Informe
No. 2

Sistemas Expertos

Club de Investigación Tecnológica

Sistemas Expertos

Preparado por: Claudio Gutiérrez Carranza
Enero 1989

Editado y Publicado por Rho-Sigma, S.A.
a nombre del Club de Investigación Tecnológica.
Todos los derechos reservados.
Prohibida la reproducción total o parcial.
San José, Costa Rica
Marzo, 1989

Contenido Página

I Introducción....1
Reseña Histórica....1
II Sistemas Expertos....4
1. Introducción....4
Papel del Conocimiento en los Sistemas Expertos....5
2. Arquitectura de los Sistemas Expertos....6
La Representación del Conocimiento....7
El Aparato de Inferencia....8
3. Clases de Sistemas Expertos....9
4. Atributos Fundamentales de los Sistemas Expertos....10
Abiertos a la inspección....10
Fácilmente modificables....10
Heurísticos....10
III Implantación....12
1. Selección de Dominio....12
2. Consideraciones de Costo y Personal....12
3. El Proceso de Adquisición de Conocimiento....13
4. El Papel de las Conchas en el Desarrollo de un Sistema Experto....14
IV Funcionamiento Formal....15
1. Sistemas de Reglas de Producción....15
¿Por qué usar reglas de producción?....16
Operación con indices de confiabilidad....17
2. Programación Libre....17
3. Razonamiento por Casos....17
V Conchas Comerciales....20
1. Introducción....20
2. Conchas más Populares....20
3. Revista Especializada....21
4. Conchas para Micros....22
5. Conchas para Minis y Estaciones de Trabajo....23
6. Conchas para Gigantes (Computadores Centrales)....24
VI Evaluación de las Conchas de Sistemas Expertos....25
1. Introducción....25
2. Representación de Conocimiento....26
3. Separación entre base de conocimiento y estrategia de control....27
4. Facilidad de corrección y adquisición de conocimientos....27
5. Capacidad de explicar los razonamientos....28
6. Capacidad de lidiar con conocimiento incierto....29
7. Capacidad de lidiar con información incompleta....29
VII Conclusiones....31
Apéndice A: Estrategias de Control Formal....33
Apéndice B:Lenguajes de Inteligencia Artificial en relación con los Sistemas Expertos 35
Bibliografía....37

Resumen Ejecutivo

En este segundo informe de investigación del CIT analizamos las oportunidades y los problemas que presenta la utilización de la tecnología de los sistemas expertos en nuestro medio.

Explicamos qué son los sistemas expertos, cómo se desarrollan y utilizan, cómo se implantan y cómo funcionan. Igualmente describimos el tipo de ayuda que el mercado ofrece para desarrollarlos y usarlos, y examinamos hasta qué punto es posible beneficiarse de esta tecnología sin apoyo de personal informático especializado o con apoyo limitado.

También analizamos cuáles son los límites de los sistemas expertos, qué tipo de problemas son susceptibles de este enfoque y cuáles son los pasos necesarios para su empleo. Finalmente tocamos el tema de las posibles ventajas que la aplicación de esta tecnología tiene potencialmente para los empresarios del medio costarricense.

El informe está dirigido a aquellas personas encargadas de planificar la aplicación de la tecnología, aunque también contiene cierto nivel de detalle que será de interés para aquellos encargados de llevar a cabo la implantación y operación de esta novedosa tecnología.

El informe ofrece una presentación conceptual para que la persona no especialista en Inteligencia Artificial perciba los fundamentos teóricos y los antecedentes históricos de la tecnología de los sistemas expertos. Además, expone cuáles son las expectativas asociadas con ella y explora los requisitos para aplicarla con fruto.

Finalmente, se hace una evaluación de las expectativas asociadas con las conchas ("shells") para desarrollar sistemas expertos, a la luz de la experiencia con este tipo de programa, tal y como se refleja en la más reciente literatura de este campo especializado de las ciencias de la computación.

Reconocimientos

Se agradece la colaboración durante la investigación y preparación de este informe de las siguientes personas: Dr. Daniel Chester, Profesor de la Universidad de Delaware, por el suministro de algunos materiales importantes para la investigación; Licda. Marlene Castro, quien leyó minuciosamente dos versiones preliminares del informe y ofreció valiosos comentarios; Dr. Roberto Sasso, quien aportó información útil e hizo importantes sugerencias para mejorar el informe; Lic. Jorge W. Bolaños, y Lic. Enrique Odio, quienes revisaron la versión semifinal y propusieron mejoras.

La labor de investigación correspondió al autor del informe, Dr. Claudio Gutiérrez, a quien debe achacarse cualquier error u omisión que contenga.

Del Autor

El Dr. Gutiérrez, filósofo, abogado y científico de la computación, sirvió más de trienta años en la Universidad de Costa Rica en distintas posiciones, incluyendo la de Rector de 1974 a 1981. Fue también Vicerrector de Docencia, Decano de la Facultad de Ciencias y Letras y Director del Departamento de Estudios Generales.

Durante muchos años ejerció las cátedras de Filosofía de la Ciencia y de Computación e Informática en la Universidad de Costa Rica. Ha sido Filósofo en Residencia de la Universidad de Texas en Austin, Profesor Invitado en la Universidad de Nuevo México, y ha ocupado la Cátedra Langton Hughes de la Universidad de Kansas. Desde 1983 es Catedrático de Ciencias de la Computación en la Universidad de Delaware, donde ha estado a cargo también del Departamento de Ciencias de la Computación.

I Introducción

Llamamos sistemas expertos a programas de computación electrónica que colocan dentro de una máquina, al alcance de un usuario sólo moderadamente adiestrado, los conocimientos acumulados a lo largo de muchos años de experiencia por expertos humanos de gran variedad de dominios.[1]

En agosto de 1988 se reunieron en Saint Paul, Minnesota, alrededor de 5.000 científicos de la computación dedicados a la empresa de tratar de emular en computadores el funcionamiento del intelecto humano, en la conferencia anual de la Asociación Americana de Inteligencia Artificial (AAAI).

Durante esa conferencia, el profeta de los sistemas expertos, Edward Feigenbaum, de la Universidad de Stanford, informó a la concurrencia que el número de sistemas expertos ha crecido de unos cuantos cientos que había por la misma fecha del año anterior a más de 2.000 ese año.

Feigenbaum dirigió una mesa redonda titulada "Resultados de un Estudio de la Primera Ola de Aplicaciones de Sistemas Expertos a los Negocios". Durante esa mesa redonda, Feigenbaum recordó su primera conferencia importante sobre sistemas expertos, en 1977, cuando todavía era posible enumerar por su nombre los más importantes ejemplares de esta tecnología.

"El crecimiento durante la pasada década y el pasado año ha sido fenomenal", dijo Feigenbaum. Agregó que de los 2.000 y resto de sistemas en existencia, unos 1.500 están en los Estados Unidos. Van desde los extremadamente simples, como sistemas expertos para archivar documentos en una oficina, hasta los muy complejos, como los de organización de la producción en una fábrica, y con ayuda de conchas[2][2] ("shells") funcionan como asistentes para profesionales y semiprofesionales en los negocios.

> "El primer principio de la ingeniería del conocimiento es que la capacidad para resolver problemas propia de un agente inteligente es primordialmente una consecuencia de sus conocimientos y sólo secundariamente una consecuencia del método de inferencia empleado. Los sistemas expertos deben ser ricos en conocimientos aunque pobres en métodos.... El poder reside en el conocimiento."
>
> Edward Feigenbaum

"Hemos creado una tecnología que facilita el crecimiento de la riqueza nacional", terminó diciendo.

Reseña Histórica

La tecnología de los sistemas expertos es un subproducto de la investigación en una disciplina nueva y esotérica llamada Inteligencia Artificial.

Para entender mejor la naturaleza y el papel de los sistemas expertos, debemos hacer una breve reseña histórica de la Inteligencia Artificial. Podemos definir esta disciplina como el intento de producir por medios electrónicos comportamientos que, si los viéramos realizados por un ser humano, diríamos que requieren inteligencia.

[1] Entendemos por dominio un campo limitado del conocimiento que coincide con la especialidad de un experto humano.

[2] Se les da el nombre de conchas ("shells") a programas que encarnan el intérprete de un sistema experto de manera generalizada, es decir, con independencia del dominio de aplicación. La existencia de conchas supone el axioma básico de la tecnología de los sistemas expertos, a saber, la separación entre el aparato inferencial y el conocimiento.

El aparato inferencial (también llamado control o intérprete) es el mecanismo que aplica el conocimiento del dominio respectivo al problema concreto que se le pide al sistema resolver.

El conocimiento, por su parte, es el conjunto de reglas que representa la experticia del campo de aplicación de cada sistema experto concreto. Una sola concha puede servir para encarnar diversos sistemas expertos, si se la dota de otras tantas bases de conocimientos.

[2]

La Inteligencia Artificial se inicia en teoría en la década de los 50, con las reflexiones de Alan Turing -célebre matemático inglés- sobre la posibilidad de que las máquinas llegaren algún día a exhibir inteligencia. Más en concreto, la disciplina comienza a existir con los trabajos pioneros de A. Newell y H. Simon, de las universidades de Carnegie Mellon y Pittsburgh.

Por esas mismas fechas, estos investigadores crean un programa que prueba teoremas lógicos, y sobre todo un programa muy influyente llamado el Solucionador General de Problemas ("General Problem Solver" o GPS).

El GPS funciona a base de *análisis de fines y medios*, método en que se comienza por especificar un estado inicial, del que se parte, y un estado final, al que se quiere llegar, en el espacio lógico donde se representa el problema.

El programa usa un conjunto de operadores [3] capaces de eliminar diferencias entre los dos estados, que se consignan en una *tabla de operadores y diferencias* específica para cada tipo de problema.

Los operadores tienen prerrequisitos para su aplicación; si estos prerrequisitos no se dan en el estado presente del problema, es posible invocar recursivamente[4] el mismo programa GPS para lograr un estado en que los operadores puedan aplicarse.

En otras palabras, el GPS comienza por comparar el estado inicial y el estado deseado para extraer una lista de diferencias entre los dos. Obtenida la lista de diferencias entre los dos estados, el programa consulta la tabla de operadores y diferencias para identificar los operadores que sirven para eliminar las diferencias encontradas.

Cada entrada en la tabla contiene una indicación de los prerrequisitos para la aplicación del respectivo operador (por ejemplo, para enroscar una tuerca en un tornillo la tuerca no debe estar enroscada ya en otro).

Puede suceder una de dos cosas:

1) o los requisitos están ya cumplidos, en cuyo caso el GPS agrega el respectivo operador a la solución que esté en proceso de construcción;

2) de lo contrario, es necesario hacer efectivos esos requisitos, problema que puede resolverse con otra invocación (recursiva) del mismo GPS (la solución que logre esta llamada recursiva se intercalará en el lugar adecuado en la solución que se esté formando para el problema principal).

La secuencia de operadores creada por la llamada principal y las llamadas recursivas a GPS refleja el proceso que puede transformar el estado inicial en el estado final y constituye la solución al problema.

El método de análisis de fines y medios funciona bien en la medida en que lo proveamos, para cada tipo de problema, de una adecuada tabla de operadores y diferencias. Por sí mismo es un método débil, incapaz de "solucionar toda clase de problemas" como el ambicioso nombre GPS sugiere.

El programa GPS no tuvo éxito como instrumento para replicar la capacidad general de resolver problemas que admiramos en los seres humanos; en cambio, resultó sumamente provechoso para revelar los constreñimientos a que está sometida la inteligencia.

Fue muy productivo en un doble sentido. En primer lugar permitió descubrir una ley de la inteligencia, que Newell y Simon bautizaron

---

[3] Se llama operador a un procedimiento o función que transforma un estado en otro distinto.

[4] El término recursivo, muy usado en ciencias de la computación, se refiere al hecho de que dentro de un procedimiento o función puede invocarse ese mismo procedimiento o función. Un ejemplo trivial puede ilustrar el sentido de este término técnico.
Imaginemos que vamos a definir lo que quiere decir **REIR**. Una posible manera de hacerlo se ofrece a continuación.

Para REIR haga en secuencia lo siguiente:
1- diga JA
2- invoque el procedimiento REIR.

Con esta definición de reiír, cualquiera que invoque el procedimiento REIR dirá JA JA JA... indefinidamente.

Normalmente las definiciones recursivas incluyen un enunciado condicional para detener la recursión. Por ejemplo, podríamos modificar nuestro fantasioso ejemplo como sigue.

Para REIR haga en secuencia lo siguiente mientras siga alegre:
1- diga JA
2- invoque el procedimiento REIR.

como "ley de proporcionalidad inversa entre la generalidad y la potencia":

A mayor generalidad, menor potencia; a mayor potencia ,menor generalidad.

Esta ley afirma que los métodos que empleamos para resolver problemas no pueden ser generales y poderosos al mismo tiempo; cuanto más generales sean (como análisis de fines y medios desprovisto de tablas de operadores y diferencias) más débiles son, y cuanto más poderosos, menos generales (como el GPS provisto de una tabla de operadores y diferencias para resolver cierto tipo de problemas).

Si el estado actual es el estado deseado **entonces** terminar, y la secuencia de operadores obtenida hasta ahora es la solución; **de otro modo**, buscar en la tabla los operadores que eliminen las diferencias.
Para cada operador,
si los requisitos de aplicación están cumplidos, agregarlo a la solución que se construye;
de lo contrario, invocar a GPS para obtener un estado en que los requisitos se cumplan, y agregar a la solución la secuencia de operadores que esta llamada produzca.
Repetir, tomando como estado actual el que resulte de aplicar los operadores seleccionados en esta vuelta.

Algoritmo de análisis de Fines y Medios

Esto llevó a distinguir de una manera muy precisa lo que es la habilidad general del entendimiento humano de la habilidad que resulta de poseer conocimientos específicos, como los que se pueden representar en una tabla de operadores y diferencias (podríamos rebautizar la tabla de operadores y diferencias y llamarla "tabla de conocimientos").

El segundo sentido en que el GPS fue productivo está relacionado con lo dicho anteriormente: demostró que el poder está en el conocimiento, como más tarde lo expondría dramáticamente Ed Feigenbaum. Dicho de otra manera, *las tablas de diferencias y operadores pueden considerarse como el antecedente de los sistemas expertos.*

A partir de las experiencias de Newell y Simon, los practicantes de la disciplina de la Inteligencia Artificial se concentraron en diseñar maneras de representar masas de conocimiento de dominios particulares, y descubrir cuál fuera la mejor manera de aprovechar los conocimientos así representados para sacar conclusiones especializadas como las que producen los expertos.

¡Estaban a punto de nacer los sistemas expertos!

II Sistemas Expertos

La década del 70 vio nacer la nueva tecnología. En su desarrollo fue muy importante el legado del GPS. La labor de desarrollo de la idea germinal, sin embargo, fue realizada principalmente en la Universidad de Stanford.

Un sistema experto es un programa de computación basado en conocimiento y que provee soluciones de calidad experta a problemas de un dominio específico.

1. Introducción

La principal herencia ganada del período anterior de desarrollo de la Inteligencia Artificial fue una apreciación de la importancia del conocimiento dependiente del dominio específico en que se trate de resolver problemas.

Un médico, por ejemplo, no es eficaz en el diagnóstico de enfermedades simplemente porque posee un don innato para resolver cierto tipo de problemas. Su eficacia depende de que sabe muchísimo sobre el dominio de la medicina. Igualmente, un geólogo es eficaz en el descubrimiento de yacimientos minerales porque ha estudiado mucha geología y tiene experiencia en ese campo.

La experticia es una combinación de la comprensión teórica de una clase de problemas y de la acumulación de reglas producto de la experiencia, también llamadas heurísticas.[5]

Los llamados sistemas expertos son programas de computación que tratan de replicar la experticia de los profesionales de distintos ramos mediante el expediente de reproducir la acumulación de buenas heurísticas (el factor de comprensión sistemática, en cambio, es mucho más difícil de introducir en un programa de computación y no ha sido incorporado a la tecnología). Las heurísticas, naturalmente, tienen que ser extraídas de los expertos humanos.

Uno de los primeros sistemas de esta clase fue DENDRAL, desarrollado en la Universidad de Stanford a fines de la década del 60. Este sistema fue diseñado para inferir la estructura de las moléculas orgánicas a partir de las fórmulas químicas y de información espectrográfica sobre masas.

Como las moléculas orgánicas tienden a ser grandes, las posibles conexiones entre sus elementos son muy abundantes y no es factible explorar todas las combinaciones. Los químicos usan heurísticas para simplificar la búsqueda del modelo correcto.

La réplica electrónica de los métodos de los expertos en química orgánica tuvo tanto éxito que ello dio lugar a la generalización de la tecnología durante la década del 70.

Probablemente el sistema experto más famoso es MYCIN, desarrollado también en Stanford. Fue el primer sistema que razonó con base en información insegura e incompleta. Gran cantidad de técnicas que hoy se aplican en todos los sistemas expertos fueron desarrolladas en el curso de la construcción de este sistema. MYCIN usa conocimiento médico para diagnosticar y recetar tratamiento sobre meningitis de la médula espinal e infecciones bacteriales de la sangre.

Otro sistema experto clásico es PROSPECTOR, un programa para determinar la posible localización y naturaleza de yacimientos

[5] Una heurística es una regla extraída de la experiencia que ha probado ser efectiva en una gran cantidad de casos de cierto tipo. No garantiza la solución del problema porque no se basa en leyes científicas o algoritmos matemáticos para recomendar una solución, sino sólo en el hecho de que muchas veces ha tenido éxito en la solución de problemas de ese tipo.

Normalmente una heurística es un método para cribar un árbol de posibilidades, descartando las alternativas menos promisorias. Por ejemplo: si estamos perdidos en un bosque, una buena heurística para salir de él es caminar en línea recta; una buena heurística para encontrar un río es seguir la gradiente del terreno hacia abajo; una buena heurística para alcanzar un pico elevado desde donde divisar el paisaje es seguir la gradiente hacia arriba. Buenas heurísticas para jugar ajedrez son: tratar de dominar las casillas del centro, no doblar peones, tener protegido al rey.

Ninguna de esas heurísticas garantiza el éxito, pero a falta de mejores razones para escoger una cierta línea de acción nos ofrecen buenas posibilidades de conseguir nuestros propósitos.

minerales, con base en información geológica sobre la respectiva región.

Si
la infección es bacteremia primaria
y el cultivo fue hecho en un lugar estéril
y la supuesta vía de ingreso es gastrointestinal
entonces
hay evidencia que sugiere (índice de confiabilidad 0,7) que la infección es bacteroide

Una regla del sistema MYCIN

MYCIN y PROSPECTOR tienen en común que su propósito es hacer diagnóstico; sin embargo, como explicaremos más adelante, existen muchos otros posibles propósitos para un sistema experto.

Papel del Conocimiento en los Sistemas Expertos

Dijimos antes que los sistemas expertos son programas basados en conocimiento que proveen soluciones de calidad experta a problemas de un dominio específico. A diferencia de lo que sucede con otros tipos de programas, para escribir sistemas expertos uno le dice al computador qué debe saber, no cómo lo debe hacer. Es por ello precisamente que se les identifica como "sistemas basados en conocimiento".

En los programas tradicionales, se suministra al computador, en el lenguaje adecuado, un algoritmo, es decir, una serie de pasos o instrucciones que el computador debe realizar para resolver el problema.

Si tal algoritmo existe, debe usarse, pues llevará a la solución infaliblemente. Pero en los problemas que un experto humano resuelve no hay tal tipo de instrumento disponible, sino que el experto usa su experiencia, acumulada a lo largo de los años para resolver el problema. En vez de algoritmo disponemos de conocimientos. Esos conocimientos son los que deben trasplantarse al sistema experto computacional.

El tipo de problemas para los que no existe un algoritmo infalible suelen llamarse problemas no estructurados; en relación con ellos es imposible escribir una serie de especificaciones que den lugar a la construcción de un programa. En cambio, su solución puede basarse en un amplio cuerpo de conocimientos que no pueden ser reducidos fácilmente a números, o que trascienden las categorías de cantidad totalmente.

La manera en que la Inteligencia Artificial enfoca los problemas no estructurados es emulando lo que las personas hacen. Se trata de acumular conocimientos y almacenarlos de tal manera que puedan usarse para solucionar problemas del mismo tipo.

Un sistema experto colecciona, formaliza y representa en forma de reglas algún cuerpo amplio de destrezas específicas en relación con una tarea, y permite usar tal conocimiento como si estuviera en la propia cabeza.[6]

Es importante tener en cuenta que, a diferencia del experto humano, los sistemas expertos comercialmente realizables no aprenden por experiencia;[7] su conocimiento debe ser extraído de los humanos y codificado en un lenguaje formal, lo cual viene a ser la tarea más pesada que debe enfrentar el constructor de sistemas expertos.

Considerados como arquitectura computacional, los sistemas expertos tienen dos partes fundamentales: el **aparato inferencial** y la **base de conocimientos**. La base de conocimientos contiene la información específica de la tarea o dominio. El aparato inferencial es el mecanismo para usar la información de la base de conocimientos.

El conocimiento permanente es complementado con conocimiento sobre el problema concreto que se quiere resolver; este conocimiento, también llamado los datos del problema, se introduce en la

---

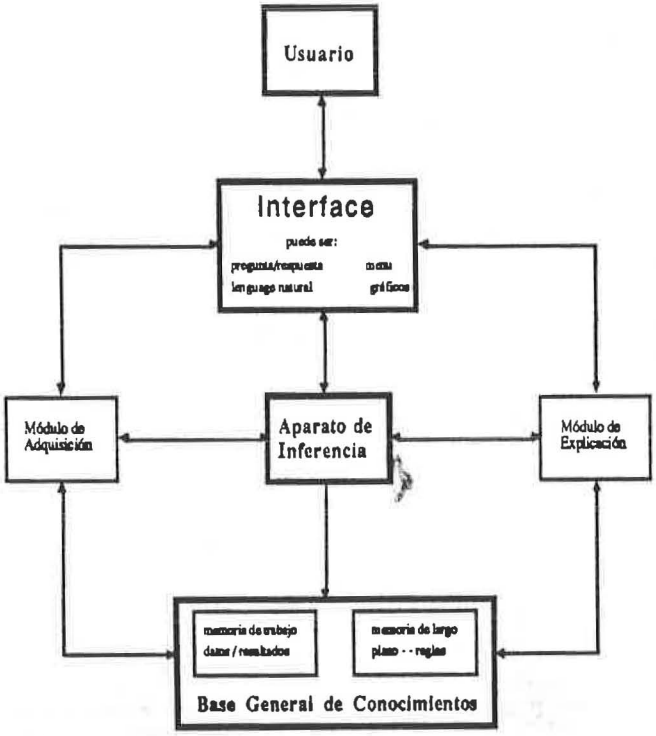
6 Debe tenerse presente, sin embargo, que la tarea debe ser de naturaleza intelectual, susceptible de recibir una representación simbólica. Los sistemas expertos actuales no son capaces de reproducir una destreza no representable, como por ejemplo nadar o andar en bicicleta.

7 Existe una gran cantidad de investigación en el campo de la Inteligencia Artificial que se ocupa precisamente de dotar a los sistemas expertos de la capacidad de aprender, pero todavía sistemas que aprenden son solamente habitantes de laboratorios.

memoria de trabajo o memoria de corto plazo, donde ocurrirá todo el proceso que conducirá eventualmente a la derivación del consejo o recomendación que el sistema otorgará al usuario como respuesta a su consulta.

En la **memoria de largo plazo** se almacena el conocimiento experto en forma de reglas, a diferencia de la memoria de trabajo, donde se introducen los datos, se producen los resultados intermedios y el resultado final.

Es frecuente que el sistema experto tenga también un **módulo de explicación**, y en los mejores ejemplares, también un **módulo de adquisición** del conocimiento que ayuda al usuario a crear las reglas que representan la experticia.



Arquitectura de un Sistema Experto Típico

2. Arquitectura de los Sistemas Expertos

El cuadro anterior muestra los módulos más importantes de un sistema experto típico.

El usuario interactúa con el sistema por medio de la *interface* que hace más cómodo el acceso al programa, escondiendo los detalles de su complejidad. En los programas comerciales ocurren muy diversas modalidades de interacción, desde la simple pregunta y respuesta hasta la expresión gráfica, pasando por el menú y el lenguaje natural estereotipado.

En relación con esto último, debemos advertir que está muy lejano el día en que los sistemas expertos puedan interactuar con el usuario en inglés o español, sin constreñimientos; las interfaces de lenguaje natural usan un subconjunto del idioma del usuario, sometido a reglas de formación de frases muy rígidas y limitadas.

El sistema tiene que mantener los datos específicos del problema y los resultados intermedios en una *memoria de trabajo* o memoria de corto plazo, que es un submódulo separado de la *memoria de largo plazo* donde se conservan las reglas o unidades de conocimiento del sistema.

La memoria de trabajo consta de los hechos que describen la situación presente. Al comienzo contendrá muy pocos hechos. Pero estos se expanderán por obra de la acción de las reglas, conforme el sistema va sacando consecuencias que enriquecen la comprensión de la situación.

La memoria de largo plazo contiene el conocimiento propiamente dicho, formado por reglas de condición/consecuencia, también llamadas reglas de producción. El número de reglas es muy grande, yendo de varios cientos para sistemas simples hasta muchos miles para los más elaborados.[8]

El *módulo explicatorio* es el componente del sistema que deja saber al usuario de qué manera el sistema ha alcanzado una conclusión particular. Sirve a la necesidad, por ejemplo, de que en cierto punto del proceso el usuario quiera preguntar cómo fue que el sistema llegó a cierta conclusión intermedia.

---

[8] Newell y Simon han calculado que una experticia importante, como jugar ajedrez, requiere alrededor de cincuenta mil unidades de conocimiento; asimismo estiman que el tiempo promedio en que los humanos obtienen esa cantidad de conocimiento es de diez años.

El módulo explicatorio sirve también otras necesidades; por ejemplo, ciertas reglas, al ejecutarse, piden al usuario información adicional, digamos en un sistema de diagnóstico médico puede preguntar si el paciente sufre temperatura alta. La interface suele permitir al usuario en esos casos preguntar a su vez por qué requiere el sistema tal información.

El módulo explicatorio contestaría que la información se requiere para determinar si una regla sobre dolor agudo en el abdomen que diagnostica apendicitis debe activarse.[9]

En suma, el módulo de explicación permite al programa justificar su razonamiento ante el usuario. Las justificaciones suelen ser una transcripción a lenguaje natural del contenido de las unidades de conocimiento o reglas usadas en el razonamiento. Para este efecto, las conchas comerciales proveen un mecanismo que asocia un texto en lenguaje natural a cada una de las reglas, el cual es usado si la explicación se requiere.

Muchas conchas incluyen también un *módulo de adquisición de conocimientos* que permite agregar o eliminar reglas y también corregir las existentes. En los sistemas más simples, que carecen de este módulo, el usuario debe emplear para esta función un procesador de textos independiente del sistema. En sistemas más complejos, es posible corregir las reglas sobre la marcha, en medio de una derivación de conclusiones.

Es frecuente que este ambiente permita navegar por las ramas de la inferencia, con el fin de localizar causas de errores en el razonamiento y hacer los ajustes necesarios en la base de conocimientos. Esta capacidad es una ayuda invaluable para el trabajo de depurar una base de conocimientos a fin de obtener la mejora en rendimiento del sistema.

La Representación del Conocimiento

El corazón del sistema es la base general de conocimientos que contiene la experticia para resolver problemas, en la forma de *reglas de condición/ consecuencia* (también llamadas en la literatura técnica *reglas de producción*), o algún equivalente.

Las reglas de un sistema de producción tienen dos partes: el antecedente o premisa (lado izquierdo de la regla) y el consecuente o conclusión o recomendación (lado derecho de la regla).

Una regla se selecciona si la información en la memoria de corto plazo contiene todos los datos que calzan con el antecedente de la regla. La parte consecuente es generalmente una hipótesis o un consejo.

Aunque menos usadas, son también formas de representar el conocimiento, funcionalmente equivalentes a las reglas de producción, aunque técnicamente distintas: los marcos, las redes semánticas, las jerarquías de tipos y los objetos que intercambian mensajes.[10]

[9] Decimos que una regla se dispara cuando la parte de consecuente se ejecuta, o sea, cuando el sistema acepta como probable o verdadera (dentro de cierto índice de confiabilidad) la hipótesis contenida en ella.

[10] Los marcos son asociaciones de atributos que pertenecen a un mismo tipo de objeto y que se tratan corporativamente; pueden por ejemplo representar una serie de síntomas relacionados con un tipo de malfunción de una máquina.

Las redes semánticas asocian conceptos unos con otros, con base en conexiones con etiquetas que representan relaciones lógicas importantes, como "es parte de", "es intrumento para", "es indicador de", etcétera.

Las jerarquías de tipos son un caso especial de redes semánticas donde las conexiones son: "es un subconjunto de" o "es un ejemplar de".

Los objetos que intercambian mensajes (ver Apéndice B) son representaciones activas de los entes importantes en un campo de conocimiento, que son capaces de aplicarse a sí mismos métodos especializados como reacción a mensajes enviados por otros objetos (por ejemplo, una regla puede recibir el mensaje "explíquese", y como reacción a él desplegar en la pantalla una oración prefabricada que describe la regla en lenguaje natural).

R1 Si hay gas en el tanque y hay gas en el carburador entonces
el motor recibe gasolina

R2 Si el motor recibe gasolina y el motor se mueve entonces
el problema es de candelas

R3 Si el motor no se mueve y las luces alumbran entonces
el problema es de arrancador

R4 Si el motor no se mueve y las luces no alumbran entonces el problema es de batería o cables

Ejemplo de reglas de producción en un sistema experto para diagnosticar fallas de un automovil

El Aparato de Inferencia

El aparato inferencial ("inference engine") aplica el conocimiento a la solución de los problemas concretos. Es el intérprete de la base de conocimientos.

En un sistema de reglas de producción el aparato inferencial o razonador ejecuta un ciclo de control de dos fases: *reconocimiento* y *acción*. Durante la primera fase, el intérprete localiza las reglas que calzan con el actual contenido de la memoria de trabajo; durante la segunda, se selecciona entre esas reglas la que convenga activar y se ejecuta esa regla.

El aparato inferencial o intérprete tiene varias tareas. Primero, examinar los hechos en la memoria de trabajo y las reglas de la base de conocimientos y añadir nuevos hechos cuando ello resulta posible; es decir, ejecutar las reglas.

Además, el aparato inferencial determina el orden en que las reglas se examinan y se activan, mediante la aplicación de diversos criterios; dos de los más usados son el criterio de continuidad y el criterio de especificidad.

Según el criterio de continuidad, se tiende a activar reglas que hayan sido usadas recientemente o cuyas condiciones calcen con hechos de más reciente creación; el sentido de este criterio es asegurar que un conjunto congruente de reglas pueda continuar y terminar un trabajo empezado.

El criterio de especificidad da preferencia a la regla más específica sobre la más general, con el propósito de hacer el proceso lo más concreto posible.

Si el sistema es interactivo, el intérprete es además responsable de hacer preguntas al usuario tendientes a enriquecer el conjunto de datos pertinentes en posesión del sistema. También es responsabilidad del intérprete decidir cuándo ha obtenido una solución al problema planteado y suministrarla al usuario.

Suele ser también responsabilidad del aparato de inferencia la conexión del sistema con otros programas, por ejemplo bases de datos u hojas electrónicas, con el objeto de requerir o aportar datos. Esta capacidad da gran flexibilidad al sistema para integrarse con herramientas algorítmicas a disposición del usuario.

Debe recalcarse que los procedimientos que sustentan al intérprete son completamente distintos y separados de las cláusulas declarativas que representan el conocimiento; esto es importante por varias razones:

1 La separación del conocimiento para resolver problemas y el aparato inferencial hace posible representar el conocimiento en una forma más natural, por ejemplo con reglas del tipo "Si... entonces".

2 Los constructores de sistemas pueden concentrarse completamente en la recolección de las bases de conocimiento. La ingeniería del conocimiento consiste precisamente en ese proceso de recolección.

3 La separación del conocimiento y el control permite desarrollar los sistemas sin consideración a detalles computacionales y en forma modular, incorporando reglas de forma declarativa una por una, sin tener que afectar el programa mismo (el intérprete o aparato inferencial).

4 Es posible diseñar conchas o intérpretes que sirven en muchos dominios de conocimiento, pues solamente se necesita cambiar la base de

conocimientos para trasladarse a otro dominio de aplicación.

5 Finalmente,[11] es posible usar la misma base de conocimientos con distintos intérpretes, con el propósito de experimentar con el desarrollo de mejores intérpretes.

3. Clases de Sistemas Expertos

Se han construido sistemas expertos para resolver problemas en dominios tan variados como la medicina, la matemática, la ingeniería, la química, la geología, los negocios, el derecho, la defensa, y la educación.

El sistema XCON de la Digital Equipment, por ejemplo, fue desarrollado para automatizar la organización de los componentes de un sistema de cómputo a fin de asegurar que los envíos incluyeran todas la piezas y cables necesarios.

El sistema Cooker de Campbell Soup fue desarrollado cuando un empleado con cuarenta años de estar en la compañía se acercó a la fecha de retiro; la compañía incorporó el conocimiento de este empleado sobre sus máquinas en este programa a fin de proveer ayuda experta para los nuevos empleados de mantenimiento.

La firma international de auditores Coopers & Lybrand desarrolló un sistema experto (Expertax) para resolver consultas sobre problemas de impuestos de sus clientes de los Estados Unidos que ha tenido mucho éxito.

En varios países europeos hay sistemas expertos para apoyar las decisiones sobre préstamos personales en las sucursales de los bancos.

La empresa Renault de Francia tiene un sistema experto (Sitere) para ayudar en el diagnóstico de fallas en las transmisiones automáticas.

La American Express usa un sistema experto como apoyo de sus decisiones sobre autorización de crédito. Se ha comprobado que el sistema se desempeña un 11% mejor que los autorizadores humanos (produce decisiones correctas en 96% de los casos, en vez de en 85% que es lo que logran los expertos humanos).

La Autoridad Portuaria de Singapur tiene un sistema experto que ayuda a escoger la secuencia óptima de carga y descarga de los buques de contenedores.

En Costa Rica, un sistema experto en uso en el consultorio del Dr. Con Wong ayuda a diagnosticar dolencias del aparato digestivo.

La Escuela de Estadística de la Universidad de Costa Rica está explorando la posibilidad de armar un sistema experto para contestar consultas rutinarias sobre métodos estadísticos que les plantean investigadores y estudiantes de otros departamentos.

El Centro de Investigaciones de Ciencias del Mar (CIMAR) de esa misma Universidad está desarrollando, en colaboración con la Universidad de Maryland, un sistema experto (escrito en Prolog) para asesorar al administrador de una empresa de pesquería.

Los problemas que resuelven los sistemas expertos son pues de muy diversa índole; una posible taxonomía se debe a Waterman (1986):

1 Interpretación --formar conclusiones de alto nivel a partir de colecciones de datos brutos.

2 Predicción --proyectar consecuencias probables de situaciones dadas.

3 Diagnóstico --determinar la causa del mal funcionamiento en situaciones complejas, con base en la observación de síntomas.

4 Diseño --determinar una configuración de los componentes de un sistema que llene ciertos objetivos de rendimiento a la vez que se somete a ciertos constreñimientos.

5 Planeamiento --concebir una secuencia de acciones que alcancen un conjunto de metas a partir de ciertas condiciones iniciales.

6 Supervisión --comparar el comportamiento observado de un sistema con el comportamiento esperado.

[11] Este punto es de interés para el creador de "software", más que para el usuario interesado en sistemas particulares.

7 Reparación --prescribir remedios para desperfectos.

8 Instrucción --descubrir y compensar deficiencias en la comprensión de una materia por un estudiante.

9 Control --regir el comportamiento de un ambiente complejo.

4. Atributos Fundamentales de los Sistemas Expertos

El atributo fundamental de los sistemas expertos es su naturaleza heurística. La etimología de la palabra "heurístico" la asocia al proceso de descubrimiento; indica una característica del pensamiento que es emergente o creativa, es decir que no podemos forzar, que puede darse o no dependiendo de las circunstancias. En su aplicación a una regla, la etimología alude al carácter aleatorio de su éxito: puede resultar o no, aunque hay una probabilidad alta de que sea productiva.

Este carácter heurístico diferencia los sistemas expertos de los programas ordinarios, que son deterministas, como las hojas electrónicas ("spread sheets") en que los datos en una parte del programa determinan de manera inevitable los resultados de las fórmulas matemáticas que aparecen en otra parte de la hoja.

El calificativo de heurístico ha pasado a ser sinónimo de "basado en conocimiento" para calificar a los sistemas computacionales, y se refiere a la importancia que tiene la información en poder del sistema para orientar la búsqueda entre muchas alternativas que podrían llevar a la solución de un problema.

Con base en el hecho de que los sistemas expertos obtienen su poder de la concentración de conocimiento, podemos reconocer en ellos los siguientes atributos:

Abiertos a la inspección
tanto con respecto a las soluciones últimas que ofrecen como a los resultados intermedios.

Fácilmente modificables
tanto por la adición como por la eliminación de sus capacidades, que corresponden a reglas en la base de conocimientos.

Heurísticos
en el sentido de que usan conocimientos específicos (muy a menudo imperfectos) para guiarse en la obtención de soluciones, y en consecuencia esos conocimientos tienen un cierto grado de confiabilidad, pero no seguridad absoluta.

Un sistema experto está abierto a la inspección en cuanto que el usuario puede, en cualquier momento durante la ejecución del programa, inspeccionar el estado del razonamiento y entender el porqué de las selecciones que el programa ha hecho (que el computador lo produjo no es suficiente justificación para seguir un consejo).

Esta apertura permite, además, en algunos casos, que el usuario participe en el razonamiento y que agregue nuevos datos o reglas para mejorar el rendimiento del sistema.

La naturaleza interactiva de los sistemas expertos requiere que los programas puedan ser armados, probados y cambiados rápidamente. Ello supone ambientes que permitan gran flexibilidad para la inclusión, remoción o corrección de reglas o unidades de conocimiento.

El concepto clave aquí es *modularidad*, es decir, la condición de que la adición, corrección o remoción de una unidad no implique ninguna modificación al resto del programa.

La naturaleza heurística de los sistemas expertos implica que el cumplimiento de la función de cada regla o unidad de conocimiento no está garantizado, pues es fruto de la experiencia y no de algún proceso lógico infalible.

Las reglas deben haber demostrado su utilidad en muchos casos, y por eso se incorporan al programa, pero habrá casos en que fallen, aunque sean pocos.

El poder del sistema radica en la pluralidad de reglas, de modo que el programa acierte en sus recomendaciones por acumulación de indicios, al igual que un juez llega a una conclusión por

acumulación de pruebas, ninguna de las cuales es por sí sola suficiente para decidir el caso.

La naturaleza heurística de los sistemas expertos impone la necesidad de introducir un criterio de aceptación para sus recomendaciones: ¿consideraremos bueno el programa cuando acierta en el 98% de los casos, o en el 78%? La respuesta a esta pregunta depende de la importancia de la información que se maneje, de las consecuencias de una decisión errónea, de la relativa importancia de la rapidez en la respuesta contra la seguridad de la misma, etcétera. El criterio no puede fijarse a priori para todos los casos.

Pareciera razonable tratar de aplicar a los sistemas expertos un nivel de calidad conmensurado con el que exigimos a los expertos humanos en el mismo campo: los investigadores de la Universidad de Stanford juzgaron bueno al programa MYCIN que diagnostica enfermedades infectocontagiosas cuando fue imposible discernir sus respuestas de las de estudiantes avanzados de medicina y médicos practicantes; la compañía Digital decidió que XCON, un programa para configurar computadores VAX, era adecuado cuando su rendimiento resultó comparable con el de los ingenieros que antes realizaban el trabajo.

III Implantación

Trataremos ahora de las consideraciones que deben tenerse en cuenta durante la implantación de un sistema experto, e incluso antes de embarcarse en tal empresa.

1. Selección de Dominio

Un aspecto importante en el uso de esta tecnología consiste en la selección de problemas que sean aptos para ser solucionados por sistemas expertos.

En primer lugar, un problema es apto si no existe solución algorítmica para él (problemas del tipo que puede solucionarse con una hoja electrónica, por ejemplo, son inadecuados para sistemas expertos).

En segundo lugar, el problema debe poder ser solucionado por un experto humano, y los expertos humanos deben estar disponibles para usarse como fuente de donde extraer la base de conocimientos; un problema que ninguna persona tenga experticia para resolver no es apto para esta tecnología.

En estos momentos, a la altura del estado del arte, es también un requisito para el problema que su solución consista únicamente en manipulación simbólica (razonamiento y uso de información), no de manipulación manual o de percepción sensorial; un problema que requiera de estos últimos estaría más allá del estado del arte, puesto que la robótica y la visión artificial son áreas de la inteligencia artificial en que no se cuenta todavía con tecnologías comercialmente utilizables.

No es compatible con el estado del arte el razonamiento de sentido común, como opuesto al razonamiento experto: un sistema experto puede configurar computadores o hacer prospección geológica, porque se trata en estos casos de dominios cerrados en un área limitada del conocimiento empírico. Pero no sería adecuado confiar en un sistema experto en relación, por ejemplo, con problemas de relaciones laborales, porque el campo es vastísimo y muy poco constreñido, aunque claramente existan expertos humanos en el dominio; los sistemas expertos electrónicos funcionan bien en lo que es representable formalmente, problemas como los de las relaciones laborales difícilmente lo son (en cambio, probablemente tendría sentido armar un sistema experto que diera consejo sobre el contexto legal de la relaciones laborales, donde el campo es menor y la oportunidad para formalización más acentuada).

Finalmente, el problema debe ser de tal naturaleza que las reglas útiles para solucionarlo sean muchas y variadas, con gran independencia entre sí; el caso típico es el diagnóstico médico, donde innumerables factores contribuyen a la identificación de una enfermedad y donde esos factores pueden considerarse cada uno por aparte como materia para una regla separada que contribuye en algún grado de probabilidad al diagnóstico basándose en síntomas independientes.

2. Consideraciones de Costo y Personal

Otras consideraciones muy importantes que tomar en cuenta son la relación costo/beneficio y la disponibilidad de personal.

Desarrollar un sistema experto implica normalmente la inversión de medio año hombre de un experto humano que esté tiempo completo disponible para la extracción de conocimiento, y de otro medio año hombre de un ingeniero del conocimiento que realice la transformación de ese conocimiento en reglas de la forma adecuada.

El empresario o administrador de la empresa debe plantearse la siguiente pregunta: ¿es la tarea en cuestión suficientemente crítica en la línea de producción, suficientemente repetitiva, absorbe recursos tan abundantes como para justificar la inversión de ese tiempo de recursos personales caros en la preparación del sistema experto?

La contestación será muchas veces positiva, como lo indica la rápida proliferación de estos sistemas en los últimos años, mencionada por Feigenbaum en la última conferencia de AAAI, como se citó antes.

Otra consideración importante se refiere al tipo de disponibilidad del experto humano: ¿son muy abundantes las personas de esta clase, o por el contrario la empresa tiene que adiestrarlas a lo largo de muchos años de experiencia, con el consiguiente riesgo de pérdida de la inversión por muerte, retiro o renuncia? ¿Es posible disponer del experto sólo por períodos cortos -por ejemplo, importándolo del extranjero mientras que la necesidad que cubre es permanente y extendida? ¿Tiene la empresa uno o dos expertos humanos que constantemente deben desplazarse a distintas instalaciones geográficamente separadas para cumplir su función? ¿Puede en este último caso un sistema de red electrónica proyectar su presencia a distintos lugares simultáneamente, o sería más práctico replicar la experticia con la tecnología de los sistemas expertos?

La construcción de un sistema experto hace uso esencial tanto de un *experto humano* como de un *ingeniero del conocimiento*. Este último sería idealmente un experto en lenguajes de Inteligencia Artificial (lenguajes computacionales de alto nivel, ver Apéndice B) y en las técnicas de representación del conocimiento.

Más modestamente, el ingeniero del conocimiento puede ser un graduado en ciencias de la computación con adecuada preparación general en Inteligencia Artificial y suficiente experiencia o capacidad en la utilización de conchas de sistemas expertos.

La función de esta persona es recomendar el "software" y el "hardware", extraer el conocimiento del experto humano, e implantar ese conocimiento en forma correcta y eficiente en una base de conocimientos. La persona en cuestión no necesita ser versada en el dominio de la aplicación, puesto que deberá contar con la contribución dedicada del experto humano.

En nuestro país es razonable pensar que la Escuela de Ciencias de la Computación e Informática de la Universidad de Costa Rica, así como la correspondiente dependencia del Instituto Tecnológico de Costa Rica, han producido ya un suficiente número de graduados con un nivel adecuado de preparación para servir decentemente esta función.

Unas palabras sobre el experto humano en el dominio de la aplicación. Normalmente será una sola persona, aunque pueden ser varias que complementen su experticia entre sí. Debe haber trabajado en el área de aplicación por bastante tiempo, suficiente para merecer confianza de la empresa en su juicio experto. Debe comprender adecuadamente las técnicas de resolución de problemas en el dominio, incluyendo el uso de atajos en el razonamiento y el manejo de datos imprecisos o incompletos. Debe también tener la capacidad para evaluar soluciones parciales, y en general todas las habilidades que normalmente asociamos con un experto.

La responsabilidad del experto en el desarrollo del sistema está en articular las reglas de su oficio, de la manera más clara y completa posible. Aunque algunos teóricos sostienen que gran parte del conocimiento de un experto es inarticulable (no puede expresarse en forma de reglas), la experiencia de una década de confección y uso de sistemas expertos ha demostrado palpablemente que un ciclo iterativo de consulta, implantación y prueba hace posible extraer si no el 100% del conocimiento inicialmente inarticulado del experto humano, digamos por ejemplo el 95% de ese conocimiento.

3. El Proceso de Adquisición de Conocimiento

El proceso de formación de la base de conocimientos para una aplicación concreta (adquisición de conocimiento) opera así: en primer lugar, el ingeniero de conocimiento entrevista al experto humano tratando de extraer de él o ella toda la información posible sobre su manera de resolver los problemas dentro del dominio de experticia.

El hecho de que el ingeniero sea básicamente ignorante del campo de aplicación resulta beneficioso, en el sentido de que hará preguntas ingenuas que obligarán al experto del dominio a exteriorizar una gran cantidad de conocimiento implícito y que normalmente sería incapaz de percibir, dado su carácter obvio para sí mismo.

En seguida el ingeniero formula esa información en el formato de reglas de condición/consecuencia. A continuación, el programa intérprete (la concha seleccionada para encarnar el sistema) es ejecutado, contra la base de conocimientos

recientemente formada con el conocimiento extraído del experto.

Para correr el programa, debe suministrársele al neófito sistema experto un problema de alguna complejidad para que trate de resolverlo. La solución obtenida (o la imposibilidad de obtenerla) es entonces sometida al análisis del experto humano, que debe evaluar su calidad. Si, como es de esperar al principio, la calidad es deficiente, o el resultado totalmente erróneo, el experto humano se sentirá estimulado por ese fracaso a enunciar una nueva regla que es necesaria para cubrir la situación y que todavía la base de conocimientos no contiene, o para detectar qué defectos tienen las reglas formuladas hasta el momento.

Hecha la corrección por el ingeniero en la emergente base de conocimientos, de conformidad con las indicaciones del experto humano, el programa deberá ejecutarse de nuevo, se producirán nuevos resultados que de nuevo serán evaluados por el experto humano, y así sucesivamente, a lo largo de muchas reiteraciones, durante las cuales el sistema mejorará y ampliará en su cobertura de cada vez más complicados casos dentro del dominio de la aplicación.

Todo esto es posible gracias a la modularidad de la arquitectura del sistema en construcción, como se explicó anteriormente.

El usuario final del sistema tiene también responsabilidades en este desarrollo, a saber: debe definir muy bien el dominio de la aplicación, explicitar el nivel de explicación de parte del programa que se requiera en las interacciones, determinar qué tipo de información el sistema puede inquirir del usuario durante la ejecución del programa, y en general qué tipo de preguntas podrán o deberán hacerse en los dos sentidos (del computador para el usuario y del usuario para el computador) durante el proceso concreto de resolución de un problema.

4. El Papel de las Conchas en el Desarrollo de un Sistema Experto

Se encuentran en el comercio programas especiales diseñados para ayudar en la construcción de un sistema experto, llamados conchas posiblemente porque sirven de apoyo al sistema mismo, representado por la base de conocimientos. Esencialmente, las conchas proveen de un marco dentro del cual el usuario puede insertar los detalles de su aplicación.

Teóricamente no se requiere conocimiento especializado para usar las conchas, con tal de que el usuario disponga de tiempo suficiente para estudiar los manuales y experimentar con los ejemplos que los fabricantes proveen. En la práctica, estudio o experiencia anterior pueden ser más que una ayuda para asimilar los manuales y tener éxito en la implantación concreta del sistema.

Aunque hay conchas con precios hasta de $9.000 y más, es posible encontrar algunas de buena calidad con precios inferiores a los $500, como puede colegirse de los cuadros que se acompañan a la respectiva sección.

IV Funcionamiento Formal

1. Sistemas de Reglas de Producción

La inmensa mayoría de los sistemas expertos actualmente en existencia pueden describirse, desde el punto de vista de su operación formal, *como sistemas de reglas de producción.*

Un sistema de reglas de producción es una disciplina de control computacional muy diferente a la organización jerárquica ejemplificada por los programas convencionales, en que un programa principal invoca a programas secundarios o subrutinas, que a su vez invocan a otros programas más elementales, y cada uno regresa resultados al programa superior que los emplea para continuar sus cálculos. Decimos de esos programas tradicionales que son jerárquicos, por esa tendencia a distribuir el trabajo entre todo un árbol burocrático de subordinados.

Por el contrario, en un sistema de reglas de producción lo que para el caso podemos considerar como programa principal es el intérprete o aparato de inferencia (la concha); su responsabilidad es poner en contacto los datos de la memoria de trabajo con las reglas de la memoria de largo plazo. La comunicación entre los procedimientos o reglas ocurre aquí en forma pública, mediante la memoria de trabajo, donde están los datos del problema y donde las reglas encuentran sus elementos activadores y dejan las consecuencias de su acción.

La memoria de trabajo actúa como el medio de comunicación, un medio muy estrecho, entre todas las reglas. Ninguna regla tiene autorización para comunicarse con otra regla, a diferencia de lo que sucede en la programación tradicional, en que un programa puede llamar a otro, que para el caso actúa como subprograma. Las reglas se activan no por órdenes superiores, como en una jerarquía, sino en forma oportunista: siempre que las condiciones de sus lados izquierdos calcen con algún dato existente en la memoria de trabajo.

Alguien ha comparado la forma en que se comunican las reglas en un sistema de reglas de producción con la forma en que compradores y vendedores de ocasión se comunican entre sí por medio de los avisos económicos de los periódicos.

Un elemento indispensable del sistema es el *intérprete*, que en cada ciclo pone en relación cada regla de la memoria de largo plazo con cada dato de la memoria de trabajo, a fin de seleccionar un conjunto de reglas que calcen con los datos. Entre ellas deberá escoger solamente una para que ejecute en el próximo ciclo.

La forma en que selecciona esa regla, escogiéndola de entre todas las reglas que calzan con los datos de la memoria de trabajo, se puede realizar de muy diversas maneras: en las conchas más elaboradas, por medio de otro sistema de reglas de nivel superior; más corrientemente, por decisión prefijada en el programa del intérprete de acuerdo con algún criterio que se considera importante, normalmente la circunstancia de que la regla (o el dato) ha sido usada recientemente, o que la regla es más específica que sus competidoras.

Los sistemas de reglas de producción usan una de dos estrategias en el control del razonamiento: predominio de los datos o predominio de las metas.

En el predominio de los datos, cada vez que un nuevo dato es agregado a la memoria de trabajo, las reglas deducen todas las consecuencias útiles que de este dato se pueden deducir, agregándolas como nuevos datos a esa misma memoria; el efecto se propaga recursivamente hasta que no haya más consecuencias que deducir.

En el predominio de las metas, lo que se agrega en la memoria de trabajo es una meta por realizar (una hipótesis por probar); la primera de ellas corresponde a la consulta del usuario, o sea al planteo del problema.

Alguna regla tendrá como consecuente una información que calza con esa meta, lo cual será señal suficiente para que el intérprete trate de realizar las submetas contenidas en el respectivo antecedente, es decir, las coloque en la memoria de trabajo como un nuevo dato; el intérprete actuará entonces recursivamente para probar que esas submetas son obtenibles, mediante la

activación en cadena de otras reglas por este mismo procedimiento.

Distintas aplicaciones pueden beneficiarse de las dos estrategias; el control de procesos fabriles, por ejemplo, admite mejor una estrategia de encadenamiento hacia adelante o razonamiento con predominio de los datos. Cada vez que pasa algo en la fábrica, las reglas que tengan eso como condición son activadas.

En problemas de diagnóstico, en que el modo hipotético es la forma de razonar más natural, el predominio de las metas o encadenamiento hacia atrás (desde las metas) es lo más indicado.

En general, se debe usar predominio de datos (encadenamiento hacia adelante) cuando la colección de datos es pequeña y se espera que muchos datos lleven a pocas conclusiones. Se debe usar predominio de las metas (encadenamiento hacia atrás) para aplicaciones con muchos datos, donde cada dato puede llevar a muchas conclusiones.

Hacer lo siguiente cada vez que se añade un nuevo hecho a la memoria de trabajo:
1 Revisar la base de conocimientos para encontrar todas las reglas que usan este hecho en su antecedente.
2 Escoger de esas reglas las que estén listas para disparar y marcarlas como activas. En las reglas que tengan otras condiciones, marcar el hecho como cumplido para facilitar el disparo de la regla cuando se den esas condiciones.
3 Disparar una a una las reglas marcadas como activas, interrumpiendo para volver a hacer este proceso cada vez que se agregue una nuevo hecho a la memoria de trabajo como resultado de la ejecución de las reglas.
4 El algoritmo termina cuando no hay más reglas activas ni nuevos hechos agregados a la memoria de trabajo.

Algoritmo de encadenamiento hacia adelante (desde los datos)

Normalmente un sistema con encadenamiento hacia atrás pedirá al usuario información durante la ejecución del proceso ("¿tiene fiebre el enfermo?", por ejemplo). Un sistema con encadenamiento hacia adelante, en cambio, deberá contar con todos los datos al comienzo del proceso (las lecturas de los indicadores de presión, temperatura, etcétera en una planta química supervisada por un sistema experto, por ejemplo).

1 Obtener de la base de conocimientos la lista de reglas cuyo consecuente unifica con el problema en curso.
2 Si no hay reglas, preguntar al usuario por la solución al problema en curso.
3 Si las hay, por cada una de ellas tomar las condiciones en orden, considerarlas una por una como problema en curso y ejecutar este mismo algoritmo desde el comienzo (cada vez que reiniciamos el algoritmo decimos que descendemos un nivel de recursión).
4 Si se agotan las condiciones de una regla sin obtener éxito, tomar la siguiente regla del mismo nivel y hacer lo mismo; si se agota el nivel, subir al nivel anterior y hacer lo mismo, y así repetidamente hasta resolver el problema principal o rendirse.

Algoritmo de encadenamiento hacia atrás (desde las metas)

El Apéndice A ofrece más detalles sobre el funcionamiento formal del intérprete, en especial la forma en que trabajan las dos estrategias de control.

¿Por qué usar reglas de producción?

La estructura formal de los sistema de reglas de producción ofrece un buen fundamento para algunos de los requisitos esenciales a los sistemas expertos. Veamos:

Un sistema experto debe ser *fácilmente modificable*; por su parte, un sistema de reglas de producción se compone de reglas independientes, cada una de las cuales se puede introducir, retirar o cambiar sin afectar el resto.

Un sistema experto debe ser *heurístico*; por su parte, un sistema de reglas de producción se presta para introducir factores de incertidumbre en las hipótesis, que la ejecución de las reglas puede combinar entre sí a fin de producir el índice de confiabilidad de una hipótesis.

Finalmente, un sistema experto debe estar *abierto a la inspección* por el usuario; por su parte, la naturaleza conectiva de la inferencia por reglas de producción, en que una meta se conecta con submetas y éstas con otras submetas, etcétera, hace posible dar explicaciones por medio del simple expediente de seguir los punteros direccionales que conectan todas esas estructuras en la memoria del sistema.

Operación con índices de confiabilidad

Las reglas de un sistema experto pueden ser dotadas de un índice de confiabilidad que exprese el grado de firmeza con que el constructor de la base de conocimientos confía en el contenido de la regla. Tales índices son números entre 1 y 0, donde la cercanía al 1 indica mayor confiabilidad. Además, el usuario del sistema puede normalmente introducir sus datos con una indicación del grado de certidumbre con que los acepta.

Los índices de las reglas pueden aparecer en dos distintas posiciones: o como confiabilidad de la regla en sí, o como confiabilidad de alguna o todas las premisas de la regla. Por ejemplo, una regla puede expresar la idea de que, con 80% de confiabilidad, si hay una confiabilidad del 90% de que la batería esté descargada, el motor no arrancará.

En todo caso, el intérprete combina los grados de confiabilidad con que tenga que lidiar para ofrecer sus resultados también bajo un índice que señale el grado de confiabilidad de la respuesta.

Los sistemas expertos usan distintos métodos para realizar el cómputo de índices de confiabilidad, siendo los más usados la teoría bayesiana de probabilidad y los factores de certeza de Stanford; unos pocos sistemas usan una variación de la teoría de conjuntos borrosos de Zadeh ("fuzzy sets").

2. Programación Libre

La metodología de reglas de producción, con todo y ser de uso generalizado, no es la única posible para la construcción de sistemas expertos. La literatura de Inteligencia Artificial está llena de sistemas escritos directamente en alguno de los lenguajes especiales para esta disciplina (ver Apéndice B), sobre temas muy diversos: aprendices de programador que ayudan al profesional en tareas rutinarias, agentes de viajes, detectores de fallas en el diseño de circuitos eléctricos, y hasta un Matemático Artificial (AM) que descubre teoremas nuevos usando las heurísticas que emplean los matemáticos para hacer sus descubrimientos.

Casi todos estos programas son ítemes de laboratorio. Una notable excepción es MACSYMA, escrito en Lisp, que es un sistema experto de uso generalizado para la manipulación de expresiones algebraicas.

Un empresario puede contratar a un programador excelente, con experiencia en las técnicas de la Inteligencia Artificial, y encargarle el diseño de un sistema a la medida de su tipo de problemas. Sin embargo, no es un curso de acción recomendable, por su costo y riesgo, excepto para empresas poderosas dispuestas a hacer inversiones cuantiosas en investigación y desarrollo.

La alternativa económica es usar las conchas comerciales que se comentan en la próxima sección, todas las cuales trabajan dentro de la metodología de las reglas de producción.

3. Razonamiento por Casos

A pesar de la difusión de la metodología de reglas, un enfoque alternativo que recientemente ha comenzado a ser explorado por la comunidad de Inteligencia Artificial es el razonamiento por casos. En efecto, en el año 1988, por primera vez se celebró un taller sobre este tipo de enfoque durante la conferencia anual de la AAAI. El autor de este informe presentó un trabajo en ese taller, al cual se hará referencia en la secuela.

Una analogía puede aclarar el contraste entre el razonamiento por reglas y el razonamiento por casos. Considérese el sistema jurídico latinoamericano, basado en la idea de codificación, de raigambre napoleónica, en

contraste con las instituciones legales de los países anglosajones, basadas en la tradición o sistema de la ley común ("common law").

En nuestros países las leyes son dictadas solamente por el órgano legislador (la Asamblea Legislativa) y tales leyes tienen el carácter de reglas de condición/consecuencia ("quien se encontrare en tal y tal situacion, tendrá tales y tales derechos u obligaciones").

En los países anglosajones los tribunales de justicia resuelven las contiendas apoyándose primordialmente no en la ley dictada formalmente por un parlamento o congreso, sino con base en los precedentes, es decir, en los casos resueltos con anterioridad.

La tecnología de los sistemas expertos busca fundamentalmente aprovechar la experticia acumulada por la experiencia. Esa experiencia puede ser formulada en reglas, que de ordinario son reglas inconscientes que el experto aplica automáticamente. Es incluso necesario un esfuerzo, y la ayuda del ingeniero del conocimiento, para que el experto pueda articular esas reglas.

La metodología alternativa del razonamiento por casos trata de usar la experiencia acumulada de manera directa, como fuente de casos que se aplican a la nueva situación de manera directa, sin intervención de reglas.

El problema teórico con este enfoque es que los casos son muchos más que las reglas, siendo la ventaja teórica que los casos calzan más directamente con la nueva situación, sin intervención de cadenas de razonamiento.

La aparición reciente de máquinas con memorias cada vez más grandes ha hecho que esta metodología parezca prometedora, pues es "intensa en memoria", mientras que la metodología alternativa es "intensa en cómputo".

La metodología es nueva en Inteligencia Artificial, y aun no ha sido aplicada de manera sistemática a la tecnología de sistemas expertos. Intentaremos hacerlo aquí, tratando de compararla con la metodología rival de las reglas de producción.

La cuestión es cómo organizar y poder utilizar la experiencia obtenida con casos del pasado para resolver problemas del presente. Podemos postular, como en el caso de los sistemas de reglas, dos memorias y un intérprete.

El papel de las reglas lo representan los casos, de modo que la memoria de largo plazo esta formada por los casos experimentados en el pasado, puestos en una conveniente representación.

La similitud llega incluso hasta la estructura de la unidad de conocimiento que se almacena en la memoria de largo plazo; pero en vez del par condición/ consecuencia de las reglas, postulamos como forma de representación un par identificador/descriptor. Cada caso se identifica con un identificador significativo (veremos en seguida algunos ejemplos) y en forma contigua se describe de la manera más cuantitativa posible, según la naturaleza del problema.

El autor de este informe tiene en su haber el diseño de dos sistemas expertos que siguen básicamente esta metodología. El primero de ellos fue implantado y ha estado en uso por muchos años en la Universidad de Costa Rica, donde se ha empleado para predecir el crecimiento de las distintas Escuelas y conceder becas para preparar en el extranjero a los profesores necesarios para hacer frente a ese crecimiento.

El programa fue escrito en 1970, originalmente en lenguaje ensamblador, se llama SIMULA (sin relación con el lenguaje de computación del mismo nombre), que significa "Simulación de una Universidad Latino Americana".

En SIMULA los casos acumulados en la base de conocimientos son compendios de la manera en que los estudiantes han distribuido sus créditos a lo largo de sus carreras entre las distintas escuelas de la universidad; la información es fácilmente obtenible, pues es computadorizada de todas maneras para otros propósitos.

El identificador en este caso son las siglas que distinguen los cursos de las distintas escuelas, y el descriptor contiene los números de créditos impartidos bajo esa sigla en los distintos años calendario.

Como en el año 70 la memoria no era abundante, especialmente en el computador de la Universidad de Costa Rica, los casos eran compendiados (promediados) por un programa preparatorio.

Una versión moderna del programa tendría que basarse en el almacenamiento directo de los casos.

Eso es precisamente lo que se propone en el diseño de otro sistema experto, preparado a principios de 1988 y aceptado como ponencia en el mencionado taller de la AAAI, para predecir el tamaño de las cortas en una plantación de banano, con base en información sobre cortas anteriores.

En este programa, llamado BANANA, el identificador de los casos es una etiqueta que expresa las condiciones climáticas imperantes en un período de diez semanas; el descriptor enuncia los números de cajas de banano cortados en las tres semanas siguientes a esas diez, para la misma cohorte de plantas (plantas paridas en la misma semana, que se identifican en la finca con una cinta de cierto color).

La acción del intérprete se reduce en lo esencial a combinar los valores de corta proyectados para las distintas semanas, tomándolos de los datos computados para las distintas cohortes, para lo cual se usa un algoritmo muy sencillo.

Las posibilidades de este enfoque metodológico están todavía por explorar. Lo único claro es que requiere más memoria y menos tiempo de cómputo que la metodología de reglas. No es impensable, incluso, la creación de conchas que encarnen la metodología, a fin de dar al usuario la misma facilidad de trabajo que le ofrece la metodología alternativa.

Otro punto por explorar es el rango de aplicación del enfoque metodológico del razonamiento por casos. Es presumible que, si ha resultado aplicable a casos tan disímiles como la estimación de necesidades de profesores y la predicción del tamaño de las cortas de banano, sea aplicable también a multitud de otros muy variados dominios.

V Conchas Comerciales

1. Introducción

Existen en el mercado diferentes conchas para el desarrollo de sistemas expertos que pueden ejecutarse en muy diversas máquinas.

Distintas conchas ofrecen diversos métodos de razonamiento, que pueden brindar ventajas particulares para determinadas aplicaciones; por ejemplo, hay algunos intérpretes que dan la alternativa de "encadenamiento hacia adelante", además de la capacidad más generalizada de "encadenamiento hacia atrás".

A continuación se presenta un resumen de las conchas más notables en el comercio, según la literatura técnica más reciente. Todas aplican encadenamiento hacia atrás como estrategia de control básica y, en consecuencia, son interactivas.

2. Conchas más Populares

ESIE, de Lightwave; una disqueta con 19 archivos. Incluye documentación que consiste en manual, tutor, manual para el ingeniero de conocimiento, historia de la inteligencia artificial y guía para el novicio (usuario no familiarizado con sistemas expertos).

Precio: $ 145.

Requiere MS DOS 2.0 o mayor.

RAM: 128 kbs (se recomienda 256). Requiere procesador de texto que produzca archivos ASCII.

La base de conocimientos puede aceptar hasta mil líneas (entre 300 y 500 reglas).

La memoria de trabajo puede aceptar hasta 200 hechos.

El sistema no tiene módulo de adquisición de conocimiento (se usa un procesador de texto externo), ni módulo de explicación (lo sustituye el mecanismo de "trace").

Solamente admite cuatro comandos: Go para empezar una consulta; Exit para terminarla; Trace On para observar el razonamiento; Trace Off para no verlo.

ESIE es un buen sistema para alguien que quiera ensayar con sistemas expertos y tenga una aplicación sin reglas complejas.

EXSYS (versión 3), de Exsys Inc., es más poderoso.

Requiere 320 kbs. de RAM.

Puede ejecutar hasta 5.000 reglas, absorbiendo 64 kbs. por cada 700 reglas (suponiendo un promedio de siete condiciones por regla).

En un computador VAX prácticamente no hay límite en el número de reglas que puede ejecutar.

Precio de la versión profesional: $795 para la computadora personal, $12.500 para la VAX con derecho a 16 usuarios.

Tiene módulo de explicación que contesta preguntas de "por qué" y de "qué pasaría si"; las primeras muestran la cadena de reglas usada para lograr la respuesta, mientras que las segundas permiten al usuario cambiar las respuestas dadas por él durante la sesión de consulta.

Es posible especificar probabilidades en las reglas, hacer comparaciones numéricas en las condiciones y realizar computaciones variadas en el consecuente.

Las reglas pueden plantear dos tipos de preguntas: selección múltiple o numéricas.

El sistema tiene la capacidad de interactuar con otros programas para recibir datos o realizar computaciones, o incluso para exponer resultados.

Tiene módulo de adquisición de conocimiento.

Ofrece flexibilidad para emplear encadenamiento hacia atrás o hacia adelante, así como muchas otras capacidades que lo convierten en una buena

elección para el que quiere llegar a tener un sistema grande que se desarrolle a partir de un sistema de escala menor.

Personal Consultant de Texas Instruments, ofrece dos modelos básicos.

Personal Consultant Easy, el modelo simple, con un precio de $ 495, requiere 512 kbs. en una micro PC o XT.

Personal Consultant Plus, el modelo más desarrollado, con un precio de $ 2.950, requiere 640 kbs. más 2 mbs. de memoria extendida en una micro AT.

Su forma normal de inferencia es encadenamiento hacia atrás, aunque puede realizar también encadenamiento hacia adelante.

Acepta índices de confiabilidad.

El modelo Plus incluye un mecanismo de selección de reglas basado en meta reglas, o sea reglas de superior nivel que resuelven conflictos entre las reglas básicas para decidir cuál debe activarse en el próximo ciclo.

La documentación es muy amplia y de muy buena calidad.

El módulo de explicación consiste básicamente en un rastreo ("trace"), y el módulo de adquisición de conocimientos está orientado al ingeniero de conocimiento y no al experto humano.

Estas conchas tienen como mayor ventaja la posibilidad de pasar del modelo menor al mayor; si se hace dentro de los seis meses de la compra de PC Easy, el precio pagado se descuenta del precio de PC Plus.

VP Expert (versión 1.2), de Paperback Software, es una de las conchas más representativas de la nueva generación de software para computadores personales.

Requiere 384 kbs. de RAM y MSDOS 2.0 o mayor. Pueden construirse bases tan grandes como con Exsys, con la ventaja de ser eminentemente compatible con otros productos para computador personal.

El precio oscila entre $ 65 y $ 100 de acuerdo con el vendedor.

El programa tiene un límite de 20 condiciones por regla y 20 niveles de encadenamiento.

Tiene la capacidad de encadenar bases de conocimientos, como una forma de superar la limitación de memoria, con base en alternar las bases de conocimiento en la memoria de largo plazo con continuidad en el contenido de la memoria de trabajo.

Tiene módulo de adquisición de conocimiento (con capacidad para producir reglas automáticamente a partir de tablas) y módulo de explicación, con capacidades parecidas a Exsys.

Acepta índices de certidumbre.

Es posible interconectar una ejecución del sistema con consultas basadas en datos u hojas electrónicas.

En suma, es un paquete excelente, especialmente adecuado para personas familiarizadas con los sistemas de menú con base en barras luminosas, típicos de los computadores personales.

La documentación es excelente, y en general tiene una potencia comparable a la de sistemas de mucho mayor precio.[12]

3. Revista Especializada

Conviene hacer notar la existencia de una revista especializada de especial interés para constructores y administradores de sistemas expertos. Es Expert Systems Strategies, una hoja noticiosa mensual de la Cutter Information Corporation. La dirección postal de su oficina de circulación es: 1100 Massachusetts Avenue, Arlington, MA 02174 U.S.A. Teléfono: (617) 648-8700. Fax: (617)648 8707.

12 El autor de este informe donó una copia de VP Expert a la Vicerrectoría de Investigación de la Universidad de Costa Rica, donde probablemente podría ser consultada.

A continuación se incluye un panorama más general sobre el estado del mercado, con clasificación por el tamaño de las conchas.

4. Conchas para Micros

Estas herramientas están dirigidas primordialmente a no programadores para desarrollar sistemas desde pequeños hasta medianos. La mayor parte de estos productos son adquiridos por personas que quieren explorar el uso de la tecnología.

En muchos casos las empresas estimulan el desarrollo de pequeños sistemas por muchos de sus empleados, para lo cual les procuran herramientas con licencias para uso múltiple.

Un rasgo del desarrollo reciente es la aparición de sistemas basados en la MAC, que vienen a hacer compañía a los sistemas más viejos para las IBM PCs y compatibles.

A mediados del año 1988 las conchas pequeñas de más venta eran Personal Consultan Easy y VP Expert.

CONCHA	COMPAÑIA	MICRO	LENGUAJE	PRECIO $
Cognate	Peridom	MAC	C	150
Crystal	Intelligen Environments	IBM	C	695
ESP Advisor	Expert Systems International	IBM	Prolog	895 a 3.000
ExperOPS5+	Expertelligence	MAC	Lisp	495
ExperOPS Common5	Expertelligence	MAC	Lisp	625
Expert Edge	Helix Exp. Sys	IBM	C	795
Humble	Xerox	IBM, MAC	Smalltalk	400 a 1.500
Instant-Expert+	Human Intellect Systems	MAC	Modula II	495
MICE	Machine Intelligence	IBM	C	750 a 9.000
P.C. Easy	Texas Instrum .	IBM	C, Scheme	495
VP Expert	Paperback Soft.	IBM	C	99
XI Plus	Expertech	IBM	M. Prolog	1.995

Conchas de Sistemas Expertos para Microcomputadores

5. Conchas para Minis y Estaciones de Trabajo

Este es el nicho más grande del mercado. Uno de los programas más populares en esta clasificación es Personal Consultant Plus. Exsys es un competidor fuerte.

CONCHA	COMPAÑIA	COMPUTER	LENGUAJE	PRECIO $
Exsys	Exsys Inc.	Vax, IBM-AT	C	795 a 12.000
GoldWorks	Gold Hill Comp.	Sun, IBM-AT	Lisp	7.500
Guru	Micro Data Base Systs.	MicroVax	C	6.500
Kee	Intellicorp	Vax, Sun, 386	Lisp	9.900 a 30.000
Laser	Bell Atlantic	MicroVax, Sun IBM-AT, MAC-II	C	5.000 a 25.000
LevelFive	Info. Builders	Vax, IBM-AT	Pascal	9.600 a 28.000
M.1	Teknowledge/ Framentec	IBM-AT	C Prolog	5.000
Nexpert	Neuron Data	IBM-AT, MAC	Assem	5.000
OPS5+	Computer Thought	Sun, IBM-AT	C	1.800
OPS/83	Production Syst. Techn.	IBM-AT Vax, Apollo	C	1.900 a 20.000
Personal Cons. Plus	Texas Instr.	Vax, IBM-AT Explorer	Scheme C	2.950
Rulemaster	Radian Corp.	Vax, IBM-AT	C	495 a 28.000
SD Advisor	Sys. Designers	Vax	Prolog	2.000 a 5.000
VaxOPS5	Digital Equip.	Vax	C	7.500
Wisdom XS	Machine Intel.	Apollo	C	5.000

Conchas de tamaño mediano para Minis, ATs, 386s y Estaciones de Trabajo

6. Conchas para Gigantes (Computadores Centrales)

Este nicho está a punto de quebrarse en clasificaciones más finas. Por lo pronto, el ganador ha sido ADS de Aion, seguido de cerca por ESE de IBM. El mercado es cambiante y se esperan sorpresas para 1989.

CONCHA	COMPAÑIA	SIS. OPERATIVO	LENGUAJE	PRECIO $
Aion Dev. System	Aion Corp.	MVS/CICS, TSO IMS, VM/CMS	Pascal	70.000
AE (Application Expert)	Cullinet S.	MVS/CICS	Cobol	41.000 a 60.000
ESE (Expert System Environ.)	IBM	MVS/TSO VM/CMS	Pascal	21.000 a 42.500
KBMS	AICorp.	VM/CMS MVS/XA	C	90.000 a 200.000
KES II	Unisys	NOS/VE, MVS/TSO	C	60.000
TIMM	General Research	MVS/VM	Fortran	1.900 a 19.000
TWAICE	LogicWare	VM/CMS	Prolog	48.900

Conchas de Sistemas Expertos para Computadores Centrales

VI Evaluación de las Conchas de Sistemas Expertos

El uso de sistemas expertos está aumentando en la industria en una amplia variedad de aplicaciones. La promesa fundamental de esta tecnología es la habilidad de desarrollar programas de computación que capten el conocimiento y la estrategia de resolver problemas de los expertos humanos.

Los sistemas expertos se desarrollan normalmente usando herramientas de software llamadas conchas de sistemas expertos. Estas conchas ofrecen mecanismos que las distinguen de los lenguajes de cómputo tradicionales y hacen practicable el desarrollo de aplicaciones concretas de la tecnología, a bajo costo.

1. Introducción

Lo que las conchas prometen al usuario es específicamente lo siguiente:

1. Facilidad de representación del conocimiento.

2. Separación entre base de conocimientos y control.

3. Facilidad de incorporar y corregir conocimientos.

4. Capacidad de explicar los razonamientos.

5. Capacidad de lidiar con conocimiento incierto.

6. Capacidad de lidiar con información incompleta.

Es ya posible evaluar hasta qué punto las conchas que están en el mercado han podido llenar estas expectativas.

Como fue explicado antes, los sistemas expertos consisten fundamentalmente de dos partes: la base de conocimientos, que es un conjunto de reglas creadas por el diseñador de la aplicación, y el intérprete que selecciona y ejecuta las reglas.

La base de conocimientos es pasiva y requiere de la segunda parte, el intérprete, que incluye un aparato inferencial para hacer asociaciones y pareamientos entre los símbolos usados en las reglas y en los datos del problema concreto.

Mediante estas asociaciones, el aparato inferencial navega por la base de conocimientos buscando satisfacer la meta introducida por el usuario. Encontrar esas asociaciones es la base de las técnicas de inferencia que se denominan encadenamiento hacia atrás y encadenamiento hacia adelante.

Aunque las conchas comerciales difieren en muchos aspectos, su denominador común es el estar basadas en la estrategia inferencial de encadenamiento hacia atrás; sólo en contados casos las conchas poseen, además, capacidades de encadenamiento hacia adelante.

La razón de la preferencia de las conchas por el encadenamiento hacia atrás (predominio de las metas) es definitiva: el encadenamiento hacia adelante es dispensioso de recursos, pues produce muchas asociaciones no atinentes que consumen memoria y tiempo sin conducir a la solución de ningún problema interesante. En cambio, el encadenamiento hacia atrás, por partir de la meta buscada, es eminentemente dirigido y mucho más económico en recursos.

Los usuarios de las conchas varían mucho en destreza y preparación, desde aquellos que no tienen adiestramiento alguno en programación hasta los *ingenieros del conocimiento*, especializados en el desarrollo de sistemas expertos de dimensiones importantes. Como se dijo anteriormente, el empleo de un sistema experto es indicado siempre que se necesite una aplicación para duplicar, sustituir o complementar la acción de un experto humano en un campo determinado y limitado de conocimiento.

La selección concreta de una concha depende de la aplicación que se tenga en mente, pero definitivamente incluye la consideración de tres factores:

* la destreza de los usuarios, tanto del diseñador potencial del sistema como de su operador, en la solución concreta de problemas;

* la dificultad y complejidad de dominio de la aplicación;

* y las características particulares de la concha en cuestión.

Pero cualquiera que sea la selección, el usuario tiene derecho a esperar que las promesas identificadas arriba sean cumplidas ampliamente.

2. Representación de Conocimiento

Lo primero que se espera que la concha ofrezca es la oportunidad de formular el conocimiento del experto de manera cómoda y sin ambigüedades o complicaciones innecesarias.

Hay una inmensa diferencia entre tener que escribir un programa en lenguajes tradicionales como Fortran o Cobol para representar una determinada capacidad de un experto, y simplemente formular de manera declarativa[13] y en unidades discretas cada aspecto de la experticia respectiva.

La forma de representación ofrecida por las conchas es declarativa y no procedimental, no es un programa sino una proposición que describe "cómo es el mundo", en un ropaje simbólico muy cercano a la forma en que el experto normalmente expresaría su conocimiento.

El formato de reglas, con una parte formada por condiciones que deben cumplirse y otra parte formada por hipótesis que quedan confirmadas o acciones que son recomendadas, ofrece una apariencia más simple, clara y robusta que un programa escrito en un lenguaje de cómputo tradicional.

Las reglas deben ser válidas por sí mismas, independientemente de ningún lenguaje, y así debe poder pensarlas el experto humano; crear una base de conocimientos, entonces, será más parecido a fabricar una base de datos que a escribir un programa.

La diferencia entre una base de datos y una base de conocimientos estriba en que en una base de datos se especifica toda la información en forma de hechos debidamente formateados; en una base de conocimientos, en cambio, se formulan además reglas que pueden considerarse hechos condicionales, que sólo son verdaderos cuando sus condiciones se cumplen; en general, una base de conocimientos contiene muchísimo más información que una base de datos, pues las reglas representan información implícita que solamente se hace explícita cuando se producen las circunstancias adecuadas.

No obstante, de acuerdo con algunos estudiosos (por ejemplo Edwin J. Miller, ver bibliografía), la insistencia en el carácter declarativo de las reglas no nos debe hacer perder de vista que, en todo caso, el conjunto de reglas articuladas por el experto humano debe funcionar como un programa; por ejemplo, debe ser depurado como se depuraría un programa, ejecutándolo muchas veces con datos de prueba variados y representativos.

Pero en todo caso, siempre será cierto que es mucho más fácil tratar de arreglar un programa que consiste de reglas transparentemente claras, independientes entre sí, y expresadas en una forma muy cercana al lenguaje común, que tratar de corregir errores en un lenguaje como Fortran, Cobol o Pascal en que uno debe examinar cada paso del programa.

En un sistema experto uno se preocupa solamente por la veracidad de las reglas y su potencial para crear conexiones lógicas que le permiten contestar preguntas importantes dentro del dominio de que se trate.

En suma, es claro que las conchas liberan al programador de la carga de escribir los detalles de funcionamiento del intérprete o aparato de inferencia, así como de los detalles de la interface entre el usuario y el sistema; no lo liberan de la necesidad de programar. Su labor, sin embargo, facilitada por el carácter lúcido, modular, y

[13] La oposición entre lo declarativo y lo procedimental es tradicional tanto en la psicología del conocimiento como en las ciencias de la computación. Básicamente lo declarativo es lo que contesta a la pregunda ¿qué? y lo procedimental es lo que contesta a la pregunda ¿cómo?

cercano al lenguaje natural de las líneas de código que tiene que producir, es decir las reglas.

Las reglas deben siempre considerarse como instrucciones de un lenguaje de cómputo, aunque de un lenguaje de altísimo nivel, mucho más poderoso que los lenguajes computacionales tradicionales.

3. Separación entre Base de Conocimiento y Estrategia de Control

Las expectativas que se refieren a la separación de la base de conocimientos y la estrategia de control o aparato inferencial son como sigue:

1. La base de conocimiento contiene solamente reglas que expresan relaciones entre hechos del dominio. El control del proceso reside en el aparato inferencial que es lo que determina la senda por la que transcurren las reglas. Incluso el orden en que las reglas están en la base de conocimientos carece de importancia.

2. Siendo así que el aparato inferencial encuentra su senda entre las reglas, el ingeniero de conocimiento sólo debe preocuparse por el contenido de la base y no por la estrategia de control.

El punto crítico aquí es cuán inteligente sea el aparato de control. Uno esperaría que fuera más inteligente que lo que implica el encadenamiento hacia atrás ofrecido por la mayor parte de las conchas, que hubiera alguna manera de conducir la asociación entre las reglas con más criterio que el puramente asociativo.

Cuando un experto humano trata de llenar una submeta muchas veces emplea estrategias de control que son independientes de la representación misma del conocimiento. Tal cosa puede involucrar el orden en que se ataquen las submetas del mismo nivel; tal vez una cadena contenga más información que otra, y convenga proseguir la cadena más corta primero, pues fallando ésta no necesitaría proseguir la otra y la meta podría declararse de una vez fallida.

Pero a pesar de estas consideraciones, lo corriente es que las conchas respeten el orden físico en que aparecen las reglas, sin más. De ahí que resulte importante preordenarlas de acuerdo con criterios como el expresado anteriormente, y que el orden de las reglas sea una consideración de diseño importante.

También es posible que el programador decida incluir algunas reglas que se refieran a otras reglas, por ejemplo para determinar un orden de preferencia según las circunstancias; en estos casos el funcionamiento del sistema resulta superior, pero ese logro se deberá al ingenio del programador y no ocurrirá por rasgos constitutivos de la concha.

En rigor, el uso de tales recursos contamina la pureza de los medios de representación del conocimiento, que no se supone que contengan nada relativo a estrategia de control. Se violan así las expectativas "oficiales" en relación con las conchas, aunque se violan para obtener mejor rendimiento.

Estos aspectos deben tenerse en cuenta cuando se juzga la necesidad de tener diseñadores de sistemas expertos de un nivel elemental ó alto: cuanto mejor programador sea el diseñador, se obtendrá mayor eficiencia del sistema.

¿Cuál es entonces la diferencia con el empleo de lenguajes tradicionales? Probablemente la diferencia estribe en esto: el lenguaje es más poderoso, permite hacer más con los mismos medios, en el mismo tiempo. Por lo demás, probablemente el usuario final quedará más satisfecho, por su posibilidad de inspeccionar las reglas y entender, sin entrenamiento especial, cómo se obtienen las conclusiones.

4. Facilidad de Corrección y Adquisición de Conocimientos

Además de ser fáciles de programar, los sistemas basados en reglas se supone que son fáciles de mantener. La teoría es que, siendo el aparato inferencial un módulo separado de la base de conocimiento, uno no tiene que preocuparse por los detalles de mantenimiento, necesita sólo preocuparse por la parte del sistema que contiene el conocimiento mismo en reglas claras y concisas.

Las expectativas son las siguientes:

1. Para afectar el programa lo que se hace es agregar reglas, que son independientes unas de las otras. Cuidar que las reglas sean correctas es más fácil que lograr que un programa tradicional sea correcto.

2. Los expertos humanos, u operadores debidamente adiestrados por éste, pueden encargarse del mantenimiento del sistema, sin intervención de un programador.

Las conchas comerciales ofrecen, es verdad, una serie de mecanismos, como el módulo de adquisición de conocimiento, que ciertamente ayudan al mantenimiento. Tales ayudas constituyen un ambiente para depurar programas mucho más poderoso que los que ordinariamente pueden encontrarse asociados con los lenguajes tradicionales.

Debemos sin embargo tener en cuenta que la expectativa de que basta añadir o cambiar reglas, sin preocuparse por el efecto que estos cambios tienen en otras reglas, es válida solamente bajo la condición de que las reglas se hayan mantenido estrictamente ajenas a responsabilidades de control, de acuerdo con la expectativa analizada en la sección anterior.

Es fácil corregir o adquirir reglas solamente si todas las reglas del sistema se refieren únicamente al campo del dominio y no al funcionamiento de otras reglas. Dicho de otro modo, esto es sólo cierto en la medida en que del diseño de la base de conocimientos se hayan excluido rigurosa-mente consideraciones de aumento de eficiencia del intérprete.

Por tal razón tal vez sea ilusorio creer que el ideal de sistemas libres de mantenimiento pueda realizarse totalmente. Esto tiene como consecuencia que el ideal de prescindir de un programador que esté disponible para sacar de apuros a un sistema sea también probablemente irrealizable.

De todas maneras, siempre seguirá siendo cierto que la intensidad de mantenimiento será considerablemente menor comparada con la necesaria en relación con programas tradicionales.

5. Capacidad de Explicar los Razonamientos

Uno de los aspectos más atractivos de los sistemas expertos es su capacidad de dar explicaciones sobre sus procesos, inteligibles para el usuario. Esta habilidad descansa en la separación entre la estrategia de control y el conocimiento.

Suponiendo la separación entre control y conocimiento, el rastro de la ejecución del programa debería teóricamente ser suficiente como una explicación del porqué de los resultados obtenidos. De ahí que los más sencillos sistemas ofrezcan ese rastro ("trace") como su único tipo de explicación.

Las conchas comerciales ofrecen dos tipos principales de facilidades explicatorias: la impresión optativa del rastro del razonamiento, de modo que el usuario puede visualizar el "cómo" del razonamiento, o la exhibición de un texto prefabricado asociado con cada una de las reglas. Este texto se mostrará cada vez que el usuario pregunta el "porqué" de una conclusión intermedia que depende directamente de la regla que contiene el texto.

Ahora bien, incluso en estos mecanismos tan sencillos pueden fallar las expectativas, puesto que si el sistema ejecuta reglas que sólo han sido incluidas por razones de control, sin representar conocimiento sobre el dominio de aplicación, su invocación aparecerá tan opaca para el usuario corriente como lo sería el uso de cualquier abstruso comando de Cobol o Pascal.

Pero hay razones más profundas para considerar limitadas las capacidades explicatorias de los sistemas expertos basados solamente en reglas de producción. Mucha de la capacidad de explicar que tienen los expertos humanos se asienta en conocimientos teóricos sobre la naturaleza de los procesos implicados, por ejemplo, leyes físicas o químicas del dominio. En esa forma, las explicaciones que un usuario puede obtener del experto humano pueden a veces consistir en breves "conferencias" sobre elementos de ciencia o vulgarizaciones de los mismos que son atinentes al problema en cuestión. Los sistemas expertos, salvo raras excepciones, no tienen la

posibilidad de dar ese tipo de explicación.[14]

6. Capacidad de Lidiar con Conocimiento Incierto

Una de las ventajas propias del aparato inferencial de los sistemas expertos es la capacidad de lidiar con la incertidumbre (algunos autores señalan esta capacidad como el aspecto más distintivo de la nueva tecnología).

La incertidumbre puede referirse tanto al grado de confianza que el experto deposita en las reglas, como al que el usuario tiene sobre sus datos.

Muchas de las conchas comerciales ofrecen medios para lidiar con reglas o hechos inciertos. Las expectativas aquí son que el intérprete sea capaz de combinar los índices de confiabilidad de acuerdo con las mejores teorías disponibles, y que los investigadores de Inteligencia Artificial hayan sabido acuñar esas teorías de modo sólido y congruente.

Lo cierto es que en este aspecto el terreno es mucho menos seguro que lo que fuera deseable, siendo un síntoma de ello la enorme variedad de soluciones al problema que ofrecen las distintas conchas. Como se indicó anteriormente, las soluciones más populares son la teoría bayesiana de la probabilidad y la teoría de los factores de certeza de Stanford.

La teoría bayesiana es matemáticamente rigurosa, pero tiende a sufrir de la dificultad de que sus presupuestos raramente se satisfacen en la realidad, especialmente el requisito de independencia de los elementos confirmatorios y la necesidad de conocer todos los valores de probabilidades alternativas.

Por su parte, la lógica de factores de certeza tiende a ser más abierta al proveer al diseñador la oportunidad de escoger cómo combinar los valores durante la inferencia, pero por ello mismo tiende a ser menos rigurosa.

Podemos sin duda afirmar que es mucho más fácil lidiar con la incertidumbre dentro de la tecnología de sistemas expertos que en la programación tradicional, donde no existen formalismos para ello. Pero debe tomarse en cuenta que la situación en Inteligencia Artificial no es todavía clara al respecto, hay mucha investigación en curso tratando de aclarar el problema, y los respectivos modelos matemáticos se hacen cada vez más sutiles y complicados.

La experiencia enseña que los factores de certeza son aptos para incorporar los métodos usados por los expertos humanos; el problema práctico está en hacer asignaciones de índices de confiabilidad que sean predictivos de la seguridad de las hipótesis, lo que sólo puede decidirse con repetidas pruebas de las reglas en ejemplos representativos.

Pero en todo caso, es una gran cosa no tener que programar la forma en que los índices se integran partiendo de cero; se puede descansar para esa tarea en quienes diseñaron el intérprete.

7. Capacidad de Lidiar con Información Incompleta

Normalmente se espera de los sistemas expertos que sean capaces de razonar a pesar de una respuesta "no sé" a una pregunta dirigida al usuario. Las expectativas son al respecto las siguientes:

1. El aparato inferencial debe ser capaz de ajustar su razonamiento a las condiciones de ignorancia que encuentre. La respuesta "no sé" debe ser tratada diferente que "sí" o "no".

2. El razonamiento no debe terminar abruptamente frente a la ignorancia de ciertos hechos, sino más bien degradarse de manera grácil, produciendo a pesar de ello algún resultado útil.

Desgraciadamente, en este terreno la evaluación no puede menos que ser negativa: las conchas no llenan las expectativas. La razón es muy simple: el intérprete no tiene la capacidad lógica de

[14] Las excepciones son sistemas expertos basados en modelos (en vez de en reglas), los cuales están apenas siendo probados en condiciones experimentales limitadas; un ejemplo interesante es el sistema FALCON, preparado por investigadores de la Universidad de Delaware, que diagnostica fallas en una planta química de la Compañía Dupont y se basa en el modelo matemático del funcionamiento de la planta.

distinguir "no sé" de cualquier otra respuesta; dicho de otro modo, es incapaz de percibir su ignorancia en cuanto tal. Sólo el diseñador de la aplicación puede hacer la distinción entre un "no sé" y un "no", lo que refuerza los argumentos dados anteriormente sobre la importancia de la programación en la confección de la base de conocimientos.

El problema es mucho más agudo si tomamos en cuenta que los sistemas deductivos electrónicos operan a menudo bajo el "principio del mundo clauso", también llamado "de la negación por fracaso". De acuerdo con este principio, todo lo que el sistema no puede probar a partir de sus premisas y usando sus reglas, debe ser considerado como falso.

Una interpretación más modesta del fracaso del sistema sería decir "por lo que yo sé, esto es falso". Pero una tal cualificación haría muy endeble el funcionamiento de un aparato inferencial, de conformidad con el estado actual del arte.

Así pues, el diseñador debe tener sumo cuidado de incorporar a la base de conocimientos todos los hechos que son atinentes en el dominio; la falta de uno de ellos podría ser interpretada como la negación del hecho, excepto en los casos en que el intérprete tome la precaución de preguntar al usuario antes de darse por vencido (si lo hace muy a menudo puede hacer la ejecución del programa insoportable para el usuario).

En resumen, hay consenso en que las expectativas de las conchas de sistemas expertos han sido satisfechas en lo siguiente:

1. Las reglas son medios efectivos de representar el conocimiento. El carácter explícito de la representación permite la inspección de la base de conocimientos por un experto, lo que sirve de validación al funcionamiento del sistema.

2. La separación de la base de conocimientos y el aparato inferencial, aunque en la práctica se da sólo parcialmente, reduce grandemente el esfuerzo de desarrollo al proveer formatos para las reglas, una estructura de control y una interface con el usuario.

3. El mantenimiento de la base de conocimientos se facilita por los mecanismos que proveen las conchas. En el caso de las bases simples, el mantenimiento es trivial.

4. Los formalismos para lidiar con la incertidumbre son adecuados para la mayor parte de las aplicaciones.

Algunas expectativas, sin embargo, no se confirman. Sobre ello podemos hacer las siguientes afirmaciones:

1. La construcción de una base de conocimientos se parece más a la programación en un lenguaje de computación ordinario que al llenado de una base de datos.

2. La separación del conocimiento y el control no es completa ni tajante. Muy a menudo deben incluirse en la base de conocimientos reglas que desempeñan funciones de control, so pena de importantes ineficiencias en el funcionamiento.

3. Las capacidades de explicación son limitadas, tanto por la naturaleza misma de los sistemas basados en reglas (ausencia de conocimientos sistemáticos sobre el dominio) como por la contaminación de las reglas con mecanismos de control, productores de opacidad en los rastreos.

4. Es ilusorio pretender que una base de conocimientos esté totalmente libre de mantenimiento, excepto en los casos de los sistemas expertos más triviales.

5. Existen limitaciones importantes en la forma en que los sistemas expertos pueden lidiar con la ausencia de conocimiento en el intento de solución de problemas particulares.

Algunas de estas deficiencias podrán ser superadas a corto plazo con mejoras en las conchas que se ofrecen en el mercado. Algunas otras son más sustanciales, y habrá que esperar más para su superación pues depende del avance de la Inteligencia Artificial como un todo.

VII Conclusiones

La tecnología de los sistemas expertos, un subproducto de veinticinco años de investigación en el campo de la Inteligencia Artificial, es un recurso productivo digno de ser explorado por los empresarios en nuestro medio.

Un sistema experto es un programa de computación basado en conocimiento y que provee soluciones de calidad experta para problemas de un dominio específico. Es un medio de poner en un computador la experiencia y la habilidad acumulada por un experto humano durante decenios, al alcance de la mano de un usuario moderadamente adiestrado.

Se han creado sistemas expertos, que han probado su utilidad, en dominios tan diversos como la medicina, la prospección geológica, la manipulación algebraica, la ingeniería, los negocios, el derecho, la agricultura o la educación.

También el tipo de experticia es variado: hay sistemas para interpretación, predicción, diagnóstico, diseño, planeamiento, reparación, instrucción, supervisión y control.

Hay diferencias importantes entre un sistema experto y un experto humano que deben tenerse muy en cuenta: en primer lugar, los sistemas expertos no pueden contener conocimientos sistemáticos sobre una materia, no tienen comprensión científica sobre los problemas. Son simples compendios de reglas empíricas o de experiencia; no debemos esperar de ellos explicaciones profundas sobre lo que hacen, porque lo que hacen es aplicar reglas, muchas de ellas cuya utilidad descansa en el hecho de que han tenido éxito en el pasado.

La otra diferencia muy importante es que los sistemas expertos son incapaces de aprender: el conocimiento que tengan deberá habérseles proporcionado previamente, y toda adición al mismo implicará una ampliación de su base de conocimientos.

Punto fundamental de esta tecnología es el haber logrado separar el control del razonamiento del experto y el contenido de su experticia. Ello ha permitido producción comercial de conchas ("shells") que se encargan del razonamiento, de modo que el constructor de un sistema experto sólo tiene que preocuparse de levantar una base de conocimientos. En los sistemas expertos "el poder está en el conocimiento".

Existe una amplia gama de conchas para sistemas expertos en el mercado, muchas de ellas para microcomputadores y con un precio inferior a los $ 500. Esencialmente, las conchas proveen de un marco dentro del cual el usuario puede insertar los detalles de su aplicación. Significan una ayuda muy grande en el empleo de esta tecnología, que la pone al alcance del empresario promedio .

La inversión necesaria para construir un sistema experto, además de la adquisición del computador y la concha, es de alrededor de medio año hombre de un experto humano que esté tiempo completo disponible para suplir el conocimiento, y de otro medio año hombre de un ingeniero del conocimiento que deberá extraer la experticia, formatearla en forma de base de conocimientos, y depurarla con uso del intérprete y de los recursos de apoyo que ofrece la concha.

El empresario costarricense debería preguntarse: ¿existe en mi empresa alguna persona cuya experticia, adquirida a través de muchos años de estudio y experiencia, no quisiera perder en el momento de su retiro?, ¿existe una persona que si fuera físicamente posible me gustaría duplicar, como se duplica un documento importante, para tenerla en todas las oficinas regionales de la empresa?

Si la respuesta a esas preguntas es afirmativa, la tecnología de sistemas expertos puede ser que sea capaz de realizar el milagro deseado. Lo siguiente que tiene que hacer el empresario es considerar cuidadosamente si la materia de que se trata es susceptible de ser implantada como sistema experto.

Un problema es apto para la tecnología de sistemas expertos si no existe solución algorítmica para él. Si el problema puede resolverse con un programa tipo hoja electrónica, por ejemplo, no es materia para sistema experto (tampoco para experto humano, y debería haber sido mecanizada hace tiempo).

Por hipótesis, el problema debe poder ser solucionado por un experto humano. No debe caerse en la tentación de creer que un sistema experto puede resolver lo que ningún experto humano puede resolver. Los computadores no son superhombres, a lo más que pueden aspirar es a ser amplificadores o replicadores de las capacidades humanas.

Debe tenerse cuidado también de que la persona cuyas capacidades queremos replicar no sea valiosa para la empresa por su sentido común, más que por su experticia. El sentido común todavía y probablemente por mucho tiempo no puede ser imitado por los computadores.

La mejor manera de diferenciar el sentido común de la experticia es recordar que la experticia se da siempre en un campo muy limitado de cosas y en relación con problemas bien estructurados, mientras que el sentido común se refiere a cosas muy variadas y a temas tan poco estructurados como la vida sentimental o las relaciones interpersonales.

Si ya el experto humano dejó la empresa o por alguna causa no está disponible, no es posible replicarlo con un sistema experto. El experto humano debe estar disponible. Si la empresa lo ha perdido, debe comenzarse por importar a uno de afuera para los efectos de la construcción del sistema experto.

Otro prerrequisito que el empresario debe tomar en cuenta es que la experticia sea de carácter intelectual (manipulación simbólica) y no de manipulación física o de percepción. Si la persona que quisiéramos replicar es una persona que sobresale por la agudeza extraordinaria de su vista o por la rapidez de sus movimientos, la tecnología de los sistemas expertos no nos puede ayudar a hacerlo.

No es necesario para armar un sistema experto que el conocimiento del experto humano sea seguro y completo; al contrario, lo normal es que las reglas que el experto aplique sean heurísticas (que no tienen garantizado el éxito), probabilísticas e incompletas. En cambio, deben ser muchas y variadas, independientes entre sí.

La siguiente cuestión que debe considerarse es la disponibilidad de un ingeniero del conocimiento para la tarea de desarrollar la base de conocimientos, y de personal adecuado para darle mantenimiento.

El ingeniero del conocimiento puede ser un graduado en ciencias de la computación con adecuada preparación general en Inteligencia Artificial y suficiente experiencia o capacidad en la utilización de conchas de sistemas expertos. Muy probablemente algunos graduados de nuestras instituciones de educación superior están capacitados para desempeñar esta función.

Aunque teóricamente no se requiere conocimiento especializado para usar las conchas y mantener las bases de conocimientos, con tal de que el operador disponga del tiempo suficiente para estudiar los manuales y experimentar con los ejemplos que los fabricantes proveen, en la práctica, estudio o experiencia anterior pueden ser más que una ayuda para asimilar los manuales y tener éxito en la implantación concreta del sistema.

Pareciera sumamente recomendable que quien opere y mantenga el sistema sea también un graduado en informática, aunque no del nivel necesario para desarrollar el sistema.

En suma, la tecnología de los sistemas expertos ofrece un reto y una oportunidad a la empresa costarricense que no debe pasarse por alto. El aprovechamiento de tal tecnología exige imaginación y cuidado de parte del empresario, así como gastos moderados. Los problemas que pueden presentarse son claros y manejables. Los frutos que pueden recogerse son numerosos y justifican la aventura.

¡Buena Suerte!

Apéndice A: Estrategias de Control Formal

El control que se encarna en el intérprete puede ser de predominio de las metas (también llamado "encadenamiento hacia atrás" o "búsqueda inversa"), o bien de predominio de los datos, (también llamado "encadenamiento hacia adelante" o "búsqueda directa").

Si es de predominio de las metas, la meta inicial se coloca en la memoria de trabajo. El sistema compara las consecuencias o parte derecha de las reglas con la meta, seleccionando una regla en cada ciclo y poniendo en la memoria de trabajo su condición o parte izquierda. Este proceso corresponde a una descomposición del problema en submetas.

El proceso continúa, con estas condiciones, jugando ahora el papel de nuevas metas que deben compararse con las consecuencias de las reglas. Así, el sistema actúa desde la meta original hacia atrás, hasta que todas las submetas en la memoria de corto plazo son confirmadas como verdaderas por este proceso de pareamiento, lo que indica que la hipótesis inicial es verdadera.

Esta búsqueda invertida corresponde a la forma en que los seres humanos probamos nuestras hipótesis.

Si el proceso es de predominio de los datos, cada ítem que existe en la memoria de corto plazo se compara con las condiciones de las reglas, separando sus consecuencias si la comparación tiene éxito, y las consecuencias se convierten en nuevos datos que deben ser comparados con nuevas condiciones, hasta que el sistema descansa porque se ha logrado producir lo que quería el usuario.

Esta búsqueda directa es menos popular en los sistemas expertos que la búsqueda inversa; la razón es que una inferencia hacia adelante puede producir mucho material no deseado, mientras que la inferencia hacia atrás, dirigida por la meta buscada, se concentra más en la dirección de los objetivos del usuario.

En el caso de la búsqueda inversa, es frecuente que algunas reglas interroguen al usuario sobre la validez de información no registrada en la memoria de corto plazo y que sin embargo es requerida para la ejecución, complementando así el proceso de inferencia.

Tomemos por ejemplo el caso de un sistema con las siguientes reglas en la memoria de largo plazo:

regla 1 Si hay gasolina en el tanque y hay gasolina en el carburador entonces el motor recibe gasolina

regla 2 Si el motor recibe gasolina entonces el problema es de candelas

regla 3 Si el carburador huele a gasolina entonces hay gasolina en el carburador

y con los siguientes hechos en la memoria de trabajo:

Hay gasolina en el tanque.
El carburador huele a gasolina.

El encadenamiento hacia adelante envuelve probar las reglas en el orden dado para ver cuáles pueden activarse con base en la existencia del antecedente en la memoria de trabajo.

La regla 3 es la única que puede seleccionarse, puesto que "el carburador huele a gasolina" está disponible, y esto produce la adición del hecho "hay gasolina en el carburador" a la memoria de trabajo.

Las reglas son probadas de nuevo, y esta vez la regla 1 se activa, pues tanto "hay gasolina en el tanque" como "hay gasolina en el carburador" están disponibles, con lo que "el motor recibe gasolina" se agrega a la memoria de trabajo.

Ahora puede activar la regla 2, el hecho "el problema es de candelas" se agrega y si ese es el resultado buscado por el usuario, el problema queda resuelto.

El encadenamiento hacia atrás comienza por tratar de probar "el problema es de candelas", que sería la pregunta entrada por el usuario (habría que interpretar el enunciado en su forma interrogativa: "¿el problema es de candelas?").

El sistema encuentra la regla 2, con "el problema es de candelas" como consecuente. Esto provoca que el sistema se pregunte a sí mismo "¿el motor recibe gasolina?", pues esta submeta es el antecedente de la regla.

La búsqueda encuentra la regla 1 que tiene "el motor recibe gasolina" como consecuente, lo que lleva al sistema a preguntarse ahora si "hay gasolina en el tanque" y "hay gasolina en el carburador", en calidad de subproblemas.

"Hay gasolina en el tanque" está ya en la memoria de trabajo; para obtener "hay gasolina en el carburador" debemos buscar una regla que tenga "hay gasolina en el carburador" como consecuente, y tratar de establecer el antecedente de esa regla.

La búsqueda encuentra la regla 3, con "el carburador huele a gasolina" como antecedente. Siendo así que "el carburador huele a gasolina" está en la memoria de trabajo, podemos ahora agregar a ella "hay gasolina en el carburador".

Ahora la regla 1 puede activarse y "el motor recibe gasolina" queda probado y puede presentarse al usuario.

En el ejemplo anterior todos los enunciados son plenamente informativos. Normalmente, sin embargo, las expresiones contienen variables, especie de huecos en la información que se unifican con constantes durante el proceso. Por ejemplo, la siguiente variante de la estructura proposicional anterior es posible:

regla 1 Si hay gasolina en el tanque y hay gasolina en el carburador entonces el motor recibe gasolina

regla 2 Si el motor recibe gasolina entonces el problema es de candelas

regla 3 Si ?X huele a ?Z entonces hay ?Z en ?X

y con los siguientes hechos en la memoria de trabajo:

Hay gasolina en el tanque.
El carburador huele a gasolina.

Los procesos ocurrirán igual que como se explicó antes, con las siguientes diferencias.

En el encadenamiento hacia adelante, la regla 3 puede seleccionarse, puesto que "el carburador huele a gasolina" está disponible y ese enunciado se unifica con el antecedente de la regla.

La unificación se propaga al consecuente, y esto produce la adición del hecho "hay gasolina en el carburador" a la memoria de trabajo.

En el encadenamiento hacia atrás, hay un momento en que necesitamos obtener "hay gasolina en el carburador", debiendo buscar una regla que tenga "hay gasolina en el carburador" como consecuente y tratar entonces de establecer el antecedente de esa regla.

La búsqueda encuentra la regla 3, cuyo consecuente, "hay ?Z en ?X", unifica con "hay gasolina en el carburador". La unificación se propaga al antecedente, produciendo "el carburador huele a gasolina".

Siendo así que "el carburador huele a gasolina" está en la memoria de trabajo, el antecedente está realizado y podemos desprender "hay gasolina en el carburador", que es la submeta que queríamos sustentar.

Apéndice B: Lenguajes de Inteligencia Artificial en Relación con los Sistemas Expertos

Las conchas para sistemas expertos constituyen programas electrónicos, construidos dentro de los lineamientos analizados en este informe, básicamente de acuerdo con el paradigma de los sistemas de producción. Como tales programas electrónicos, deben estar escritas en algún lenguaje computacional. Pero muy frecuentemente no están escritas en lenguajes tradicionales como Fortran o Pascal, sino más bien en lenguajes de muy alto nivel propios de las investigaciones de Inteligencia Artificial.

Algunos están escritos en C, por razones de eficiencia (C es un lenguaje especialmente apropiado para la construcción de "software de sistemas", como por ejemplo sistemas operativos), pero incluso en estos casos, si bien la versión definitiva del programa está escrita en C, muy probablemente el programa original fue desarrollado en alguno de los lenguajes de Inteligencia Artificial.

La razón es clara: los lenguajes de Inteligencia Artificial tienen una flexibilidad muy grande que los hace eminentemente aptos para el desarrollo de programas de gran complejidad y versatilidad.

Los lenguajes de Inteligencia Artificial se caracterizan por combinar dos atributos que normalmente se encuentran separados, en distintos lenguajes. Al igual que los lenguajes de programación de alto nivel, permiten al programador expresarse con parsimonia de medios para producir efectos de importancia, pero de la misma manera que los lenguajes ensambladores, permiten al programador tratar partes del programa como si fueran datos, modificables por otros elementos del programa. La capacidad de un programa para modificarse a sí mismo es probablemente el factor más importante en un lenguaje para escribir programas de Inteligencia Artificial.

Existen tres clases principales de lenguajes de Inteligencia Artificial, basados en concepciones fundamentalmente diferentes. Por una parte está la programación funcional, de la cual el lenguaje Lisp ("list processing language") es el ejemplar más conspicuo. En segundo lugar tenemos la programación lógica, cuyo ejemplar más representativo es Prolog ("programming in logic"). Finalmente está la programación orientada a los objetos, cuyo ejemplar más importante es Smalltalk (originalmente diseñado como un lenguaje para niños que resultó ser inesperadamente poderoso).

El lenguaje Lisp es el decano de los lenguajes de Inteligencia Artificial, siendo el segundo más viejo de los lenguajes de programación de alto nivel (después de Fortran); fue inventado por John McCarthy en la década del 50. Como todos los lenguajes de programación funcional, tiene la particularidad de que el programador no escribe instrucciones para ser ejecutadas una después de la otra, sino más bien expresiones que son evaluadas, y contestan con un valor.

Una expresión en Lisp puede ser atómica o compuesta; las expresiones compuestas son normalmente una función (en sentido técnico matemático) con sus respectivos argumentos, que al ser evaluada contesta con un valor. Como las expresiones pueden estar formadas por otras expresiones, el valor de contestación de las expresiones internas sirve de argumento (o dato) para la evaluación de las expresiones que las abarcan.

En dialectos de Lisp como Scheme, uno de los más puros, los argumentos para las funciones pueden ser funciones ellos mismos, y los mismos valores de contestación pueden también ser funciones; el resultado neto es un altísimo grado de abstracción y un enorme poder de programación.

Prolog, un elegante lenguaje inventado en Europa en la década del 70, y por cierto adaptado como lenguaje oficial para los computadores de "quinta generación" del gobierno del Japón, tiene la particularidad de ser una aplicación directa de la lógica de predicados.

Se programa en Prolog simplemente expresando en lógica de predicados lo que se quiere que el programa realice, sin preocuparse por detalles del cómo, es decir del control de la computación.

Esta peculiaridad lo hace especialmente apto para representar las bases de conocimientos de los sistemas expertos.

Por otra parte, el intérprete de Prolog usa como estrategia de control el encadenamiento hacia atrás, la estrategia preferida por las conchas comerciales, como hemos visto. Finalmente, el intérprete está dotado de capacidades para comparar expresiones y unificar variables, lo que hace innecesario programar estos detalles al construir una concha.

En Smalltalk, un lenguaje desarrollado por la compañía Xerox, el elemento fundamental es el objeto, que sirve para representar, de manera uniforme, tanto datos como procedimientos. La programación se hace con base en definir los objetos que existen en el dominio, y los mensajes que son capaces de recibir y de enviar esos objetos (algunos de los mensajes pueden ser ellos mismos algunos de los objetos definidos).

Este avanzado tipo de programación se conoce también con el nombre de programación por intercambio de mensajes. Cuando un objeto recibe un mensaje, busca en su interior si existe un método adecuado para tramitar ese mensaje; el trámite puede consistir por ejemplo en el envío de otro mensaje a otro objeto, en un cambio del estado interno del propio objeto, o en una actividad exterior como desplegar algo en la pantalla. La modularidad de los objetos se presta muy bien para la representación de reglas de sistemas expertos.

Una característica importante de los objetos es que pueden organizarse jerárquicamente, y unos objetos heredar de otros diversos atributos, entre ellos métodos para tramitar mensajes. Este tipo de programación es de mucho poder y ofrece excepcional comodidad al programador.

Es posible decir que Lisp dominó el pasado de la Inteligencia Artificial y fue determinante en la revolución intelectual que creó los sistemas expertos. Prolog, por su parte, es el lenguaje más adaptado al estado del arte actual en esta tecnología. Es probable que los sistemas expertos del futuro tiendan a ser construidos usando programación orientada a los objetos, al estilo de Smalltalk.

Bibliografía

Barr, Avron et alia, The Handbook of Artificial Intelligence (Reading, Mass.: Addison Wesley, 1982).

Barr, Valerie, "Exploring Expert Systems" (Computer, noviembre 1988).

Buchanan, B.G. and Shortliffe, E.H. (eds.), Rule Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project (Reading, Mass.: Addison Wesley, 1984).

Citrenbaum, Ronald et alia, "Selecting a Shell" (AI Expert, setiembre 1987).

Davis, R. y Lenat, D.B., Knowledge Based Systems in Artificial Intelligence (New York: McGraw Hill, 1982).

(Forgy, Charles L., "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem" (Artificial Intelligence, enero 1982).

Freedman, Roy, "27 Product Wrap Up: Evaluating Shells" (AI Expert, setiembre 1987).

Gevarter, William B., An Overview of Expert Systems (Fanconia, Va.: National Tech. Info. Service, mayo 1982).

Gevarter, William B., "The Nature and Evaluation of Commercial Expert System Building Tools" (Computer, mayo 1987).

Gutiérrez, Claudio, SIMULA (San José, CR: OPLAU Universidad de Costa Rica, 1973).

Gutiérrez, Claudio, "An Expert System to Predict Yield in a Banana Plantation" (Minneapolis-St. Paul, Minnesota: Proceedings Case-Based Reasoning Workshop, AAAI-88, 1988).

Gutiérrez, Claudio y Castro, Marlene, Informática y Sociedad (San José: Educa, 1987)

Harmon Associates, "Currently available Expert System Building Tools" (Expert System Strategies, Vol. 4, No. 7, 1988).

Harmon, P. y King, D., Expert Systems: Artificial Intelligence in Business (New York: Wiley, 1985).

Harmon, P. et alia, Expert Systems: Tools and Applications (New York: Wiley, 1988)

Jackson, Peter, Introduction to Expert Systems (Reading, Mass.: Addison Wesley, 1986).

Johnson, L. y Keravnou, E.T. Expert Systems Technology: A Guide (Cambridge, Mass.: Abacus Press, 1985).

Keravnou, E.T. y Johnson, L., Competent Expert Systems: A Case Study in Fault Diagnosis (London: Kegan Paul, 1985).

Klahr, P. and Waterman, D.A. (eds.), Expert Systems: Techniques, Tools, and Applications (Reading, Mass.: Addison Wesley, 1186)

Krutch, John, Experiments in Artificial Intelligence for Small Computers (Indianapolis, Ind.: Howard W. Sams, 1981)

Linsay, R. K. et alia, Applications of Artificial Intelligence for Organic Chemistry (New York: McGraw Hill, 1980)

Michie, D. (ed.), Expert Systems in the Micro Electronic Age (Edinburgh: Edinburgh University Press, 1979).

Miller, Edwin J. et alia, "Expert System Shells: Do They Deliver What They Promise?" (Chemical Engineering Progress, octubre 1988).

Naylor, Chris, Build Your Own Expert System (New York: Halsted Press, 1985).

Negoita, C.V., Expert Systems and Fuzzy Systems (Menlo Park, California: Benjamin Cummings, 1984).

Raeth, Peter G., "Two PC based Expert System Shells for the First Time Developer" (Computer, noviembre 1988)

Rowe, Neil C., Artificial Intelligence Through Prolog (Englewood Cliffs, NJ: Prentice Hall, 1988).

Smart, G. y Langeland Knudsen, J., The CRI Directory of Expert Systems (Oxford: Learned Information Ltd., 1986).

Walker, A. et alia, Knowledge Systems and PROLOG: A Logical Approach to Expert Systems and Natural Language Processing (Reading, Mass.: Addison Wesley, 1987).

Waterman, D.A., A Guide to Expert Systems (Reading, Mass.: Addison Wesley, 1986).

Patrick Henry Winston, Artificial Intelligence (Reading, Mass.: Addison-Wesley, 1984).