PROLOG COMO HERRAMIENTA PARA CONSTRUIR SISTEMAS EXPERTOS

Myriam Blanco Vargas
Profesora Asociada
Departamento de Información y Sistemas
Universidad del Valle
Cali - Colombia

INTRODUCCION

Hace dos años participé en el **V ENCUENTRO NACIONAL DE INFORMATICA** con la ponencia los Sistemas Expertos en el sector Salud. En aquella ocasión mi objetivo era presentar los sistemas expertos como una tecnología de gran utilidad en los problemas de diagnóstico.

Durante estos dos años he estado trabajando con **PROLOG**, uno de los lenguajes tradicionales en inteligencia artificial y hoy me gustaría poder transmitirles a ustedes, parte de mi experiencia en el empleo del mismo como una herramienta para construir sistemas expertos.

Esta experiencia proviene en gran parte de mi trabajo en el proyecto de investigación *Desarrollo en Prolog de un entorno generalizado para la construcción de sistemas expertos,* el cual se enmarca dentro de la línea de investigación que en inteligencia artificial y sistemas expertos, se desarrollan actualmente en la Universidad del Valle.

Otros de los proyectos que se realizan en esta misma área, es el proyecto: *Construcción de un entorno generalizado para el Desarrollo de sistemas expertos consultores,* a cargo del profesor **José Antonio Abadía**.

Los puntos que presentaré a continuación son los siguientes:

* Introducción sobre sistemas expertos y herramientas para construirlos.
* ENDI: un entorno generalizado para construir sistemas expertos desarrollado en UNIVALLE.
* PROLOG: herramienta para construir sistemas expertos.
* Conclusiones.

1. SISTEMAS EXPERTOS

La tecnología de los Sistemas Expertos tiene como finalidad almacenar los conocimientos de un experto en un computador y emplearlos en la solución de problemas.

Una definición formal (NAYLOR, 1985) de Sistemas Expertos es la siguiente:

> *Se considera que un sistema experto es la incorporación en un ordenador de un componente basado en conocimiento que se obtiene a partir de la función de procesamiento. Una característica adicional que es deseable, que para muchos es fundamental, es la capacidad del sistema para, bajo demanda, JUSTIFICAR SU PROPIA LINEA DE RAZONAMIENTO de una forma inmediatamente inteligible para el que pregunta. El estilo de programación que se adopta para conseguir estas características es la PROGRAMACION BASADA EN REGLAS".*

(Definición formal de Sistemas Expertos aprobada por el comité del grupo especialista en sistemas expertos de la British Computer Society).

> *La palabra EXPERTO hace referencia a un individuo, el cual es ampliamente reconocido para solucionar determinado tipo de problemas, los cuales, otras personas no podrían solucionar tan efectiva y eficazmente. Para producir una mayor claridad se definió el término "SISTEMAS BASADOS EN CONOCIMIENTOS" pues es posible construir sistemas muy útiles e importantes, sin que ellos representen conocimientos de un experto.*

Se ha propuesto que el término SISTEMAS EXPERTOS sea empleado para sistemas basados efectivamente en conocimientos de expertos o especialistas, y el término SISTEMAS BASADOS EN CONOCIMIENTOS para sistemas que no necesariamente se apoyan en conocimientos de expertos.

Los componentes básicos de un sistema experto son: **(figura 1)**

El MOTOR DE INFERENCIA O MECANISMO DE INFERENCIA (Inference Engine).

El mecanismo de inferencia es un interpretador de reglas, estructuras o cláusulas, el cual contiene la estrategia de búsqueda de solución.

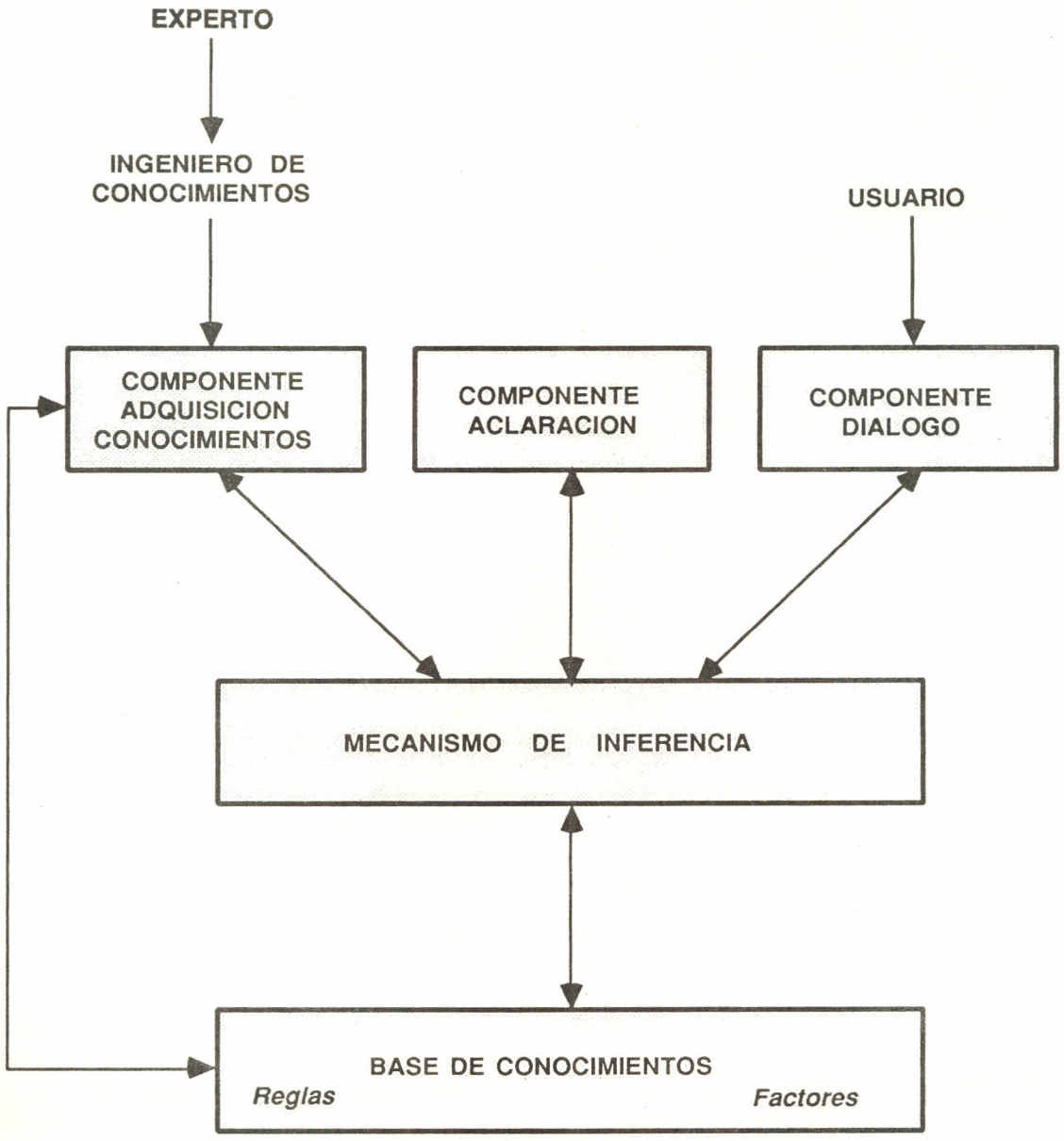

FIGURA 1. COMPONENTES DE UN SISTEMA EXPERTO

- LA BASE DE CONOCIMIENTOS (Knowledge Base)

La base de conocimientos contiene los conocimientos asociados al problema, representados a través de un esquema especial.

En el caso particular de un entorno no existe dicha base de conocimientos, hasta tanto el usuario no introduzca el conocimiento a través del componente de adquisición diseñado para tal fin.

La intersección del sistema con el usuario es representada a través del COMPONENTE DE DIALOGO Y EL COMPONENTE DE ACLARACION, el cual es comparable con la capacidad del hombre para sustentar sus decisiones.

El ingeniero de conocimientos **(Knowledge Engineer)** es alguien, quien se ha especializado en analizar problemas, juntar conocimientos y desarrollar sistemas de conocimientos.

Normalmente implica formación en el campo de la informática o investigaciones en Inteligencia Artificial a lo que se añade experiencias en la realización de Sistemas Expertos.

El campo de aplicación de los Sistemas Expertos es muy amplio y variado: diagnóstico (médico y técnico), planeación, asesoría, enseñanza, entrenamiento entre otros.

Los primeros sistemas expertos fueron escritos directamente en lenguajes de programación simbólica como Lisp, pero actualmente la mayoría son desarrollados con la ayuda de software especialmente diseñado para tal fin.

Este software son los denominados "shells", conchas o entornos y su objetivo es facilitar el desarrollo de los sistemas expertos; disminuir sus costos.

2. HERRAMIENTAS PARA CONSTRUIR SISTEMAS EXPERTOS

Como herramientas para construir sistemas expertos se consideran aquellos recursos de programación que faciliten el desarrollo de Sistemas Expertos o de conocimientos. Comprende desde herramientas muy específicas como los entornos hasta los lenguajes de propósito general. **(figura 2)**

2.1 LENGUAJES DE PROGRAMACION O ENTORNOS GENERALES

Los lenguajes pueden ser clasificados en "imperativos o procedurales" y "declarativos".

Aquellos lenguajes como Basic, Fortran, Pascal, Cobol, etc. en los cuales debe definirse la forma como debe buscarse la solución, se denominan "imperativos".

FIGURA 2. HERRAMIENTAS PARA CONSTRUIR SISTEMAS EXPERTOS

Estos lenguajes funcionan muy bien cuando se puede establecer de manera previa el algoritmo que resuelve el problema. Su objetivo es el procesamiento de datos. Los procesos están totalmente definidos.

En los lenguajes "declarativos" los programas se construyen con base en definiciones que describen la relación entre los elementos que se están manipulando. Cuando se ejecuta el programa, el procesador utiliza las definiciones para responder satisfactoriamente a una pregunta. Ellos se fundamentan en los conceptos de lógica formal y procesamiento de expresiones y no de datos. Lenguajes de este tipo son Lisp y Prolog.

Aún cuando se han desarrollado Sistemas Expertos con lenguajes imperativos, no son estos los más adecuados para escribir programas que simulen comportamiento lógico.

Las aplicaciones de inteligencia artificial y en particular los Sistemas Expertos exigen poder simular la inteligencia, siendo necesario relacionar entre si libre y simultáneamente un grupo de elementos.

Como *"lenguajes de alto nivel"* puede catalogarse el Lisp, lenguajes de programación simbólica y pueden también incluirse algunos lenguajes imperativos en los cuales se han realizado algunos desarrollos como Fortran, Pascal y Basic.

Prolog y OPSS son clasificados por algunos como lenguajes y por otros como ambientes de programación. Son menos flexibles que el *lisp*, pero con enfoques no tan específicos como el de los entornos. Lo mismo sucede con el *interlisp* el cual es una versión del lisp, pero con una cantidad de rutinas ya definidas.

2.2 AMBIENTES DE PROGRAMACION

En general incluye software que permite mezclar diversas formas de programación desde el punto de vista de representación del conocimiento y de diseño de sistemas de deducción, con lo cual se puede enfrentar una gama más amplia de situaciones con esquemas de razonamiento de representación del conocimiento no tan particulares como en los entornos o *"shells"*.

KEE (Knowledge Engineering Environment) pertenece al grupo de los grandes híbridos, diseñados para el desarrollo de sistemas expertos con más de 1000 reglas y no para una específica forma de representación del conocimiento como es el caso de los entornos EMYCIN, AL/X y otros.

2.3 ENTORNOS ESPECIFICOS O SHELLS

Son abstracciones de aplicaciones ya realizadas que facilitan en alto grado el desarrollo de Sistemas Expertos para situaciones específicas.

La mayoría de los entornos específicos proceden de sistemas ya instalados. De ellos el que más influencia ha tenido, ha sido *MYCIN,* del que se derivó como entorno abstracto de *EMYCIN,* con base en el cual se construyen, entre otros los sistemas *PUFF y SACON.*

Basado en Prospector, sistemas expertos para la localización de minerales, se desarrolló *AL/X.*

Aprender a usar los entornos no es más difícil que conocer un nuevo lenguaje de programación, pero para hacerlo con éxito es necesario tener conocimientos generales sobre conceptos básicos en inteligencia artificial tal como:

* Arquitectura de un sistema experto
* Limitación de estos sistemas
* Método de representación del conocimiento
* Problemas más adecuados para la tecnología de sistemas expertos.

Estas herramientas corren en una variedad de computadores, desde máquinas para procesamiento simbólico, hasta microcomputadores, ofreciendo una gama amplia de capacidades y utilidades que faciliten su uso.

Las herramientas orientadas a micros son simples y es fácil entender como usarlas para construir un sistema. Ellos tienden a representar el conocimiento de sistemas de reglas IF THEN; usan solamente un sistema de inferencia y soportan mecanismos de control muy limitados.

Algunos especifican en su sistema de aclaración, el método de razonamiento utilizado para resolver un problema.

3. PROLOG (Progamming Logic Language)

Fue desarrollado alrededor de los años 70 por **Alain Colmerauer** y sus colegas de la Universidad de Marcella. Prolog se convirtió rápidamente en el lenguaje principal para la inteligencia artificial en Europa, mientras que LIPS era empleado principalmente en Estados Unidos.

El PROLOG procede de una etapa de mayor desarrollo de las técnicas basadas en inteligencia artificial. Mientras que el LISP fue desarrollado a la medida de los pro-

blemas de los primeros tiempos en que los programas inteligentes se planteaban como búsqueda general.

El PROLOG es una consecuencia del desarrollo de las técnicas de deducción automáticas y por tanto incorpora una mayor cantidad de teoría que el LISP, aunque éste sea más general.

El PROLOG es un lenguaje "declarativo" en el cual hechos y reglas se definen sobre símbolos específicos y luego se formulan preguntas sobre si un objetivo concreto se deduce lógicamente de estos hechos / reglas. El mecanismo que realiza la comprobación es parte del mismo PROLOG.

El PROLOG es el lenguaje elegido en el Japón para el proyecto de computadores de quinta generación.

Este lenguaje de programación es hasta la fecha la mejor implementación para programación lógica. Simultáneamente es su sintaxis menos compleja que la mayoría de los lenguajes.Es un lenguaje orientado a problemas que involucran objetos y relaciones entre los mismos.

La descripción de un programa en PROLOG se diferencia notablemente de la de un programa escrito en un lenguaje tradicional como FORTRAN o COBOL.

Un programa en PROLOG consiste básicamente de:

- Declaración de FACTORES asociados a objetos y sus relaciones.

- Definición de REGLAS.

A continuación se presenta en PASCAL y PROLOG la descripción de un programa que lee números y prueba su validez. El número será válido, si se encuentra entre los rangos (0 ,3) y (5 ,10).

PROGRAMA EN PASCAL

```
PROGRAM ejemplo (Input, Output);

CONST     inf-1 = 0;          sup-1 =3;
          inf-2 = 5;          sup-2 = 1
TYPE      si - no = (si,no) ;
VAR    valor    : INTERG
       valido   : BOOLEAN ;
       respues: si-no
BEGIN
   REPEAT
   WRITE  (Número:  ) ;
   READLN  (valor) ;
   valido:= ((valor › inf-1) AND (valor › sup-1 OR
            (valor › inf-2) AND (valor › sup-2)) ;
   IF    valido
         THEN WRITELN  ('valor valido')
         ELSE WRITELN  ('valor no valido' ) ;
   WRITE  ('Mas números? Escriba si o no:  );
         READLN (respues) ;
   UNTIL respues = no ;
END
```

PROGRAMA EN PROLOG

```
rango - posible  (0,  3).
rango - posible  (5,10).

valor - valido (x): -
     rango-posible  (Inf, Sup),  Inf < x,  x <Sup.
```

En este caso, deberá formularse la pregunta, con el número cuya validez se desee conocer. Por ejemplo:

```
? valor - valido(6)
```

PROLOG responderá en forma afirmativa.

Entre las características propias del PROLOG, merecen destacarse las siguientes:

* Posee un mecanismo propio de inferencia conocido como encadenamiento hacia atrás.

* El orden de las reglas determina la forma como se realiza el encadenamiento.

* Permite especificar una tarea como un conjunto de restricciones que deben satisfacerse y no requiere que se especifique la forma como se realiza dicha tarea.

* Permite programar en forma declarativa, proporcionando al sistema los datos necesarios para que éste opere de forma procedimental.

* Ofrece el manejo de listas y revisión en forma natural.

4. ENDI (ENTORNO GENERALIZADO PARA CONSTRUCCION DE SISTEMAS EXPERTOS)

ENDI es un entorno generalizado o shell para el desarrollo de sistemas expertos en diagnósticos, el cual permite que usuarios no familiarizados con las técnicas de sistemas expertos, construyan sistemas expertos en

diagnósticos sin tener que recurrir a la programación directa. ENDI ofrece dos alternativas para representar el conocimiento del experto.

En el primer caso el conocimiento es representado mediante un árbol binario de síntomas o factores el cual se va abriendo (nodos intermedios) hasta determinar exactamente la causa de la falla (nodo final) y poder formular una recomendación. **(figura 3)**

En el segundo caso, **(figura 4)** los síntomas pueden tomar diferentes valores es decir se tienen opciones múltiples y el árbol no necesariamente es binario.

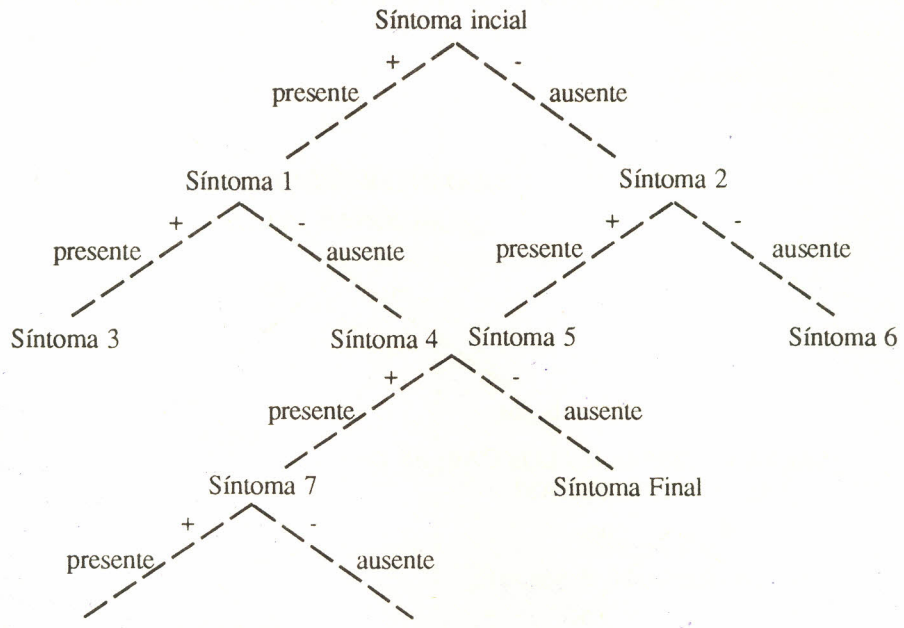

FIGURA 3. REPRESENTACION DEL CONOCIMIENTO. MODELO TIPO 1

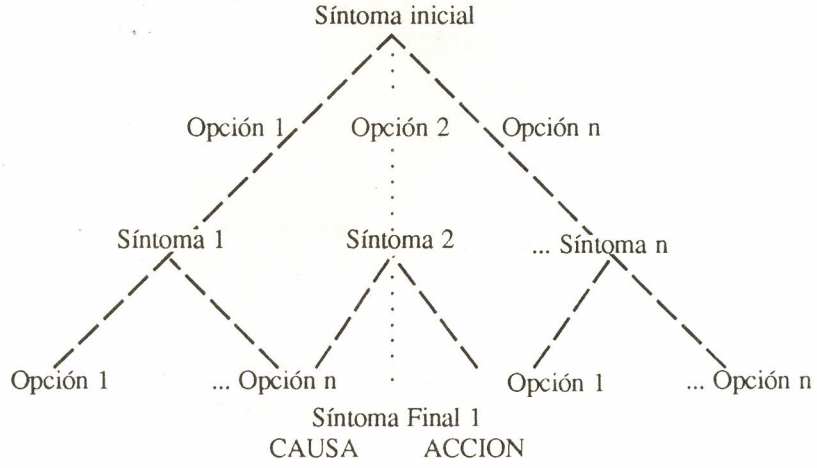

FIGURA 4. REPRESENTACION DEL CONOCIMIENTO. MODELO TIPO 2

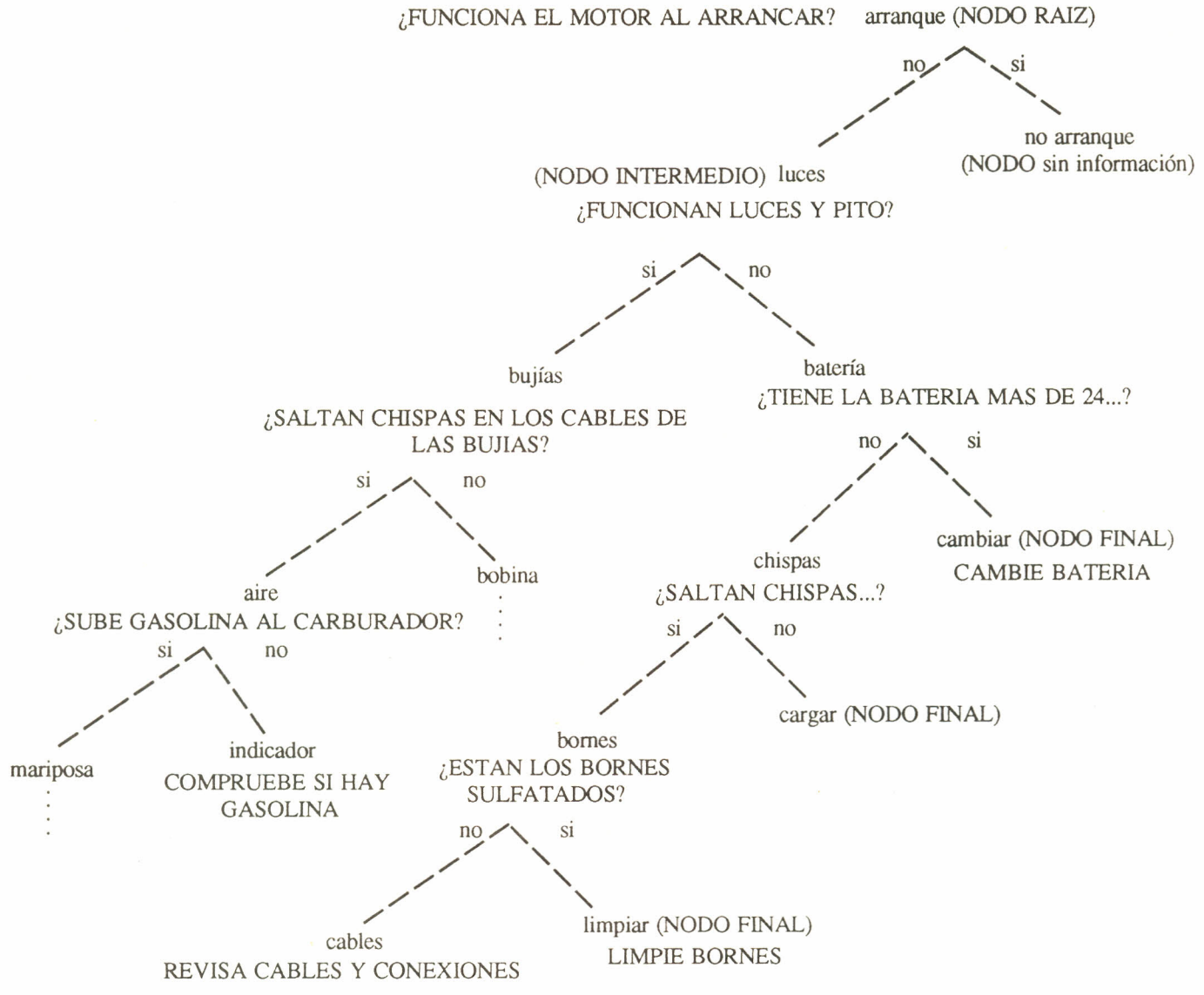

FIGURA 4A. EJEMPLO MODELO TIPO1

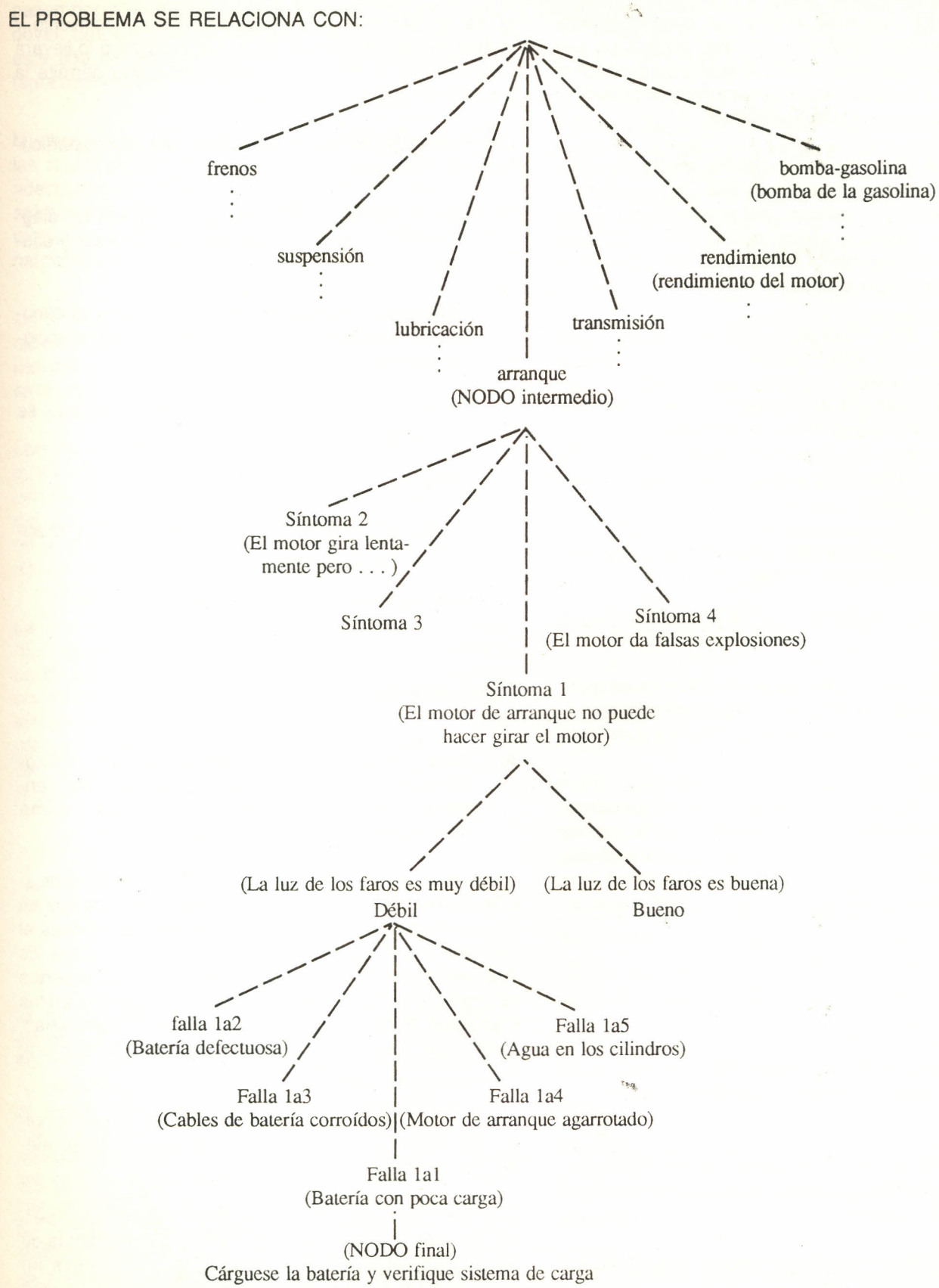

FIGURA 5A. EJEMPLO DE UN MODELO TIPO 2

El árbol de síntomas o factores es un modelo gráfico-lógico que muestra las distintas formas en que los síntomas o factores primarios se combinan para producir el síntoma tope y cuya identificación equivale al encontrar la causa que produce la falla o problema.

Los árboles de síntomas son excelente medio para que los expertos humanos puedan transmitir sus conocimientos formales y empíricos del problema bajo estudio. Con estos modelos resulta muy sencillo, al presentarse un síntoma recorrerlos hacia abajo, para encontrar la causa que pueda motivarlo.

Con base en esta representación, se desarrolló un COMPONENTE DE INFERENCIA muy sencillo, el cual permite recorrer el árbol según las respuestas del usuario. Este componente fue ampliándose gradualmente con las funciones que permiten suministrar las ACLARACIONES durante las consultas y mejorar el sistema de DIALOGO a través de ventanas y menúes especiales.

El sistema de menúes ha sido seleccionado para ofrecerle al usuario mayor rapidez y precisión.

El menú principal ofrece las siguientes opciones básicas:

CONSULTA: permite al usuario que lo lleve a una recomendación para la falla diagnósticada. El diálogo se realiza mediante la formulación de preguntas y la selección de la respuesta de un menú.

CONSULTA Y ACLARA: el usuario obtiene adicionalmente a la recomendación, un resumen de todas las preguntas formuladas y respuestas dadas, y simultáneamente con cada pregunta observará las posibles preguntas o recomendaciones, según la opción que se elija.

BASE DE CONOCIMIENTO: el usuario puede modificar, consultar, agregar o borrar información de la base de conocimientos.

SALVAR LA BASE DE CONOCIMIENTOS: solicita el nombre con el cual se desea salvar la base de conocimientos.

CAMBIAR TIPO O BASE DE CONOCIMIENTOS: permite al usuario cambiar a una nueva base sin necesidad de abandonar el sistema.

SALIR: le permite al usuario abandonar el sistema de diagnóstico.

Antes de presentar el menú principal, el sistema pregunta al usuario el tipo de la base de conocimiento y si ésta existe o es nueva. En el segundo caso lo llevará directamente al módulo de adquisición que permite la creación de una base.

Las siguientes son algunas de las características particulares ofrecidas por ENDI:

- El conocimiento necesario para efectuar los diagnósticos es llevado a un árbol de síntomas y causas.
- Ofrece dos alternativas para representar el conocimiento: árboles binarios o diagnóstico con opciones múltiples.
- La adquisición del conocimiento del experto se realiza en forma sistemática y fácil.
- Sencillas interfases con el usuario.
- Empleo de menues para mayor rapidez y precisión en las consultas.
- Interfases de diálogo y aclaración en español.

CONCLUSIONES

A. Con relación a la selección del "Diagnóstico" como área de aplicación del entorno y a los "árboles de síntomas" como forma para almacenar el conocimiento.

Una forma para definir los posibles campos de aplicación de los sistemas expertos es definir situaciones en las cuales existe un proceso de toma de decisiones el cual puede ser reducido a una sucesión de juicios. Es decir el campo de aplicación de los sistemas expertos es amplio y variado. Pueden citarse aplicaciones en las siguientes areas: diagnóstico, planeación, asesorías, análisis de mercados, entrenamiento, capacitaciones, etc.

Sin embargo el área en la cual han tenido mayor impulso los sistemas expertos es el diagnóstico, tanto médico como técnico.

En el campo del diagnóstico médico los sistemas expertos han sido muy útiles quizas porque los problemas de diagnóstico requieren frecuentemente mucha más información de la que es posible almacenar y exigen además una valoración secuencial de un gran número de hipótesis. En este proceso puede ser fácil que se

pasen por alto un dato o una posible alternativa. Parece obvio, que un sistema computarizado puede ayudar a este tipo de proceso librando al usuario de una dura tarea memorística.

Muchos de los sistemas expertos en diagnóstico utiliza las reglas de tipo if-then. Aunque estos sistemas son adecuados para algunos problemas de diagnóstico, presentan limitaciones, por ejemplo el no utilizar un buen modelo que incluya la descripción estructural y funcional del sistema.

Un árbol de síntomas es un modelo gráfico-lógico que muestra las distintas maneras en que pueden combinarse las causas primarias para producir el síntoma que se modela (síntoma tope).

Los árboles de síntomas son un excelente medio para que los expertos humanos puedan transmitir sus conocimientos formales e informales del problema bajo estudio. Con estos modelos es muy fácil ante la presencia de un síntoma, recorrer el árbol hacia abajo hasta encontrar la solución o recomendación al problema.

La construcción de estos árboles requiere conocer muy bien la estructura del sistema, sus procedimientos, condiciones de operación, etc. Por otra parte los árboles de síntomas facilitan la labor de mantenimiento de las bases de conocimiento, pues proporcionan la forma para que otras personas fácilmente retroalimenten (revisen, corrijan y aumenten) el conocimiento existente en la base.

B. Con relación a la herramienta empleada para la creación del entorno

Si se considera el "mecanismo de inferencia" de un sistema experto como la estrategia de búsqueda de solución, podría afirmarse que cuando el conocimiento ha sido elevado a árboles de síntomas, dicha estrategia se limita simplemente a recorrer el árbol hacia abajo hasta encontrar la causa del problema. Esto puede plantearse fácilmente en PROLOG y en muy pocas líneas. Esta es una de las grandes ventajas de la utilización de PROLOG para la realización del entorno ENDI. Sin embargo dicha búsqueda en el árbol no se origina automáticamente por el mecanismo de inferencia dirigida por objetivos (propio del PROLOG), sino que se programa mediante reglas y con base en la forma como se almacena la información en la base de conocimientos.

Entre las desventajas puede citarse la necesidad de máquinas de procesamiento simbólico, para obtener la mayor eficiencia de estos programas.

Actualmente ENDI se halla implementando en CS-PROLOG, el cual es un interpretador y se tienen problemas de velocidad. Por ejemplo cargar una base de conocimientos TIPO 1, de 20 nodos (1874 bytes) consume 55 segundos en un computador con procesador 286 y 8 segundos con un procesador 386.

CS-PROLOG (Comunication Sequential Prolog) ofrece las siguientes características de gran ayuda en el desarrollo o construcción de entornos:

1) Facilidad para el manejo de ventanas y menues con predicados particulars para tal fin.

2) Facilidad para la elaboración de g´raficos: CS-PROLOG permite al usuario debujar puntos, líneas, arcos, figuras geométricas y trabajar en el espacio tridimensional, de una forma muy sencilla.

3) Ejecución paralela: el concepto del tiempo ha sido introducido en CS-PROLOG en forma abstracta y se ha denominado tiempo virtual. Pero este no corresponde a tiempo real, ni al tiempo requerido para la evaluación de claúsulas en CS-PROLOG.

Aplicaciones para el uso explícito de tiempo incluyen simulaciones de procesos que pueden lógicamente dividirse en eventos separados pero comunicados secuencialmente.

"Paralelo" en el contexto de CS-PROLOG, significa que pueden existir varios procesos al mismo tiempo, pero sólo hay uno activo en un momento dado,pues los demás son suspendidos.

Esta facilidad particular de CS-PROLOG puede ser muy útil en la solución de algunos problemas, sin embargo no se utilizó en el desarrollo del entorno ENDI.

Para disminuir los problemas de velocidad que actualmente se tienen, se desea implementar el entorno ENDI en Turbo-Prolog, con lo cual se podrá compilar el programa.

Posteriormente se hará una evaluación comparativa en aspectos como: uso de memoria, velocidad, facilidad de adaptación, amigabilidad, etc.

REFERENCIAS BIBLIOGRAFICAS

1. **ABADIA, José Antonio**. Propuesta de Investigación "*Construcción de un entorno generalizado para el desarrollo de sistemas expertos consultores*" . Febrero de 1987.

2. **ALTY** y **COOMBS**. *"Sistemas Expertos". Conceptos y Ejemplos.* Díaz de Santos, S. A. 1986.

3. **ARELLANO, Juan SCHWARZBLAT, Morris, LARRE Mónica.** *"Un nuevo enfoque para la construcción de Sistemas Expertos en diagnóstico".* Artículos Técnicos.

4. **BLANCO VARGAS, Myriam.** *"Sistemas Expertos".* Centro de publicaciones de regionalización de la Universidad del Valle. 1988.

5. **BOERINGER, Bernhard, CHIOPRIS Carlo, FUTO IVAN.** *Wissensbasierte Systeme Mit Prolog.* Addison-Wesley 1988.

6. **BONNET, Alain**. *Artificial Intelligence.* Prentice/ Hall. 1985.

7. **CENTRO PARA EL DESARROLLO TECNOLOGICO INDUSTRIAL**. Ministerio de Industria y Energía (España). "*Sistemas Expertos: Consideraciones Teóricas y Sugerencias de uso.*

8. **FISCHER, Martín A, FIRSCHEIN Oscar.** *Intelligence.* Addison-wesley 1987.

9. **HARMON P. / KING D.** *Experten Systeme in der Praxis.* R. Oldenbourg Verlag Munchen Wien 1987.

10. **HARTNELL, Tim.** *"Inteligencia Artificial".* Anaga Multimedia 1986.

11. **JACKSON, Peter.** *"Experten Systeme".* Addison-wesley 1985.

12. **NAYLOR Chris.** *"Construya su propio sistema experto "* . Díaz de Santos, S. A. 1985.

13. **RAUCH-HINDIN, Wendy B.** *"Artificial Intelligence in Business, Science, and Industry".* Prentice Hall.

14. **ROBINSON, R. Philip.** *"Aplique Turbo Prolog".* McGraww-Hill 1987.