"),nl,
      write("\n\tdrive del disco, controladores, y cable del disco."),nl,
      write("\n\tEsto es simplificado si tiene un segundo disco en la"),nl,
      write("\n\tunidad " un disco sobrante para usar."),nl,
      write("\n\tTiene uno disponible o 2 floppy-disk en la"),nl,
      write("\n\tunidad (s/n) ? "),
      respuesta(Reply),
      asserta(dbase(dosdiscos,Reply)),Reply='s'.
    check(memoria) :-
      titulo,
      write("\n\tTiene una prueba la memoria (s/n) ? "),
      respuesta(Reply),
      asserta(dbase(memoria,Reply)),Reply='s'.
    check(dos) :-
      titulo,
      write("\n\t tiene instalado la versi"n 2.0 en su"),nl,
      write("\n\tdisco duro (s/n) ? "),
      respuesta(Reply),
      asserta(dbase(dos,Reply)),Reply='s'.
    check(disco_duro) :-
      titulo,
      write("\n\test  usted tratando de reiniciar desde el disco duro (s/n)?"),
      respuesta(Reply),
      asserta(dbase(disco_duro,Reply)),Reply='s'.
    check(duro) :-
      titulo,
      write("\n\t El disco toca el panel. se siente el motor del drive "),nl,
```

```
    write("\n\tcuando gira y no enciende el LED del drive cuando busca "),nl,
    write("\n\talguna informaci"n en el disco (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(duro,Reply)),Reply='s'.
  check(drive) :-
    titulo,
    write("\n\tHaga un chequeo visual del drive. Asegurese que el disco"),nl,
    write("\n\tcarge apropiadamente y se active el motor del drive."),nl,
    write("\n\tLimpie la cabeza lectura. Use un limpiador especial. "),nl,
    write("\n\tFunciona bin ahora (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(drive,Reply)),Reply='s'.
  check(floppy_disk_boot) :-
    titulo,
    write("\n\tTrate de inicializar el sistema desde el disco flexible "),nl,
    write("\n\tasegurese que el disco este bueno con una copia del DOS."),nl,
    write("\n\tEs esto acertado (s/n) ? "),nl,
    respuesta(Reply),
    asserta(dbase(floppy_disk_boot,Reply)),Reply='s'.
  check(fisico_disk) :-
    titulo,
    write("\n\tSu siguiente tarea es de aislar el problema para ambos"),nl,
    write("\n\t(el disco f'sico o el hardware). Asegurese de que el "),nl,
    write("\n\tdisco fisicamente no este da$ado. Use otro disco en el "),nl,
    write("\n\tmismo drive. Asegurese de que el disco est formateado y"),nl,
    write("\n\tcontenga el DOS, pruebe el mismo disco-flexible en otra "),nl,
    write("\n\tcomputadora. El disco est  defectuoso (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(fisico_disk,Reply)),Reply='s'.
  check(boot_lee_duro) :-
    titulo,
    write("\n\tDespues de inicializar desde el disco flexible, trate de"),nl,
    write("\n\tleer el directorio en el disco duro y comienze un  "),nl,
    write("\n\tprograma desde el disco duro. "),nl,
    write("\n\t\tEs esto acertado (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(boot_lee_duro,Reply)),Reply='s'.
  check(boot_escritura_duro) :-
    titulo,
    write("\n\tTrate de restaurar el DOS para el disco duro desde el "),nl,
    write("\n\tdisco flexible colocando un disco DOS en la unidad A "),nl,
    write("\n\ty teclee siguiente: SYS C: . "),nl,
    write("\n\t\tEs esto acertado (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(boot_escritura_duro,Reply)),Reply='s'.
  check(autoexec) :-
    titulo,
    write("\n\tReinicialice desde el disco flexible, renombre el "),nl,
    write("\n\tarchivo AUTOEXEC.BAT en el disco duro con otro nombre "),nl,
    write("\n\ty trate de reinicializar desde el disco duro."),nl,
    write("\n\t\t Es esto acertado (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(autoexec,Reply)),Reply='s'.
  check(config) :-
    titulo,
    write("\n\tReinicialice desde el disco flexible, renombre el "),nl,
    write("\n\tarchivo CONFIG.SYS en el disco duro con otro nombre. "),nl,
    write("\n\tTrate de inicializar otra vez desde el disco duro."),nl,
    write("\n\t\tEs esto acertado (s/n) ? "),
    respuesta(Reply),
```

```
    asserta(dbase(config,Reply)),Reply='s'.
  check(reboot) :-
    titulo,
    write("\n\tTrate de tomar un reinicio acertado, use CTRL/ALT/DEL "),nl,
    write("\n\tprimero. Si esto no reinicializa apague la computadora y"),nl,
    write("\n\tpruebe otra vez, si esto no trabaja, apague la "),nl,
    write("\n\tcomputadora un rato (15 minutos), entonces reinicialice"),nl,
    write("\n\totra vez. Con el reinicio, es acertado eventualmente (s/n) "),
    respuesta(Reply),
    asserta(dbase(reboot,Reply)),Reply='s'.
  check(repeat) :-
    titulo,
    write("\n\tTrate de repetir exactamente todos los pasos con LOCK-UP"),nl,
    write("\n\texactamente y en el mismo orden."),nl,
    write("\n\t\tLa computadora sigue igual con LOCK-UP (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(repeat,Reply)),Reply='s'.
  check(clavija) :-
    titulo,
    write("\n\tAsegurese de que la clavija del teclado este bin "),nl,
    write("\n\tconectado en el sistema y que el cable este en bun "),nl,
    write("\n\testado. Estan los cables bin (s/n) "),
    respuesta(Reply),
    asserta(dbase(clavija,Reply)),Reply='s'.
  check(ocupado) :-
    titulo,
    write("\n\tAsegurese de que la computadora no est haciendo otra "),nl,
    write("\n\tcosa y el teclado este intensionalmente desconectado,"),nl,
    write("\n\tal menos que el trabajo este completamente terminado. "),nl,
    write("\n\tMire la luz del drive."),nl,
    write("\n\tEsta la computadora realmente ocupada -trabajando- (s/n) ?"),
    respuesta(Reply),
    asserta(dbase(ocupado,Reply)),Reply='s'.
  check(consistent) :-
    titulo,
    write("\n\tEs algo consistente acerca del problema, "),nl,
    write("\n\testo es, tiene el mismo problema en el mismo programa "),nl,
    write("\n\tcuando usted hace la misma cosa (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(consistent,Reply)),Reply='s'.
  check(configura) :-
    titulo,
    write("\n\tEs la temperatura ambiente de la computadora "),nl,
    write("\n\t<= 15 grados y >= 22 grados centigrados,"),nl,
    write("\n\ttodas las ventilaciones estan abiertas "),nl,
    write("\n\ty todas llegan a la computadora (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(configura,Reply)),Reply='s'.
  check(run_programa) :-
    titulo,
    write("\n\tEs este un programa DOS (s/n) ? "),nl,
    respuesta(Reply),
    asserta(dbase(run_programa,Reply)),Reply='s'.
  check(expansion) :-
    titulo,
    write("\n\tEst  el disco duro en una unidad de expansi"n (s/n) ?"),
    respuesta(Reply),
    asserta(dbase(expansion,Reply)),Reply='s'.
  check(fisico_movimiento) :-
    titulo,
```

```
    write("\n\tMueve usted fisicamente la computadora de su lugar (s/n) ?"),
    respuesta(Reply),
    asserta(dbase(fisico_movimiento,Reply)),Reply='s'.
  check(run_disk_software) :-
    titulo,
    write("\n\tHace esto solamente con un programa (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(run_disk_sotfware,Reply)),Reply='s'.
  check(run_resident) :-
    titulo,
    write("\n\tTrate de mover todos los programas residentes y comenzar"),nl,
    write("\n\tsin el AUTOEXEC.BAT (renombre este y comience). "),nl,
    write("\n\tPersiste el problema (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(run_resident,Reply)),Reply='s'.
  check(run_floppy_fisico) :-
    titulo,
    write("\n\tChecar el disco visualmente para verificar daño fisico"),nl,
    write("\n\tEl disco flexible esta bin (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(run_floppy_fisico,Reply)),Reply='s'.
  check(run_dosdiscos) :-
    titulo,
    write("\n\tTiene un segundo disco flexible en el drive o accesa "),nl,
    write("\n\tpara otra computadora compatible (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(run_dosdiscos,Reply)),Reply='s'.
  check(second_disk) :-
    titulo,
    write("\n\tHace el disco de trabajo el mismo trabajo con los "),nl,
    write("\n\tmismos resultados en otro drive (s/n) "),
    respuesta(Reply),
    asserta(dbase(second_disk,Reply)),Reply='s'.
  check(run_un_floppy) :-
    titulo,
    write("\n\tTiene el disco flexible el mismo problema en particular"),nl,
    write("\n\t" con disco flexible de una marca (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(run_un_floppy,Reply)),Reply='s'.
  check(escritura_protecc) :-
    titulo,
    write("\n\tEsta la ranura de protecci"n contra escritura "),nl,
    write("\n\tdescubierta (s/n) ? "),
    respuesta(Reply),
    assert(dbase(escritura_protecc,Reply)),Reply='s'.
  check(impresor_programa) :-
    titulo,
    write("\n\tTrate de dar DIR, liste en la impresora usando "),nl,
    write("\n\tDIR CTRL/P, y pulse <ENTER>."),nl,
    write("\n\t\t Esto imprime correctamente (s/n) ? "),
    respuesta(Reply),
    asserta(dbase(impresor_programa,Reply)),Reply='s'.
  check(un_beep) :-
    titulo,
    write("\n\tUn [Beep] corto es un sonido normal despues de que el "),nl,
    write("\n\tsistema es inicializado desde el disco flexible. "),nl,
    write("\n\tMarca el problema al iniciar (s/n) ? "),
    respuesta(Reply),Reply='s'.

/* SELECCIONA EL TIPO DE MENU SEGUN LA BUSQUEDA */
```

```
checar(Z,X) :-
  dbase(Z,Y),X=Y,!.

checar(sistema,X) :-
  not(dbase(sistema,_)),
  repeat,
  titulo,
  write("\n\t\t\t     QUE TIPO BASICO DE PROBLEMA TIENE? "),nl,
  write("\n\n\t\t\t\t 1) Problema al inicializar"),nl,
  write("\n\n\t\t\t\t 2) Problemas de corrida"),nl,
  write("\n\n\t\t\t\t 3) Problemas de pantalla"),nl,
  write("\n\n\t\t\t\t 4) Problemas de teclado"),nl,
  write("\n\n\t\t\t\t 5) Problemas de impresora"),nl,
  write("\n\n\t\t\t\t\t SELECIONE :"),
  respuesta(Reply),
  char_int(Reply,Z),
  Z<54,Z>48,!,
  asserta(dbase(sistema,Reply)),
  X=Reply.
checar(inicio,X) :-
  not(dbase(inicio,_)),
  repeat,
  titulo,
  write("\n\t\tQue tipo de problema de inicio tiene? "),nl,
  write("\n\n\t\t 1) Error del sistema durante el inicio "),nl,
  write("\n\t\t 2) Cuando inicializa, no enciende el drive (LED)"),nl,
  write("\t\t    no visualiza nada en pantalla"),nl,
  write("\n\t\t 3) Cuando inicializa, enciende el drive y no "),nl,
  write("\t\t    se visualiza nada en pantalla"),nl,
  write("\n\t\t 4) Trabaja bin, pero no inicializa"),nl,
  write("\n\n\t\t\t SELECCIONE: "),
  respuesta(Reply),
  Char_int(Reply,Z),
  Z<53,Z>48,!,
  asserta(dbase(inicio,Reply)),
  X=Reply.
checar(beep,X) :-
  not(dbase(beep,_)),
  repeat,
  titulo,
  write("\n\tCon algunos beeps, puede oir cuando el diagn"stico"),nl,
  write("\t\t inicial corre? Esto es:"),nl,
  write("\n\t    1) Esto es, no suena el beep."),nl,
  write("\t    2) Suena un beep corto y enciende el LED del drive "),nl,
  write("\t    3) Suena beep continuos "),nl,
  write("\t    4) Suena un beep largo y uno corto "),nl,
  write("\t    5) Suena un beep largo y dos cortos "),nl,
  write("\t    6) Suena un beep corto y el display est \en blanco "),nl,
  write("\t    7) Repite Beep cortos "),nl,
  write("\t    8) Un beep corto. "),nl,
  write("\n\n\t\t\t SELECCIONE : "),
  respuesta(Reply),
  char_int(Reply,Z),
  Z<57,Z>48,!,
  asserta(dbase(beep,Reply)),
  X=Reply.
checar(errorcodigo,X) :-
  not(dbase(errorcodigo,_)),
  repeat,
```

```
    titulo,
    write("\n\tSi un mensaje de error es desplegado. Que es ese mensaje"),nl,
    write("\t\t     a) 101 o 131 o 1xx"),nl,
    write("\t\t     b) 201 o xxxx201 checar la paridad x o 20x o xx20x"),nl,
    write("\t\t     c) 301 o xx301 o teclado no funcionando o 30x o "),nl,
    write("\t\t        xx30x"),nl,
    write("\t\t     d) 601 o 6xx"),nl,
    write("\t\t     e) 4xx"),nl,
    write("\t\t     f) 9xx"),nl,
    write("\t\t     g) 13xx"),nl,
    write("\t\t     h) 14xx o problemas de impresi"n"),nl,
    write("\t\t     i) 1701 o 17xx"),nl,
    write("\t\t     j) 1801 o 18xx"),nl,
    write("\t\t     k) 02x"),nl,
    write("\n\t\t\t\t\t SELECCIONE : "),
    respuesta(Reply),
    char_int(Reply,Z),
    Z<110,Z>96,!,
    asserta(dbase(errorcodigo,Reply)),
    X=Reply.
  checar(floppy_boot,X) :-
    not(dbase(floppy_boot,_)),
    repeat,
    titulo,
    write("\n\tLo siguiente es tratar de diagnosticar el problema con"),nl,
    write("\tel disco flexible al inicializar, decidiendo si el "),nl,
    write("\tproblema es el sistema de la computadora."),
    write("\n\t\t  Que s/ntomas tiene la unidad de drive?"),nl,
    write("\n\t1) El drive est  muerto ,no enciende "),nl,
    write("\t   la luz del LED cuando se inicializa"),nl,
    write("\n\t2) No enciende el LED cuando el motor est  "),nl,
    write("\t   funcionando "),nl,
    write("\n\t3) Despliega el siguiente mensaje NON-SYSTEM DISK o "),nl,
    write("\t   DISK ERROR o el mensaje DISK BOOT FAILURE "),nl,
    write("\n\t4) Despliega el mensaje DRIVE NOT READY "),nl,
    write("\n\t5) Despliega el mensaje BAD MISSING COMMAND INTERPRETER "),nl,
    write("\n\t\t\t SELECCIONE: "),
    respuesta(Reply),
    char_int(Reply,Z),
    Z<54,Z>48,!,
    assert(dbase(floppy_boot,Reply)),
    X=Reply.
  checar(run_disco_duro,X) :-
    not(dbase(run_disco_duro,_)),
    repeat,
    titulo,
    write("\n\n\t\tQue s/ntomas tiene el disco ? "),nl,
    write("\n\t\t   1) El disco est  'muerto' no carga "),nl,
    write("\n\t\t       cuando es accesado "),nl,
    write("\n\t\t   2) No lee " no escribe apropiadamente "),nl,
    write("\n\t\t   3) Lee correctamente pero no escribe "),nl,
    write("\n\t\t   4) el archivo CHKDSK genera mensaje de error "),nl,
    write("\n\t\t   5) El disco tiene un sonido extra/o "),nl,
    write("\n\t\t   6) El disco pierde informaci"n previamente "),nl,
    write("\n\n\t\t\t\t SELECCIONA : "),
    respuesta(Reply),
    char_int(Reply,Z),
    Z<55,Z>48,!,
    asserta(dbase(run_disco_duro,Reply)),
    X=Reply.
```

```
    checar(run_floppy_disk,X) :-
      not(dbase(run_floppy_disk,_)),
      repeat,
      titulo,
      write("\n\n\t\tQue sintomas tiene el disco ?"),nl,
      write("\n\t\t    1) El disco est  'muerto', no carga "),nl,
      write("\t\t       cuando es accesado. "),nl,
      write("\t\t    2) Despliega el siguiente mensaje: "),nl,
      write("\t\t       NOT READY READING DRIVE X"),nl,
      write("\t\t       ABORT,RETRY,IGNORE?"),nl,
      write("\t\t    3) No lee o escribe apropiadamente "),nl,
      write("\t\t    4) Lee correctamente pero no escribe "),nl,
      write("\t\t    5) El disco hace un ruido extraño "),nl,
      write("\t\t    6) Dificultad para cargar o descargar un disco "),nl,
      write("\n\n\t\t\t\t SELECCIONE : "),
      respuesta(Reply),
      char_int(Reply,Z),
      Z<55,Z>48,!,
      asserta(dbase(run_floppy_disk,Reply)),
      X=Reply.
    checar(run,X) :-
      not(dbase(run,_)),
      repeat,
      titulo,
      write("\n\n\t\t Que tipo de problema ha experimentado: "),nl,
      write("\n\t  1) No funciona el Encendido de la computadora "),nl,
      write("\t      el teclado est  muerto "),nl,
      write("\t  2) Al encender la computadora esta no funciona bin"),nl,
      write("\t  3) Mensajes de error de paridad. "),nl,
      write("\t  4) Hace ruidos extraños al encender la computadora "),nl,
      write("\t  5) Operaci"n erratica y fallas generales "),nl,
      write("\t  6) Problemas en operaci"n el algun programa especifico."),nl,
      write("\t  7) Problemas en disco duro "),nl,
      write("\t  8) Problemas de disco flexible "),nl,
      write("\t  9) Sobrecargamientos "),nl,
      write("\n\t\t\t SELECIONE : "),
      respuesta(Reply),
      char_int(Reply,Z),
      Z<58,Z>48,!,
      asserta(dbase(run,Reply)),
      X=Reply.
    checar(despliegue,X) :-
      not(dbase(despliegue,_)),
      repeat,
      titulo,
      write("\n\n\t\tQue tipo de despliegue ha experimentado ?"),nl,
      write("\n\n\t\t    1) No despliega informacion "),nl,
      write("\t\t    2) No despliega entrada y salida de datos"),nl,
      write("\t\t    3) El monitor se sobrecarga "),nl,
      write("\t\t    4) Desincronizaci"n vertical "),nl,
      write("\t\t    5) Desincronizaci"n horizontal"),nl,
      write("\t\t    6) despliega basura (Caracteres extraños) "),nl,
      write("\t\t    7) Los colores son malos "),nl,
      write("\n\n\t\t\t SELECCIONE : "),
      respuesta(Reply),
      char_int(Reply,Z),
      Z<56,Z>48,!,
      asserta(dbase(despliegue,Reply)),
      X=Reply.
    checar(teclado,X) :-
```

```
      not(dbase(teclado,_)),
      repeat,
      titulo,
      write("\n\t\t\t\tAntes de continuar, asegurese de que no "),nl,
      write("\t\t tiene da$o fisico el teclado. Si se "),nl,
      write("\t\t derram" caf en el teclado, se necesitar  "),nl,
      write("\t\t ayuda profesional "),nl,
      write("\n\t\tQue tipo de problemas tiene su teclado "),nl,
      write("\t\t     1) El teclado no hace nada "),nl,
      write("\t\t     2) Despliega otros caracteres que usted no teclea "),nl,
      write("\t\t     3) No funcionan una " mas teclas "),nl,
      write("\t\t     d) Cuando usted presiona una tecla, aparecen "),nl,
      write("\t\t        dos " m s caracteres "),nl,
      write("\t\t     5) Tiene objetos ajenos en el teclado "),nl,
      write("\t\t     6) Se ha derramado algo en el teclado "),nl,
      write("\n\t\t\t SELECCIONE :"),
      respuesta(Reply),
      char_int(Reply,Z),
      Z<56,Z>48,      Z<56,Z>48,!,
      asserta(dbase(teclado,Reply)),
      X=Reply.
    checar(impresor,X) :-
      not(dbase(impresor,_)),
      repeat,
      titulo,
      write("\n\n\tQue tipo de problemas tiene en la impresion?"),nl,
      write("\n\t\t     a) Impresora 'muerta' "),nl,
      write("\t\t     b) Configuraci"n de la impresora (ver manual) "),nl,
      write("\t\t     c) Falla al ponerse en ON-LINE "),nl,
      write("\t\t     d) Imprime basura "),nl,
      write("\t\t     e) Ocasionalmente imprime errores "),nl,
      write("\t\t     f) Se autointerrumpe cuando est  imprimiendo "),nl,
      write("\t\t     g) La cabeza de impresi"n no corre correctamente "),nl,
      write("\t\t     h) Corre correctamente pero no imprime "),nl,
      write("\t\t     i) La impresora se sobre-carga "),nl,
      write("\t\t     j) Imprime caracteres diferentes "),nl,
      write("\t\t     k) No imprime ciertos caracteres al imprimir "),nl,
      write("\n\t\t\t  SELECCIONE :" ),
      respuesta(Reply),
      char_int(Reply,Z),
      Z<109,Z>96,!,
      asserta(dbase(impresor,Reply)),
      X=Reply.

/* POSICION DEL CURSOR EN ESPERA DE UNA RESPUESTA. */

  respuesta(Reply) :-
    readchar(Reply),
    write(Reply),nl.

/* DESPLIEGA LAS CAUSAS */

  causa(ac_power) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,14,74),
    write("\n\tLa causa probablemente es un poder bajo "),nl,
    write("\n\tChecar para estar seguro de que el cable est    "),nl,
    write("\n\tbin conectado al sistema de la computadora. "),nl,
    write("\n\t Asegurese de que exista corriente en la salida "),nl,
    write("\n\t para la impresora ").
  causa(switch) :-
```

```
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tLa causa probablemente es el switch de ON/OFF "),nl,
    write("\n\tChecar el sistema interno del switch de ON/OFF "),nl,
    write("\n\tDe no ser as/, cambie el switch ").
  causa(no_poder) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,14,74),
    write("\n\tProbablemente sean los conectores de la fuente de poder."),nl,
    write("\n\tCheque los cables de la fuente de poder."),nl,
    write("\n\tRevise el tipo de adpatador de video que tiene su PC. "),nl,
    write("\n\tChecar los circuitos del microprocesador y el circuito "),nl,
    write("\n\tROM. Posiblemente sea la falla del sistema.").
  causa(fuente_poder) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tProbablemente sea un problema interno"),nl,
    write("\n\tCheque la fuente de poder y los cables.").
  causa(sistema_tarjeta) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,06,74),
    write("\n\tTiene un sistema defectuoso. Repita o remplace ").
  causa(sistema_tarjeta2) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tEl sistema de teclado de switches est  dañado. "),nl,
    write("\n\tCheque los switches del sistema, puede haber un falso "),nl,
    write("\n\tcontacto, de no ser as/ remplacelo. ").
  causa(despliegue) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tPuede estar defectuoso el cable del monitor o "),nl,
    write("\n\tposiblemente el propio monitor. "),nl,
    write("\n\tRemplace o repare. ").
  causa(disk_drive) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tDeben estar desconectados (internamente) los cables del "),nl,
    write("\n\t de las unidades del drive. Tambien puede ser que "),nl,
    write("\n\tdescontrolados o descincronizados "),nl,
    write("\n\tRemplace o conecte las unidades del drive. ").
  causa(memoria) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tPueden ser uno o mas defectos en los circuitos de la "),nl,
    write("\n\tmemoria, asegurese que todos los 'chips' esten bin "),nl,
    write("\n\tconectados en la tarjeta principal, haga un rastreo "),nl,
    write("\n\tespec/fico para localizar los circuitos en mal estado.").
  causa(teclado) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tEl teclado puede estar mal conectado "),nl,
    write("\n\t o dañado. Revise.").
  causa(mono_monitor) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tCheque los cables del monitor monocromo "),nl,
    write("\n\tVerifique la intensidad del monitor "),nl,
    write("\n\tVerifique la tarjeta del adaptador "),nl,
    write("\n\tSi esto persiste consulte un especialista.").
  causa(impresor) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tLa impresora no est  conectada o defectuosa "),nl,
    write("\n\tCheque la impresora, los adaptadores y los cables.").
  causa(disco_duro) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tTiene un defecto o el disco duro est  desconectado"),nl,
    write("\n\t cheque el disco y la fuente de poder.").
  causa(adaptador_juegos) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
```

```
    write("\n\tEl adaptador de juegos posiblemente este mal "),nl,
    write("\n\tconectado. Verifique los cables.").
  causa(expansion_unit) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,14,74),
    write("\n\tLa unidad de expansi"n puede estar defectuosa "),nl,
    write("\n\tCheque si la unidad de expansi"n esta conectada "),nl,
    write("\n\t al sistema. Posiblemente existe un cable en mal "),nl,
    write("\n\testado. Inicialice el sistema desde un disco flexible "),nl,
    write("\n\ty verifique de nuevo la unidad del disco duro.").
  causa(config) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,14,74),
    write("\n\tAlgo en el archivo CONFIG.SYS est  bloqueando que "),nl,
    write("\n\t el sistema se inicialice correctamente,"),nl,
    write("\n\t esto es, verifique que este bien los comandos de "),nl,
    write("\n\t<DEVICE> que hayan que no funcione bien el "),nl,
    write("\n\t Hardware y el Software.").
  causa(disco_duro) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tEl disco duro puede estar da¤ado. Posiblemente "),nl,
    write("\n\tno tenga el encendido el disco duro. Si el drive "),nl,
    write("\n\test  en la unidad de expansi"n, asegurese de que "),nl,
    write("\n\test encendido para poder trabajar.").
  causa(dos) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tInstale el DOS 2.0 en el disco duro,"),nl,
    write("\n\tsi no existe ningun directorio en el disco "),nl,
    write("\n\tes que posiblemente est da¤ado. "),nl,
    write("\n\tHabra que formatear el disco duro para que funcione.").
  causa(mal_dos) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tEsta mal la versi"n DOS del disco duro "),nl,
    write("\n\tcambie de veri"n y verifique.").
  causa(autoexec) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tPosiblemente hay alguna instrucci"n que haga "),nl,
    write("\n\t que no est funcionando bien el archivo. Revise ").
  causa(desconocido) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tSupongo que se necesita un nuevo Hardware o que se"),nl,
    write("\n\t instale un Software, reinicialice otra vez. "),nl,
    write("\n\t Cheque los sistemas de encendido ").
  causa(drive_power) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tNo est  encendido el drive, puede estar defectuoso."),nl,
    write("\n\tCheque la fuente de poder. Si persiste remplace o repare ").
  causa(fisico_disk) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tSi est  fisicamente da¤ado el disco, cuelgelo en su "),nl,
    write("\n\tarbol de navidad. Pero antes trate de respaldar la "),nl,
    write("\n\tinformaci"n usando el archvivo RECOVER del DOS, " las "),nl,
    write("\n\tutilerias del NORTON ").
  causa(sucio) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tPosiblemente este da¤ado la cabeza lectora o no est  "),nl,
    write("\n\tbin colocado o cerrado la ranura del drive. "),nl,
    write("\n\tLimpie las cabezas con un limpiador especial ").
  causa(dosdiscos) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,14,74),
    write("\n\tTrate de inicializar el sistema desde otro drive si "),nl,
    write("\n\tsu computadora lo permite, es que estan mal configurados "),nl,
```

```
    write("\n\tlas unidades de disco. Cheque esto con el controlador "),nl,
    write("\n\tinterno de la m quina. Verifique cables. "),nl,
    write("\n\tPosiblemente estn los motores mal alineados.").
causa(floppy_drive) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tAparentemente el problema es de la unidad de drive "),nl,
    write("\n\tpueden ser los cables internos o el controlador."),nl,
    write("\n\tAsegurese de que al encender el motor encienda el LED "),nl,
    write("\n\tIntercambie los drives, y verifique los cables.").
causa(format) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\t El disco no est  formateado, o no contiene "),nl,
    write("\n\t el sistema DOS, use un sistema operativo con "),nl,
    write("\n\t un formato mas potente, ese puede ser el problema."),nl,
    write("\n\t Revise que el disco contenga el S.O. correctamente.").
causa(puerta) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tEl disco no est  formateado. "),nl,
    write("\n\tVerifique que las puertas del drive este bin cerradas "),nl,
    write("\n\t" remplace el disco."),nl,
    write("\n\tTal vez comienza a fallar la unidad de disco, revise.").
causa(command) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tEl archivo COMMAND.COM no est  en el disco "),nl,
    write("\n\tUse el DOS con el archivo COMMAND.COM.").
causa(clavija) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,06,74),
    write("\n\t\tCambie los cables.").
causa(hardware) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,14,74),
    write("\n\tTrate de inicializar el sistema desde un disco flexible "),nl,
    write("\n\tCheque si est  mal el DOS, repita la operaci"n "),nl,
    write("\n\tSi el problema persiste es que est  mal algunos circuitos."),nl,
    write("\n\tRevise el teclado, verifique la memoria. Pruebe con "),nl,
    write("\n\tun bun teclado.").
causa(software) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tCheque si el problema periste en su programa."),nl,
    write("\n\tCorralo en otra m quina. Use un identificador "),nl,
    write("\n\t para verificar si tiene algunos sectores del "),nl,
    write("\n\t programa que esten da*ados.").
causa(statica) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tEl problema puede ser originado por electricidad "),nl,
    write("\n\test tica, rocie un 'spray' especial para este problema.").
causa(ocupado) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,16,74),
    write("\n\tExisten algunos programas que al ser ejecutados "),nl,
    write("\n\tno corren adecuadamente o marcan error a la entrada "),nl,
    write("\n\tde datos, puede ser porque solo acepta may#sculas "),nl,
    write("\n\tactive el CAPS-LOCK del teclado o tambin puede ser "),nl,
    write("\n\tque no se compil" bin " en otra computadora con otras "),nl,
    write("\n\tlibrerias, osea problemas de ejecuci"n (LINK). ").
causa(caer_power) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tPuede ser que la fuente de poder est da*ado " en "),nl,
    write("\n\talgunos casos puede tener electricidad est tica."),nl,
    write("\n\tCheque la tarjeta principal.").
causa(parity_error) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
```

```
    write("\n\tPor lo general la falla est  en un chip de la memoria "),nl,
    write("\n\tlocalice este chip y remplace. Otra causa puede ser "),nl,
    write("\n\tque el Hardware est dañado en los interruptores de "),nl,
    write("\n\terror de paridad, instale uno nuevo. ").
  causa(caesistema) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tLas causas pueden ser : "),nl,
    write("\n\t1. un programa en particular."),nl,
    write("\n\t2. Un problema en el Hardware o drives."),nl,
    write("\n\t3. Un problema en la tarjeta principal.").
  causa(erratico_software) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tCheque si existe algun problema con los programas "),nl,
    write("\n\tresidentes, si esto no es, cree una nueva copia del "),nl,
    write("\n\tprograma. Cheque todos los programas residentes si "),nl,
    write("\n\tel problema persiste.").
  causa(configura) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tLa temperatura ambiente de la computadora debe de "),nl,
    write("\n\testar entre 15 y 22 grados Centigrados. Esto puede variar "),nl,
    write("\n\tcheque los componentes de su computadora ").
  causa(run_disco_duro) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tPueden estar mal los cables o el drive "),nl,
    write("\n\t est  defectuoso.").
  causa(run_expansion) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tAsegurese que la unidad de expansi"n este en ON "),nl,
    write("\n\t si lo est , entonces el disco duro est  defectuoso.").
  causa(run_durodisk_write) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tEl problema puede ser que est un drive en mal estado "),nl,
    write("\n\t " que los cables del controlador de drives esten malos.").
  causa(run_durodisk_lee) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tEl problema puede ser el controlador de drives "),nl,
    write("\n\t" que los cables de ste esten mal.").
  causa(run_durodisk_chkdsk) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,16,74),
    write("\n\tPueden existir algunos Clusters perdidos."),nl,
    write("\n\tEl problema puede ser causa de defecto en el drive "),nl,
    write("\n\t" tambien puede ser por diseño (incompatibilidad) con otro"),nl,
    write("\n\tprograma. Si corre CHKDSK y no lo encuentra es que "),nl,
    write("\n\tsu sistema operativo es una versi"n vieja. "),nl,
    write("\n\tUtilice la versi"n 1985 del DOS en adelante.").
  causa(run_durodisk_ruido) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tLo mas probable es que el error sea en el drive "),nl,
    write("\n\trespalde el disco y cheque automaticamente ste. ").
  causa(run_duro) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tSaque el disco del drive, RESPALDE EL DISCO INMEDIATAMENTE"),nl,
    write("\n\tchequelo.").
  causa(fisico_movimiento) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tEn algunas computadoras, El drive del disco todavía "),nl,
    write("\n\test  -duro- afloje esto dando DIR o corriendo unos "),nl,
    write("\n\tprogramas para que tenga m s facilidad de movimiento"),nl,
    write("\n\tEstacione el disco usando el archivo PARK. Revise.").
  causa(run_disk_software) :-
```

```
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tAlgunos programas tienen dificultad para trabajar con "),nl,
    write("\n\talgunos discos. Si el problema es con algun programa "),nl,
    write("\n\ten especial, cheque el programa.").
causa(run_resident) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tAlg#n software residente puede interferir con la lectura "),nl,
    write("\n\ten el disco. Trate de mover especificamente el programa "),nl,
    write("\n\tresidente " instale en otro "rden. ").
causa(run_floppy_fisico) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tRecupere la informaci"n desde otro disco y "),nl,
    write("\n\tdestruya el disco. ").
causa(mal_floppy) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tRecupere la informaci"n del disco y formatee "),nl,
    write("\n\tel otro disco. Si el problema persiste destruyalo "),nl,
    write("\n\tVerifique la marca que este libre de error o si lo desea "),nl,
    write("\n\tcambie de marca. ").
causa(mal_floppy_drive) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tSi el problema no es del disco flexible entonces "),nl,
    write("\n\tcheque el drive si est  bin alineado, Tal vez necesite "),nl,
    write("\n\treparaci"n. ").
causa(mal_floppy2) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tSi el problema es consistente con algun disco "),nl,
    write("\n\trespalde y remplace el disco. Si el problema persiste "),nl,
    write("\n\tcambie de marca. ").
causa(run_floppy_disk_power) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tProbablemente es que el drive no tenga suficiente poder "),nl,
    write("\n\tcambie y revise.").
causa(run_floppy_no_lee) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,14,74),
    write("\n\tLas causas mas probables es que el el disco no este "),nl,
    write("\n\tbin colocado en el drive, o verifique las puertas del "),nl,
    write("\n\tdrive que este bin cerrados. Asegurese que su disco est"),nl,
    write("\n\tformateado. Si persiste la falla cheque en otra"),nl,
    write("\n\tcomputadora " cambie de disco. ").
causa(escritura_protecc) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tDesproteja su disco quitando la protecci"n que tiene en "),nl,
    write("\n\t la ranura de la parte derecha del disco. ").
causa(floppy_ruido) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tProbablemente es que exista un objeto que afecte "),nl,
    write("\n\t el drive, o que el motor comienza a fallar. Revise. ").
causa(floppy_cargar) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tProbablemente es que exista un objeto que afecte "),nl,
    write("\n\t el drive, revise. ").
causa(run_recalentamiento) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tAsegurese que no est defectuoso un elemento del "),nl,
    write("\n\tsistema, o verifique que tenga ventilaci"n su equipo "),nl,
    write("\n\t" la parte donde se encuentra. ").
causa(run_software_dos) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tVea el tipo de mensaje del DOS y vea el manual "),nl,
```

```
    write("\n\t y continue con su proceso. ").
  causa(run_software_aplicacion) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tContacte la compania del software para pedir ayuda "),nl,
    write("\n\tverifique el manual de operacion e intente de  nuevo.").
  causa(no_despliegue) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tAsegurese que el monitor este encendido "),nl,
    write("\n\tmueva los controles de contraste y brillantes "),nl,
    write("\n\to verifique que el cable del monitor este bin "),nl,
    write("\n\tconectado. ").
  causa(despliegue_desvanece) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tLa fuente de poder puede estar defectuoso "),nl,
    write("\n\tsustituya otro monitor, si la falla es con el monitor "),nl,
    write("\n\tes que este est  defectuoso. ").
  causa(despliegue_recalentamiento) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tAseugurese que el monitor tenga ventilaci"n "),nl,
    write("\n\tEl monitor puede estar fallando. ").
  causa(despliegue_vertical) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tAjuste los controles que estan atras del monitor "),nl,
    write("\n\tsi la falla persiste, revise los controles. ").
  causa(despliegue_horizontal) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tPuede ser la incompatibilidad del adaptador de video"),nl,
    write("\n\tSi tiene un software para video, uselo."),nl,
    write("\n\tAjuste los controles que se encuentran atras del "),nl,
    write("\n\t monitor, si la falla persiste, revise los controles. ").
  causa(despliegue_basura) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,13,74),
    write("\n\tPrimero asegurese de que el teclado este bin "),nl,
    write("\tdesconecte y revise con otra computadora. Si el teclado "),nl,
    write("\test  bin, entonces pruebe el monitor, revise que no est "),nl,
    write("\tdesplegando caracteres extaraños, desconecte y pruebe con "),nl,
    write("\totra computadora. Si el monitor est  bin, es que la "),nl,
    write("\tparte central de la memoria no est  captando bin la "),nl,
    write("\tinformaci"n del teclado. Otro problema puede ser que  "),nl,
    write("\tcomiencen a fallar los PIXELES del monitor.").
  causa(despliegue_color) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tAjuste los controles de color, si los controles no "),nl,
    write("\n\tresponden, revise el adaptador de la tarjeta de video. ").
  causa(teclado_muerto) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tAsugurese que el teclado est bin conectado a la "),nl,
    write("\n\tcomputadora verifique que la pantalla despliegue cualquier"),nl,
    write("\n\tmensaje cuando usted teclee cualquier cosa. Cheque los "),nl,
    write("\n\tcables e instale uno nuevo. ").
  causa(teclado_basura) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,18,74),
    write("\n\tAsegure que las teclas esten en buena posici"n "),nl,
    write("\n\trevise que despliegue los caracteres que usted da "),nl,
    write("\n\tPuede ser que est configurado a otro formato, por lo"),nl,
    write("\n\tregular al formato USA, use los comandos KEYxxx para "),nl,
    write("\n\tcambiar el tipo de formato."),nl,
    write("\n\tPruebe con otro teclado compatible. Si la falla persiste"),nl,
    write("\n\tes que su teclado est  en malas condiciones. ").
  causa(teclado_tecla) :-
```

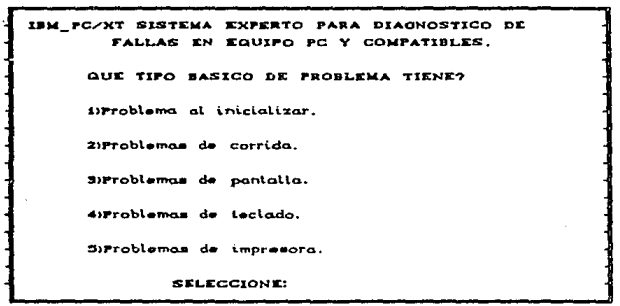
```
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tQuite las teclas y limpie la base de estas, cuidando de"),nl,
    write("\n\tno maltratar los resortes, coloque las teclas verificando"),nl,
    write("\n\tque esten bin colocadas. ").
causa(teclado_doble) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tUse un DOS desde la unidad A del disco flexible"),nl,
    write("\n\tReinicialice primero, usando CTRL/ALT/DEL, si desde "),nl,
    write("\n\tel reinicio es malo, es que el teclado no corresponde "),nl,
    write("\n\t" no es compatible con el sistema que usted tiene. ").
causa(teclado_ajeno) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tVerifique que el teclado sea de la configuraci"n de la "),nl,
    write("\n\tcomputadora, pruebe dando CTRL/ALT/DEL para ver si "),nl,
    write("\n\t las teclas funcionan correctamente con el sistema. ").
causa(teclado_derrame) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,12,74),
    write("\n\tRemueva la potencia inmediatamente, enjuage con una "),nl,
    write("\n\tfranela y pongale aire presurizado. Deje "),nl,
    write("\n\tsecar 48 horas y chequelo, esto no necesita mas "),nl,
    write("\n\treparaci"n. ").
causa(impresor_muerto) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tRemplace la fuente de poder de la impresora, "),nl,
    write("\n\trevise si llega el volaje adecuado. ").
causa(impresor_notest) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,06,74),
    write("\n\tRevise el manual de operaci"n de su impresora.").
causa(impresor_noline) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tEl cable est  defectuoso o el adaptador. "),nl,
    write("\n\tTal vez el interruptor de ON-LINE comience a fallar. Revise.").
causa(impresor_viejo) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tVerifique los interruptores de su impresora "),nl,
    write("\n\t y coloquelos adecuadamente a su sistema. ").
causa(impresor_programa) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tNo seleccion" adecuadamente el drive donde se "),nl,
    write("\n\tencuentra el programa a imprimir, use la opcion "),nl,
    write("\n\t MODE del DOS para imprimir. ").
causa(impresor_erratico) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tLa impresora no est  ajustada correctamente. "),nl,
    write("\n\tLos cables pueden estar mal ajustados."),nl,
    write("\n\tRevise el programa que est  imprimiendo.").
causa(impresor_stop) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
    write("\n\tPuede existir bloqueo en algo , que no tenga papel"),nl,
    write("\n\to que se no este en ON-LINE, verifique los cables. ").
 causa(impresor_carro) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tCheque la fuente de potencia, vea si no existe objetos "),nl,
    write("\n\tque obstruyan la impresi"n, verifique que la cabeza de "),nl,
    write("\n\timpresi"n corre bin en el rodillo. ").
 causa(impresor_noprint) :-
    makewindow(1,113,112,"HIPOTESIS",5,2,11,74),
    write("\n\tLa impresora no est  instalada correctamente "),nl,
    write("\n\tajuste los interruptores de control, si esto no "),nl,
    write("\n\tfunciona es que la impresora esta defectuosa.").
```

```
causa(impresor_recalentar) :-
  makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
  write("\n\tLa impresora se sobrecarga, necesita ventilaci"n"),nl,
  write("\n\ty descanso. ").
causa(impresor_perdido) :-
  makewindow(1,113,112,"HIPOTESIS",5,2,14,74),
  write("\n\tCheque la configuraci"n de impresi"n de su programa"),nl,
  write("\n\tes decir que el programa est configurado a otro tipo"),nl,
  write("\n\timpresora."),nl,
  write("\n\tcheque la configuraci"n de la impresora, si es de "),nl,
  write("\n\tmatriz, verifique este bin alineada.").
causa(papel_bloqueo) :-
  makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
  write("\n\tCheque que el papel este en buen estado y no "),nl,
  write("\n\tlleve objetos extraños. ").
causa(impresor_desigual) :-
  makewindow(1,113,112,"HIPOTESIS",5,2,08,74),
  write("\n\tLa impresora necesita ajustamiento en los interruptores. "),nl,
  write("\n\tRevise el programa, ste tambin puede ser la causa.").
```

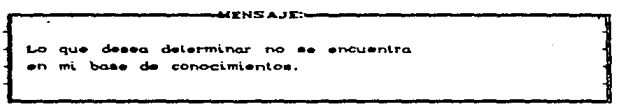
VI.1 ENTRADA DE DATOS.

La entrada de datos que presenta el sistema es a modo de menús. Conforme va haciendo preguntas, va evaluando las respuestas para llegar a una conclusión.

La pantalla principal presenta las fallas más características de una computadora personal. He aquí la pantalla.



Este es el menú principal para la entrada de datos del sistema. Aquí es donde comienza a manejar las reglas de búsqueda para obtener una respuesta. Como la búsqueda es variable, habrá casos en que no encuentre una solución. Cuando esto sucede presenta el siguiente mensaje:



Este mensaje indica que en la búsqueda no encontró una respuesta. Pero este mensaje es desplegado un mínimo de 5 ocasiones para diferentes problemas.

Cabe aclarar que la entrada de datos es en minúsculas, es decir que la tecla CAPS-LOCK debe estar desactivada. A continuación se despliega todas las posibles entradas de datos, sin un órden específico, la intensión es conocer los submenús de cada problema así como las preguntas denominadas *de redondeo*, para encontrar la solución más adecuada.

MENU PRINCIPAL

```
IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO
                        PC Y COMPATIBLE

                  QUE TIPO BASICO DE PROBLEMA TIENE?
                      1) Problema al inicializar
                      2) Problemas de corrida
                      3) Problemas de pantalla
                      4) Problemas de teclado
                      5) Problemas de impresora
                           SELECIONE :
```

MENU PARA PROBLEMAS DE INICIO

```
IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO
                        PC Y COMPATIBLE

              Que tipo de problema de inicio tiene?
               1) Error del sistema durante el inicio
               2) Cuando inicializa, no enciende el drive (LED)
                  no visualiza nada en pantalla
               3) Cuando inicializa, enciende el drive y no
                  se visualiza nada en pantalla
               4) Trabaja bién, pero no inicializa
                     SELECCIONE:
```

MENU PARA DIAGNOSTICO DE BEEPS

```
IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO
                        PC Y COMPATIBLE

          Con algunos beeps, puede oir cuando el diagnóstico
                inicial corre? Esto es:
            1) Esto es, no suena el beep.
            2) Suena un beep corto y enciende el LED del drive
            3) Suena beep continuos
            4) Suena un beep largo y uno corto
            5) Suena un beep largo y dos cortos
            6) Suena un beep corto y el display est  en blanco
            7) Repite Beep cortos
            8) Un beep corto.
                        SELECCIONE :
```

MENU DE DIAGNOSTICO DE CORRIDAS.

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Lo siguiente es tratar de diagnosticar el problema con el disco flexible al inicializar, decidiendo si el problema es el sistema de la computadora.
Que síntomas tiene la unidad de drive?

1) El drive está muerto ,no enciende la luz del LED cuando se inicializa
2) No enciende el LED cuando el motor est funcionando
3) Despliega el siguiente mensaje NON-SYSTEM DISK o DISK ERROR o el mensaje DISK BOOT FAILURE
4) Despliega el mensaje DRIVE NOT READY
5) Despliega el mensaje BAD MISSING COMMAND INTERPRETER

SELECCIONE:

MENU PARA DIAGNOSTICO DE MENSAJES DE PARIDAD.

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Si un mensaje de error es desplegado. Que es ese mensaje

a) 101 o 131 o 1xx
b) 201 o xxxx201 checar la paridad x o 20x o xx20x
c) 301 o xx301 o teclado no funcionando o 30x o xx30x
d) 601 o 6xx
e) 4xx
f) 9xx
g) 13xx
h) 14xx o problemas de impresión
i) 1701 o 17xx
j) 1801 o 18xx
k) 02x

SELECCIONE :

PREGUNTAS PARA REDONDEAR EL DIAGNOSTICO.

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Un [Beep] corto es un sonido normal despues de que el sistema es inicializado desde el disco flexible.
Marca el problema al iniciar (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Su siguiente tarea es de aislar el problema para ambos (el disco fásico o el hardware). Asegurese de que el disco fisicamente no este dañado. Use otro disco en el mismo drive. Asegurese de que el disco est formateado y contenga el DOS, pruebe el mismo disco-flexible en otra computadora. El disco está defectuoso (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Haga un chequeo visual del drive. Asegurese que el disco carge apropiadamente y se active el motor del drive. Limpie la cabeza lectura. Use un limpiador especial. Funciona bién ahora (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Su próxima tarea, es para aislar el problema entre el drive del disco, controladores, y cable del disco. Esto es simplificado si tiene un segundo disco en la unidad ó un disco sobrante para usar. Tiene uno disponible o 2 floppy-disk en la unidad (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Tiene una prueba la memoria (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Checar el cable principal, puede estar mal colocado. Esta bién colocado el cable principal (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

está usted tratando de reiniciar desde el disco duro (s/n)?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Trate de inicializar el sistema desde el disco flexible asegurese que el disco este bueno con una copia del DOS. Es esto acertado (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

tiene instalado la versión 2.0 en su disco duro (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

El disco toca el panel. se siente el motor del drive cuando gira y no enciende el LED del drive cuando busca alguna información en el disco (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Despues de inicializar desde el disco flexible, trate de leer el directorio en el disco duro y comienze un programa desde el disco duro.
Es esto acertado (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Trate de restaurar el DOS para el disco duro desde el disco flexible colocando un disco DOS en la unidad A y teclee siguiente: SYS C: .
Es esto acertado (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Reinicialice desde el disco flexible, renombre el archivo AUTOEXEC.BAT en el disco duro con otro nombre y trate de reinicializar desde el disco duro.
Es esto acertado (s/n) ?

ENTRADA DE DATOS PARA LOS PROBLEMAS DE CORRIDA.

```
IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO
                          PC Y COMPATIBLE

          Que tipo de problema ha experimentado:

  1) No funciona el Encendido de la computadora
     el teclado está muerto
  2) Al encender la computadora esta no funciona bién
  3) Mensajes de error de paridad.
  4) Hace ruidos extraños al encender la computadora
  5) Operación erratica y fallas generales
  6) Problemas en operación el algun programa específico.
  7) Problemas en disco duro
  8) Problemas de disco flexible
  9) Sobrecargamientos

                    SELECIONE :
```

ENTRADA DE DATOS PARA DIAGNOSTICO DE DISCO DURO.

```
IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO
                          PC Y COMPATIBLE

        Que síntomas tiene el disco ?
           1) El disco está 'muerto' no carga
              cuando es accesado
           2) No lee ó no escribe apropiadamente
           3) Lee correctamente pero no escribe
           4) el archivo CHKDSK genera mensaje de error
           5) El disco tiene un sonido extraño
           6) El disco pierde información previamente
                      SELECCIONA:
```

ENTRADA DE DATOS PARA DISCO FLEXIBLE.

```
IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO
                          PC Y COMPATIBLE

        Que síntomas tiene el disco ?
           1) El disco está 'muerto', no carga
              cuando es accesado.
           2) Despliega el siguiente mensaje:
              NOT READY READING DRIVE X
              ABORT,RETRY,IGNORE?
           3) No lee o escribe apropiadamente
           4) Lee correctamente pero no escribe
           5) El disco hace un ruido extraño
           6) Dificultad para cargar o descargar un disco

                SELECCIONE :
```

REDONDEA EL DIAGNOSTICO PARA LOS PROBLEMAS DE CORRIDA.

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Es la temperatura ambiente de la computadora
<= 15 grados y >= 22 grados centigrados,
todas las ventilaciones estan abiertas
y todas llegan a la computadora (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Asegurese de que la computadora no est haciendo otra
cosa y el teclado este intensionalmente desconectado,
al menos que el trabajo este completamente terminado.
Mire la luz del drive.
Esta la computadora realmente ocupada -trabajando- (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Asegurese de que la clavija del teclado este bién
conectado en el sistema y que el cable este en buén
estado. Estan los cables bién (s/n)

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Trate de tomar un reinicio acertado,. use CTRL/ALT/DEL
primero. Si esto no reinicializa apague la computadora y
pruebe otra vez, si esto no trabaja, apague la
computadora un rato (15 minutos), entonces reinicialice
otra vez. Con el reinicio, es acertado eventualmente (s/n)

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Trate de repetir exactamente todos los pasos con LOCK-UP
exactamente y en el mismo orden.
La computadora sigue igual con LOCK-UP (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Está el disco duro en una unidad de expansión (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Hace esto solamente con un programa (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Trate de mover todos los programas residentes y comenzar
sin el AUTOEXEC.BAT (renombre este y comience).
Persiste el problema (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Es algo consistente acerca del problema,
esto es, tiene el mismo problema en el mismo programa
cuando usted hace la misma cosa (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Mueve usted físicamente la computadora de su lugar (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Checar el disco visualmente para verificar daño físico
El disco flexible esta bién (s/n) ?

IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE FALLAS EN EQUIPO PC Y COMPATIBLE

Trate de mover todos los programas residentes y comenzar
sin el AUTOEXEC.BAT (renombre este y comience).
Persiste el problema (s/n) ?

VI.2.- RESPUESTAS DEL SISTEMA EXPERTO.

El programa presenta un total de 87 posibles respuestas, establecidas en las rutinas de causa. Cada rutina de respuesta tiene una solución, ejemplo:

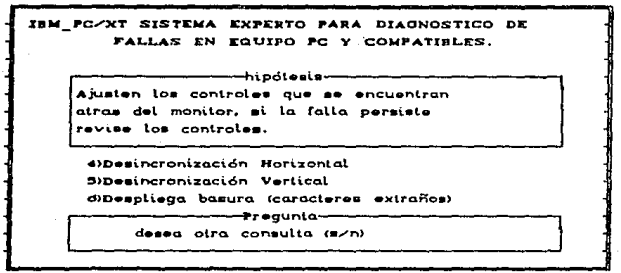
```
causa(despliegue_vertical) :-
makewindow(1,113,112,'hipotesis',5,2,8,74),
write('ajuste los controles que estan atras del monitor '),nl,
write('si la falla persiste revise los controles').
```

Esta rutina despliegua la causa por la que pueda existir un despliegue vertical.

Al momento de desplegar la causa -hipótesis-, esta aparece en una ventana según el tamaño del texto. Ejemplo:

```
IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE
        FALLAS EN EQUIPO PC Y COMPATIBLES.

      QUE TIPO BASICO DE PROBLEMA TIENE?

      1)Problema al inicializar.
      2)Problemas de corrida.
      3)Problemas de pantalla.
      4)Problemas de teclado.
      5)Problemas de impresora.
            SELECCIONE: 3
```

```
IBM_PC/XT SISTEMA EXPERTO PARA DIAGNOSTICO DE
        FALLAS EN EQUIPO PC Y COMPATIBLES.

    Que tipo de despliegue ha experimentado ?

      1)No despliega información
      2)No despliega Entrada o salida de datos
      3)El monitor se sobrecarga
      4)Desincronización Horizontal
      5)Desincronización Vertical
      6)Despliega basura (caracteres extraños)
      7)Los colores son malos

            SELECCIONE : 5
```



También despliega en una ventana aparte, la opción a otra consulta (se recuerda usar minúsculas).

A continuación se presentan las salidas del sistema a modo de ventanas (tal y como lo presenta el sistema). Dichas salidas estan organizadas en los temas ya vistos.

RESPUESTAS PARA PROBLEMAS DE INICIO

La causa probablemente es un poder bajo ó
Checar para estar seguro de que el cable esté
bién conectado al sistema de la computadora.
Asegurese de que exista corriente en la salida
para la impresora.

La causa probablemente es el switch de ON/OFF
Checar el sistema interno del switch de ON/OFF
De no ser así, cambie el switch

Probablemente sean los conectores de la fuente de poder.
Cheque los cables de la fuente de poder.
Revise el tipo de adpatador de video que tiene su PC.
Checar los circuitos del microprocesador y el circuito
ROM. Posiblemente sea la falla del sistema.

Probablemente sea un problema interno
Cheque la fuente de poder y los cables.
Tiene un sistema defectuoso. Repita o remplace.

El sistema de teclado de switches esté dañado.
Cheque los switches del sistema, puede haber un falso
contacto, de no ser así remplacelo.

Puede estar defectuoso el cable del monitor o
posiblemente el propio monitor.
Remplace o repare.

Deben estar desconectados (internamente) los cables
de las unidades del drive. También puede ser que
descontrolados o descincronizados
Remplace o conecte las unidades del drive.

Pueden ser uno o mas defectos en los circuitos de la
memoria, asegurese que todos los 'chips' esten bién
conectados en la tarjeta principal, haga un rastreo
específico para localizar los circuitos en mal estado.

El teclado puede estar mal conectado
o dañado. Revise.

Cheque los cables del monitor monocromo
Verifique la intensidad del monitor
Verifique la tarjeta del adaptador
Si esto persiste consulte un especialista.

La impresora no está conectada o defectuosa
Cheque la impresora, los adaptadores y los cables.

Tiene un defecto o el disco duro está desconectado
cheque el disco y la fuente de poder.

El adaptador de juegos posiblemente este mal
conectado. Verifique los cables.

La unidad de expansión puede estar defectuosa
Cheque si la unidad de expansión esta conectada
al sistema. Posiblemente exista un cable en mal
estado. Inicialice el sistema desde un disco flexible
y verifique de nuevo la unidad del disco duro.

Algo en el archivo CONFIG.SYS est bloqueando que
el sistema se inicialice correctamente,
esto es, verifique que este bién los comandos de
<DEVICE> que hagan que no funcione bién el
Hardware y el Software.

RESPUESTAS PARA PROBLEMAS DE CORRIDA.

El disco duro puede estar dañado. Posiblemente
no tenga el encendido el disco duro. Si el drive
está en la unidad de expansión, asegurese de que
está encendido para poder trabajar.

Instale el DOS 2.0 en el disco duro,
si no existe ningun directorio en el disco
es que posiblemente esté dañado.
Habra que formatear el disco duro para que funcione.

Esta mal la versión DOS del disco duro
cambie de versión y verifique.

Posiblemente hay alguna instrucción que haga
que no esté funcionando bién el archivo. Revise

Supongo que se necesita un nuevo Hardware o que se
instale un Software, reinicialice otra vez.
Cheque los sistemas de encendido.

No está encendido el drive, puede estar defectuoso.
Cheque la fuente de poder. Si persiste remplace o repare.

Si está fisicamente danado el disco, cuelgelo en su arbol de navidad. Pero antes trate de respaldar la información usando el archivo RECOVER del DOS, ó las utilerías del NORTON.

Posiblemente este dañado la cabeza lectora o no está bién colocado o cerrado la ranura del drive.
Limpie las cabezas con un limpiador especial.

Trate de inicializar el sistema desde otro drive si su computadora lo permite, ó es que estan mal configurados las unidades de disco. Cheque esto con el controlador interno de la máquina. Verifique cables.
Posiblemente estén los motores mal alineados.

Aparentemente el problema es de la unidad de drive pueden ser los cables internos o el controlador.
Asegurese de que al encender el motor encienda el LED
Intercambie los drives, y verifique los cables.

El disco no está formateado, o no contiene el sistema DOS, use un sistema operativo con un formato mas potente, ese puede ser el problema.
Revise que el disco contenga el S.O. correctamente.

El disco no está formateado.
Verifique que las puertas del drive este bién cerradas
remplace el disco.
Tal vez comienza a fallar la unidad de disco, revise.

El archivo COMMAND.COM no está en el disco
Use el DOS con el archivo COMMAND.COM.

Cambie los cables.

Trate de inicializar el sistema desde un disco flexible
Cheque si está mal el DOS, repita la operación
Si el problema persiste es que está mal algunos circuitos
Revise el teclado, verifique la memoria. Pruebe con un buén teclado.

Cheque si el problema periste en su programa.
Corralo en otra máquina. Use un identificador para verificar si tiene algunos sectores del programa que esten dañados.

El problema puede ser originado por electricidad estática, rocie un 'spray' especial para este problema.

Existen algunos programas que al ser ejecutados no corren adecuadamente o marcan error a la entrada de datos, puede ser porque solo acepta mayúsculas active el CAPS-LOCK del teclado o también puede ser que no se compiló bién ó en otra computadora con otras librerias, osea problemas de ejecución (LINK).

Puede ser que la fuente de poder esté dañado ó en algunos casos puede tener electricidad estática. Cheque la tarjeta principal.

Por lo general la falla está en un chip de la memoria localice este chip y remplace. Otra causa puede ser que el Hardware esté dañado en los interruptores de error de paridad, instale uno nuevo.

las causas pueden ser :
1. un programa en particular.
2. Un problema en el Hardware o drives.
3. Un problema en la tarjeta principal.

Cheque si existe algun problema con los programas residentes, si esto no es, cree una nueva copia del programa. Cheque todos los programas residentes si el problema persiste.

La temperatura ambiente de la computadora debe de estar entre 15 y 22 grados Centigrados. Esto puede variar cheque los componentes de su computadora.

Pueden estar mal los cables o el drive está defectuoso.

Asegurese que la unidad de expansión este en ON si lo está, entonces el disco duro está defectuoso.

El problema puede ser que esté un drive en mal estado ó que los cables del controlador de drives esten malos.

El problema puedo ser el controlador de drives que los cables de éste esten mal.

Pueden existir algunos Clusters perdidos.
El problema puede ser causa de defecto en el drive
ó tambien puede ser por diseño (incompatibilidad) con otro
programa. Si corre CHKDSK y no lo encuentra es que
su sistema operativo es una versión vieja.
Utilice la versión 1985 del DOS en adelante.

Lo mas probable es que el error sea en el drive
respalde el disco y cheque automaticamente éste.

Saque el disco del drive, RESPALDE EL DISCO INMEDIATAMENTE
chequelo.

En algunas computadoras, El drive del disco todavía
está -duro- afloje esto dando DIR o corriendo unos
programas para que tenga más facilidad de movimiento
Estacione el disco usando el archivo PARK. Revise.

Algunos programas tienen dificultad para trabajar con
algunos discos. Si el problema es con algun programa
en especial, cheque el programa.

Algún software residente puede interferír con la lectura
en el disco. Trate de mover específicamente el programa
residente ó instale en otro órden.

Recupere la información desde otro disco y
destruya el disco.

Recupere la información del disco y formatee
el otro disco. Si el problema persiste destruyalo
Verifique la marca que este libre de error o si lo desea
cambie de marca.

Si el problema no es del disco flexible entonces
cheque el drive si está bién alineado, Tal vez necesite
reparación.

Si el problema es consistente con algun disco
respalde y remplace el disco. Si el problema persiste
cambie de marca.

Probablemente es que el drive no tenga suficiente poder
cambie y revise.

Las causas mas probables es que el el disco no este bién colocado en el drive, o verifique las puertas del drive que este bién cerrados. Asegurese que su disco esté formateado. Si persiste la falla cheque en otra computadora ó cambie de disco.

Desproteja su disco quitando la protección que tiene en la ranura de la parte derecha del disco.

Probablemente es que exista un objeto que afecte el drive, o que el motor comienza a fallar. Revise.

Probablemente es que exista un objeto que afecte el drive, revise.

Asegurese que no esté defectuoso un elemento del sistema, o verifique que tenga ventilación su equipo la parte donde se encuentra.

Vea el tipo de mensaje del DOS y vea el manual y continue con su proceso.

Contacte la compañía del software para pedir ayuda verifique el manual de operación e intente de nuevo.

RESPUESTAS PARA PROBLEMAS DE PANTALLA.

Asegurese que el monitor este encendido nueva los controles de contraste y brillantes o verifique que el cable del monitor este bién conectado.

La fuente de poder puede estar defectuoso sustituya otro monitor, si la falla es con el monitor es que este está defectuoso.

Aseugurese que el monitor tenga ventilación El monitor puede estar fallando.

Ajuste los controles que estan atras del monitor si la falla persiste, revise los controles.

Puede ser la incompatibilidad del adaptador de video Si tiene un software para video, uselo. Ajuste los controles que se encuentran atras del monitor, si la falla persiste, revise los controles.

Primero asegurese de que el teclado este bién desconecte y revise con otra computadora. Si el teclado está bién, entonces pruebe el monitor, revise que no esté desplegando caracteres extaraños, desconecte y pruebe con otra computadora. Si el monitor está bién, es que la parte central de la memoria no está captando bién la información del teclado. Otro problema puede ser que comiencen a fallar los PIXELES del monitor.

Ajuste los controles de color, si los controles no responden, revise el adaptador de la tarjeta de video.

RESPUESTAS PARA PROBLEMAS DE TECLADO.

Asugurese que el teclado esté bién conectado a la computadora verifique que la pantalla despliegue cualquier mensaje cuando usted teclee cualquier cosa. Cheque los cables e instale uno nuevo.

Asegure que las teclas esten en buena posición revise que despliegue los caracteres que usted da Puede ser que est configurado a otro formato, por lo regular al formato USA, use los comandos KEYxxx para cambiar el tipo de formato.

Pruebe con otro teclado compatible. Si la falla persiste es que su teclado está en malas condiciones.

Quite las teclas y limpie la base de estas, cuidando de no maltratar los resortes, coloque las teclas verificando que esten bién colocadas.

Use un DOS desde la unidad A del disco flexible Reinicialice primero, usando CTRL/ALT/DEL, si desde el reinicio es malo, es que el teclado no corresponde u no es compatible con el sistema que usted tiene.

Verifique que el teclado sea de la configuración de la computadora, pruebe dando CTRL/ALT/DEL para ver si las teclas funcionan correctamente con el sistema.

Remueva la potencia inmediatamente, enjuage con una franela y pongale aire presurizado. Deje secar 48 horas y chequelo, esto no necesita mas reparación.

RESPUESTAS PARA PROBLEMAS DE IMPRESORA.

Remplace la fuente de poder de la impresora,
revise si llega el volaje adecuado.

Revise el manual de operación de su impresora.

El cable está defectuoso o el adaptador.
Tal vez el interruptor de ON-LINE comience a fallar. Revise.

Verifique los interruptores de su impresora
y coloquelos adecuadamente a su sistema.

No seleccionó adecuadamente el drive donde se
encuentra el programa a imprimir, use la opción
MODE del DOS para imprimir.

La impresora no está ajustada correctamente.
Los cables pueden estar mal ajustados.
Revise el programa que está imprimiendo.

Puede existir bloqueo en algo , que no tenga papel
o que se no este en ON-LINE, verifique los cables.

Cheque la fuente de potencia, vea si no existe objetos
que obstruyan la impresión, verifique que la cabeza de
impresión corre bién en el rodillo.

La impresora no está instalada correctamente
ajuste los interruptores de control, si esto no
funciona es que la impresora esta defectuosa.

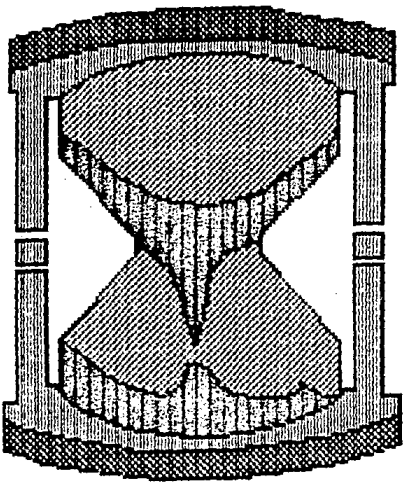
La impresora se sobrecarga, necesita ventilación
y descanso.

Cheque la configuración de impresión de su programa
es decir que el programa est configurado a otro tipo
impresora.
cheque la configuración de la impresora, si es de
matriz, verifique este bién alineada.

Cheque que el papel este en buen estado y no
lleve objetos extraños.

La impresora necesita ajustamiento en los interruptores.
Revise el programa, éste también puede ser la causa.

PARTE IV



ANALISIS

VII.- ANALISIS

Cuando se desarrolla un sistema, ó se va desarrollar se toma mucho en cuenta la inversión de lo que se está haciendo.

Esto se hace con el fin de que si se quiere comercializar el producto, saber cuanto es lo que valora, y para obtener esa cantidad, hay una serie de pasos a tomar para llegar a la cifra adecuada.

Dicha inversión toma en cuenta lo siguiente:

VII.1 COSTO DEL SISTEMA.

Dado que el costo es uno de los aspectos que determina si un sistema es aceptado, los analistas se aseguran de que se identifiquen y estimen apropiadamente todos los costos.

Los costos varian según el tipo, y consisten en varios elementos, distintos.

Dichos costos incluyen:

MEDIDA DE LOS RECURSOR	UNIDAD DE MAQUINA.
Cada hora de lote	.250
Cada hora de procesador	870.000
Cada mil escrituras en disco	.400
Cada mil escrituras en cinta	.750
Cada mil lineas impresas	.800
Cada mil transmisiones de trabajo	.200
Cada hora conectada en linea	1.375

FACTORES DE COSTO.

la unidad de máquina indican la utilización relativa de los componentes de una computadora. Cada unidad de máquina se carga a razón de 1.20 dolares.

Para determinar el costo, se multiplica las unidades de máquina para esa actividad por el cargo unitario de máquina (ejemplo: Una hora de uso de una unidad central de proceso cuesta 670.000 unidades de máquina por 1.2, resultado $ 804.00).

Puede solicitarse servicio prioritario, se utilizan las horas/hombre, hombre/máquina, etc. y ajustarse dependiendo de la prioridad deseada de proceso.

Analizaremos el costo del sistema en horas/hombre, horas/pesos y en horas/dolares.

* COTIZAR EL SISTEMA EN HORAS/HOMBRE (M. N.).

Para el desarrollo de sistemas por hora existe un intervalo de pago de honorarios . Dicho pago es según la capacidad del programador (principiante, intermedio ó avanzado). La hora se valua en:

* $ 1,500.00 Hora/hombre ——————→ principiante
* $ 3,000.000 Hora/hombre ——————→ intermedio
* $ 5,000.000 - $ 20,000.00 Hora/hombre→ profesionista ó avanzado.

Tomando un nivel entre el intermedio y avanzado tenemos un costo total de horas/hombre de:

$ 3,000.00 hora/hombre

El sistema fue desarrollado en un lapso de 1 semana laboral es decir de lunes a viernes, multiplicando las horas por días tenemos que:

24 hrs. X 5 días ..... = 120 hrs.

Este es el total de horas a la semana, multpiplicando por el costo horas hombre tenemos:

120 hrs. x $3,000.00 .....= $360,000.00

Es decir que el costo del sistema en horas/hombre tiene un total de:

$ 360,000.00 horas/hombre	...(1)

- COTIZAR EL TIEMPO EN HORAS/HOMBRE (Dolares).

Del resultado anterior resta hacer un simple proceso para convertirse a dolares.

El dolar controlado tiene un costo de:

$ 1 dolar .....= $ 2,250.00*

* valor según fecha de 10 de julio de 1989.

Convirtiendo (1) a dolares:

$ 360,000.00 / $2,250.00 .....= $ 141.00

Es decir que el costo del sistema en horas/hombre en dolares tiene un total de:

$ 141.00 horas/hombre (Dolares)

Para determinar los costos estimados para un sistema requiere de la identificación de los elementos arriba mencionados, y que conforman el costo en su totalidad, es decir, costos directos y tangibles.

Son tangibles, ya que por alguna razón hubo salidas en efectivo, es decir gastos (diskettes, cinta, papel, libros, etc.).

Son directos porque el costo no se distribuye a terceras personas, ni mucho menos a mantenimiento de instalaciones usadas durante el proyecto.

VII.2.- BENEFICIOS DEL SISTEMA

Los beneficios son las ventajas que se obtienen de la instalación y utilización del mismo. Existen tres tipos de beneficios: tangibles o intangibles, Fijos o variables y directos o indirectos. Estas categorias no se excluyen en el presente trabajo, por lo tanto, se puede clasificar el beneficio dentro de lo tangible y lo directo. Puede ser fijo y variable, ya que la primera puede resultar si el sistema se compra y se paga por equipo de computo, el costo es fijo, no cambia, ya sea que el sistema se utilice mucho o poco, es decir, es único. Puede ser variable si el sistema se le da mantenimiento en periodos constantes, según la variación de las necesidades del usuario.

Dentro de los beneficios podemos captar lo siguiente, El sistema de diagnóstico, aunque aparentemente es sencillo, muy pocos lograrán realizar un sistema de este tipo en otro lenguaje.

Digo que es un sistema -aparentemente facil-, ya que realizado con otro lenguaje de programación sería más difícil y más complicado. (a excepción de PASCAL y lenguaje C), ya que su lógica es variable para todas las respuestas.

Ademas el lenguaje de Turbo Prolog pocos lo saben, y uno de los beneficios que más se encuentran en el *sistema experto de diagnóstico* es la recopilación de información de los expertos de la materia, es decir, que dicho sistema puede ser confiable.

• COSTO/BENEFICIO.

Analizando los costos en sotfware se puede decir que el sistema presenta un costo igual ó poco alto en comparación con otros programas de diagnóstico, pero cabe mencionar que el sistema es confiable y ademas por su lógica de programación puede expanderse a un nivel más alto, es decir, puede llegar a ser un sistema experto al cien por ciento de su capacidad.

Otro beneficio es que presenta una base de conocimiento bastante alta, dicho en otras palabras, es como si estuviera consultando a varios expertos en la materia. En comparación con los experto humanos, es que muchas veces no coiciden cual pueda ser la solución más optima.

Repitiendo una frase del libro de Turbo Prolog -programación avanzada- de Herbert Schildt, pag. 89, dice:

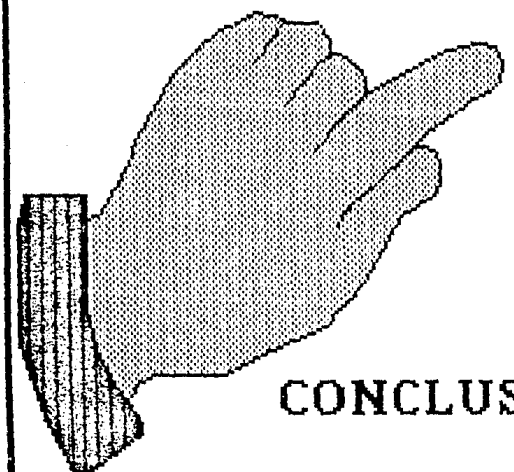
> [...la mayoria de los expertos humanos no saben lo que saben; los humanos no pueden hacer un volcado de memoria de la misma forma que una computadora. En consecuencia, puede ser dificil extraer toda la información necesaria. Ademas algúnos expertos estarán simplemente menos inclinados a perder tiempo diciendole todo lo que saben sobre la materia.].

Para beneficio de todos los que tienen un equipo PC este sistema presenta una forma de evaluar los posibles problemas de operación que puedan surgir tanto de hardware como de software.

En conclusión la información de la base de conocimiento concuerda consigo misma. Aunque no encontrará todos los problemas, si encontrará un 85 % de respuesta.

Sin embargo, analizando como está implementado la lógica de conocimiento, se puede alcanzar otros niveles de programación más compleja abarcando el 100 % de los problemas existentes para todos los nuevos equipos de computadoras personales.

PARTE V



CONCLUSIONES

CAPITULO VIII

VIII.- CONCLUSIONES GENERALES.

Dentro de las conclusiones generales se puede abarcar (en forma resumida) comentarios acerca de la computadora y sus periféricos y otros factores que ayudan para un máximo aprovechamiento del equipo.

VIII.1 RECOMENDACIONES GENERALES PARA EL BUEN FUNCIONAMIENTO DEL EQUIPO.

VIII.1.1 ASPECTOS SOBRE LA MICROCOMPUTADORA.

A grandes rasgos la microcomputadora personal (PC) está compuesto de tres partes principales:

* El teclado.
* La unidad base.
* El monitor.

En términos generales existen tres franjas en las que se encuentran ubicados los microprocesadores.

La franja baja comprende a todos aquellos llamados PC o XT, y tienen un microprocesador 8088/8086.

En la franja media se encuentran los llamados AT, máquinas que incluyen un microprocesador 80288/80286, es más veloz que las anteriores y pueden contener una unidad de disco flexible de hasta 1.2 MB.

Por último en la franja alta se encuentran las supermicros, mucho más veloces que las anteriores, con capacidades de memoria más grandes y un microprocesador 80388/80386.

EL TECLADO.

A través del teclado se introduce toda la información que se desea procesar. Se le denomina un dispositivo de entrada.

LA UNIDAD BASE.

La unidad base es la que contiene la unidad de proceso real en un módulo llamado tarjeta madre, también contiene la fuente de poder y las unidades de disco magnético.

EL MONITOR.

El monitor es la parte por la que la unidad central se comunica con el usuario. Se trata de un dispositivo de salida.

LAS IMPRESORAS.

Uno de los periféricos de mayor uso con los sistemas de computo son las impresoras. Con ellas se obtiene los reportes de todo tipo de trabajo por lo que vienen a resultar 'el cuello de la botella' de muchos centros de computo.

La impresora posee tres grupos perfectamente definidos. El grupo de impresión llamado '*cabeza de impresión*', el grupo de motores para el avance del papel y la cabeza, y el grupo electrónico que es la tarjeta de circuitos impreso que lleva el control de los dos grupos antes citados, así como la interfaz hacia el usuario.

Adicionalmente se tiene una serie de sensores que controlan la presencia o ausencia del papel, de un posible introductor de hojas sueltas, etc.

LOS GRAFICADORES.

Algunas veces, es necesario mostrar los resultados en forma gráfica (y en color), para esto se utilizan los plotters o graficadores.

A manera similar a las impresoras, en este tipo de periféricos existen tres grupos. La diferencia que existe es en el grupo de impresión de la imagen y las cintas es por impacto de la cabeza sobre la cinta. El graficador tiene cartuchos de tinta (o carretes de plumillas) que se abren o cierran de acuerdo al color requerido.

VIII.1.2 FACTORES PARA EL APROVECHAMIENTO DEL EQUIPO.

MEDIO AMBIENTE.

Aun cuando el microprocesador, sea 'personal', no requiere mayor cuidado que el de un mueble más en el interior de una oficina, es recomendable tener ciertas consideraciones para con ellas. En general se puede decir que el lugar debe ser confortable para la persona que lo opera y que el equipo sea adecuado también, cuando se habla de cuidados evidentemente se está hablando también de comidas, cigarros y café.

TABUS DE LA COMIDA, BEBIDA Y HUMO DEL CIGARRO.

Por lo que se refiere a la comida y bebidas, en su total ausencia lo que realmente importa, se dice que todos los líquidos son atraidos por el teclado, así como las migajas de pan, mientras que el humo y ceniza de cigarros son irresistiblemente atraidos por las unidades de disco.

En un escritorio el derrame de café se puede limpiar con una esponja, pero cuando éste cae al interior del teclado, hay que pensar en la asistencia de personal calificado ó hasta en un teclado nuevo. Pero... ¿ Que pararía si el derrame fluye hacia los diskettes?

El disco puede remplazarse con poco dinero, pero la información..., hay que considerar todo el trabajo y tiempo para reuperar la información (si es que no existe respaldo).

NECESIDADES DE VENTILACION.

Aunque la PC no tenga gran necesidad de ventilación , hay que recordar que es un dispositivo electrónico y que con algunas de sus partes generan una buena cantidad de calor. La PC deberá estar instalada en una area donde circule aire o que ésta no esté reestringida.

VENTILACION DE LA UNIDAD BASE

Para una ventilación adecuada , hay que asegurarse que cada ranura de opciones no ocupada exista una cubierta de metalo plastico ya que el aire circula desde la parte frontal del equipo y sale a traves del ventilador, y no deberá existir ninguna trayectoria adicional.

VENTILACION DEL MONITOR.

Evitar la tentación de usar la parte superior del monitor como un lugar para almacenar papeles . Las regillas que existen es esa parte no son decorativas; permiten que el aire caliente salga, de no ser así el monitor podría llegar a calentarse demasiado.

NECESIDADES DE VENTILACION.

Para la mayoría de los casos será suficiente una fuente de energía con tierra física y regulador. Si la computadora va a ser una parte de una estación de trabajo grande donde habrá impresoras, monitores y otros dispositivos relacionados, entonces se deberá verificar la capacidad de corriente eléctrica de las líneas de suministro.

La tierra física es un requisito que por lo general no se le da importancia suficiente y sin embargo ha sido la causa de daños irreparables en un buen número de equipos.

La tierra física deberá cumplir ciertos valores de resistencias óhmicas máximas entre las terminales de contacto, para que a la presencia de la alimentación la diferencia de potencial entre neutro y tierra física sea casi cero.

De acuerdo a las especificaciones, de los equipos de oficina operan sobre un rango de 115 V. + 10% lo que da una máxima de 127 Volts aproximadamente; sin embargo el suministro de energía en nuestro país varía entre 100 a 135 Voltios, lo que hace totalmente inoperante el rango de trabajo de los equipos sin la necesidad de un regulador de voltaje.

Es muy importante que las clavijas de los equipos que normalmente tienen tres posiciones sean insertadas en contactos de dos entradas y mucho menos cortada la terminal de tierra física. Una vez más, la tierra es para la protección del equipo y del usuario.

ELECTRICIDAD ESTATICA.

Cualquiera que camine sobre una alfombra en un día seco generará suficiente electricidad estática, y esta es suficiente para la computadora que recibe la descarga, pudiendo alterar o danar los componentes electrónicos.

CARPETAS ANTIESTATICAS.

En las areas con muy baja humedad y alfombrado deberá considerarse la adquisición de 'carpetas' antiestáticas para pisos y escritorios. Una caminata de 10 segundos sobre un piso alfombrado puede generar de 20,000 a 30,000 voltios de electricidad estática, y una desarga de 2,000 V. puede ocasionar la perdida de datos.

EL USUARIO.

Por lo que se refiere al usuario, la manera en que interviene para su máximo aprovechamiento, es llevado al cabo todas las recomendaciones anteriores. Muchas veces el usuario es la causa de que el equipo no funcione adecuadamente, ya que la mayoría de los casos es un inexperto sobre el funcionamiento del equipo.

VIII.1.3 EL SISTEMA OPERATIVO.

El sistema operativo en un equipo microprocesador es la interfaz que permite conectar al usuario con el sistema y ademas le proporciona las herramientas para el mantenimiento de los archivos y programas.

El sistema operativo de mayor uso en los equipos PC son los llamados MS-DOS, por sus siglas en ingles *MicroSoft Disk Operating System*, el cual está formado por tres archivos:

IBMDOS	oculto
IBMBIO	oculto
COMMAND.COM	contiene los comandos internos.

Es necesario aclarar que no todas las versiones son compatibles con todos los programas de aplicación que se usan debido a los cambios en los archivos ocultos. Por lo que se recomienda informarse si los programas que se usan pueden ser soportados por versiones posteriores del S. O. en uso.

Hasta la actualidad las versiones más comunes del MS-DOS son las que a continuación se mencionan:

Versión 2.11
Versión 3.10
Versión 3.20
Versión 3.30
Versión 4.01

CONTAMINACION DE SOPORTES MAGNETICOS, (Virus).

Muchos equipos PC se encuentran contaminados con un software específico denominado virus.

Aunque durante el presente trabajo no se menciona como funciona y como cuidarse del mismo, en esta parte se resume como trabaja y como podemos solucionar un problema en caso de virus, ya que el virus es un problema reciente de equipos PC y existe poca información sobre el tema.

El virus es una secuencia de instrucciones que entre otras cosas, se pueden propagar contamindo HDU's y diskettes de otros sistemas.

El virus típico, toma control del sistema operativo ya sea del HDU o diskette y permanece latente hasta que se cumplen ciertas condiciones luego de las cuales se hacen copias hacia otros archivos y discos.

De esta manera se hace la propagación de computadoras a computadoras y aún en redes. En la mayoría de los casos esta contaminación causa daños irreversibles.

Esta contaminación se inició con discos adquiridos en tiendas donde se venden programas *piratas* a bajo precio. El motivo aparente fue de castigar a los usuarios de este tipo de programas.

ALGUNOS TIPOS DE VIRUS.

Virus de Pakistan: Este virus produce una mezcla de la información de los archivos y no existe forma de recuperació.

Pelotita de Ping-Pong: Aparece en la pantalla haciendo un efecto de pelota que rebota en los bordes de la pantalla y sobre los caracteres. No se puede determinar cuando destruye la información en forma parcial o total de la información residente.

CONSTITUCION DEL VIRUS.

El virus está formado de lo siguiente:

1.- Código que instala el programa en memoria a través de un TRS (Terminate and State Resident) MS-DOS interrupt.

2.- Código que copia el programa en un diskette, altera un archivo, modifica el boot sector o contamina un archivo/programa actualmente en uso.

3.- Código que controla el número de veces que el virus se ha copiado sobre sí mismo.

4.- La activación del virus a través de varias condiciones: fecha, hora, número de acceso al HDU, etc.

PREVENCION DEL VIRUS.

De acuerdo a la experiencia, se pueden citar dos formas de prevenir la contaminación de virus, estos son:

+ Prevención por el usuario
+ Programas anti-virus.

Prevención por el usuario. La mayor parte de los virus se alojane en el boot sector y contaminan al equipo al momento de realizar la carga desde un sistema operativo contaminado que es el conducto principal de contaminación.

Para esto es importante seguir las siguientes reglas.

a) Si se tiene un sistema sin HDU, efectuar la carga desde un diskette dedicado exclusivamente para la carga del S.O.

b) Si se tiene un sistema con HDU, no efectuar jamás la carga de un diskette que no sea un original certificado. Es recomendable, ademas, en estos sistemas trabajar con subdirectorios por tipos de programas. De esta manera se reduce el riesgo de contaminación considerablemente debido a que la mayoría de los virus operan sobre el directorio raíz.

Algunas situaciones que comunmente indican la presencia de virus son las que a continuación se mencionan:

* La carga del programa se realiza en un tiempo superior al normal.

* Se verifica un número de acceso anormal al HDU, para determinado programa.

* Mensaje de error no usuales.

* Se verifican accesos a unidades cuando se supone que dichas unidades no deben activarse.

* Menor memoria disponible en el sistema.

* Programas o archivos que desaparecen.

Para esto se debe prohibirse correr programas de juegos de video, ya que a través de estos es donde se verifica el mayor indice de contaminación.

Como sugerencia, se puede hacer un respaldo regularmente de la información en los sistemas con disco duro.

PROGRAMAS ANTIVIRUS.

Existen dos categorias de programas anti-virus.

a) Programas que preveen la contaminación del sistema.

b) Programas que ayudan a identificar la presencia del virus.

Los primeros son programas residentes que controlan periodicamente el sistema bloqueando al virus antes de que éste pueda contaminarlo.

La segunda categoría opera en dos fases:

La primera vez que se ejecutan registran parámetros del sistema como el número de archivos ocultos. Es necesario que en esta primera fase el sistema esté descontaminado.

Posteriores ejecuciones comparan el estado actual del sistema con el registrado y detectan diferencias.

Dado que estos virus dependen de la información del ser humano, jamas existirá un programa capaz de detectar corregir todos los casos de virus que se presentan (al menos no en esta decada.), en la actualidad se estima que existen 200 tipos distintos de virus en el mundo de la informática.

CURA DEL VIRUS.

Una vez contaminado el sistema , la secuencia de restablecimiento por teclado no es una solución como tampoco lo es el desactivar el equipo y volverlo a encender si la carga del sistema se va a ser desde el mismo disco flexible o disco duro.

Cuando se haya verificado la presencia de virus en sistemas con disco duro, es necesario llevar los siguientes procedimientos:

1.- Si la contaminación no permite la carga del sistema, vaya al punto 'El sistema no permite la carga', si solamente se ha registrado presencia de virus. Pase al punto siguiente.

2.- Apague el equipo.

3.- Coloque un diskette de sistema operativo que no se encuentre contaminado en la unidad de diskette.

4.- Asegurese que la carga se realice normalmente.

5.- Haga un respaldo de todos los archivos de trabajo. (no ejecutables), ya que el virus puede estar en un archivo EXE, COM, SYS o BAT.

6.- Visualizar el contenido de todos los archivos BAT si alguno presenta líneas incorrectas. No haga respaldo de tales archivos.

7.- Llamar a la sección de informática de la empresa, para realizar la preparación del disco duro.

8.- Recrear la estructura del diskette.

9.- Reinstalar todos los archivos ejecutables tomandolos del paquete original.

10.- Efectuar la restauración de los archivos respaldados.

El sistema no permite la carga.

Si el sistema no permite la carga es muy problable que el dano se encuentre en la zona física de carga del disco, esto se corrige de la siguiente manera.

1.- Apagar el equipo.

2.- Hacer la carga desde diskette con un disco no contaminado.

3.- Ejecutar el comando: SYS C:

4.- Copiar el archivo COMMAND.COM del diskette mediante el siguiente comando: A>COPY COMMAND.COM C:\

5.- Verificar la carga desde el disco duro.

6.- Si persiste la carga llamar al servicio técnico del área de informática de su empresa.

VIII.1.4 DIAGNOSTICO DE FALLAS.

Si bién es cierto que todo lo que se ha comentado hasta aquí ayuda a que el equipo funcione satisfactoriamente, también es cierto que en algunas ocasiones presente algunos problemas que no son siempre tan 'graves'.

LISTA PRELIMINAR DE DIAGNOSTICO.

Cuando se lleve a cabo una lista de todo lo que pueda fallar en una computadora, no debemos olvidar de ponernos al principio de la lista. Enseguida existen varios dispositivos externos, luego internos con partes móviles. Después de eso debe estar el monitor. Finalmente, cuando todo esto falle -o mejor dicho, cuando hayamos verificado cuidadosamente que nada de eso falle- solo entonces debemos pensar que la falla es la computadora.

De tal forma esta lista se deberá ver de la siguiente forma:

1.- Errores humanos (no conectar la energía, colocar los conectores en posición equivocada, etc.).

2.- Dispositivos externos (impresora sin papel, liquidos sobre diskette, etc.).

3.- Dispositivos con partes móviles (unidades de disco flexibles y duro).

4.- Monitor (controles de contraste y/o brillantes).

5.- La computadora (algún componente electrónico).

VIII.1.5 RECOMENDACIONES FINALES PARA EL CUIDADO DEL EQUIPO.

Como recomendaciones finales para el cuidado del equipo, se citan a continuación algunos ejemplos.

La impresora.

Uno de los problemas principales de daños de las impresoras son el tipo se cinta que se usa. Deberá siempre preferirse por cintas original del fabricante, ya que no solamente sirve para que los caracteres impresos se vean claros sino que del material de que está hecho el soporte entintado depende mucho la vida del módulo de impresión.

Jamas deberán usarse cintas recicladas. Una cinta reciclada es aquella que mediante un proceso de re-entintado vuelve a venderse para su uso. Estas cintas siempre serán las causas de que una cabeza de impresión se dañe.

Deberá evitarse el avance del papel mediante la perilla, estando el equipo encendido, y jamas deberá detenerse el avance del papel cuando éste está controlado por la impresora de manera automática. Cualquiera de las dos situaciones anteriores provocará la rotura de la perilla.

El siministro de papel deberá estar libre de cualquier obstrucción.

Los discos.

Los discos flexibles o diskettes requieren un cuidado muy grande. Siempre que no se usen deberán estar dentro de su cubierta de PVC.

Cuando se manipulen evitar usar clips o grapas sobre la funda. No escribir sobre ellos, puede provocar marcas sobre el propio disco y hacerlo inutilizable.

Evite el abrir el acceso a la unidad de discos cuando éste se encuentra activo, la actividad se está llevando a cabo cuando el indicador frontal de la unidad está iluminado (LED encendido).

El monitor.

El monitor (o pantalla) es un dispositivo que requiere de mayor atención dentro del ambiente de trabajo.

Procurar no colocarlo cerca de ventanas o lugares de mucha iluminación ya que esto obliga a darle mayor intensidad disminuyendo con ello la vida útil del mismo.

Si se trabaja con hojas de cálculo como Lotus, disminuya al mínimo los controles de intensidad y contraste, esto es si por alguna razón tiene que dejar el equipo solo por tiempos largos ya que de lo contrario puede quedar marcado el fósforo de la pantalla.

Como última recomendación diré que cualquier medida adicional no será por demas y si redundará en un mayor tiempo de vida para el equipo y mayor servicio.

El determinar con certeza la falla podrá permitir al usuario proporcionar mayor información al personal de servicio y en no pocas ocasiones corregir el problema.

¿Que se requiere para poder hacer un buén diagnóstico de la falla? En realidad ningún estudio especializado, solamente el sentido común.

Se debe de pensar que cualquier agregado que se tenga que hacer al microcomputador deberá ser hecho por un especialista ya que implica una tarea en el interior del equipo. Un agregado mal puesto puede ser causa de problemas como por ejemplo que el sistemas no opere correctamente de una manera total.

El objetivo que se logró en este trabajo, fué el de desarrollar un sistema experto capaz de detectar anomalías generales en el funcionamiento de computadoras personales y sus compatibles. Dichas anomalías comprenden el monitor, CPU, teclado, Periféricos, etc.

El sistema fué pensado exclusivamente para operar a modo de consulta (como se definió al inició del trabajo), de tal manera que si el usuario llega a tener problemas de operación en su equipo, el sistema experto lo puede ayudar a detectar la falla; esto a traves de menús que presenta dicho sistema.

La finalidad de hacer este sistema es el aportar una ayuda sencilla, entendible, confiable y rápida sobre problemas de operación que presenten éstas máquinas en cualquier empresa, ó incluso mismo en escuelas, ya que los alumnos a igual que los empleados se van familiarizando con los problemas, a tal grado que entenderan como utilizar el equipo adecuadamente.

El presente trabajo titulado *Sistema experto para diagnóstico de Fallas en equipo PC y compatibles*, se conluye que va dirigido a todas aquellas personas que estén -de alguna manera- relacionados en el ambiente informático.

Tal vez puedan existir personal con un alto nivel de preparación que vea el trabajo (externo) como algo demasiadamente sencillo. (de hecho yo lo veo así sin ser ese tipo de personas); la cuestión es que como *software comerciable* no sería cien por ciento explotable (tal vez un setenta % sí), pero como primera versión, es muy aceptable.

Al mencionar que es acepatble, me refiero a que la lógica interna del programa se puede ampliar, y esto es una de las ventajas que presentan los *sistemas expertos*, ya que los SE son programas de computadoras que imitan a un experto humano. Si un experto humano puede ampliar sus campos de conocimientos, ¿porqué no este sistema?.

Este sistema cumple con los requisitos -como el sistema de diagnóstico experto de medicina-, para ser manejados en cualquier parte donde puedan existir computadoras.

El *Diag_pc*, (casi lo autonombré -*diag*nóstico de *pc*-), es un programa realizado en un lenguaje de la quinta generación como lo es el TURBO PROLOG.

¿Porqué el Turbo Prolog?. Los lenguajes de computadoras son raramente buenos para todos los tipos de problemas. FORTRAN, es usado principalmente por los científicos y los matemáticos para fines de cálculos y simulación; COBOL es usado principalmente en el mundo comercial. PASCAL y C son lenguajes estructurados para fines especiales de diseño y logica general, pero en su lugar TURBO PROLOG está diseñado para manejar problemas lógicos (es decir, problemas en los que se necesitan tomar desiciones de una forma ordenada) cosa que PASCAL y C no lo presentan.

Turbo Prolog, intenta hacer que la computadora *razone* la forma de encontrar una solución, y es adecuado para diferentes tipos de problemas de *inteligencia artificial*. Los más significativos de éstos son los *sistemas expertos* y *el procesamiento de lenguaje natural*. Ademas de que Turbo Prolog como lenguaje de quinta generación irá evolucionando para tener otras aplicaciones en el campo de la IA, así como sus costos y mantenimiento serán más bajos, ya que no se necesitará mucho de programación convensional, sino de lógica.

El sistema experto *Diag_pc*, presenta información (esto es, una base de datos) y una herramienta para comprender las preguntas y encontrar las respuestas correctas a ciertas cuestiones, examinando las bases de datos (es decir, el motor de *inferencia*).

El trabajo fué aplicado directamente a computadoras personales IBM, denominadas también PC's y sus compatibles, ya que son equipos que más se usan en el mercado, a comparación de otras marcas de computadoras personales como COMODORE, RADIO SHACK, APPLE entre otras.

Unos casos típicos de aplicación de este tipo de programas se encuentran en el campo de la medicina (USA desde 1979 en MEXICO a partir de 1987-88), en donde a las computadoras se les hacen preguntas para que diagnostiquen enfermedades. En Pemex y el IMP -Instituto Mexicano del Petroleo-, existían programas rutinarios desde 1975 hechos en Fortran para detectar fallas en las perforaciones en plataformas y yacimientos. A partir de 1987 estas empresas adquirieron equipo de computo (VAX, DG-1000, DG-2000, UNIVAX -UNISYS-, etc), para fines de manejar información y evitar errores en las zonas petroleras.

Para esto se usaron varios tipos de programas de sistemas expertos. Entre toda esa selección de programas y paquetes, me llegó a la mano toda la información para sistemas expertos, y es ahí donde me encontré con Turbo Prolog. En ese momento mi idea fué el hacer un sistema experto pero enfocado a mi area como lo es la computación.

La facilidad que presenta Turbo Prolog para poder realizar el *Diag_pc*, es que tiene incorporadas estructuras para la creación de base de datos y tiene un motor de inferencia listo para ejecutar. Todo lo que hubo de hacerse, es decirle al programa las reglas en las que ha de basarse y éste encuentra el camino para dar información apropiada.

Si yo lo hubiera hecho con otros lenguajes de programación, me hubiera encontrado con problemas como escribir las reglas, especificar el camino y generalmente, tendría que haber trabajado en un nivel mucho más detallado para crear el motor de inferencia.

Ademas, la base de datos de Turbo Prolog está construida con las mismas estructuras que se utilizan para escribir las reglas de una manera facil de ejecutarse.

La lógica de *Diag_pc*, se basa en una serie de manipulaciones lógicas -valgame el pleonasmo-, conocida como *lógica de predicados ó cálculo proposicional*. Por ejemplo las siguientes sentencias son hechos:

LA GENTE CON ZAPATOS DE BASKETBALL JUEGAN AL BASKETBALL

LA PERSONA NO. 1 TIENE ZAPATOS DE BASKETBALL

Lo que puede inferirse, figurarse, deducirse, calcularse ó determinarse de estos hechos es que la persona No.1 juega al basketball.

El turbo prolog hace que la computadora maneje la parte de inferir. De hecho el sistema experto tiene un motor de inferencia (de Turbo Prolog) incorporado que automáticamente busca los hechos y construye ó prueba conclusiones lógicas. Así dado, el problema puede parecerse trivial, pero si se tuviera una base de datos técnica almacenada con miles de hechos y reglas, no sería práctico entregar esa información a un humano para obtener conclusiones ó respuestas rápidas.

En otras palabras el trabajo vale más (al menos siento yo) en su forma lógica y en su manera de trabajar, que en su forma externa de responder.

IX. CONTINUIDAD (OTROS HORIZONTES A EXPLORAR).

El presente trabajo -como ya se mencionó anteriormente- tiene como ventaja de que puede ampliarse tanto en su base de conocimiento como en su estructura lógica.

Sin embargo lo más importante de éste sistema experto, es que como está hecho en un lenguaje de programación como es el Turbo Prolog, pueden hacerse unos cambios en su lógica, para que ya no funcione como técnica de -búsqueda de hacia adelante y hacia atras-, y mucho menos en modo de consulta en donde el usuario debe de pulsar la opción al tipo de problema que tiene, sino todo lo contrario.

Como ya se sabe el campo de la Inteligencia Artificial (IA), presenta varias ramas -robótica, Sistemas expertos, Reconocimiento de imágenes, etc-. Pues bién, este sistema experto puede convertirse en un programa de Lenguaje Natural. Como puede ser esto?. Con las mismas lógicas que presenta la programación en Turbo Prolog puede hacerse que *Diag_pc* sea como si estuvieramos con un experto aún más humano, ya que el lenguaje natural tiene como características pricipales -y como su nombre lo dice-, el simular una persona hablando, ejemplo:

SISTEMA:	**QUE PROBLEMA PRESENTA SU EQUIPO COMPATIBLE ?**
USUARIO:	**TIENE FALLAS EN EL TECLADO**
SISTEMA:	**QUE TIPO DE FALLAS TIENE EL TECLADO ?**
USUARIO:	**NO CORRESPONDE LAS TECLAS QUE SE PULSAN ?**
SISTEMA:	**DESPLIEGA OTRA COSA QUE USTED NO PULSA ?**
USUARIO:	**SI**
SISTEMA:	**YA VERIFICO BIEN LOS RESORTES DE CADA TECLA ?**
USUARIO:	**SI**

```
SISTEMA:        EL TECLADO CORRESPONDE A LA MAQUINA ?
USUARIO:        SI
SISTEMA:        VEO QUE SU PROBLEMA ES MAS COMPLICADO DE LO
                QUE ESPERABA.
SISTEMA:        YA USO LOS COMANDOS DE -KEYBUK, KEYSPL- ?
USUARIO:        NO
SISTEMA:        QUE ESPERA PARA USARLOS !!
SISTEMA:        DESEA ALGUNA OTRA CONSULTA ?
USUARIO:        NO
SISTEMA:        ME DESCONECTO ?
USUARIO:        SI
SISTEMA:        HASTA LUEGO...
```

Así como el sistema presenta un diálogo confiable, así puede ser los proximos sistemas de este tipo. de Hecho si se conecta un sintetizador de voz se puede hablar con ella si meter las manos en el teclado.

Este tipo de sistema de lenguaje natural, lo presenta Turbo Prolog en uno de sus discos, -examples-, como lo es el programa llamado *Geobase* que contiene toda la información posible de todo la situación geográfica de los EEUU.

De hecho el campo de la Inteligencia Artificial es tan grande que no solo puede abarcarse esa área, teniendo una buena imaginación se puede lograr eso y mucho más, y lograr que esta área tenga más continuidad y rebasar fronteras.

APENDICE -A-

Standard ASCII Character Set

DECIMAL	OCTAL	HEX	ASCII SYMBOL	COMMENTS
0	000	00	↑@	NUL
1	001	01	↑A	SOH
2	002	02	↑B	STX
3	003	03	↑C	ETX
4	004	04	↑D	EOT
5	005	05	↑E	ENQ
6	006	06	↑F	ACK
7	007	07	↑G	BEL
8	010	08	↑H	BS
9	011	09	↑I	TAB
10	012	0A	↑J	
11	013	0B	↑K	
12	014	0C	↑L	
13	015	0D	↑M	
14	016	0E	↑N	SO
15	017	0F	↑O	SI
16	020	10	↑P	DLE
17	021	11	↑Q	DC1
18	022	12	↑R	DC2
19	023	13	↑S	DC3
20	024	14	↑T	DC4
21	025	15	↑U	NAK
22	026	16	↑V	SYN
23	027	17	↑W	ETB
24	030	18	↑X	CAN
25	031	19	↑Y	EM
26	032	1A	↑Z	SUB
27	033	1B	ESC	ESCAPE
28	034	1C	↑\	FS
29	035	1D	↑]	GS
30	036	1E	↑↑	RS
31	037	1F	↑—	US
32	040	20		SPACE
33	041	21	!	
34	042	22	"	
35	043	23	#	
36	044	24	$	
37	045	25	%	
38	046	26	&	
39	047	27	'	
40	050	28	(	
41	051	29	)	
42	052	2A	*	
43	053	2B	+	
44	054	2C	,	
45	055	2D	-	
46	056	2E	.	
47	057	2F	/	
48	060	30	0	
49	061	31	1	
50	062	32	2	
51	063	33	3	
52	064	34	4	
53	065	35	5	
54	066	36	6	
55	067	37	7	
56	070	38	8	
57	071	39	9	
58	072	3A	:	
59	073	3B	;	
60	074	3C	<	
61	075	3D	=	
62	076	3E	>	
63	077	3F	?	
64	100	40	@	
65	101	41	A	
66	102	42	B	
67	103	43	C	
68	104	44	D	
69	105	45	E	
70	106	46	F	
71	107	47	G	
72	110	48	H	
73	111	49	I	
74	112	4A	J	
75	113	4B	K	
76	114	4C	L	
77	115	4D	M	
78	116	4E	N	
79	117	4F	O	
80	120	50	P	
81	121	51	Q	
82	122	52	R	
83	123	53	S	
84	124	54	T	
85	125	55	U	
86	126	56	V	
87	127	57	W	
88	130	58	X	
89	131	59	Y	
90	132	5A	Z	
91	133	5B	[	
92	134	5C	\	
93	135	5D	]	
94	136	5E	↑^	
95	137	5F	← _	
96	140	60	`	
97	141	61	a	
98	142	62	b	
99	143	63	c	
100	144	64	d	
101	145	65	e	
102	146	66	f	
103	147	67	g	
104	150	68	h	
105	151	69	i	
106	152	6A	j	
107	153	6B	k	
108	154	6C	l	
109	155	6D	m	
110	156	6E	n	
111	157	6F	o	
112	160	70	p	
113	161	71	q	
114	162	72	r	
115	163	73	s	
116	164	74	t	
117	165	75	u	
118	166	76	v	
119	167	77	w	
120	170	78	x	
121	171	79	y	
122	172	7A	z	
123	173	7B	{	
124	174	7C	|	
125	175	7D	}	
126	176	7E	~	
127	177	7F	DEL	

B I B L I O G R A F I A

CAPITULO I

1.- SACERDOTI E. D. 'A STRUCTURE FOR PLANS AND BEHAVIOR'
Elsevier, NY, 1977.

2.- SAMUEL A. L. 'INTELIGENCE ARTIFICIAL'
1152, IJCHI-83, 1983.

3.- FEINGENBAUM E. A. and FELDMAN J. 'COMPUTERS AND THOUGHT'
MC-Graw-Hill , N. Y., 1963.

4.- FEINGENBAUM E. A. 'KNOWLEDGE ENGINEERING FOR THE 80'S'
Computer Science Dept., Stanford University 1982.

5.- MICHIE DONALD 'KNOWLEDGE-BASED SYSTEMS'
Jan. 1980.

6.- STEFIK M. J. 'THE ORGANIZATION OF EXPERT SYSTEM'
A perspective tutorial.

7. CHAPA V. SERGIO V. 'ARQUITECTURA DE SISTEMAS EXPERTOS'
Centro de Investigación de Estudios Avanzados IPN
Jun. 1984.

CAPITULO II

1.- CHAPA V. SERGIO V. 'ARQUITECTURA DE SISTEMAS EXPERTOS'
Centro de Investigación de Estudios Avanzados IPN
Jun. 1984.

2.- HEBERT SCHILDT 'TURBO PROLOG -Programación Avanzada-'
Mc. Graw-Hill, México 19828.

3.- STEFIK M. J. 'UNDERSTAND EXPERT SYSTEM'
Borland Inc., N. Y., 1986

CAPITULO III

1.- W. F. CLOSCKIN, C. S. MELISH 'PROGRAMING IN TURBO PROLOG'
Borland Inc., N.Y. 1984.

2.- PHILLIP R. ROBINSON 'APLIQUE EL TURBO PROLOG'
MC. Graw-Hill, México, 1988

CAPITULO IV

1.- RUTH ASHLEY, J. N. HDEZ. 'COMPUTADORA PERSONAL -Sistema Operativo en disco DOS'
Limusa, México, 1986.

2.- GRAHAM C. 'IBM/PC GUIA DEL IMB/PC (DOS 2.0) XT'
MC. Graw-Hill, México, 1988.

3.- SACHS T. 'EL IBM/PC'
MC. Graw-Hill, México, 1988.

4.- MORGAN 'INTRODUCCION AL MICROPROCESADOR 8086/8088'
MC. Graw-Hill, México, 1987.

5.- J. L. HDEZ. 'IBM-PC + MS-DOS -Principios de Operación-'
Paraninfo S.A., Madrid España, 1988.

CAPITULO V

1.- STEFIK M. J. 'UNDERSTAND EXPERT SYSTEM'
Borland Inc., N. Y., 1986

2.- HEBERT SCHILDT 'TURBO PROLOG -Programación Avanzada-'
Mc. Graw-Hill, México 1988.

CAPITULO VI

1.- PHILLIP R. ROBINSON 'APLIQUE EL TURBO PROLOG'
MC. Graw-Hill, México, 1988

2.- HEBERT SCHILDT 'TURBO PROLOG -Programación Avanzada-'
Mc. Graw-Hill, México 19828.

CAPITULO VII

1.- 'INGENIERIA DEL SOFTWARE'
Manual del Centro de Estudios Avanzados
Arcos de Belen 10, México D. F.

2.- HEBERT SCHILDT 'TURBO PROLOG -Programación Avanzada-'
Mc. Graw-Hill, México 19828.

CAPITULO VIII

1.- 'SEMINARIO A USUARIOS DE MICROCOMPUTADORAS'
PEMEX (Unidad de Informática y Difusión)
México, 1989.