Presttun, K.
Medidas de seguridad en redes de comunicación
Comunicaciones Eléctricas (1986), volumen 60, nº 1, págs. 63–70

Aunque se ha escrito bastante sobre seguridad en redes, nadie ha presentado todavía una aplicación de las técnicas propuestas a la RDSI y a los sistemas de tratamiento de mensajes. Este artículo describe los principios de integración de la seguridad como un servicio opcional en los casos antes mencionados, considerando tanto la distribución de claves como la transferencia de información.

Immendörfer, M.
Aplicaciones del proceso de voz a equipos de telecomunicación y oficinas
Comunicaciones Eléctricas (1986), volumen 60, nº 1, págs. 71–78

La introducción del reconocimiento y síntesis de voz en equipos de comunicación y oficinas permitirá a los abonados acceder a servicios de telecomunicación actuales y futuros, de forma grata para el usuario, mediante la palabra hablada. Sin embargo, ante las limitaciones actuales de la tecnología – sobre todo en el reconocimiento de la voz –, los diseñadores de sistemas han de resolver algunos problemas básicos. El autor examina tales problemas y presenta algunas directrices para un apropiado desarrollo del sistema. Algunas casas de ITT han desarrollado varios modelos que demuestran cómo mejorar el interfaz hombre-máquina mediante la tecnología del proceso de voz.

Jakatdar, P.; Mulla, H. D.
Comunicación oral en ordenadores personales
Comunicaciones Eléctricas (1986), volumen 60, nº 1, págs. 79–86

La aplicación de la tecnología de proceso de voz para mejorar la flexibilidad y utilidad de los ordenadores personales es un factor clave en la automatización de oficinas. El sistema de comunicación oral de ITT, opción del ordenador personal ITT XTRA*, ofrece al usuario una gran variedad de facilidades, como la entrada oral de órdenes, el correo de voz y recuperación de datos y los contestadores telefónicos. También incluye la gestión de una agenda de trabajo con avisador acústico y en pantalla, y la marcación rápida. El artículo describe las características más relevantes de este producto y muestra cómo utilizarlo para mejorar la productividad en una oficina.

Canter, S.; Crossland, W. A.
Dispositivo esméctico de direccionamiento eléctrico con almacenamiento para pantallas planas grandes
Comunicaciones Eléctricas (1986), volumen 60, nº 1, págs. 87–93

Las pantallas de cristal líquido han mejorado muchísimo en los últimos años, aunque las actuales pantallas de rotación nemática, direccionadas por una multiplexación de barrido rápido, son complejas y todavía ofrecen una visualización peor que las de tubos de rayos catódicos convencionales. Los autores describen un nuevo tipo de pantalla basada en la dispersión dinámica y en la reorientación dieléctrica de cristales líquidos esmécticos A. Como este efecto presenta biestabilidad, o sea almacenamiento, resuelve los problemas asociados a la multiplexación de barrido rápido. Las pantallas de grandes dimensiones producidas con este material dan un excelente contraste, siendo apropiadas para múltiples aplicaciones.

Oficinas Editoriales

La correspondencia relacionada con las diferentes versiones de Electrical Communication debe dirigirse al editor correspondiente:

Michael Deason
Electrical Communication
Great Eastern House
Edinburgh Way
Harlow, Essex
England

Wolfgang Schmid
Elektrisches Nachrichtenwesen
Lorenzstrasse 10
7000 Stuttgart 40
Deutsche Bundesrepublik

Antonio Soto
Comunicaciones Eléctricas
Ramírez de Prado, 5
28045 Madrid
España

Jean-Pierre Dartois
Revue des Télécommunications
30 Avenue George V
75008 París
France

Comunicaciones Eléctricas

VOLUMEN 60 Nº 2 1986

REVISTA TECNICA DE ITT

Volumen 60
Número 2 · 1986

Comunicaciones Eléctricas

Edición española de ELECTRICAL COMMUNICATION
Revista técnica publicada trimestralmente por ITT Corporation

Comunicaciones Eléctricas presenta las investigaciones, los desarrollos y las realizaciones conseguidas por ITT y sus compañías asociadas.

Publicada desde 1922 en versión inglesa, se edita actualmente en cuatro idiomas y se distribuye en el mundo entero.

Se invita a los ingenieros de ITT a proponer proyectos de artículos, cuyos resúmenes deben enviarse al editor internacional para su consideración.

Publicado en octubre de 1986

Las direcciones de los editores de dan en la página 192

Tecnología de sistemas expertos

Comunicaciones Eléctricas, Director en Español: A. Soto, Ramírez de Prado, 5
28045 Madrid (España)

ITT

Se utilizan ya sistemas expertos para aplicar el saber de los especialistas a la resolución de problemas en diversas áreas de la industria. Sus posibilidades son vastas, abarcando todo campo en el que exista una base de experiencias reconocida, y ofreciendo soluciones a problemas que han resultado inabordables con la tecnología de programación convencional.

Presentación

Pocas innovaciones ha habido en la tecnología de programación tan excitantes como los sistemas expertos: hasta los no especialistas pueden apreciar el alcance de una tecnología que aparentemente otorga a las máquinas las facultades humanas de pensar y razonar. Por desgracia, el interés despertado ha hecho caer también en exageraciones, manifestadas por una parte en la pretensión de sobrevalorar el potencial de los sistemas actuales y, por el otro extremo, en la conservadora negativa a reconocer su indudable poder. ITT ha elaborado ya sistemas expertos que están en uso cotidiano, a la par que tiene en desarrollo otros más experimentales y además prosigue la investigación sobre herramientas y métodos que ensancharán las posibilidades hasta campos de aplicación enteramente nuevos. Este número de *Comunicaciones Eléctricas* presenta las experiencias de ITT en esta tecnología y los resultados conseguidos en una diversidad de aplicaciones, encarando futuras áreas de desarrollo.

El sistema experto es un programa de ordenador que capta el conocimiento en un campo concreto del saber (llamado dominio) y lo aplica luego a la resolución de un problema. La denominación de experto obedece a dos razones: imita el comportamiento de los expertos humanos en un determinado dominio, y sirve a los especialistas en ese dominio para complementar su conocimiento y ayudarles a realizar su trabajo con mayor eficacia.

A un elevado número de áreas tienen aplicación potencial estos sistemas. Las que se describen en este número abarcan desde la instalación de centrales telefónicas hasta los sistemas de guerra electrónica en tiempo real para aviones de caza, habiendo demostrado ciertos campos de aplicación una gran receptividad a la nueva tecnología. Los sistemas de diagnóstico, como el programa de pruebas en fábrica del Sistema 12 y el sistema de localización de fallos en centrales TXE4A, se consideran ya asimismo dentro del estado del arte.

En la ingeniería de aplicación para el Sistema 12, los sistemas basados en reglas han demostrado notables ventajas con respecto a la programación convencional. En otros dominios, todavía no operativos, se esperan muy próximos resultados, y aquí se incluyen el sistema de seguridad en tiempo real para señalización ferroviaria y el encaminamiento dinámico en redes militares de comunicación por paquetes. Sin duda existen aplicaciones en que las condiciones físicas restringen severamente la potencia de procesamiento — ejemplo es el sistema de defensa a bordo de aviones de combate —, y la viabilidad de las mismas exigirá progresos en otras tecnologías. Pese a todo, la gran variedad de sistemas

que las casas de ITT desarrollan en Europa y Norteamérica evidencia que la gama de aplicaciones de los sistemas expertos es ya bastante extensa.

ITT ha optado por seguir un abanico de métodos en la investigación sobre sistemas expertos, juzgando prematuro estandarizar un método concreto en el presente estado de la tecnología, si bien a más largo plazo se pretende hacerlo por las usuales razones del coste del mantenimiento y soporte de múltiples metodologías. El sistema ESSAI, actualmente en desarrollo en el Engineering Support Centre de Harlow (Inglaterra) es el más destacado aspirante a sistema primario, ya que no requiere unos circuitos especializados ni unas raras aptitudes de programación, condiciones ambas fundamentales para la industrialización de esta tecnología.

La estructura de los programas de sistemas expertos difiere mucho de la tradicional; las familiares nociones de subrutinas y ficheros de datos se complementan ahora con bases de conocimientos, grupos de reglas y máquinas de inferencia. Rasgo destacado de la nueva estructura es que permite separar el saber-hacer habitualmente incorporado a cualquier programa de ordenador del esquema de control que determina cómo trabaja el programa, gracias a lo cual se pueden añadir nuevos conocimientos sin alterar el modo de ejecución. En la programación clásica, cualquier cambio suele exigir una prueba de regresión completa del programa entero para detectar efectos secundarios no previstos; en cambio un programa basado en conocimiento se suplementa con nuevos elementos autónomos sin afectar en absoluto a la integridad del programa de control (habitualmente llamado máquina de inferencia). Esto tiene una enorme repercusión en la productividad y la capacidad de mantenimiento.

Un sistema experto auténtico habrá de ser capaz de defender sus decisiones. Para ello guarda registro de todas las reglas que le han servido para llegar a una conclusión, y puede exponer, si así se le solicita, el razonamiento seguido hasta la recomendación final. Esto es importante, ya que aterroriza pensar en unas máquinas inteligentes que controlen procesos complejos y sin embargo sean incapaces de explicar a los operadores lo que realmente están haciendo.

Quedan todavía muchos problemas por resolver antes de disponer con facilidad de programas de estas características. En los centros de investigación de ITT se estudian nuevas representaciones del conocimiento y de los procesos de razonamiento, medios de potenciar el funcionamiento de estos sistemas – intrínsecamente lentos –, y maneras de mejorar los interfaces humanos. Hoy día, la representación del conocimiento se vale de un conjunto de reglas previamente determinadas: SI se cumplen tales condiciones ENTONCES se tiene tal respuesta. Cuando se penetre más en la naturaleza del conocimiento, se podrán escudriñar las causas que sustentan esas reglas y comprender mejor el fondo del saber humano. En este campo realiza investigación básica el Advanced Technology Center de Shelton (EE. UU.). En cuanto a la mejora a corto plazo de prestaciones, transportabilidad y facilidad de uso, el Engineering Support Centre se dedica muy activamente a la creación de un juego de herramientas completo y sólido que recoja los últimos avances en tales aspectos.

Casi nadie duda que la tecnología de sistemas expertos dará acceso a una gama nueva de aplicaciones del ordenador, hoy sencillamente impracticables. Al seguir bajando el coste de la potencia de tratamiento, se desarrollarán más y más sistemas en muchos dominios diferentes. Las nuevas tecnologías del soporte físico, como el procesamiento en paralelo, acelerarán la tendencia hacia arquitecturas enteramente nuevas para los programas. Estamos pisando el umbral de una verdadera revolución en los tipos de programas de aplicación producidos y en los métodos de su desarrollo, y en este camino avanza ITT con firmeza.

A. D. Kenny
Director, Ingeniería de Programación
ITT Europe, Inc, Bruselas, Bélgica

Introducción a los sistemas expertos

Los sistemas expertos, diseñados para captar el conocimiento de los expertos humanos y aplicarlo para resolver problemas específicos, ofrecen ya sustanciales ventajas en aplicaciones donde la tecnología convencional se ha demostrado ineficaz. ITT utiliza la tecnología actual a la par que desarrolla técnicas avanzadas que sean capaces de resolver problemas más complejos.

J. J. Harvey
ITT Europe Engineering Support Centre,
Harlow, Inglaterra

Introducción

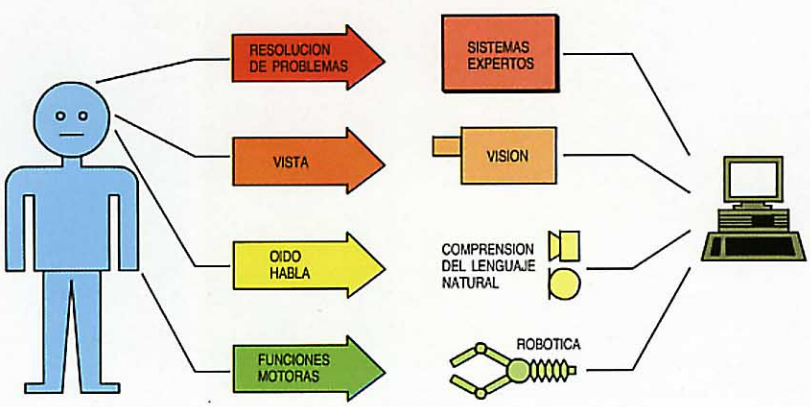
La tecnología de los sistemas expertos, todavía en su infancia, surge desde el nuevo y cada vez más amplio campo de la IA (inteligencia artificial). Hay dos maneras de concebir la IA: desde el punto de vista teórico, esta disciplina se preocupa por comprender el modo de conseguir ordenadores que perciban y entiendan a nivel humano (Fig. 1), mientras que desde un enfoque de ingeniería su objetivo es el desarrollo de ordenadores que demuestren facultades humanas sin requerir un fundamento teórico. Así como han podido construirse puentes antes de haber alcanzado la mecánica su completo desarrollo, también es posible elaborar sistemas inteligentes que contribuyan a la resolución de problemas y a la toma de decisiones sin disponer todavía de un fundamento teórico completo.

Como se ilustra en la figura 1, las grandes áreas de IA son fiel reflejo de las facultades humanas: la robótica imita las de locomoción y manipulación; la comprensión de lenguaje natural y el habla, las de comunicación; la de visión emula la capacidad de distinguir y reconocer imágenes, y finalmente la de sistemas expertos asume la facultad de resolver problemas.

Con estas orientaciones se entrecruzan las técnicas de IA utilizadas para desarrollar modelos informáticos, las cuales pueden clasificarse en varios grupos. El primero es el de representación de los conocimientos, que cubre los formalismos utilizados para describir tanto los conocimientos declarativos como los de procedimiento; son declarativos los conocimientos poseídos (hechos, teorías, hipótesis), y de procedimiento los que describen cómo utilizar aquellos conocimientos (estrategias y tácticas). El segundo grupo es el de proceso del conocimiento, es decir, los mecanismos que exploran el conocimiento y hacen deducciones razonadas para llegar a una conclusión. El tercero comprende las técnicas de aprendizaje, que producen nuevos conocimientos observando la eficacia del conocimiento existente en su interacción con el mundo. Las estrategias de planificación para organizar la resolución de problemas constituyen el cuarto grupo, y el quinto (interfaz de usuario) determina cómo interactúa el sistema con los usuarios.

Los sistemas expertos abarcan la parte del talento humano relacionada con la resolución de problemas y aplicación de aptitudes. Son sistemas informáticos que utilizan conocimientos y métodos de inferencia o razonamiento para resolver problemas normalmente tratados por expertos. De ahí que un sistema experto esté concebido para recoger el conocimiento de los especialistas en un cierto campo de aplicación (llamado dominio), y ponerlo a disposición de los menos experimentados.

Figura 1
La inteligencia artificial modela la conducta humana.



Componentes de un sistema experto

Hay tres componentes principales en un sistema experto: interfaz de usuario,

máquina de inferencia y base de conocimientos (Fig. 2).

Interfaz de usuario

Este interfaz permite al usuario interactuar con el sistema para presentar el problema y enterarse de las conclusiones. Un aspecto esencial es que el sistema experto, igual que un experto humano, pueda justificar sus conclusiones y explicar por qué se han tenido en cuenta o pasado por alto determinadas opciones.

Una consideración importante para el diseñador del interfaz de usuario es cómo repartir la iniciativa entre sistema y usuario. Cuando el sistema toma la iniciativa, dirige el diálogo y hace preguntas al usuario, quien, a menos que se le solicite, no puede

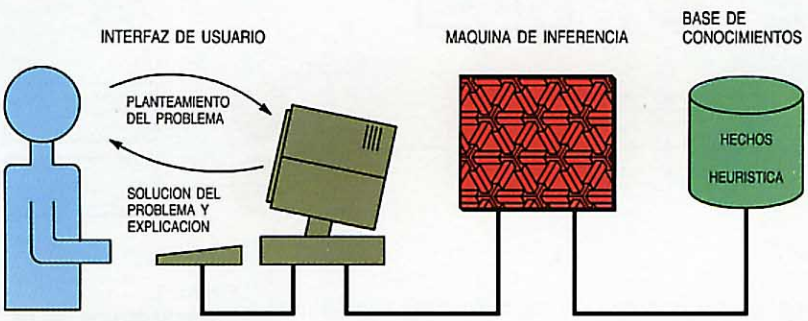


Figura 2
Componentes del sistema experto: interfaz de usuario, máquina de inferencia y base de conocimientos.

presentar información al sistema. Si es, por ejemplo, un sistema experto de diagnóstico el que toma la iniciativa, este sistema selecciona una hipótesis e interroga al usuario hasta que tal hipótesis se confirma o falla. Esto es adecuado para usuarios inexpertos que carezcan de opinión propia, pero tal vez es frustrante para usuarios expertos capaces de aportar una hipótesis alternativa o información adicional, que se sentirían menospreciados por el sistema.

Cuando la iniciativa se comparte, el usuario puede aportar información sin que el sistema se lo haya pedido, e intervenir en la toma de decisión. Así, en el citado ejemplo del diagnóstico, el sistema invita al usuario a seleccionar una de las hipótesis disponibles. Además, en cada oportunidad de decisión se pide al usuario que comente la hipótesis actual para sugerir otro posible modo de actuar.

Como cabe esperar, es más complejo de diseñar un sistema experto con iniciativa compartida que otro donde ésta corresponda siempre al sistema. Aún más complejo sería un sistema con iniciativa totalmente a cargo del usuario, en el que la interacción se viera forzada a cubrir una amplia gama de entradas de usuario, con toda la ambigüedad inherente al lenguaje natural, y por ello mismo más allá de las posibilidades tecnológicas actuales.

Máquina de inferencia

La máquina de inferencia, o mecanismo de razonamiento, es similar a la estructura de control en un programa convencional; opera por deducción y selecciona el conocimiento relevante para llegar a una conclusión. De ahí que el sistema pueda contestar a preguntas del usuario aun cuando la respuesta no esté explícitamente almacenada en la base de conocimientos. El objetivo es imitar el razonamiento humano de forma que el usuario pueda comprender los pasos que da el sistema. Una de las principales tareas del diseñador es elegir una técnica de inferencia apropiada.

Existen varias técnicas de inferencia, cada una de las cuales modela un procedimiento diferente de razonamiento. Por ejemplo, la concatenación regresiva o técnica guiada por el objetivo modela la forma de pensar de un experto en diagnóstico, que comienza en un presunto fallo y retrocede en etapas lógicas para hallar sus posibles causas. Por el contrario, la concatenación progresiva está dirigida por datos y modela el razonamiento de un experto en configuración que, partiendo de una lista de requisitos, determina la lista detallada de materiales necesarios para satisfacer tales requisitos. En ambas formas suele utilizarse una red de inferencia para modelar el proceso de razonamiento (Fig. 3).

Caracterizar así la técnica de inferencia permite una amplia generalización de su uso. En la práctica, sin embargo, pueden necesitarse ambas técnicas; una parte del problema quizá encaje con una estrategia progresiva, mientras que a otra le convenga más una concatenación regresiva. El reto consiste en ofrecer estas técnicas de un modo tal que el ingeniero del conocimiento pueda experimentar diferentes opciones para llegar a una estrategia óptima. ITT ha elaborado una solución que adopta un enfoque integrado de las técnicas de inferencia múltiple[1,2].

Otro importante aspecto de la inferencia es la capacidad de tratar problemas cuyos datos no son seguros. Ello podría necesitarse, por ejemplo, en la diagnosis si no son fiables los datos o evidencias de la causa de un fallo. El usuario expresará esto normalmente como la probabilidad o grado de certeza de que se presente cierto síntoma. Las técnicas de razonamiento con incertidumbre serán parte integrante de la máquina de inferencia.

Base de conocimientos

La base de conocimientos es tal vez el componente más importante, pues contiene el conocimiento y aptitudes de los expertos. Por esta razón, a los sistemas expertos a menudo se les llama sistemas

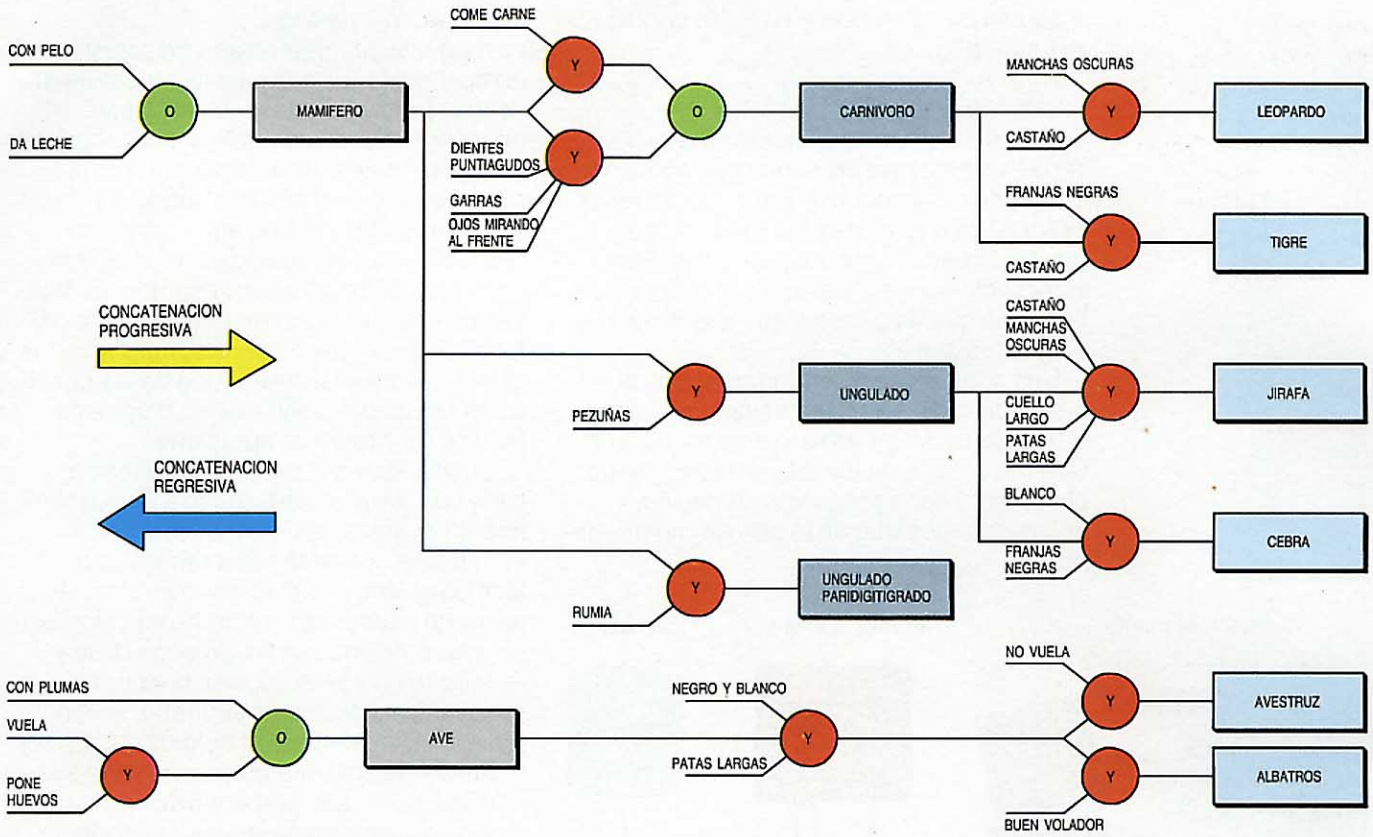


Figura 3
Ejemplo de una red de inferencia, utilizable con concatenación progresiva y regresiva.

basados en conocimientos. Es una gran ventaja que la base de conocimientos esté separada de la parte de control y del mecanismo de inferencia, pues ello permite añadir o cambiar conocimientos sin preocuparse del control ni recorrer el largo proceso de desarrollo que requieren los programas convencionales.

El diseñador de la base de conocimientos debe elegir una técnica de representación adecuada para describir el conocimiento de los expertos. Factores importantes son el poder expresivo de la representación (facilidad de descripción y lectura del conocimiento de los expertos) y la eficiencia de cálculo (sobrecarga en tiempo de ejecución que origina el proceso de la representación adoptada). Por un lado, utilizando el lenguaje natural de modo espontáneo para describir los conocimientos se conseguiría una gran expresividad, mientras que por el otro extremo una representación basada en un lenguaje de programación aseguraría una ejecución rápida. En general, se elige una técnica comprensible para los expertos, que les aliente a mantener la base de conocimientos y validar sus resultados, ofreciendo sin embargo una velocidad de ejecución aceptable.

Esta elección se apoya en las principales técnicas de representación de conocimientos utilizadas en sistemas expertos: reglas, redes semánticas, tramas y objetos.

Las reglas son la forma más común de representación. Cada regla consta de una o más condiciones, que al cumplirse dan lugar a una o más acciones (Fig. 4). Las bases de conocimientos que utilizan reglas tienen la ventaja de ser fácilmente modificables, pues cada regla es en principio una instrucción declarativa de conocimientos que está aislada de las otras reglas. Además, las reglas parecen concordar con el modo de formular los expertos sus conocimientos, en una clásica relación causa-efecto. Típicamente, los expertos crean un repertorio heurístico o normas "de andar por casa", que se describen fácilmente por una representación a base de reglas. Sin embargo, cuando el número de reglas pasa de algu-

Figura 4
Representación de conocimientos por medio de reglas, forma de representación más común en los actuales sistemas expertos.

REGLAS EJEMPLO – CASO ELEMENTAL	
SI EL ANIMAL TIENE PELO O SI EL ANIMAL DA LECHE ENTONCES EL ANIMAL ES MAMIFERO	SI EL ANIMAL TIENE PLUMAS O SI EL ANIMAL VUELA Y SI EL ANIMAL PONE HUEVOS ENTONCES EL ANIMAL ES UN AVE

REGLAS EJEMPLO – MUNDO REAL
SI EL TRAFICO POR LINEA ES NORMAL Y EL TIPO DE CIRCUITO DE LINEA ES ELC ENTONCES HAY UNA LINEA DE RESERVA POR CADA 479 LINEAS Y EL NUMERO DE LINEAS POR ASM ES IGUAL A 128 Y EL NUMERO DE LINEAS POR PBA ES IGUAL A 6

nos centenares esa misma facilidad de modificación se convierte en obstáculo, pues resulta difícil determinar cómo afectará un cambio al comportamiento global en cuanto a resolución de problemas. Este inconveniente se minimiza repartiendo la base de conocimientos en grupos de reglas, cada uno orientado a un aspecto diferente del problema.

Una variante de la representación por reglas utiliza la forma de lógica predicada propia del cálculo analítico; en ella las reglas se construyen con rigurosa lógica matemática, cuyas deducciones aplican a la técnica de inferencia. El valor de esta representación depende de que la aplicación sea adecuada a la inferencia por deducción lógica.

Las redes semánticas o las redes asociativas representan los conocimientos bajo la forma de una red de relaciones, compuesta de una serie de *nodos* interconectados por *arcos*, es decir, por líneas que conectan los nodos (Fig. 5). Los nodos representan los elementos de conocimiento, siendo los arcos los que determinan la relación entre nodos. Esta relación podría ser de herencia — cuando un nodo hereda las propiedades del otro —, o descriptiva, cuando sólo describe dichas propiedades. El problema de tales redes es su difícil actualización para reflejar nuevos conocimientos o relaciones cambiadas.

Las tramas son una forma de representación cada vez más común, que combina los conceptos de las redes semánticas y las reglas. La trama es una plantilla que contiene cierto número de *casilleros* y, opcionalmente, los valores que pueden tomar tales casilleros (Fig. 6). Estos valores pueden darse en forma de reglas, necesitándose un proceso deductivo para obtener el valor del casillero. Las tramas tienen la ventaja de representar explícitamente las relaciones de datos en una forma jerárquica, de modo que las tramas descendentes en la jerarquía puedan heredar valores de las tramas superiores.

Los objetos adoptan una representación similar a las tramas e incorporan las nociones de casilleros, valores y herencia. La diferencia clave reside en que la comunicación con objetos se hace en forma de *mensajes*, por cuyo medio unos objetos solicitan a otros que ejecuten ciertas tareas. Un objeto puede así representar un grupo de reglas, utilizando mensajes para programar la ejecución de dichas reglas.

El desarrollo de un sistema experto consiste realmente en elaborar representaciones apropiadas del conocimiento de los expertos. Las herramientas para construir tales sistemas suelen ofrecer una gama de técnicas de representación del conocimiento, a fin de que el ingeniero pueda seleccionar una que sea adecuada para la aplicación.

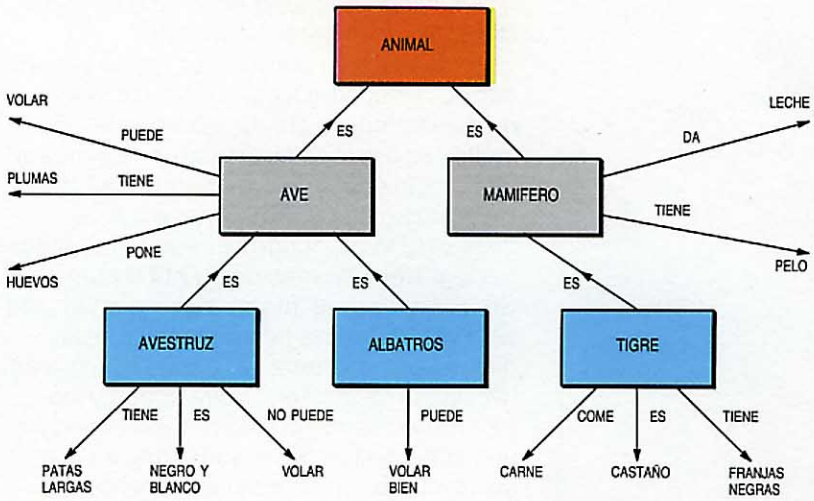


Figura 5
Representación de conocimientos utilizando redes semánticas. El conocimiento está representado por los nodos, siendo los arcos o líneas de interconexión los que determinan la relación entre nodos.

Ventajas de los sistemas expertos

La ventaja más evidente es que los sistemas expertos captan la inteligencia y el juicio humanos para utilizarlos en áreas donde los recursos son escasos y por lo tanto muy apreciados. Ello importa sobre todo en compañías de alta tecnología, donde es grande la demanda de expertos.

Como sucede con la tecnología convencional, puede plantearse en serio el uso de la tecnología de sistemas expertos si de algún modo se mide el beneficio que aporta. Ello sin embargo no es fácil, dado que no hay métodos establecidos para estimar el impacto en una organización de un nuevo sistema desarrollado. Dentro de ITT se ha trabajado en formalizar y aplicar tales medidas. Como ventajas cabe mencionar la reducción de costes por necesitar menos personal cualificado, el acortamiento del tiempo empleado en resolver un problema,

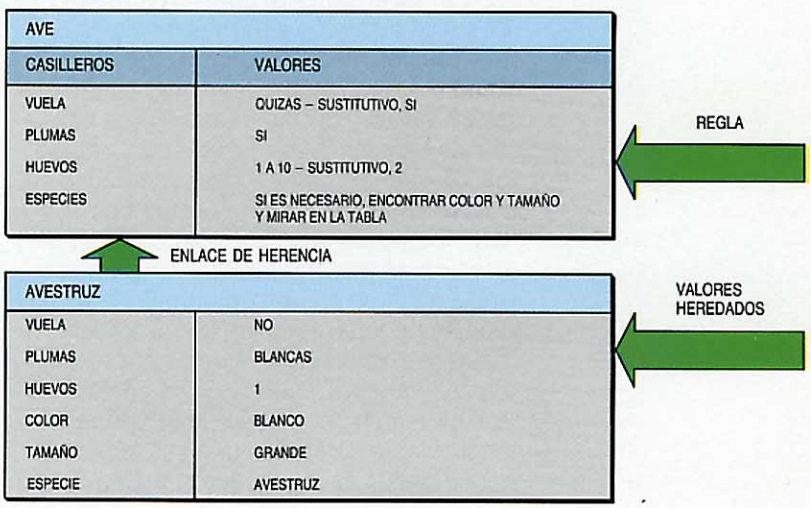


Figura 6
Representación de conocimientos por medio de tramas, que combinan los conceptos de reglas y redes semánticas.

o la mayor disponibilidad de técnica especializada en un campo concreto.

Otras ventajas potenciales son la capacidad de la tecnología para gestionar sistemas más complejos y prestar un soporte informático en áreas en que no sirve de ayuda la tecnología convencional de ordenadores. Un ejemplo de gestión compleja es el diseño de VLSI, donde se desarrollan circuitos digitales con más de 100.000 puertas. Se utiliza soporte informático automatizado para configurar las centrales telefónicas digitales del Sistema 12, pues los dispares requisitos de las Administraciones y los frecuentes cambios hacen difícil conseguir una solución por ordenador convencional satisfactoria. Un aspecto esencial de los sistemas expertos desarrollados por ITT como soporte de la configuración de las centrales es la fácil modificación de las reglas que gobiernan el proceso[3].

La tecnología de sistemas expertos ofrece la oportunidad de desarrollar nuevos productos y servicios, incluyendo herramientas para construcción de los propios sistemas (estimadas en 220 millones de dólares para 1990), y sistemas para tareas tales como aconsejar a los consumidores.

ITT ha conseguido especializarse en el desarrollo y aplicaciones de herramientas para sistemas expertos, cubriendo desde el desarrollo de programas por técnicas de IA hasta la ingeniería del conocimiento basada en dicho tipo de herramientas. Como soporte de dichas tareas se dispone de un cierto número de herramientas informáticas.

La experiencia dentro de ITT ha demostrado que los sistemas expertos tienen un gran impacto en la productividad y consiguen ahorro de costes en áreas donde la programación tradicional no ha podido penetrar. Las aplicaciones incluyen la diagnosis de fallo a nivel de placa de circuito impreso y de sistema[4,5], la configuración de las centrales telefónicas y población de las tablas de datos relacionadas[3,6], y la toma de decisiones en tiempo real[7]. La estrategia de ITT sobre sistemas expertos se centra en su utilización inmediata en la propia Compañía, asegurando al mismo tiempo la investigación y desarrollo adecuados para conseguir las técnicas que reclaman las futuras aplicaciones. Brotarán técnicas nuevas de las investigaciones sobre arquitecturas de sistemas de diagnóstico[8] y sistemas VLSI, y sobre cómo integrar mecanismos de inferencia múltiple en un conjunto de herramientas de ingeniería del conocimiento[1]. Con la realización de sistemas expertos no sólo se ha beneficiado la Compañía, sino que además ha adquirido una experiencia considerable y sentado una sólida base tecnológica. Los resultados actuales indican que la gama de aplicaciones crecerá en el futuro, aportando nuevos beneficios.

Desarrollo de sistemas expertos

El desarrollo de un sistema experto se realiza en varias etapas. Este proceso, denominado ingeniería del conocimiento, cubre dos aspectos: la adquisición de conocimientos – captación del conocimiento de expertos para almacenarlo en la base de conocimientos en una forma utilizable – y el proceso del conocimiento, en el cual éstos se aplican para resolver un problema.

Adquisición de conocimiento

Por ser una tarea lenta y muy cualificada, la adquisición de conocimiento limita mucho el uso generalizado de los sistemas expertos. ITT se ha aprovechado del desarrollo y empleo de herramientas informáticas para facilitar la labor del ingeniero del conocimiento[9], pero aún queda mucho por hacer.

Se aplican dos técnicas principales para captar el conocimiento de un experto: *artesanal* e *inductiva*.

La artesanal, que define las reglas directamente, se basa en entrevistas entre el ingeniero y los expertos para identificar los conocimientos dentro de un determinado dominio. Como resultado se obtiene una serie de reglas (u otra forma de representación) que encierran el conocimiento del experto.

En la inductiva, se utilizan herramientas informáticas para inducir reglas a partir de ejemplos suministrados por los expertos en el dominio, obteniendo un conjunto de reglas susceptibles de revisión y corrección. Como alternativa, podrían modificarse los ejemplos y repetirse el proceso inductivo.

La elección de la técnica viene determinada por el modo de representar el conocimiento en el campo de aplicación. El método artesanal es más apropiado cuando los conocimientos se representan por un conjunto de libros de reglas (como en la configuración de centrales Sistema 12, donde hay volúmenes con reglas para describir los conocimientos utilizados por los ingenieros de CAE[3]), o por un juego de normas de ingeniería y fabricación (p. ej., en diseño de circuitos hay documentos que definen las reglas de aptitud del diseño para ser probado). Sin embargo, conviene más la inducción cuando los conocimientos se representan por una serie de ejemplos de aptitudes ejercitadas por los ingenieros en el dominio, como ocurre con las pruebas en fábrica de las placas del Sistema 12, donde se ha acumulado un gran número de ejemplos de diagnosis[4].

Si se dispone de conocimientos en alguna de estas formas, la adquisición de conocimiento está bien fundamentada; en caso contrario, sin embargo, el desarrollo de una aplicación costará mucho más tiempo.

Proceso del conocimiento
Se cuenta con un cierto número de tecnologías para construir sistemas expertos, entre ellas lenguajes de uso general y especiales, capas de aplicación y juegos de herramientas. En la primera categoría figuran lenguajes como el LISP y sus dialectos, que por desgracia no proporcionan los mecanismos requeridos por los sistemas expertos y por ello exigen al ingeniero del conocimiento una labor de desarrollo lenta y difícil.

Los lenguajes de uso especial incluyen los OPS5, OPS83 y Prolog, todos los cuales incorporan técnicas específicas de los sistemas expertos. La familia OPS se ha utilizado en ITT[6] y en la industria, generalmente para construir sistemas expertos de aplicación concreta. Mientras estas técnicas sean adecuadas a la aplicación habrá pocos problemas. Sin embargo, si la aplicación requiere técnicas diferentes o mayor poder expresivo para representar conocimientos, habrá que potenciar apreciablemente el lenguaje.

Las envolturas de aplicación, como EX-TRAN, Sage y Emycin, aportan una visión a nivel más alto de las técnicas disponibles para construcción de un sistema experto, y por tanto simplifican aún más la tarea del ingeniero del conocimiento. ITT ha utilizado envolturas en varias aplicaciones[4,5].

Entre los "juegos" de herramientas para sistemas expertos figuran los Knowledgecraft, ART y ESSAI, los cuales incorporan componentes utilizados en la mayoría de sistemas expertos, como las herramientas de adquisición de conocimiento y diversos mecanismos de representación de conocimientos e inferencia. Ofrecen un conjunto común de componentes en forma de bloques constructivos utilizables para adquirir y procesar conocimientos en una extensa gama de aplicaciones. Los "juegos" de herramientas se están empleando en ITT[3], tanto en aplicaciones como para explorar algunos de los temas a investigar en sistemas expertos[1].

Se tiende a una mayor generalidad y flexibilidad, ofreciendo al ingeniero del conocimiento una diversidad de técnicas que permitan desarrollar aplicaciones dentro de un marco común. Este enfoque reduce el conocimiento especializado necesario para construir sistemas expertos, haciendo que los ingenieros puedan concentrarse en la aplicación de las técnicas más que en su desarrollo.

Recursos
Los recursos requeridos dependen de la complejidad del dominio y de los objetivos fijados para los sistemas expertos, así como de la tecnología escogida. Un equipo de proyecto comprenderá expertos en el dominio (aunque no con plena dedicación), ingenieros del conocimiento y soporte tecnológico. Los ingenieros mencionados son el nexo de unión entre los expertos y el sistema; se asemejan a los ingenieros de sistemas, siendo su cometido adquirir los conocimientos relevantes al dominio y popular la base de conocimientos. El soporte tecnológico se necesita para prestar consejo y guía en el uso de la tecnología y su mejora, si ello fuere preciso.

Evaluación
Una vez desarrollado, un sistema experto debe ser evaluado para validar su comportamiento frente a problemas conocidos y previamente resueltos. En tal evaluación deberían participar expertos del dominio capaces de criticar el funcionamiento del sistema, señalando omisiones en la base de conocimientos y sugiriendo mejoras en el interfaz de usuario. Esta etapa proporciona un punto de revisión para posteriores desarrollos.

Desarrollo ulterior
Los sistemas expertos se desarrollan incrementalmente, creciendo la base de conocimientos a medida que se sabe más del dominio implicado. Gracias a este proceso de mejora, el sistema experto es utilizable y provechoso.

Futuros sistemas expertos

Para poder construir sistemas expertos capaces de resolver la gama total de problemas de un dominio, tales sistemas necesitan integrar con capacidad razonadora una amplia variedad de conocimientos sobre las cuestiones debatidas. La diagnosis de fallos es un ejemplo. La mayoría de los actuales sistemas expertos de diagnóstico utilizan bases de conocimientos empíricos, es decir, fundados en la experiencia. Esto tiene como ventaja la eficiencia en el cálculo, y a menudo es la única solución cuando no se dispone de conocimientos de tipo causal (es decir, del efecto producido en un modelo básico del problema), como ocurre en la diagnosis médica. Los sistemas expertos de base empírica son satisfactorios cuando se posee experiencia sobre la gama total de problemas detectados, pero son inadecuados para problemas nuevos, ante los cuales suelen reaccionar abandonando y llamando al experto. También se ven

limitadas las bases de conocimientos empíricas si se requiere una detallada explicación del método de resolución del problema, ya que ésta solamente puede expresarse por asociaciones empíricas y no por referencias a la causa subyacente.

Los expertos se valen del conocimiento empírico para resolver problemas por ser cognoscitivamente más fácil y más rápido que el conocimiento causal (de causa-efecto), el cual razona desde los principios primarios. Sin embargo, cuando los expertos se enfrentan con problemas nuevos en absoluto abandonan, sino que utilizan sus conocimientos de estructura, comportamiento y causa para llegar a una solución. Por tanto, los futuros sistemas expertos combinarán ambos tipos de procedimientos[10].

La investigación y el desarrollo en sistemas expertos suscitan temas fundamentales, descritos a continuación.

Representación de conocimientos

Un sistema experto necesita tratar con múltiples modelos del área en cuestión: el conocimiento de la estructura o topología del área, como es la conectividad física de componentes en un circuito VLSI o de las unidades en un ordenador; el conocimiento de funciones y comportamiento, tal como las características eléctricas de los componentes de un circuito y su respuesta en simulación; el conocimiento del diseño y de las restricciones impuestas al diseñador (ej.: las razones para utilizar determinado conjunto de componentes o ciertas conexiones en un circuito); el conocimiento de las leyes físicas que rigen el comportamiento, como la Ley de Ohm; el saber interpretar dicho comportamiento, lo que puede implicar el conocer técnicas para resolver problemas, como son la búsqueda y localización de faltas; el conocimiento de casos anteriores, en fin, o sea el conocimiento empírico.

Estos modelos se complican todavía más por la necesidad de representar el tiempo y por los distintos modos de variación del comportamiento con el tiempo.

Es preciso desarrollar técnicas para representar estos modelos; quizá también haya que ampliar las representaciones disponibles. Además, habrá que tener en cuenta los problemas asociados a las bases de conocimientos grandes y bases distribuidas.

Proceso de conocimientos

Como los futuros sistemas expertos utilizarán ambos razonamientos, empírico y causal, se necesitan técnicas que permitan pasar de un modo a otro valiéndose de los diferentes modelos de conocimiento disponibles y en respuesta a solicitudes del usuario. Puede haber fuentes múltiples de datos o evidencias, capaces de interactuar con los diferentes modelos para proporcionar soluciones parciales que unidas contribuyan a una solución total.

Los mecanismos de inferencia, técnicas de búsqueda y métodos de trabajar con evidencia incierta dependen del problema, y no puede esperarse que sean homogéneos para la base de conocimientos entera. Es preciso desarrollar técnicas en las que se refleje esta combinación de mecanismos de inferencia, y ellas podrían incluir mecanismos a los que recurra el ingeniero del conocimiento o el sistema dentro del proceso de resolución del problema.

Interfaz de usuario

Un sistema experto tiene que ofrecer medios para poder compartir con el usuario la iniciativa en la resolución de problemas, en vez de ser el sistema quien la imponga cual sucede en la mayoría de los actuales sistemas expertos. Ello implica autorizar al usuario a que aporte entradas no solicitadas y a que comparta la toma de decisión. La entrada podría tener lugar en puntos críticos del proceso, capacitando al usuario para sugerir el camino óptimo cuando el sistema tenga que elegir entre varios.

El sistema experto debe dar explicaciones al usuario referidas a la causa física subyacente: éstas deberían siempre acomodarse al nivel de conocimientos sobre el dominio que posea el usuario, del cual puede enterarse el sistema a través del interfaz de usuario.

Validación y mantenimiento del conocimiento

Las bases de conocimientos actuales son todavía modestas — desde 200 hasta unas 4.000 reglas —, mas se prevé que crezcan hasta muchos miles de reglas. El desarrollo de las bases de conocimientos es una tarea cualificada, pero la ingeniería del conocimiento está sólo en sus comienzos. En consecuencia, pueden generarse reglas que originen incoherencia o carencia de integridad en la base de conocimientos, dando lugar a errores, o al menos resultados no tan óptimos, en cualquier conclusión enunciada por el sistema activo. Las incoherencias son difíciles de detectar en una revisión manual con soporte de herramientas limitado, y en general no es posible determinar la integridad.

Cualquier proceso de validación debería, pues, examinar tanto la coherencia como la integridad de la base de conocimientos. La coherencia atañe a las reglas entre sí conflictivas (reglas con partes de decisión mutuamente exclusivas que pueden cum-

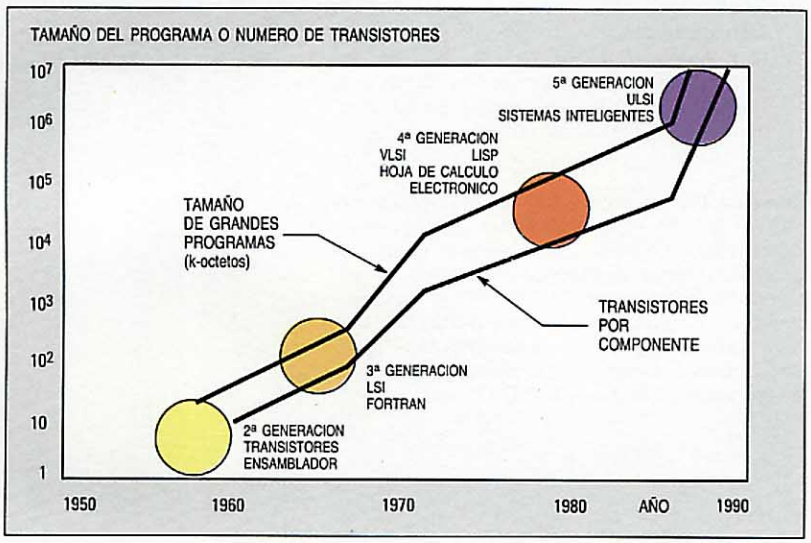
plirse al mismo tiempo), y a las redundantes (aquéllas cuyas partes de decisión tienen el mismo efecto). Una complicación adicional es la posible relación entre la incoherencia y la estrategia de control; por tanto reglas coherentes a la luz de una búsqueda transversal pueden tornarse incoherentes para una búsqueda en profundidad. La integridad asegura que existen reglas para resolver cualquier situación.

Tal vez resulte posible desarrollar técnicas que sustenten la reorganización y refinamiento automáticos de la base de conocimientos a lo largo del tiempo, y que sean capaces de aprender de la experiencia operativa.

Aprendizaje y adquisición de conocimientos

Es preciso desarrollar técnicas que reduzcan el tiempo dedicado a adquisición de conocimientos. Los modelos empíricos posiblemente podrían deducirse de modo automático, quizás por algún proceso inductivo, a partir de los modelos causales del área en cuestión, lo cual todavía deja sin resolver el problema de adquisición de conocimientos para dichos modelos causales. La investigación del proceso de aprendizaje puede conducir al desarrollo de técnicas adecuadas.

Figura 7
Crecimiento de la complejidad física y lógica de los ordenadores a lo largo de cuatro décadas.



Arquitecturas físicas

Hasta ahora solamente se han considerado las fuertes exigencias impuestas a los diseñadores de programas. Paralelamente los diseñadores de circuitos aceptan un reto igual, si no mayor, al buscar nuevas arquitecturas capaces de explotar el creciente número y diversidad de sistemas de conocimientos. Los ordenadores de la presente generación utilizan un proceso al estilo Von Neuman basado en la ejecución secuencial de las instrucciones del programa, que desde luego refleja la naturaleza de la programación convencional. Lo que se necesita es un proceso en el que los conocimientos se ejecuten en paralelo, en porciones independientes entre sí y que sin embargo puedan afectar a la solución del problema.

Los sistemas se hacen más complejos, al paso que la tecnología posibilita prestar nuevos y mejores servicios al comercio y la industria. Esto salta a la vista si nos referimos a la mayor extensión de programas desarrollados y a la densidad de transistores en una pastilla (Fig. 7), exponiendo asimismo algo sobre la mayor potencia y funcionalidad ofrecida a los usuarios. Los sistemas de la próxima generación, ya en desarrollo, ofrecerán posibilidades todavía mayores, y por primera vez se podrá gestionar tal complejidad por medio de sistemas basados en inteligencia artificial.

De esta combinación de equipos y programas y de las diversas tecnologías que los hacen posibles (Fig. 8), nacerán los sistemas inteligentes.

Conclusiones

Como se expone en este número de *Comunicaciones Eléctricas*, la tecnología de sistemas expertos ha progresado ya hasta el punto de poder ofrecer soluciones innovadoras y rentables a una extensa gama de problemas industriales. En los próximos diez años los perfeccionamientos metodológicos, las nuevas arquitecturas de equipo y un soporte lógico más potente conseguirán la introducción de sistemas expertos en casi todas las áreas que ordinariamente reclaman una técnica cualificada.

Los sistemas verdaderamente inteligentes, tales que nos permitan interactuar con ellos al mismo nivel que entre seres humanos, abrirán un camino hacia el saber muy similar al que hasta hoy ha abierto la educación formal. La diferencia capital es que los sistemas inteligentes serán fuentes de conocimiento constantemente accesibles, con capacidad de adaptar y filtrar el flujo de conocimientos así como de satisfacer las necesidades de los individuos a lo largo de sus vidas.

Referencias

1 G. Jones, R. Nuttall y K. Stone: Integración de múltiples esquemas de control: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 122–127 (en este número).

2 J. J. Harvey: ESSAI, juego de herramientas para sistemas expertos: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 109–114 (en este número).

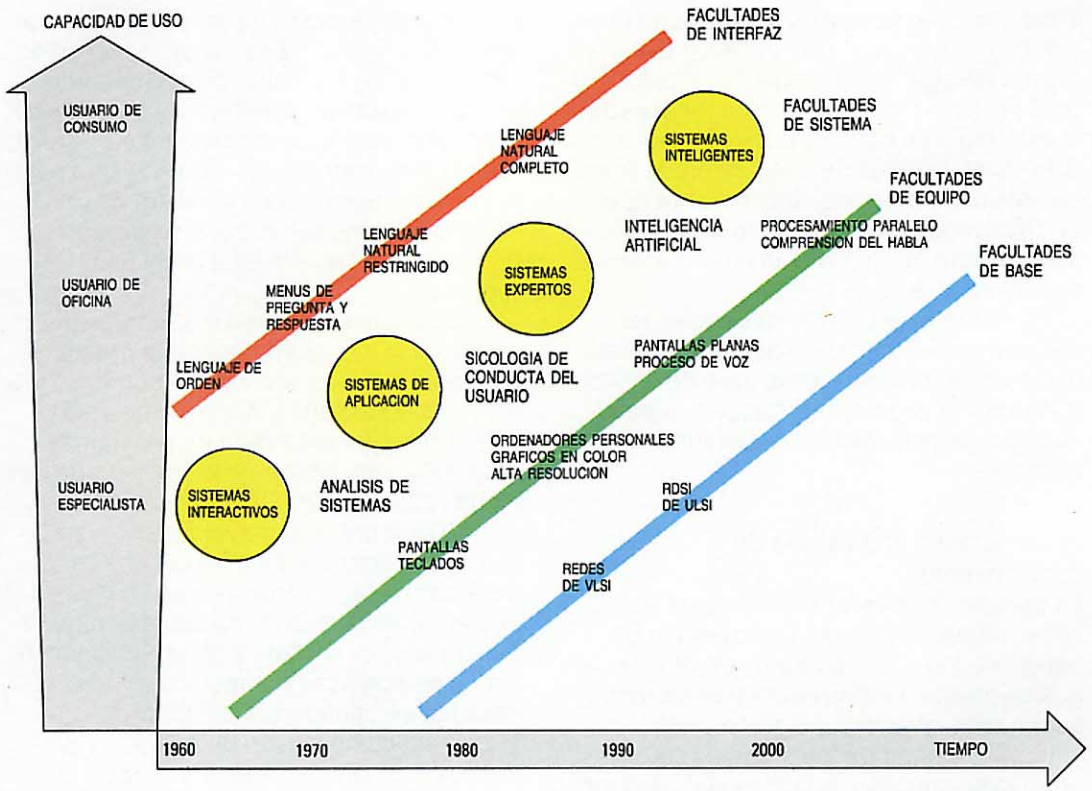


Figura 8
Desarrollo de los sistemas inteligentes en el próximo siglo.

3 R. Schelfhout: Ingeniería de aplicación para circuitos del Sistema 12: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 135–140 (en este número).

4 R. Gunhold y J. Zettel: Pruebas en fábrica del Sistema 12: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 128–134 (en este número).

5 M. Thandasseri: Sistemas expertos aplicados a centrales TXE4A: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 154–161 (en este número).

6 J. Yun Cabrera y G. D. Ketels: Generación de datos para el Sistema 12: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 141–146 (en este número).

7 N. Theuretzbacher: Tecnología de sistemas expertos para sistemas de seguridad crítica en tiempo real: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 147–153 (en este número).

8 M. E. Atwood y E. R. Radlinski: Arquitectura de sistema de diagnóstico: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 174–179 (en este número).

9 M. A. Newstead y R. Pettipher: Adquisición de conocimiento para sistemas expertos: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 115–121 (en este número).

10 M. E. Atwood, R. Brooks, y E. R. Radlinski: Modelos causales: la nueva generación de sistemas expertos: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 180–184 (en este número).

James J. Harvey entró en ITT en 1981 después de trabajar para ICL en el desarrollo de grandes sistemas operativos multifunción. Es actualmente director del grupo de sistemas basados en conocimientos en el Centro de Investigación ESC de ITTE en Harlow, Inglaterra. Este grupo se dedica al desarrollo de herramientas para construcción de sistemas expertos, así como a prestar soporte para el uso de estas herramientas en aplicaciones dentro de la ITT.

ESSAI, juego de herramientas para sistemas expertos

El construir los primeros sistemas expertos fue una larga labor que exigió desarrollar una representación del conocimiento y unas técnicas de búsqueda apropiadas. Ahora el juego de herramientas ESSAI agiliza el desarrollo al poner un surtido completo de técnicas a disposición del ingeniero del conocimiento, que podrá elegir la más adecuada para cada aplicación.

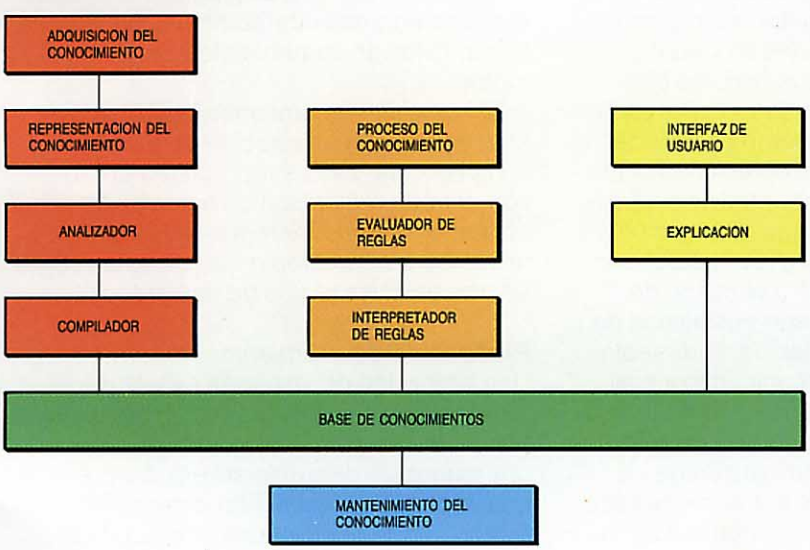
J. J. Harvey
ITT Europe Engineering Support Centre,
Harlow, Inglaterra

Introducción

Ha aumentado mucho recientemente el interés en construir sistemas expertos para resolver una gran variedad de problemas prácticos. En los primeros tiempos de la tecnología, construir un sistema experto era generalmente un proceso largo. Ahora, sin embargo, empieza a haber herramientas que lo simplifican y reducen el tiempo de desarrollo. Un ejemplo es el juego de herramientas ESSAI, utilizable para construir sistemas expertos en diversas aplicaciones, que actualmente está desarrollando el ITT Europe Engineering Support Centre, en Harlow, Inglaterra.

El ESSAI proporciona al ingeniero especializado en tratar el conocimiento distintos formalismos de representación del mismo, mecanismos asociados de inferencia y control y técnicas para interactuar con el usuario. El objetivo es minimizar el nivel de capacitación necesario para aplicar técnicas de sistemas expertos, y abrir al máximo el abanico de técnicas disponibles dentro de un contexto común.

Figura 1
Arquitectura del juego de herramientas ESSAI para construcción de sistemas expertos.



Arquitectura de ESSAI

La arquitectura del sistema ESSAI, representada en la figura 1, comprende diversos componentes e interacciones.

Representación del conocimiento
Lo esencial en un sistema experto es separar el conocimiento usado en la aplicación del mecanismo que sirve para procesar dicho conocimiento. El esquema principal de representación del conocimiento en el juego de herramientas ESSAI se basa en reglas que utilizan una forma de lógica con evaluación variable. Generalmente las reglas tienen la forma SI <*condiciones*> ENTONCES <*acciones*>, e incumbe al ingeniero definir las condiciones que, cuando resultan ser verdaderas, provocan las acciones o decisiones.

Sin embargo, en el ESSAI se utiliza una variante de este formato de reglas. En su expresión más sencilla, las reglas tienen la sintaxis siguiente:

[CONDICION] ::> [CONCLUSION]

en la cual CONDICION es una expresión formal en lógica de evaluación variable, que implica declaraciones elementales de condición (llamadas selectores), unidas por diversos operadores lógicos (que incluyen cuantificadores). CONCLUSION define la decisión o acción que se ejecuta cuando CONDICION se cumpla.

Para asegurar la independencia del lenguaje, la sintaxis utiliza notación simbólica

Figura 2
Sintaxis de reglas en ESSAI. En este caso la diagnosis obtenida se añade al conjunto de diagnosis que se está formando.

Estructuras de datos simples

Sintaxis del juego de herramientas

```
STARTER_MOTOR_RULE
[BATTERY = CHARGED]
[ENGINE_TURNS = WONT_TURN, SLOW]
[IGNITION_PROBLEM = NO]
[SOLENOID_PROBLEM = NO]
::>
[DIAGNOSIS = STARTER_MOTOR PROBLEM]
```

Lenguaje equivalente

Si la batería está cargada
y el motor no gira o gira lento,
y no hay problemas de encendido,
y no hay problemas de solenoide,
entonces
la diagnosis es: problema en el motor de arranque

Estructuras de datos complejos

Sintaxis del juego de herramientas

```
DISTRIBUTOR_RULE
[POINTS = DIRTY, WORN]
V
[DISTRIB_CAP = DAMP, CRACKED]
::>
[DIAGNOSIS = DIAGNOSIS U {DISTRIBUTOR}]
```

Lenguaje equivalente

Si los puntos están sucios o gastados
o
la tapa está húmeda o rajada,
entonces
la diagnosis es: distribuidor

en vez de palabras clave. En su forma más simple, un selector o condición de una regla consta de un nombre que identifica una variable o estructura de datos, un símbolo que denota alguna operación sobre la variable, y un valor o expresión. La figura 2 ilustra lo anterior, mostrando cómo podría definirse una regla para localizar un fallo en un coche.

Por su concepción, el lenguaje de descripción mediante reglas ha de ser legible y requerir una capacitación mínima de los expertos que construyan tales reglas. El objetivo es que el experto, generalmente no programador, sea capaz de dominar el sistema prontamente e introducir luego las reglas.

Los nombres, fijados por el ingeniero del conocimiento, pueden estar en cualquier idioma, así como los textos asociados con aquellas variables sobre las que pueda solicitarse alguna información al usuario en tiempo de ejecución.

Las variables en la regla dada como ejemplo acceden a estructuras simples de datos, es decir, las que utilizan valores enteros, reales, nominales (valores con nombres predefinidos), o cadenas de caracteres. Las reglas pueden también acceder a estructuras complejas de datos tales como "sets" (colección de elementos no duplicados y no ordenados), "bags" (colección de elementos no ordenados y con posibilidad de duplicación), "arrays" (colección de elementos ordenados y con posibilidad de duplicación), registros y tablas. Todas estas estructuras pueden estar anidadas a cualquier nivel, permitiendo que en las reglas se reúna y referencie una nutrida colección de elementos de datos. La utilización de estructuras complejas de datos se muestra en el segundo ejemplo, en el que la diagnosis efectuada se añade al conjunto (set) de diagnosis posibles (Fig. 2).

El ingeniero puede incorporar, cuando se necesiten, funciones de usuario, es decir conjuntos de instrucciones que se invocan desde las expresiones incluidas en las reglas. Corresponde al usuario elegir el lenguaje de programación para codificar dichas funciones.

Desde cualquier punto de la regla donde se permita una expresión puede llamarse a una función de usuario y utilizar ésta para evaluar una función compleja, acceder a un dispositivo externo o escribir en la pantalla. Estas funciones pueden realizar tareas que normalmente serían imposibles con los medios de representación que posee el sistema (p. ej., cálculos de tráfico telefónico, acceso a ficheros externos).

Una aplicación de sistemas expertos es divisible en varios subproblemas, cada uno de los cuales recoge el conocimiento necesario para su resolución. Por ejemplo, un sistema de diagnosis de automóviles puede dividirse en tres subproblemas: fallos eléctricos, fallos de combustible, y fallos de motor.

El juego de herramientas ESSAI apoya este enfoque de resolución de problemas, al representar cada subproblema por un conjunto de reglas con un nombre; tal conjunto, o *grupo de reglas* tiene su mecanismo de control propio que describe cómo se procesan las reglas de ese grupo.

Proceso del conocimiento

Una aplicación de sistemas expertos requiere una gama de modos de procesar el conocimiento posibles, que vayan desde una expresión determinista en un extremo a una expresión en absoluto determinista en el otro. La opción determinista faculta al

ingeniero para imponer el orden de proceso del conocimiento, mientras que la no determinista permite especificar una estrategia general para procesar las reglas. En el ejemplo de diagnosis de coches, el mecánico puede escoger inicialmente una estrategia determinista, buscando primero problemas eléctricos, después problemas de motor y, por último, problemas de combustible. Dentro de cada una de estas áreas de posible fallo se puede adoptar una estrategia no determinista realizando una búsqueda "dirigida por hipótesis", es decir, el mecánico supone una causa de fallo y busca luego la confirmación del supuesto utilizando su conocimiento de los síntomas y efectos.

El ESSAI propicia este enfoque al incorporar múltiples esquemas de control, que el ingeniero puede especificar como parámetros.

El conocimiento de control, especificado separadamente del conocimiento declarativo, se asocia con el grupo de reglas. El ingeniero es libre de elegir la opción de proceso del conocimiento, probando varios esquemas para determinar el óptimo en una aplicación concreta.

El componente procesador de conocimiento de ESSAI ejecuta estas opciones, dando valores sustitutivos en donde así convenga, de tal forma que el ingeniero no esté siempre obligado a elegir. Por ejemplo, el mecanismo de inferencia implícito es la concatenación progresiva con búsqueda "transversal". Este componente comprende las partes que contienen la lógica de selección de grupo de reglas y de reglas, así como su posterior evaluación e interpretación. Accede al contenido de la base de conocimientos, pide al usuario valores para las variables desconocidas, determina la veracidad de las porciones de conocimiento por medio del "evaluador de reglas" y formula conocimiento nuevo utilizando el "interpretador de reglas".

El evaluador de reglas determina el grado de verdad del lado izquierdo de una regla (entero o en parte) mediante un valor comprendido entre *0* y *1*, donde *0* indica falso y *1* verdadero. El peso asociado a la condición indica el nivel de confianza en la veracidad de la afirmación que dicha condición contiene. Los parámetros de evaluación especifican cómo hay que combinar los grados de verdad asignados a las variables del lado izquierdo para obtener el grado de verdad de la condición completa.

El interpretador de reglas ejecuta lo expresado en el lado derecho, es decir, interpreta la parte de decisión o acción.

Interfaz de usuario

El diseño del juego de herramientas ESSAI subraya la necesidad de simplificar la interacción entre el sistema experto y el usuario. Para lograrlo se ofrece una gama de facilidades, desde un interfaz interactivo sustitutivo, capaz de suplir la pasividad del usuario en tiempo de ejecución, hasta un interfaz que pueda ser totalmente configurado por el ingeniero. Por consiguiente, el interfaz puede ajustarse a la aplicación, utilizando valores sustitutivos cuando sea adecuado. De esta manera se consigue un formato de pantalla en tiempo de ejecución exactamente a gusto del usuario.

La interacción de tipo sustitutivo se basa en dispositivos de caracteres y utiliza un sistema de ventanas, dividiendo la pantalla en zonas utilizadas para diferentes fines. La zona superior registra el nombre del grupo de reglas que se está procesando, la inferior muestra las preguntas del sistema y aportaciones del usuario, y la zona media presenta las respuestas que el usuario da a tales preguntas. El tamaño de las zonas lo fija de un modo implícito el sistema, aunque puede variarlo el ingeniero.

El modo de interacción sustitutiva se vale de preguntas y respuestas, siendo el sistema quien pregunta. La forma de la pregunta depende de la naturaleza de la información solicitada: puede ser una lista de alternativas, un valor único, o una tabla de valores.

La selección de la pregunta depende de cómo se procesa el conocimiento. Un posible enfoque es que la selección sobrevenga como efecto de una programación no determinista, en la que el sistema intente procesar una regla y compruebe que ello no es posible porque necesita cierta información. Si ésta la puede obtener procesando otra regla, el sistema suspende el proceso de la primera y emprende el de esta nueva regla. Sin embargo, si la información sólo puede aportarla el usuario, el sistema hace una pregunta cuyo texto lo fija el ingeniero de un modo tal que facilite al usuario el introducir los valores apropiados.

El texto puede ir en el idioma que elija el ingeniero. Esta "libertad de idioma" tiene aún mayor alcance, pues todos los mensajes del sistema están contenidos en un fichero que le acompaña y por ello pueden traducirse a cualquier lengua sin más que sustituir el texto correspondiente en el citado fichero; éste se utiliza también para formar la pantalla inicial.

Otro modo de solicitar datos del usuario es mediante una secuencia determinista de preguntas, establecida por el ingeniero del conocimiento.

Antes de que comience el proceso interactivo puede introduirse en el sistema un fichero de datos que contenga cualquier número de variables con sus valores asociados.

El juego de herramientas ESSAI permite la independencia de los dispositivos, de modo que se puedan aprovechar las ventajas de los gráficos de alta resolución, sin dejar de admitir los dispositivos de caracteres.

A través de un menú se ofrecen al usuario varias opciones de visualización en pantalla durante el proceso, incluyendo la presentación de reglas y diversas características más que faciliten la explicación.

El usuario está facultado para suspender una sesión y llevarla a un fichero determinado de forma que pueda reanudarse posteriormente. Cuando se recarga el sistema el usuario tiene la opción de restablecer una sesión así guardada, en cuyo caso el sistema procesa el fichero preservado hasta el punto en el que se paró la sesión, y a partir de ahí prosigue la interacción con el usuario.

Facilidad de explicación

El modo de explicación da opción a visualizar el conjunto de reglas que está considerando el sistema, así como a obtener información adicional sobre las preguntas dirigidas al usuario mientras se procesa el conocimiento.

El ESSAI permite cuestionar la decisión tomada por el sistema mediante preguntas tales como *¿Por qué?*, *¿Cómo?*, y *¿Por qué no?*. Justifica sus propias acciones por las condiciones que dieron lugar a ellas, y es capaz de mostrar cómo estas acciones encajan en la línea de razonamiento seguida. Con esta base el sistema describe las otras alternativas de razonamiento consideradas y el porqué de su rechazo. Una vez que el sistema llega a los resultados, el usuario puede cuestionar cualquiera de ellos y descubrir cómo lo obtuvo el sistema. Para ello se rastrean las reglas en orden inverso, desde la más reciente hasta la primera, mostrando el proceso lógico – y las reglas – que ha utilizado el sistema para llegar a ese resultado concreto; en todo esto se hace uso de los textos asociados con las reglas y se da opción a exponer los detalles de las mismas.

Construcción de un sistema experto

El ingeniero del conocimiento que utiliza el ESSAI para construir un sistema experto, crea un fichero que describe los tipos de variables en el problema, las variables con las que resolverlo y los grupos de reglas.

El fichero de reglas se escribe mediante un editor de textos. Primero describe todos los datos que requiere la aplicación, junto con la declaración de todas las funciones de usuario. A continuación define las reglas que han de ejecutarse en cada grupo de reglas, junto con cualquier opción específica de control en dichos grupos. También incluye la definición del control requerido sobre los grupos de reglas, especificando así su secuencia de ejecución.

Analizador de la base de conocimientos

El analizador realiza un análisis sintáctico de las reglas, informando al usuario de los eventuales errores; toma reglas obtenidas de los expertos o bien inducidas de ejemplos, y las analiza para producir una representación intermedia que sirva de entrada al proceso a efectuar sobre el conocimiento. Normalmente se utiliza el analizador durante el desarrollo progresivo y refinamiento de la aplicación, y el compilador cuando la aplicación es ya operacional y adquiere importancia el comportamiento en tiempo de ejecución.

Compilador

El compilador incremental compila opcionalmente la base de conocimientos a código fuente, que a su vez puede compilarse y montarse, mejorando esto sustancialmente el rendimiento del proceso del sistema.

Validación de conocimiento

El proceso de validación considera dos aspectos de la base de conocimientos: coherencia e integridad. La coherencia afecta a reglas contradictorias (es decir, reglas con partes de decisión que se excluyan entre sí y que puedan satisfacerse al mismo tiempo), y reglas redundantes (reglas con partes de decisión idénticas y que tengan el mismo efecto). La integridad asegura que existen reglas para dar solución a todas las situaciones posibles.

Ejecución de la aplicación de sistema experto

Una vez completado, el sistema experto se puede ejecutar en forma interactiva o por lotes. En el primero de los supuestos, se muestra la pantalla inicial si se emplea el interfaz sustitutivo; esta pantalla se acomoda a cada aplicación, y en ella se pregunta al usuario el nombre de la base de conocimientos a utilizar en la sesión, que puede ser bien el de un fichero creado por el analizador, o bien el de un fichero creado en una sesión anterior mediante la opción de guardar antes citada.

El sistema experto carga entonces el fichero de la base de conocimientos, y tras su lectura el sistema pregunta al usuario el nombre de un fichero de datos de entrada

en el cual pueda dar el usuario todos o parte de los datos necesarios para el sistema experto. Esto será útil cuando una parte de los datos sea constante a lo largo de varias sesiones, o cuando algunos datos provengan de otro proceso.

El sistema experto procesa el fichero actualizando las variables en la base de

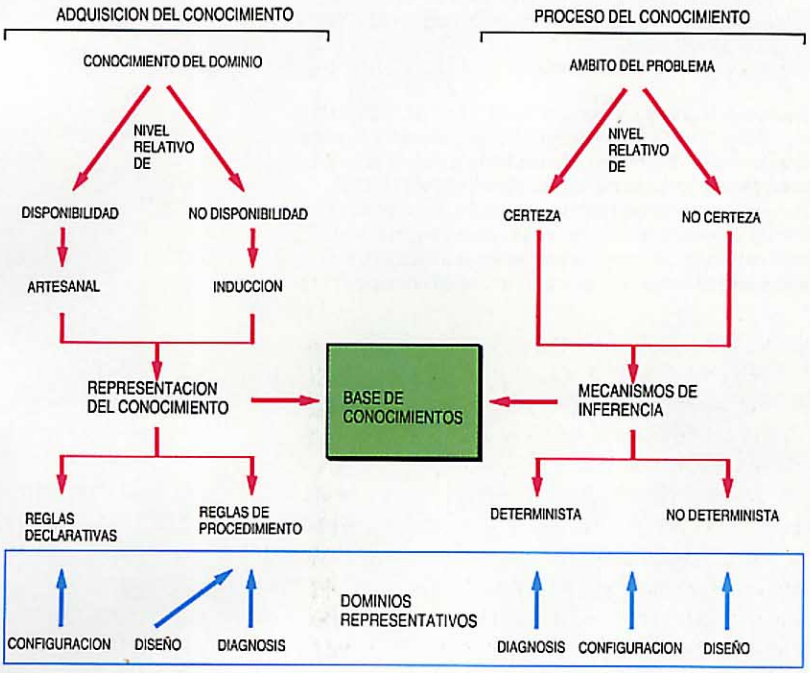


Figura 3
Objeto del juego de herramientas para sistemas expertos ESSAI.

conocimientos con los oportunos valores de datos y presentándolas en la pantalla. Cuando se han procesado todos los datos, se avisa al usuario y se le da la oportunidad de revisarlos antes de continuar.

Seguidamente el sistema experto selecciona grupos de reglas para ser procesados en la secuencia que definió el ingeniero cuando construyó el sistema. Cada grupo de reglas se identifica por su nombre durante el proceso; terminado el proceso de un grupo de reglas, se presentan todos los resultados asociados con ese grupo y el proceso continúa con el grupo siguiente.

Es posible observar cómo se va procesando el conocimiento, pues puede sacarse a un fichero información sobre el proceso de las reglas a medida que avanza su ejecución.

Facilidades del juego de herramientas

El ESSAI ofrece al ingeniero del conocimiento las facilidades que necesita durante las principales fases de la construcción de un sistema experto, o sea, durante la adquisición de conocimiento y su proceso (Fig. 3).

La adquisición de conocimiento quizás sea el aspecto que más tiempo consume en la ingeniería del conocimiento. Se han desarrollado herramientas informáticas como soporte de las dos técnicas principales de adquisición de conocimiento: la artesanal y la inducción de reglas a partir de ejemplos[1]. Uno de los objetivos del juego de herramientas es el permitir ambos enfoques, integrándolos merced al uso de una representación de conocimiento común. De esta forma, el ingeniero puede escoger una u otra técnica según su propio criterio, y combinarlas como convenga. Así, pues, los resultados de la inducción se pueden modificar por técnicas artesanales, y a su vez los obtenidos por procedimientos artesanales pueden valer como entrada a procesos inductivos para generar un conjunto de ejemplos que caracterice a las reglas artesanales.

La elección entre una y otra técnica se ve así reducida a una decisión basada en el nivel relativo de disponibilidad del conocimiento. Considerando áreas de aplicación representativas en ITT como la diagnosis, los casos suelen presentarse en forma de ejemplos de diagnosis ya completadas, y en tales situaciones la inducción es el punto de partida más natural. Por otro lado, los sistemas de configuración están normalmente bien definidos en manuales de reglas que constituyen una base sólida para el método artesanal. Los sistemas de diseño, como los de placas de circuito impreso, se sitúan en un terreno intermedio en el que suele haber reglas que establecen la capacidad de pruebas y de fabricación de los diseños, y al mismo tiempo gran parte del diseño es creativo y se apoya en ejemplos.

Desde el punto de vista de la herramienta, la decisión del ingeniero del conocimiento sobre el tipo de proceso radica en la libertad de elegir la técnica apropiada a la aplicación, y ello se apoya en el uso de técnicas de representación de conocimiento comunes, tales que se pueda pasar de una a otra según convenga.

El ingeniero se enfrenta a otro dilema cuando prepara la aplicación para el proceso del conocimiento[2]. La elección de la técnica para tratar un conocimiento específico depende del nivel relativo de certeza en llegar a una solución dentro del ámbito del problema que esa aplicación describe.

Volviendo a considerar las aplicaciones representativas, en diagnosis se tiende a problemas con solución determinista, es decir, que se resuelven a través de pruebas sistemáticas descritas por reglas. Por su parte, los sistemas de diseño tienden a ser casi enteramente no deterministas, y

en ellos la solución se alcanza mediante un conjunto de normas heurísticas modificadas por ciertas limitaciones. Los sistemas de configuración parecen ocupar el punto medio, al poder ser prescritas las fases principales de modo determinista, aunque en procesos intermedios se aplique la heurística.

Aunque estos ejemplos sean solamente generalizaciones, refuerzan la idea de que el juego de herramientas debe ofrecer una gran variedad de técnicas de proceso del conocimiento, abarcando desde el control determinista al no determinista, y proporcionando valores sustitutivos de la decisión del ingeniero del conocimiento de modo que éste no se vea obligado a elegir siempre. Esto lo consigue el ESSAI al tener la posibilidad de especificar el conocimiento de control separado del conocimiento declarativo; hay que insistir en que el disponer de varias técnicas diferentes proporciona una máxima versatilidad y permite probar diferentes caminos para determinar la técnica que mejor se adapte a la aplicación.

Referencias

1 M. A. Newstead y R. Pettipher: Adquisición de conocimiento para sistemas expertos: *Comunicaciones Eléctricas*, 1986, volumen 60, nº 2, págs. 115–121 (en este número).

2 G. Jones, R. Nuttall y K. Stone: Integración de múltiples esquemas de control: *Comunicaciones Eléctricas*, 1986, volumen 60, nº 2, págs. 122–127 (en este número).

James J. Harvey se incorporó a ITT en 1981, después de trabajar en ICL en el desarrollo de grandes sistemas operativos multifunción. Actualmente dirige el grupo de sistemas de gestión de conocimiento en el ITT-ESC Research Centre de Harlow, Inglaterra. Este grupo se dedica a desarrollar herramientas para construir sistemas expertos, así como a promover la utilización de estas herramientas en aplicaciones en el seno de ITT.

Adquisición de conocimiento para sistemas expertos

La adquisición de conocimiento es una importante causa de atascos en el desarrollo de sistemas expertos. A la solución de tal problema contribuye la adquisición automática por técnicas de inducción de reglas, que reduce el tiempo gastado en preparar las reglas a mano.

M. A. Newstead
R. Pettipher
ITT Europe Engineering Support Centre,
Harlow, Inglaterra

Introducción

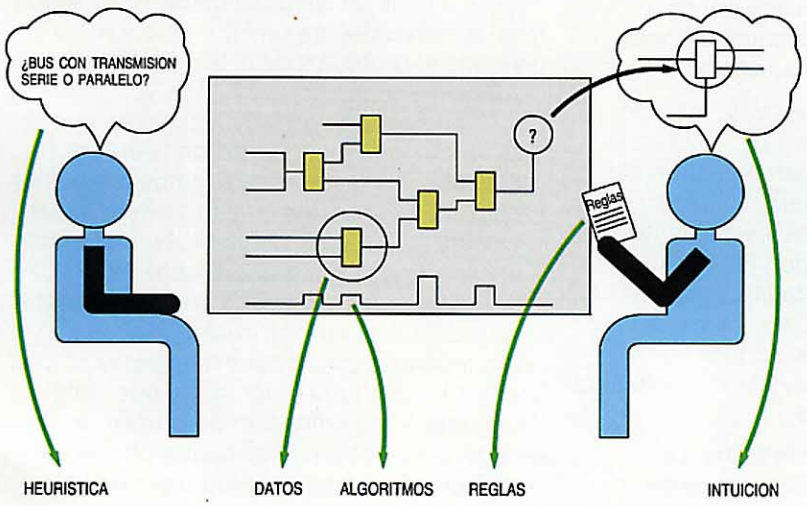
La ingeniería del conocimiento es el proceso de convertir el conocimiento de un especialista a una forma utilizable por un sistema experto para resolver problemas en un área particular o "dominio". A medida que las herramientas para elaborar sistemas expertos se refinan más y comprenden técnicas aplicables a la resolución de problemas en una amplia variedad de aplicaciones, la misión del ingeniero del conocimiento se acerca cada vez más a escoger una representación apropiada del conocimiento del experto. La adquisición de conocimientos poseídos por tales expertos es pues la clave de la ingeniería del conocimiento.

Dicha adquisición es uno de los grandes problemas que entorpecen el desarrollo de sistemas expertos, por lo que actualmente se investiga mucho para mejorar las técnicas disponibles. El problema nace de que, en general, hay que obtener conocimientos de expertos que los han adquirido a lo largo de los años, y tales conocimientos cubren un vasto campo. Como ejemplo consideremos los utilizados en el diseño de circuitos electrónicos (Fig. 1) que incluyen: datos básicos sobre las propiedades de componentes electrónicos; algoritmos de simulación del comportamiento del circuito; reglas que definen las limitaciones relativas a capacidad de fabricación y prueba; heurística o normas "caseras" que proceden de la experiencia y de aquella clase de intuición propia de los ingenieros más cualificados, capaces de dar soluciones originales a problemas no resueltos.

Las dificultades provienen de que la habilidad del experto ha sido "interiorizada" y no suele haber acceso a sus conocimientos. Normalmente los expertos no plasman su saber en reglas de fácil acceso que puedan traducirse directamente a una base de conocimientos, sino más bien en forma de reglas prácticas, asimilación a otros casos anteriores, intuición y procesos análogos, elaborados durante muchos años de experiencia. Los mecanismos realmente utilizados para resolver un problema y tomar una decisión son a menudo poco evidentes para el experto o difíciles de explicar. En tales casos, corresponde al ingeniero del conocimiento hacer supuestos razonables, que el experto ha de comprobar.

Se siguen básicamente dos métodos para la adquisición de conocimiento: el artesanal y el inductivo por ordenador. Ambos están concebidos para identificar los hechos y conceptos básicos utilizados por los expertos, y las interrelaciones entre tales hechos.

Figura 1
Conocimiento de expertos.



Método artesanal

La técnica principal de adquisición de conocimiento es artesanal, preparando a mano el ingeniero una representación del conoci-

miento que maneja el experto en la resolución de un problema. Este proceso se simplifica si el conocimiento relevante existe en forma de guías o manuales, como sucedió al utilizar un sistema experto en la ingeniería de aplicación al cliente[1]. Sin embargo, este caso no es muy común, y la ingeniería del conocimiento se convierte en un proceso de descubrimiento que requiere interacción entre el ingeniero y el experto.

Una forma de adquirir conocimientos son las entrevistas con el experto o expertos para descubrir los procesos decisorios y después, utilizando métodos manuales, escribir reglas que reflejen estos procesos. Esta técnica puede ser laboriosa, pues los expertos tienen a menudo dificultad en explicar sus procesos mentales, y en muchas áreas no se ponen de acuerdo, obligando a consultar a muchos para alcanzar el consenso. El desarrollo de la base de conocimientos es, pues, un proceso iterativo en el que se producen reglas, se revisan, se comentan con los expertos y se actualizan cuando sea oportuno.

La clave de un método artesanal eficiente es un eficaz diálogo entre el experto y el ingeniero del conocimiento. Seguidamente se señalan algunas técnicas que contribuyen a este proceso.

Entrevistas dirigidas: en ellas se le pide al experto centrarse en un aspecto de la tarea de resolver problemas, explorándolo a fondo para lograr una imagen detallada del área de aplicación. Al ingeniero del conocimiento le corresponde explorar las sucesivas soluciones aplicadas, las razones para ejecutar o evitar acciones concretas, y el uso de cualquier fuente externa como manuales, libros, herramientas u otros expertos. El proceso se puede complementar con organigramas que muestren la secuencia de decisiones propuestas, y grabaciones para un análisis posterior. A menudo será útil valerse de un caso concreto como medio para centrar la entrevista, y después reunirse con un grupo de expertos que puedan enriquecer y criticar el método de resolución propuesto por el experto para el caso en cuestión.

Entrevistas estructuradas: este método orientado a los conceptos se utiliza en la teoría de construcción personal de Kelly, desarrollada para psicoterapia. La técnica permite al psicoterapeuta y al paciente identificar y analizar rasgos del carácter, habiendo sido automatizada y ampliada su finalidad para formar la base de unos sistemas semiautomáticos de adquisición de conocimiento con los que el experto pueda interactuar directamente.

Auto-información: en esta técnica el experto piensa en voz alta y describe el protocolo que utiliza para resolver el problema. El ingeniero del conocimiento analiza luego detalladamente este protocolo para aislar los pasos clave y los criterios esenciales aplicados por el experto para llegar al resultado.

En la práctica se utilizará una combinación de estas técnicas durante el ciclo de desarrollo, terminando con una revisión por los expertos del prototipo de sistema conseguido. Esta revisión suele consistir en la presentación de un problema de prueba al prototipo, seguido de un análisis crítico de sus prestaciones por los expertos, el cual normalmente descubre faltas de conocimiento, aplicaciones incorrectas, y decisiones incompletas o erróneas. Por supuesto es importante que el experto escoja el caso de prueba y que éste concuerde con el objetivo fijado al desarrollar el sistema. Los resultados dan lugar a actualizaciones de la base de conocimientos y a la continuación del desarrollo.

Inducción

La inducción es un método diferente para adquisición de conocimiento que supera algunas de las dificultades del proceso artesanal. Se basa en la premisa de que el área de aplicación pueda describirse mediante ejemplos que caractericen la habilidad del experto en la resolución de problemas. El papel de la inducción es el de generalizar estos ejemplos en forma de reglas que puedan utilizarse en un sistema experto operacional para examinar nuevos problemas y determinar si encajan en alguno de los tipos generales. La eficacia de la inducción depende de que existan ejemplos representativos completos, adecuadamente clasificados por un experto. La inducción pretende proporcionar algoritmos informatizados capaces de interactuar directamente con el experto para el desarrollo de la base de conocimientos, reduciendo así la necesidad de un ingeniero y resolviendo el problema de la adquisición de conocimientos.

El primer paso en la inducción es que el experto identifique, junto con la propia decisión, los factores relevantes (atributos) para llegar a ella, los cuales se denominan valores o clases de decisión. Normalmente el experto no encuentra dificultad en esta tarea. Además, la mayoría de los algoritmos de inducción eliminan cualquier atributo redundante señalado por el experto, lo cual es particularmente importante pues implica un periodo de refinación de la base de conocimientos y posibilidades de mejorar los conocimientos del propio experto.

Tabla 1 — Criterios para seleccionar la técnica de adquisición de conocimiento

Usar la artesanal si:	Usar la inductiva si:
El conocimiento está ya recogido en forma de manuales, instrucciones, etc.	Se puede obtener un conjunto de ejemplos representativos
Se dispone de acceso a expertos	No se dispone fácilmente de expertos
Los expertos no tienen dificultad en explicar sus razonamientos	Los expertos tienen dificultad en explicar sus razonamientos
El dominio es grande	El dominio es finito y bien definido
El conocimiento es específico y las decisiones están delimitadas	

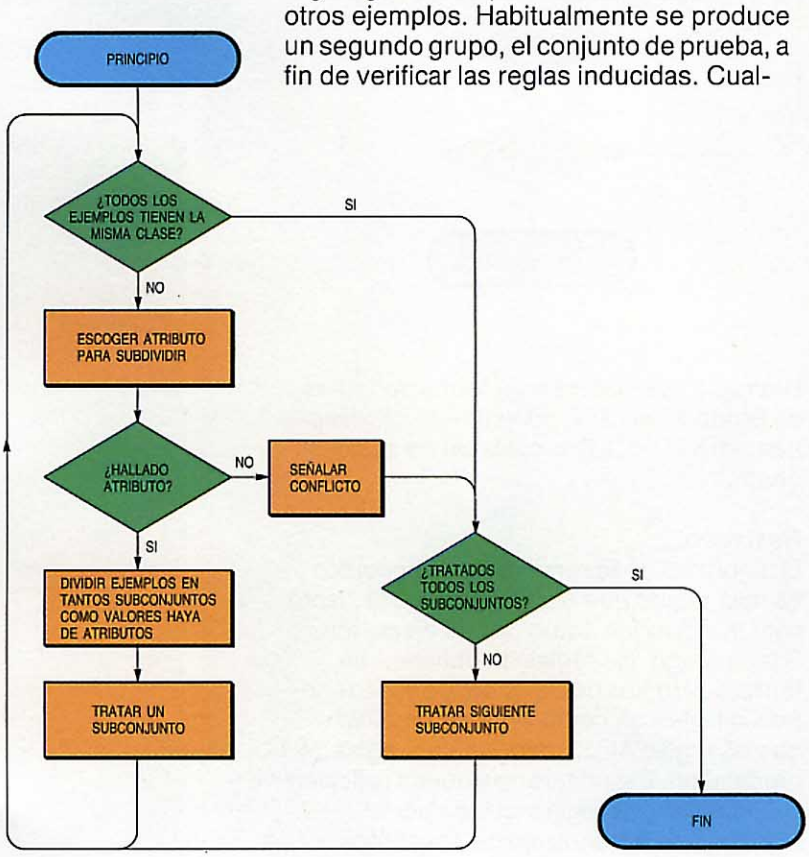
El segundo paso es obtener un grupo de ejemplos que demuestren la habilidad del experto en resolver problemas, tales como una serie de diagnósticos o un conjunto de configuraciones. Es importante elegir ejemplos lo más completos posible, que contengan todas las clases de decisión en el área correspondiente y cubran el máximo campo de aplicación; estos ejemplos pueden tomarse de casos reales donde ya los expertos hayan llegado a una conclusión. Por otro lado, se puede construir automáticamente un conjunto de ejemplos utilizando la descripción de los atributos, y luego el experto les puede asignar una clase de decisión.

El programa de inducción examina el anterior grupo de ejemplos, llamado de adiestramiento, y a partir de ellos obtiene reglas generales para la clasificación de otros ejemplos. Habitualmente se produce un segundo grupo, el conjunto de prueba, a fin de verificar las reglas inducidas. Cualquier ejemplo del conjunto de prueba no clasificado correctamente por las reglas se puede transferir al grupo de adiestramiento y luego volver a inducir las reglas. Siempre que se encuentren nuevos ejemplos, podrán éstos añadirse al conjunto de adiestramiento, permitiendo así un aprendizaje progresivo.

Algunos sistemas de inducción manejan solamente conceptos individuales, en los que la clase de decisión tiene que recibir uno de dos únicos valores, normalmente *verdadero* y *falso;* entonces los ejemplos forman casos positivos o negativos de este concepto. Esto sucedería, por ejemplo, al probar un circuito respecto a un fallo concreto, siendo el resultado simplemente la *presencia* o *ausencia* del fallo. Otros sistemas pueden tratar conceptos múltiples, donde la clase de decisión reciba uno entre muchos valores; así ocurre en los diagnósticos, que el fallo puede localizarse en uno de los numerosos componentes.

La inducción no es apropiada en todos los casos; en algunos el método artesanal es mejor. Se pueden definir algunos criterios (Tabla 1) que ayudan a seleccionar la técnica de adquisición de conocimiento idónea.

Figura 2
Algoritmo de inducción por aprendizaje de conceptos: se produce un conflicto cuando dos ejemplos del conjunto de adiestramiento tienen idénticos atributos pero clases diferentes.



Algoritmo de inducción 1: sistema de aprendizaje de conceptos

El aprendizaje de conceptos se basa en la división repetida del conjunto de ejemplos en subconjuntos que dependen de los diferentes valores de un atributo escogido, el cual es distinto para cada división, hasta que todos los ejemplos de cada subconjunto tengan la misma clase de decisión, o hasta que no haya más atributos disponibles para crear más subdivisiones. El algoritmo funciona del siguiente modo (Fig. 2):

- toma el conjunto de ejemplos
- se detiene si todos los ejemplos tienen la misma clase
- elige el mejor atributo para subdividir respecto al mismo
- divide los ejemplos en tantos subconjuntos como valores haya de ese atributo
- para cada subconjunto repite de modo recurrente desde el segundo paso.

La concisión de la regla inducida depende del criterio adoptado en la selección de un atributo idóneo para dividir el conjunto de ejemplos; en este caso se utiliza un algoritmo basado en la teoría de la información y se selecciona el atributo que maximiza la diferencia en entropía entre el sistema antes y después de la subdivisión.

Las reglas deducidas se obtienen en formato de árbol de decisión y pueden ser

modificadas por el experto, ya sea directamente o añadiendo ejemplos e induciendo de nuevo.

Algoritmo de inducción 2: AQ11
Mientras que el algoritmo por aprendizaje de conceptos empieza con una regla muy general y le aplica sucesivas restricciones, el AQ11 parte de un conjunto de condiciones muy específico que generaliza progresivamente. En el AQ11 se manejan conceptos múltiples, pero cada clase de decisión se trata como si fuese un concepto aislado; los ejemplos que resultan en la clase se toman como ejemplos positivos del concepto, y todos los demás como ejemplos negativos. El algoritmo construye una *estrella*, formada por un conjunto de términos o *compuestos* en el que cada compuesto consiste en una serie de condiciones unidas por AND lógicos. Estos compuestos llegarán a formar las partes condicionales de las reglas finales. La estrella se construye considerando un ejemplo positivo cada vez y construyéndole una *estrella parcial*, lo que se hace examinando cada uno de los ejemplos negativos y produciendo las condiciones apropiadas para poder excluirlo sin excluir el ejemplo positivo. Esta estrella parcial se reduce a un solo compuesto antes de añadirse a la lista de reglas que forman la estrella final. El algoritmo funciona así (Fig. 3):

Para cada clase,
por cada ejemplo positivo:

- Inicializar la estrella parcial a la unidad.
- Después, para cada ejemplo negativo:

  si el ejemplo negativo está considerado en la estrella parcial, excluirlo;

  utilizar algebra Booleana para excluir compuestos innecesarios en la estrella parcial;

  desechar todos los compuestos menos un cierto número definido por el usuario, aplicando el criterio de selección que él mismo establezca;

  tomar el ejemplo negativo siguiente y empezar de nuevo.
- Cuando se hayan excluido todos los ejemplos negativos, añadir "el mejor" compuesto de la estrella parcial a la lista de compuestos que formarán la estrella final.
- Quitar los ejemplos positivos considerados en este compuesto del conjunto de ejemplos positivos.
- Considerar el ejemplo positivo siguiente y empezar de nuevo.
- Tomar la clase siguiente y empezar de nuevo.

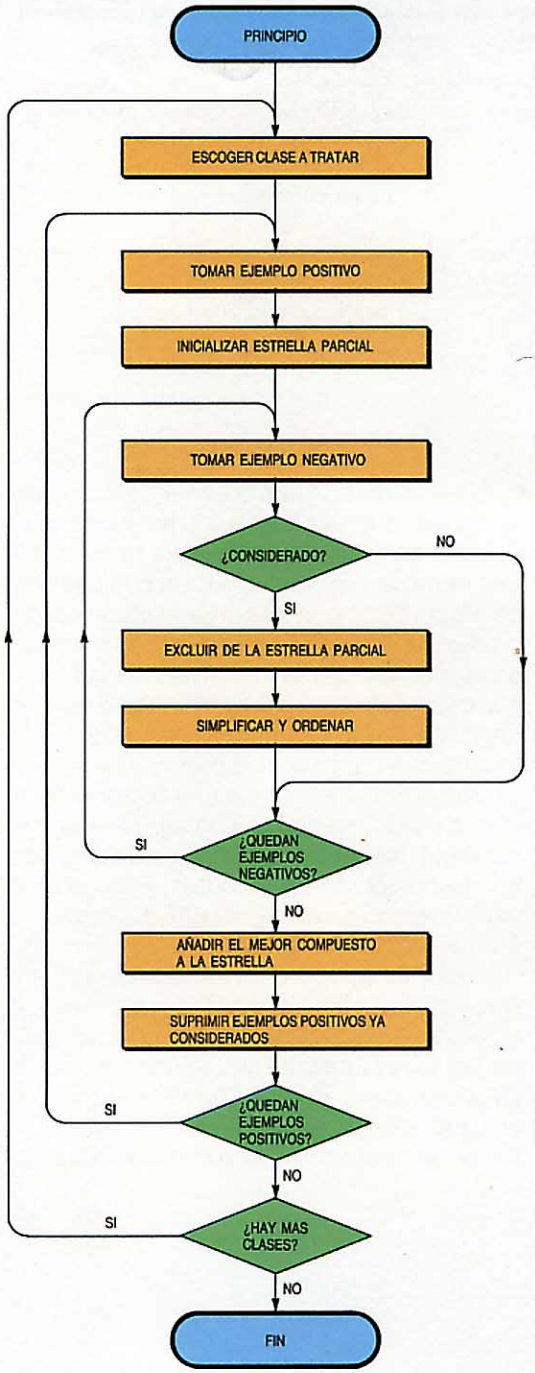


Figura 3
Algoritmo de inducción AQ11.

Las reglas se obtienen en forma de reglas de producción: SI <*conjunto de condiciones*> ENTONCES <*conjunto de acciones*>.

Resumen
El algoritmo de aprendizaje de conceptos es más rápido que el algoritmo AQ11, tanto para la inducción como para la ejecución. Sin embargo, las reglas se obtienen en forma de árboles de decisión, y por ello son más difíciles de comprender y modificar que las reglas AQ11, dadas como reglas de producción. Esta diferencia queda reflejada al comparar uns regla inducida por el algoritmo de aprendizaje de conceptos

(Fig. 4) con la inducida por el AQ11 (Fig. 5) utilizando el mismo conjunto de atributos, clases de decisión y ejemplos.

Ejemplo de adquisición de conocimiento

El sistema experto para diagnosis de placas de circuito impreso[2] es un ejemplo del uso de las dos técnicas, artesanal e inductiva, en cuyo desarrollo se utilizó una envoltura de sistemas expertos que admite ambas técnicas.

La primera etapa de adquisición de conocimiento fue examinar la tarea global, en este caso la diagnosis de fallos en placas, y dividirla en partes menores y más manejables. Se escogió la placa de interfaz terminal que contiene varios subsistemas importantes: el circuito generador de frecuencia de reloj, el circuito de verificación de redundancia cíclica (CRC) y los puertos. El sistema de diagnosis para esta placa se descompuso, por lo tanto, en cuatro partes principales, una por cada subsistema más una parte supervisora que seleccione y enlace dichos subsistemas.

Mientras los ingenieros diagnosticaban y reparaban placas averiadas, se fueron acumulando numerosos ejemplos de síntomas y diagnósticos que podían utilizarse para inducir reglas cuando fuera posible. Los ejemplos contenían gran cantidad de información: los síntomas originales en forma de códigos de fallo proporcionados por la estación de prueba, el componente en fallo identificado por el ingeniero probador, y las etapas de diagnosis, a través de las cuales se podían obtener los atributos de decisión apropiados. Era claro, viendo los ejemplos, que había información suficiente para utilizar la inducción en algunas áreas de fallo. Se expone aquí un ejemplo simplificado, que permite determinar el subsistema afectado a partir de los códigos de fallo.

Para empezar, se identificaron los atributos o factores de decisión (en este caso los códigos de fallo), y se definieron sus valores. Asimismo se señalaron y enumeraron las clases de decisión que se correspondían con los diferentes subsistemas de la placa. Se escogió un formato de los atributos y de las clases de decisión asociadas (Fig. 6) tal que permitiese su directo procesamiento por la rutina de inducción. Análogamente, los ejemplos se prepararon en un fichero formatado de acuerdo con los atributos y clases de decisión determinados previamente (Fig. 7).

Utilizando el fichero de definición de atributos y clases de decisión junto con el fichero de ejemplos, el algoritmo de inducción generó una regla que se presenta de modo simplificado en la figura 4, la cual hace uso del algoritmo de aprendizaje de conceptos. Esta regla pudo examinarse y modificarse directamente de forma apropiada, examinando luego las demás partes del sistema que trataban los detalles de la localización del fallo.

El ingeniero del conocimiento se reunió con los expertos en diagnosis, quienes con la ayuda del diagrama eléctrico de los subsistemas indicaron las etapas a seguir para diagnosticar una placa averiada. El ingeniero anotó los puntos del circuito que se debían examinar, la instrumentación a utilizar, y los resultados que cabía esperar, cada uno de los cuales se convirtió en un atributo y una posible pregunta en el sistema experto final. La gama de valores

SINTAXIS CLS	INTERPRETACION
[FCC] :	
AA : FREQ	IF FCC = AA THEN FREQ
AB : [FTCODE] :	ELSE IF FCC = AB
BFFC : CRC	AND FTCODE = BFFC THEN CRC
DFFF : FREQ	AND FTCODE = DFFF THEN FREQ
60F : PORT	AND FTCODE = 60F THEN PORT
50A : CRC	AND FTCODE = 50A THEN CRC
AC : CRC	ELSE IF FCC = AC THEN CRC
AD : PORT	ELSE IF FCC = AD THEN PORT
AE : PORT	ELSE IF FCC = AE THEN PORT
AF : PORT	ELSE IF FCC = AF THEN PORT

Figura 4
Ejemplo de una regla inducida mediante el algoritmo de aprendizaje de conceptos.
CLS - sistema de aprendizaje de conceptos.

SINTAXIS AQ11	INTERPRETACION
FREQ RULE	FREQ RULE
[FCC = AA]	IF FCC = AA
V	OR
[FTCODE = DFFF]	IF FTCODE = DFFF
[FCC = AA, AB]	AND FCC = AA OR AB
: : >	THEN
[CLASS = FREQ]	CLASS = FREQ
;	
CRC RULE	CRC RULE
[FTCODE = BFFC, 50A]	IF FTCODE = BFFC OR 50A
[FCC = AB, AC]	AND FCC = AB OR AC
V	OR
[FCC = AC]	FCC = AC
: : >	THEN
[CLASS = CRC]	CLASS = CRC
;	
PORT RULE	PORT RULE
[FCC = AD, AE, AF]	IF FCC = AD OR AE OR AF
V	OR
[FTCODE = 60F]	IF FTCODE = 60F
[FCC = AB, AD, AE, AF]	AND FCC= AB OR AD OR AE OR AF
: : >	THEN
[CLASS = PORT]	CLASS = PORT

Figura 5
Regla inducida para el ejemplo de la figura 4 por el algoritmo AQ11.

DATOS DE PLACA		
NOMBRE	TIPO	VALORES
PLACA	LOGICA	32144 22108 32146 22110
VAR	LOGICA	AAFA ABBA
PCS	LOGICA	9 1

CODIGOS DE FALLO		
NOMBRE	TIPO	VALORES
FTCODE	LOGICA	BFFC DFFF 60F 50A AE AF
FCC	LOGICA	AA AB AC AD

CLASES DE DECISION	
NOMBRE	
FREQ	
CRC	
PORT	

Figura 6
Atributos y clases de decisión asociadas para diagnosis de fallos en una placa de interfaz terminal.

EJEMPLOS DE FREQUENCIA (FREQ)					
PLACA	VAR.	PCS	FTCODE	FCC	CLASE
32144	ABBA	1	-	AA	FREQ
32144	ABBA	1	DFFF	AB	FREQ

EJEMPLOS DE CRC					
PLACA	VAR	PCS	FTCODE	FCC	CLASE
32144	ABBA	1	BFFC	AB	CRC
32144	ABBA	1	DFFF	AC	CRC
32144	ABBA	1	BFFC	AB	CRC
32144	ABBA	1	DFFF	AC	CRC
32144	ABBA	1	50A	AB	CRC

EJEMPLOS DE PUERTO (PORT)					
PLACA	VAR.	PCS	FTCODE	FCC	CLASE
32144	ABBA	1	60F	AF	PORT
32144	ABBA	1	50A	AE	PORT
32144	ABBA	1	60F	AE	PORT
32144	ABBA	1	50A	AD	PORT
32144	ABBA	1	60F	AB	PORT
32144	ABBA	1	DFFF	AD	PORT

"-" REPRESENTA TODOS LOS VALORES POSIBLES

Figura 7
Ejemplos de diagnosis de fallos formatada de acuerdo con los atributos y clases de decisión.

posibles se redujo al máximo, y generalmente se limitó a dos valores: bueno o malo. Por ejemplo, cuando el experto busca una tensión de aproximadamente 5 V en un cierto punto del circuito, el atributo se definiría como "¿tensión entre 4,5 y 5,5?", que puede contestarse con un *sí* o un *no*. Esta técnica simplifica la estructura de las reglas y suprime cualquier ambigüedad para el usuario final.

Seguidamente, el ingeniero y el experto elaboraron las reglas, aptas para ser ampliadas o modificadas en cualquier momento. Los diferentes subsistemas se integraron después en un sistema experto único.

Al poner en uso el sistema, se descubrieron algunos puntos donde las reglas eran incompletas o erróneas, por lo que hubo que sostener más entrevistas con los expertos para introducir modificaciones, y ello se hizo de una forma sencilla. Finalmente se obtuvo un sistema que podía diagnosticar correctamente más del 80% de las placas en fallo.

Técnicas de aprendizaje

La artesanal y la inductiva no son más que dos técnicas dentro del concepto general de aprendizaje. Uno de los modos más simples de aprender es memorizar por repetición, como se hace con un número de teléfono, lo cual es comparable a un ordenador "alambrado" para realizar una determinada tarea.

El aprendizaje por lecciones comunicadas, en el que alguien con experiencia en un campo específico instruye a un principiante, necesita más habilidad y es equivalente a la preparación artesanal de reglas en un sistema experto.

El aprendizaje por ejemplos se utiliza frecuentemente y equivale a la adquisición automática de conocimiento en la forma de inducción.

El aprendizaje por analogía utiliza la solución de un problema como punto de partida para resolver otros similares. Esto es aplicable tanto a la resolución de problemas humanos como a los sistemas expertos.

El aprendizaje por descubrimiento tiene actualmente una aplicación muy limitada a los sistemas expertos, salvo en ciertas áreas especializadas.

Probablemente la inducción estará siempre limitada a formas de conocimiento donde no se requiera una completa comprensión del problema de fondo, y donde puedan obtenerse resultados examinando las características del problema. Un sistema que aprenda por analogía o descubrimiento necesita conocer mucho más la naturaleza del problema. Por ello estas técnicas son mucho más difíciles de poner en práctica, pero en último término permitirán el aplicar métodos de aprendizaje automático a una gama de problemas muchísimo más extensa.

Conclusiones

A medida que aumenta la complejidad de las bases de conocimientos también crece la importancia del problema de la adquisición. Las bases de conocimientos actuales contienen solamente cientos, o como mucho, miles de reglas, y se espera que esta cantidad aumente bastante en unos pocos años. La base de conocimientos tomada como objetivo para la quinta generación de computadores contendrá más de 20.000 reglas y 100 millones de datos. Esto se aproxima a la cantidad total de conocimiento que posee un ser humano, y seguramente no podrá construirse por medios manuales. Por ello es esencial un medio automático de adquisición de conocimiento, y la inducción constituye una potente solución, al menos para ciertos tipos de problema.

La integración del método artesanal y el inductivo abre un camino que permitirá inducir algunas reglas a partir de ejemplos en el caso de que éstos existan, y hacer otras reglas artesanalmente, ayudados por herramientas como la teoría de construcción personal; las reglas obtenidas por los dos métodos pueden luego integrarse. El experto podrá examinar y modificar las reglas, o producir ejemplos todavía no cubiertos por ellas, y utilizará el método inductivo para generar reglas modificadas cuando sea preciso.

Al fructificar los trabajos de investigación sobre analogía y descubrimiento, estas técnicas podrán también integrarse. Sólo con este tipo de sistemas integrados de adquisición de conocimiento será posible construir las bases de conocimientos del futuro.

Referencias

1 H. Schelfhout: Ingeniería de aplicación para circuitos del Sistema 12: *Comunicaciones Eléctricas*, 1986, volumen 60, nº 2, págs. 135–140 (en este número)
2 R. Gunhold y J. Zettel: Pruebas en fábrica del Sistema 12: *Comunicaciones Eléctricas*, volumen 60, nº 2, págs. 128–134 (en este número)

Mark Newstead nació en Inglaterra en 1961. En 1984 se graduó BSc en informática en Hatfield Polytechnic, entrando luego en el grupo de sistemas basados en conocimiento del Engineering Support Centre de ITTE en Harlow. Como miembro del grupo de desarrollo ha trabajado en el diseño y elaboración del conjunto de herramientas ESSAI para sistemas expertos. El Sr. Newstead, especialista en aprendizaje automático, trabaja actualmente en el diseño de herramientas de inducción para utilizar con el ESSAI.

Rodney Pettipher se graduó BEng en ingeniería eléctrica en la Universidad de Liverpol, en 1967. En el mismo año comenzó a trabajar en STC, en el prototipo de la central TXE4. Después pasó algún tiempo en CGCT París trabajando en la programación de las centrales METACONTA* 11A. El Sr. Pettipher ingresó en ITTE ESC en 1980, donde hoy trabaja en el grupo de sistemas basados en conocimientos, principalmente en aplicaciones de ingeniería del conocimiento para sistemas de telecomunicación.

* Marca registrada del Sistema ITT

Integración de múltiples esquemas de control

Los sistemas expertos son un poderoso instrumento para resolver problemas complejos, pero en general el ingeniero del conocimiento tiene que construir un esquema de control específico para cada problema nuevo. Se ha desarrollo ahora un conjunto integrado de esquemas de control que faculta al ingeniero responsable para elegir el más adecuado al dominio en que esté trabajando.

G. Jones
R. Nuttall
K. Stone
ITT Europe Engineering Support Centre, Harlow, Inglaterra

Introducción

Al iniciarse esta tecnología, el diseñador de sistemas expertos, o ingeniero del conocimiento, pretendía demostrar que dichos sistemas podían resolver problemas no banales. Así, se desarrollaron sistemas para muy diversos fines – como el diagnóstico de infecciones bacterianas, la configuración de sistemas de ordenador o la evaluación de recursos minerales –, cuyos esquemas de control sin embargo (es decir, la estrategia para aplicar las reglas de la base de conocimientos) eran altamente especializados y muy vinculados al dominio del problema, perdiendo las ventajas de separar el conocimiento del control. Hoy existe un interés creciente en las aplicaciones comerciales de los sistemas expertos, y por ello el ingeniero del conocimiento debe construir un sistema capaz de operar eficazmente en una gran variedad de dominios. El uso de un esquema de control artesanal para cada problema es inflexible y consume el tiempo de un ingeniero muy capacitado, que es un recurso escaso.

El diseño de sistemas expertos se puede ver como un ejercicio de identificación de aquellas características del dominio que permiten al sistema optimizar la búsqueda de una solución en todo el ámbito del problema. Para ello hay que adecuar la estrategia de control a la estructura de los datos del dominio encerrados en la base de conocimientos. Debe, pues, suministrarse al ingeniero del conocimiento un completo y versátil conjunto de esquemas de control independientes del dominio. Al separar el esquema de control de la base de conocimientos pueden añadirse nuevas reglas sin alterar el control, y utilizar un mismo esquema para resolver problemas en dominios diversos.

Resolución de problemas mediante reglas

Los sistemas expertos son un poderoso medio para resolver problemas de muy diversa índole. Merced a esquemas de control adecuados se pueden tratar de modo automático problemas complejos para los que no se conocen métodos algorítmicos, o éstos resultan imposibles de calcular. Además esta técnica permite remodelar problemas más sencillos para hacer frente a cambios frecuentes de los conocimientos.

Esquemas de control para problemas no-deterministas

Considérese la construcción de un muro con piedras sin argamasa. Sin duda, un albañil experto no sería capaz de describir el orden apropiado de selección y colocación de las piedras. A medida que toma forma el muro, el albañil selecciona una estrategia inmediata para ir encajando en los huecos las piedras que le quedan, y esto le puede llevar a desmontar parte de lo ya construido: el llamado retroceso ("backtracking") en los esquemas de control. Aun partiendo de conjuntos de piedras idénticos, es muy raro que dos paredes sean iguales. Para que un esquema de control imite este método de resolución, ha de seleccionar y aplicar las reglas adecuadas al estado actual del problema y mostrar un cierto no-determinismo en la forma de acceder a una solución.

La resolución de problemas puede considerarse como el proceso de llegar a un estado final u objetivo, pasando a través de estados intermedios. Las transiciones entre estados se deben a la aplicación de las reglas, concretamente a las acciones que se indican en su lado derecho, desen-

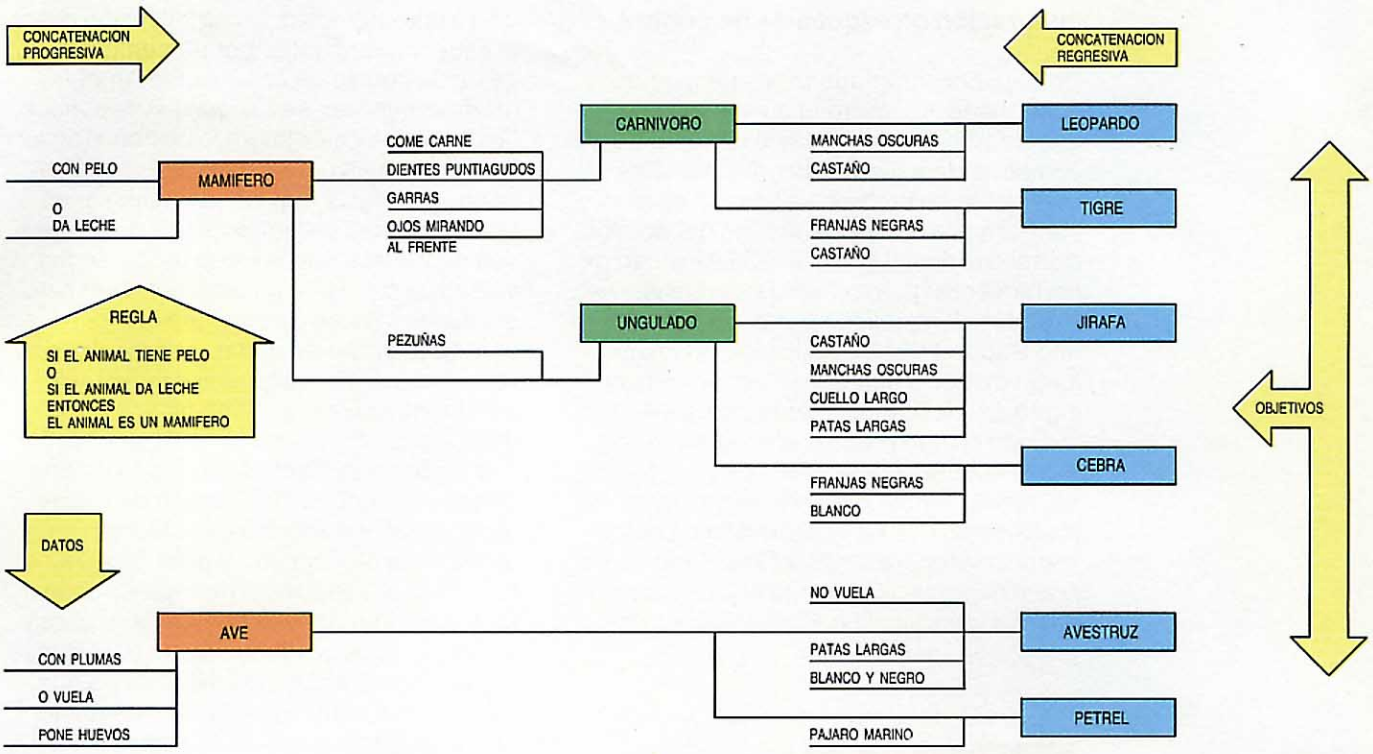


Figura 1
Ejemplo de la red de inferencia de un problema.

cadenadas al cumplir un estado determinado del problema las condiciones señaladas en el lado izquierdo. Los estados de un problema se pueden ver como nodos de una red de inferencia, y las transiciones de estado como las uniones entre nodos. Dicha red servirá para ilustrar lo que puede hacer un conjunto de reglas, y cómo las acciones de una regla sirven de condiciones para la regla siguiente. La figura 1 presenta una red de inferencia de un conjunto de reglas correspondientes al sencillo problema de clasificar un animal conocidas sus características. La manera de abrirse camino un sistema experto en esta red de estados se puede caracterizar por el mecanismo de búsqueda.

Se han utilizado con éxito numerosos tipos de búsqueda en la resolución de problemas, clasificados por la dirección en que van explorando a través de la red de inferencia. La concatenación progresiva consiste en encontrar un camino desde un estado inicial hasta un estado objetivo, como sucede en el problema de la clasificación de animales por red de inferencia (Fig. 1). La concatenación regresiva implica hallar un camino en sentido inverso, y ésta será la dirección de búsqueda utilizada en diagnosis; por ejemplo, una hipótesis sobre la naturaleza de un fallo puede probarse determinando si concuerda con los síntomas detectados.

Además del sentido en que se realiza la búsqueda, una red de estados puede ser examinada primero *en profundidad* y después *en anchura*, o viceversa. Una búsqueda prioritaria en profundidad no selecciona la siguiente rama de la red de inferencia hasta no haber analizado hasta el final la rama que está considerando, a partir de un cierto estado. Por el contrario, una búsqueda prioritaria en anchura (transversal) examina todos los posibles estados a un mismo nivel antes de pasar al nivel siguiente.

Esquemas de control para problemas deterministas

Cuando haya un método algorítmico para resolver un problema y sea de fácil descripción, la aplicación de reglas a los estados de ese problema deberá capturar dicho algoritmo. En consecuencia, el ingeniero del conocimiento tendrá que ser capaz de manipular el esquema de control en cuanto a la aplicación de reglas individuales. En el caso de problemas complejos, su partición en subproblemas suele ayudar a la resolución total.

Por ejemplo, el proceso de construir el muro antes mencionado podría dividirse en el acopio de las piedras adecuadas y la construcción en sí. El orden de estas tareas es obvio, y sería incongruente llamar a esto mecanismo de búsqueda. Un esquema de control adicional podría establecer explícitamente el orden de ejecución de las operaciones sobre el conjunto de hechos, ya sea invocando subproblemas o mediante la aplicación de reglas.

Integración de esquemas de control

Un conjunto integrado de esquemas de control que incorpore una variedad de estrategias, deterministas o no, será un componente esencial de un sistema de resolución de problemas basado en el conocimiento e independiente del dominio de aplicación, tal como el ESSAI, juego de herramientas para construir sistemas expertos desarrollado por el ITT Engineering Support Centre en Harlow[1]. Con un juego de herramientas de esta índole, el ingeniero del conocimiento podrá influir en el modo en que procesa el sistema experto.

Los esquemas de control para el proceso del conocimiento deben complementar los mecanismos de representación de dicho conocimiento. Además, el ingeniero ha de poder expresar el conocimiento del dominio con independencia del conocimiento del control. Normalmente es posible la partición en subproblemas que lleven consigo los relevantes conocimientos de resolución de problemas. En el ESSAI, estas particiones se denominan grupos de reglas. El grupo de reglas, constituido por un conjunto de reglas con nombre, es la unidad para expresar las operaciones de control.

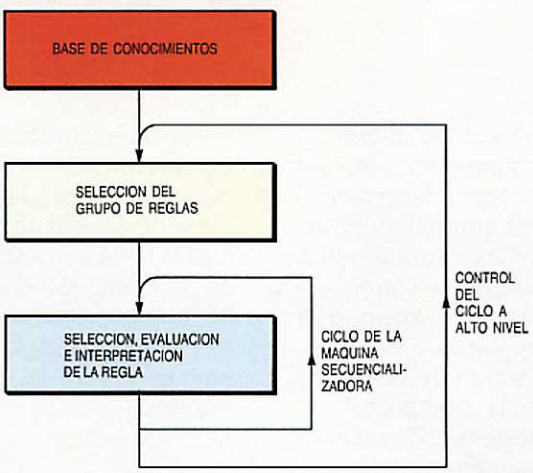


Figura 2
Proceso del conocimiento.

La estrategia de diseño de los esquemas de control en ESSAI hace explícitas todas las opciones que puede ejercitar el sistema en el proceso del conocimiento (Fig. 2). De esta forma se deja al ingeniero del conocimiento facultad de elección que prevalece sobre los supuestos del sistema. Sin embargo, para que tal elección no sea obligatoria se han previsto acciones sustitutivas en el sistema.

Los esquemas de control contienen la lógica para la selección de grupos de reglas y de reglas, así como su posterior evaluación e interpretación. Dicha selección cubre el espectro de control por procedimientos, desde sistemas de control enteramente no-deterministas en los que no se especifica nada acerca de interacción entre reglas, hasta sistemas puramente deterministas donde las reglas han de interaccionar de una sola forma predeterminada. Los referidos esquemas pueden especificarse por el ingeniero mediante parámetros, bien para el total de la base de conocimientos, o bien para cada grupo de reglas individuales. Se reconoce así que para resolver un problema determinado haya que combinar varias estrategias.

El sistema de control especifica ambos comportamientos, el del grupo de reglas (esto es, el mecanismo de selección de reglas para el proceso), y el de cada regla individual. La selección de reglas puede representarse por una sentencia explícita sobre el orden de ejecución de las mismas, o bien especificando los parámetros que caracterizan el mecanismo de búsqueda. El primer método simplifica la expresión del control determinista, mientras que el segundo expresa mejor el control no-determinista.

Esquemas de control determinista

El control determinista ofrece un nivel de control por procedimientos sobre reglas y grupos de reglas, siendo además óptimo para ejecutar operaciones de entrada y salida. En ESSAI, esas operaciones se logran pasando mensajes a los objetos, lo cual aporta también un mecanismo para herencia jerárquica. Un objeto posee un conocimiento basado en procedimientos (métodos) que describen cómo puede accederse a sus variables y que comprenden una o más reglas ejecutadas en el mismo orden en que aparecen en tales métodos.

La comunicación con los objetos supone el envío de un mensaje a uno de sus métodos, el cual puede a su vez enviar otros mensajes, bien hacia sí mismo o hacia otros objetos. Es posible utilizar métodos en la base de conocimientos para la ordenación explícita de las reglas contenidas dentro de cualquier grupo de reglas, en vez de hacer una especificación no-determinista de tal secuencia.

Una clase especial, Meta-clase, define la secuencia de las operaciones de alto nivel durante el proceso del conocimiento. Cuando se ha cargado la base de conocimientos, el sistema busca un método llamado *Inicial* en la Meta-clase, y si existe tal

método se le envía un mensaje que provoque su ejecución. El ingeniero del conocimiento puede haber escrito ese método en la Meta-clase para efectuar diversas operaciones de inicialización en el sistema, tales como introducir variables y secuencializar grupos de reglas. A falta de dicho método, o de toda la clase, los grupos de reglas se ejecutan siguiendo un control no-determinista.

En el ESSAI, los grupos de reglas se consideran objetos compuestos de reglas que a su vez son objetos. Tanto las reglas como los grupos de reglas se pueden secuencializar de modo determinista o no-determinista. La ejecución determinista de las reglas implica el envío de mensajes a las mismas en la secuencia deseada. Estas combinaciones de esquemas de control dan al ingeniero del conocimiento un control adecuado a sus necesidades.

En el siguiente ejemplo se aprecia de qué manera puede el ingeniero especificar un control determinista utilizando la sintaxis que proporciona el sistema:

```
META : CLASS =
    METHOD NEW = {
        [input_data::PROMPT]
        [rulegroup1::DO]
        ([rulegroup2::DO] V
        [rulegroup3::DO])
        ::>
        [results::OUTPUT]
    }
```

Lo anterior se interpreta así:

- En el arranque, el sistema ejecuta el método *PROMPT* asociado con el objeto *input_data*. Esto origina la petición de datos de entrada al usuario, normalmente en forma de pregunta o de menú en el cual seleccionar la respuesta.
- Ejecutar *rulegroup1:* si se cumple la condición de terminación especificada por el ingeniero del conocimiento para este conjunto de reglas, el sistema prosigue con la sentencia siguiente. Esta condición puede expresarse en forma de conclusión que deba alcanzarse, o sea una acción resultante de procesar con éxito las reglas.
- Ejecutar *rulegroup2*; en este caso, si no se satisface la condición de terminación de este conjunto de reglas, el sistema ejecuta *rulegroup3*. El símbolo "V" representa la disyunción lógica "O" y ocasiona la interpretación de *rulegroup2* O *rulegroup3*.
- Ejecutar el método asociado con el objeto *results*, lo que origina la salida de resultados por un periférico determinado, generalmente un terminal o un fichero.

Control no-determinista

El control no determinista proporciona cierto número de estrategias de búsqueda que puede seleccionar el ingeniero del conocimiento para problemas que no tengan solución prescrita o determinista. Tales estrategias, elegidas dando valores apropiados a los parámetros de la base de conocimientos, pueden ser válidas para el conjunto del sistema o bien específicas de cada grupo de reglas. Se adopta una estrategia de búsqueda con valores sustitutivos (en concatenación progresiva y con prioridad transversal) si el ingeniero no fija explícitamente los parámetros.

La forma de elegir las opciones de búsqueda se ilustra con el siguiente ejemplo:

```
FORWARD CHAINED, BREADTH-FIRST RULE GROUP = {
    ! RULES
    ! RULES
    ! RULES
     } [Search_direction = Forward]
       [Search_span = Breadth]
       [Select_askable_vars = Broad effect];
```

El lenguaje utilizado para especificar los parámetros de búsqueda tiene idéntica sintaxis que el lenguaje con el que se especifican otros atributos del sistema, y detalla la heurística que desea aplicar el ingeniero.

Proceso de las reglas

A lo largo de la operación del esquema de control, las reglas de un grupo dado pueden estar dentro de uno de los cuatro conjuntos de reglas que definen el estado del problema así como el progreso en su resolución:

Conjunto de reglas viables: al comienzo del proceso todas las reglas pertenecen a este conjunto, por lo cual son seleccionables por el esquema de control.

Conjunto de reglas activas: lo integran reglas pendientes de evaluación, es decir, en espera de que las condiciones de su lado izquierdo se evalúen en "verdadero" o "falso".

Conjunto de reglas en conflicto: constituido por reglas ya evaluadas en "verdadero" y pendientes de interpretación, o sea esperando a que se ejecuten las acciones de su lado derecho.

Conjunto no viable: reglas que no contribuyen a resolver el problema, bien por haber dado "falso", haber sido ya interpretadas, o porque el esquema de control haya determinado que no deben ser interpretadas.

Las acciones efectuadas sobre las reglas de un conjunto específico serán similares

en cualquier estrategia de búsqueda; la única diferencia entre unas y otras serán las transiciones de una regla entre estos conjuntos. La figura 3 indica la configuración de los conjuntos de reglas para búsqueda de concatenación progresiva y prioridad transversal. Las transiciones se controlan por una máquina de secuencialización de uso general que repasa de modo cíclico los conjuntos de reglas hasta llegar a la condición de terminación. Las actividades de esta máquina dependen del esquema de control elegido. La figura 4 presenta el ciclo en las mismas condiciones de búsqueda antes señaladas para la figura 3.

Las transiciones se producen como consecuencia del proceso individual de las reglas durante las dos primeras etapas, evaluación e interpretación de las reglas.

La evaluación puede describirse mediante el cálculo del grado de veracidad del lado izquierdo de una regla, a base de asignar pesos opcionales a las variables o a las condiciones, enlazando éstas por operadores lógicos y siendo los valores de aquéllas conocidos por el sistema. Las transiciones que podrían resultar son:

- *Verdadero*. Cuando la evaluación completa de una regla da "verdadero", la regla pasa desde el conjunto de reglas activas al de reglas en conflicto.
- *Falso*. Cuando la evaluación completa de una regla da como resultado "falso", la regla pasa del conjunto de reglas activas al de reglas no viables.
- *En suspenso*. Cuando no pueda evaluarse una regla, ésta podría trasladarse desde el conjunto de reglas activas al de reglas viables, o bien permanecer entre las reglas activas dependiendo del tipo de mecanismo de búsqueda escogido. Normalmente no es evaluable una regla cuando está esperando valores para las variables que aparecen en su lado izquierdo.

La interpretación se caracteriza conociendo cómo distribuir el valor de veracidad asociado con el total del lado izquierdo a través de las condiciones y variables del lado derecho, con base en los pesos asignados a las variables de este último lado y en los operadores lógicos que unen las acciones del mismo.

La selección de una regla apropiada para interpretación puede ser conflictiva, al ser posible elegir entre varias; esto se soluciona mediante un mecanismo que selecciona la regla siguiente en una cadena de posibles reglas que se está explorando. El efecto de la resolución de conflictos depende de la estrategia de búsqueda. Si ésta es con prioridad transversal, no hay conflicto pues se seleccionan todas las reglas que puedan valer, mientras que si la prioridad es en profundidad sólo se puede seleccionar una regla y por ello hay que resolver el conflicto. Más aún, en un esquema de control por concatenación progresiva, esto se realiza por selección en

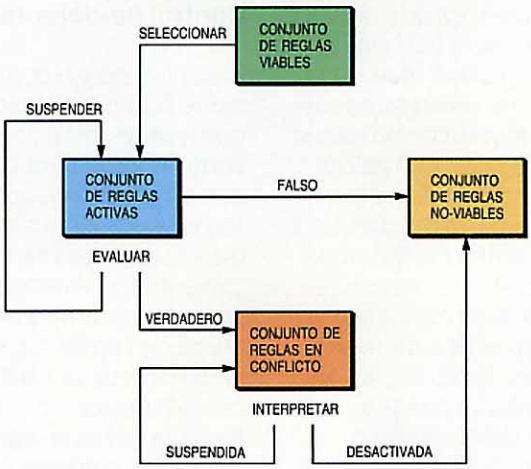


Figura 3
Estados y transiciones de un conjunto de reglas.

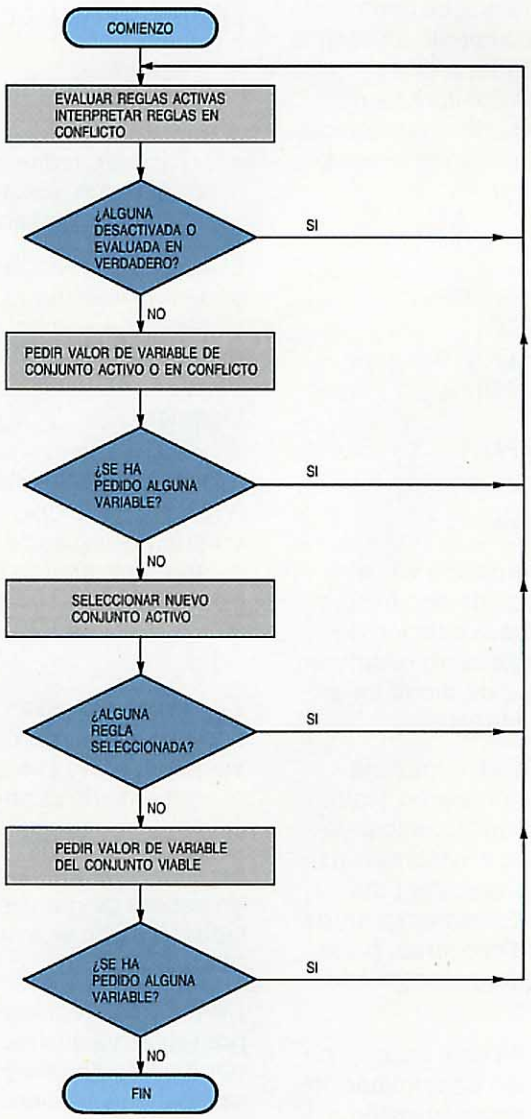


Figura 4
Ciclo del mecanismo de secuencialización para estrategia de búsqueda por concatenación progresiva, prioridad transversal.

el conjunto de reglas en conflicto, mientras que en la concatenación regresiva se escoge entre las reglas activas. La interpretación da como resultado una de las siguientes acciones en cada regla:

- *Desactivar:* cuando una regla ha sido interpretada positivamente pasa desde el conjunto de reglas en conflicto al conjunto de reglas no viables.
- *Suspender:* cuando la regla no se puede interpretar, entonces permanece en el conjunto de reglas en conflicto o pasa al de reglas no viables. Normalmente tal situación se da porque está esperando recibir el valor de una variable que necesita. El destino de las reglas en suspenso depende del mecanismo de búsqueda escogido.

Esencialmente las reglas contienen dos clases de variables: aquéllas cuyos valores pueden deducirse de las acciones realizadas en otras reglas, y las que reciben valores dados por el usuario. Si no puede procesarse una regla porque requiere un valor que suministra el usuario, queda en suspenso. En tal situación, el sistema escoge una variable entre todas las reglas suspendidas y pide al usuario que le dé un valor. Para seleccionar tal variable el ingeniero del conocimiento puede seguir una técnica que utilice el parámetro *select_askable_vars* (como en el ejemplo de la pág. 125).

Las reglas del conjunto de reglas viables que se pueden evaluar (por haber definido al menos una de las variables utilizadas en la regla) se trasladan al conjunto de reglas activas. El número exacto de reglas migradas depende de la opción de resolución de conflictos.

Conclusiones

La integración de una rica gama de esquemas de control dentro del mismo sistema general (independiente del dominio) de resolución de problemas permite al ingeniero del conocimiento elegir el esquema más adecuado para cada dominio. Además de constituir un poderoso marco dentro del cual trabajar para construir sistemas expertos con una amplia variedad de aplicaciones de requisitos de control muy diversos, la integración de múltiples esquemas de control proporciona una útil herramienta de investigación.

La ingeniería del conocimiento es una especialidad relativamente nueva y en rápido desarrollo, y convendría recogerla en un sistema experto. Sin embargo, se conoce bien poco acerca de los procesos que llevan a cabo estos ingenieros, y hay escasos ejemplos en que inspirarse. Por ello, consiguiendo herramientas que permitan a los ingenieros del conocimiento asociar métodos de resolución de problemas con dominios de aplicación, podrán apreciarse con mayor claridad las reglas que rigen su actuación, posibilitando el desarrollo de un sistema automático capaz de resolver problemas.

Referencias

1 J. J. Harvey: ESSAI, juego de herramientas para sistemas expertos: *Comunicaciones Eléctricas*, 1986, volumen 60, nº 2, págs. 109–114 (en este número).

Geraldine Jones se graduó MSc en desarrollo y análisis de programas por la Universidad Herriot-Watt, Edimburgo, en 1983. Ingresó entonces en el ITT Programming Support Centre, Harlow, donde trabajó en un entorno soporte para el desarrollo de la programación del Sistema 12. En abril de 1984 pasó al grupo de sistemas basados en el conocimiento del Engineering Support Centre, en el que ahora se encarga del diseño y desarrollo de los mecanismos de inferencia para el ESSAI.

Richard Nuttall nació en Lancashire, Inglaterra, en 1963. En 1985 obtuvo el BSc en programación por la Universidad de Manchester, entrando seguidamente en el ITTE Engineering Support Centre, Harlow, donde ahora trabaja en el diseño y desarrollo de los mecanismos de proceso del conocimiento para el ESSAI.

Kevin Stone se graduó BSc en informática por la Universidad de Essex en 1982. Desde su graduación trabaja para ITT en diversos proyectos de programación. En la actualidad participa en el diseño y desarrollo de los mecanismos de representación del conocimiento para uso en sistemas expertos.

Pruebas en fábrica del Sistema 12

Los sistemas expertos se usan cada vez más en las pruebas de fábrica para diagnosis de faltas en las placas de circuito impreso del Sistema 12 de ITT. Estos sistemas, basados en conocimientos, se utilizan hoy en día para las placas de procesador, interfaz terminal, enlace digital y línea analógica.

R. Gunhold
Standard Elektrik Lorenz AG, Stuttgart, República Federal de Alemania
J. Zettel
ITT Advanced Manufacturing Technology Center, Bruselas, Bélgica

Introducción

La prueba automática de componentes eléctricos, placas impresas, unidades funcionales y centrales completas es parte integrante del proceso de fabricación de las centrales digitales del Sistema 12 de ITT. Sin embargo, la evolución tecnológica permanente impone nuevas exigencias al personal de pruebas, especialmente en el área de diagnóstico de placas de circuito impreso y unidades funcionales. Además de un conocimiento fundamental de sistemas y circuitos eléctricos, esta actividad de prueba requiere saber aplicar pasos de diagnóstico razonados, a menudo basados en síntomas de fallos incompletos. Las exigencias son grandes, y para las placas más complejas la carencia de personal experto en pruebas limita la producción.

Los sistemas expertos ofrecen ahora una solución, ya que poseen la gran ventaja de almacenar en una base de conocimientos central todos los conocimientos de diagnóstico importantes, de manera que los técnicos menos experimentados puedan aprovecharse de la maestría de los que dominan ese tema. En consecuencia, se ha desarrollado un sistema experto para diagnosis de faltas, basado en la envoltura de sistemas expertos EX-TRAN (Expert-Translator). Los resultados en varias fábricas ITT muestran que el sistema mejora sustancialmente el conjunto de herramientas de diagnóstico disponible.

Sistemas basados en conocimientos

En los últimos años los sistemas expertos, una de las áreas más prometedoras en el campo de la inteligencia artificial, han dado el paso crucial desde el laboratorio a las aplicaciones del mundo real. La característica esencial de estos sistemas es su capacidad para almacenar el conocimiento humano (por ejemplo, reglas de decisión), en bases de conocimientos que pueden contener tanto información de hechos como heurística. Por medio de inferencias, los sistemas expertos son capaces de deducir consejos basados en un determinado conjunto de datos de entrada. El usuario puede preguntar por los hechos que respaldan cierta decisión del sistema, en cualquier momento del diálogo. La comunicación entre usuario y sistema experto, que puede incluir la introducción de nueva información a la base de conocimientos, se efectúa por diálogo hombre-máquina.

EX-TRAN
El EX-TRAN es una envoltura de sistemas expertos desarrollada en la Universidad de Edimburgo, en estrecha colaboración con el ITTE Engineering Support Centre de Harlow y Standard Elektrik Lorenz de Stuttgart. Para poder utilizarlo se le deben introducir los conocimientos específicos de cada aplicación. Se basa en reglas y representa el conocimiento disponible en forma de árbol de decisión, generado de dos principales modos. En uno de ellos el ingeniero del conocimiento introduce informaciones contenidas en ejemplos. Con este enfoque es necesario definir todos los atributos y clases (decisiones) relevantes para un problema dado y luego asignar valores reales a cada ejemplo. El EX-TRAN podrá generar entonces, por medio de su algoritmo de inducción, el árbol de decisión que representa el conocimiento contenido en los ejemplos. En el otro enfoque, se genera el árbol de decisión introduciéndolo directamente mediante una sintaxis fácil de utilizar y de comprender.

Los dos métodos encuentran aplicación en las pruebas de fábrica, dependiendo su elección de la forma en que se disponga del conocimiento requerido. En ambos, el árbol de decisión resultante se traduce a código Fortran 77, que está preparado para el uso después de la compilación y el montaje. Durante la operación el árbol de decisión es ejecutado paso a paso. Toda la información textual pertinente, tal como preguntas, instrucciones y explicaciones, está contenida en un fichero separado, llamado archivo de texto del problema. En cualquier momento puede llamarse a las subrutinas externas (p. ej., realizar cálculos numéricos).

La figura 1 muestra un diagrama de bloques del EX-TRAN.

Selección del proyecto

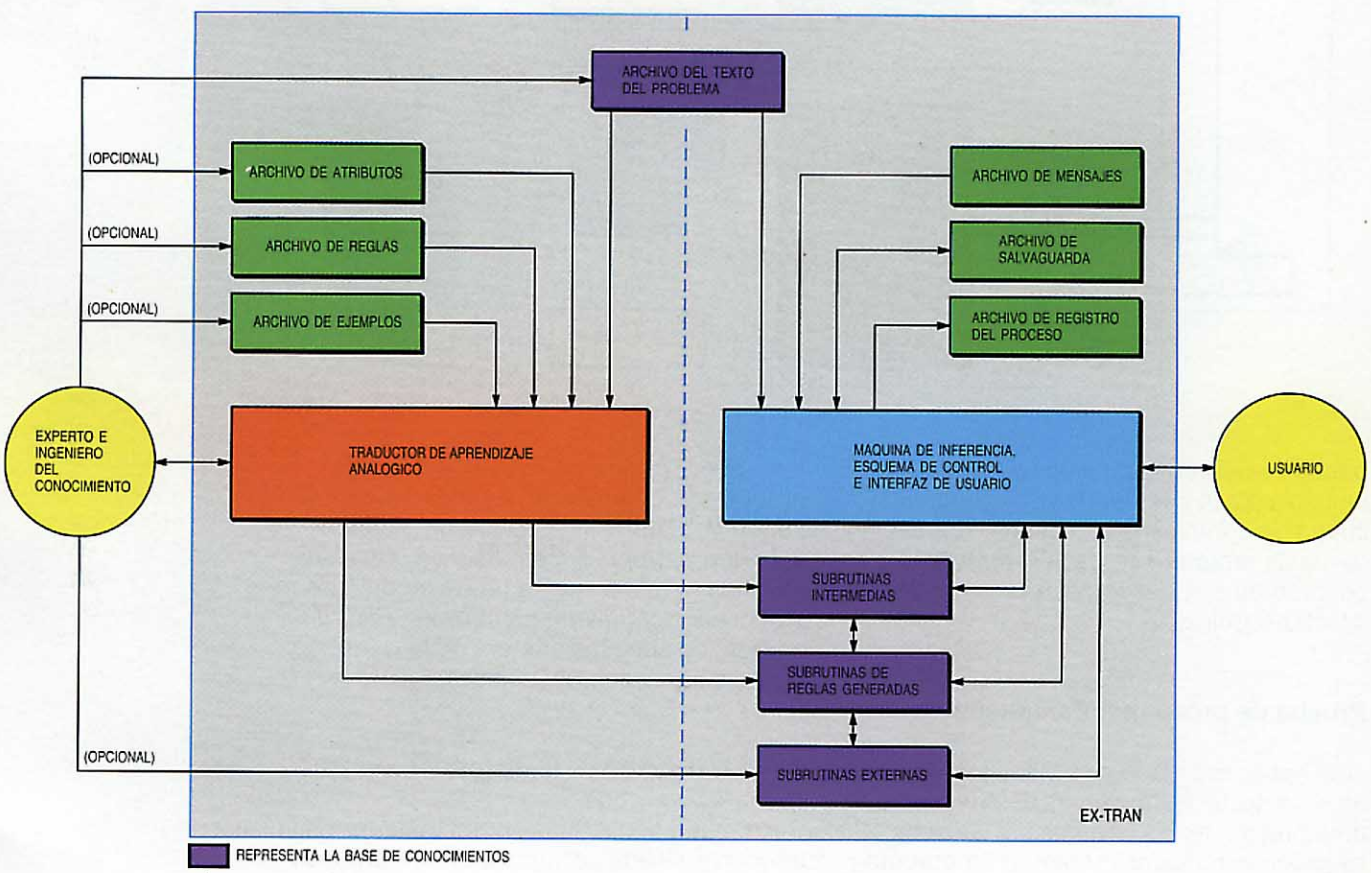
Durante la planificación de la producción del Sistema 12, fueron definidas las diferentes etapas de prueba y evaluados sus costes. Cada centro de diseño fue y sigue siendo responsable de generar y distribuir programas de prueba a los participantes. Una vez preparados tales programas, el siguiente requisito fue apoyar las actividades a pie de obra, como son los largos procesos de diagnóstico en el entorno de pruebas en fábrica. Inicialmente, se había considerado la técnica clásica de simular la función y comportamiento de la unidad, pero pronto se constató que la simulación de las principales placas difícilmente proporcionaría una diagnosis completa y precisa, y ello por dos razones:

- con las técnicas de simulación existentes era virtualmente imposible crear modelos de los complejos circuitos LSI diseñados a medida
- sólo unos pocos expertos comprenden plenamente el comportamiento funcional de sistemas de tal complejidad.

No obstante, había que mejorar la diagnosis de pruebas de fabricación, y como las técnicas de simulación están limitadas a soluciones algorítmicas, se necesitaba un nuevo enfoque. La capacidad de los sistemas expertos para captar el conocimiento de especialistas y ponerlo a disposición de cualquier usuario del sistema, se adecuaba al problema de modo ideal. Por consiguiente, se inició un proyecto piloto en cuyo marco se consideraron diversas posibilidades de aplicación de un sistema experto a las pruebas.

La primera aplicación examinada fue la de servir de soporte a las pruebas en-circuito, que básicamente denuncian las faltas típicas de fabricación, tales como componentes mal colocados o erróneos. Se

Figura 1
Diagrama de bloques mostrando la estructura de la envoltura de sistemas expertos EX-TRAN.



concluyó que esta aplicación no era bastante ambiciosa y podría resolverse mejor mediante herramientas convencionales.

La segunda fue la prueba de unidades funcionales, que implica llegar a probar todo un bastidor del Sistema 12. Sin embargo, se creyó que esta tarea era demasiado compleja para la primera aplicación de un sistema experto.

Finalmente se pensó en la prueba de procesador ampliada (EPT, *extended processor testing*); ésta es una prueba funcional a nivel de placa, y se convino en que podría ser una primera aplicación oportuna pues el esfuerzo estaría concentrado en una sola placa que aportaría el grado adecuado de complejidad funcional. Como punto de arranque se decidió apoyar el proceso de diagnóstico para la placa de interfaz terminal.

Prueba de procesador ampliada

Tras haber superado con éxito las pruebas en-circuito, todas las placas de procesador, memoria e interfaz terminal del Sistema 12 se prueban funcionalmente en un entorno sumamente parecido al del sistema. Durante esta prueba la tensión de alimentación, la frecuencia de reloj y la temperatura ambiente se ajustan automáticamente a un cierto número de estados diferentes, incrementando de este modo la probabilidad de detectar faltas marginales. Durante la EPT, la placa de procesador, la placa de interfaz terminal y la placa de memoria constituyen la unidad (conjunto) que se prueba. En la estación EPT (Fig. 2) se distinguen dos partes principales. Una es la estación GO/NOGO, capaz de probar en tiempo real hasta 10 módulos de procesador a la vez, en un tiempo máximo cercano a 60 minutos; una vez comenzado, el proceso de prueba se desarrolla automáticamente. La segunda parte es la estación de diagnóstico, que permite la diagnosis de una placa afectada por falta dentro de un módulo de procesador completo. La estación EPT proporciona un gran número de herramientas de diagnóstico específicas.

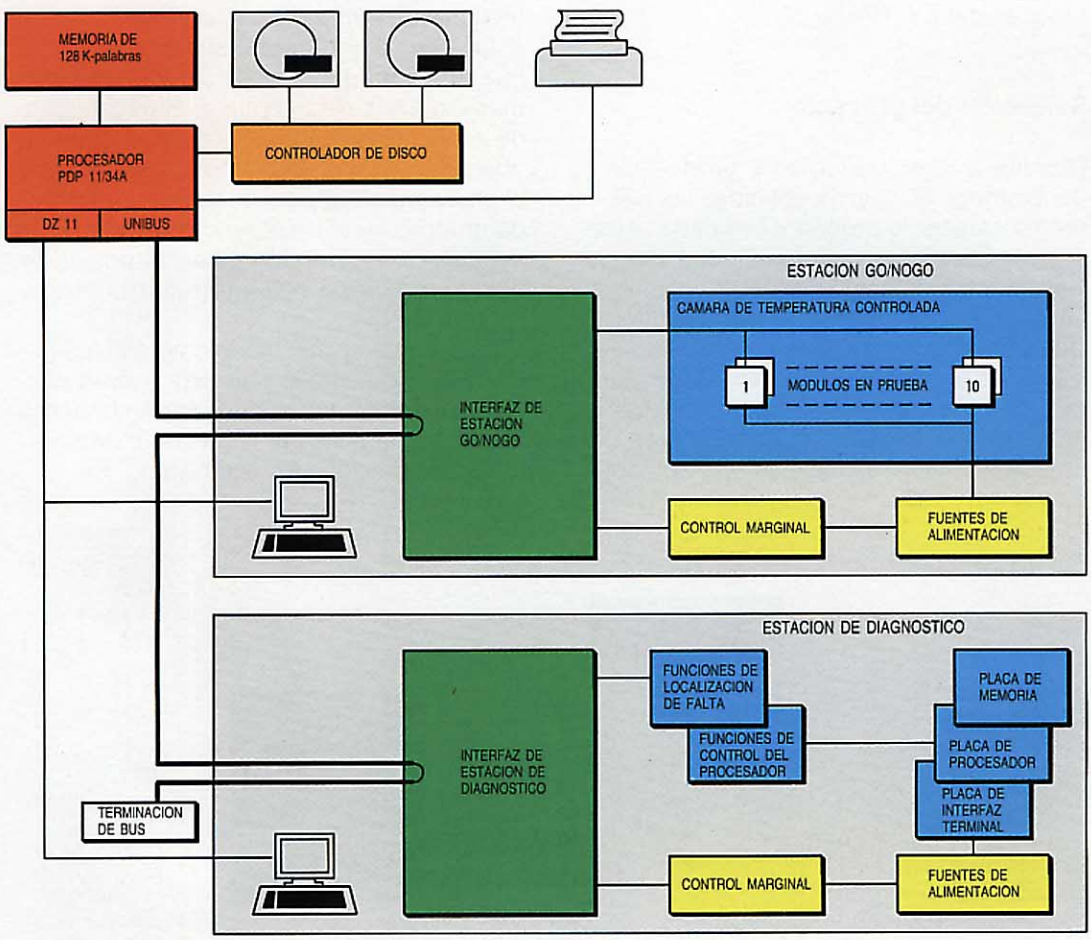


Figura 2
Diagrama de bloques de la estación de prueba de procesador ampliada.

Caracterización de la diagnosis por EPT

Cada etapa de fabricación del Sistema 12 queda verificada por un surtido de pruebas

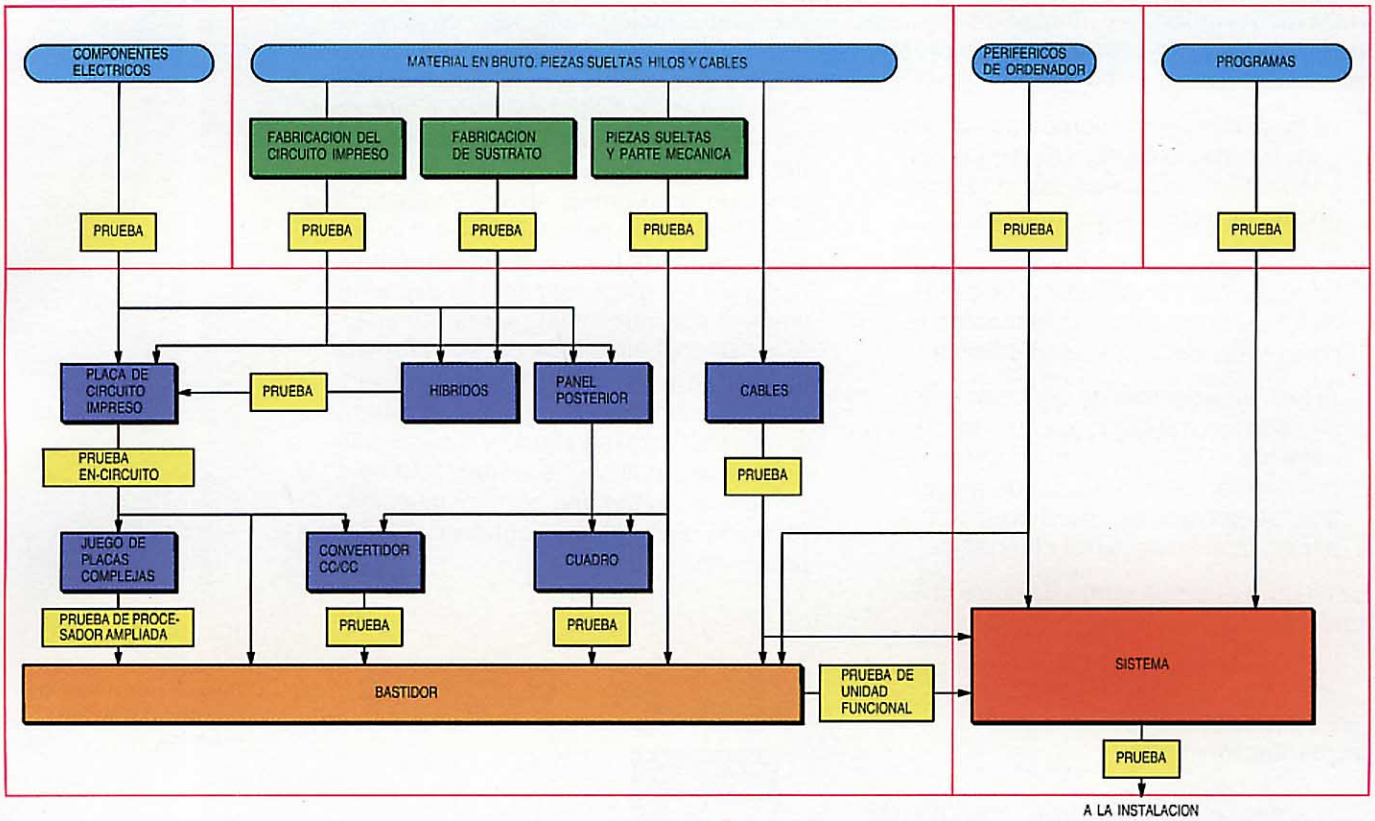
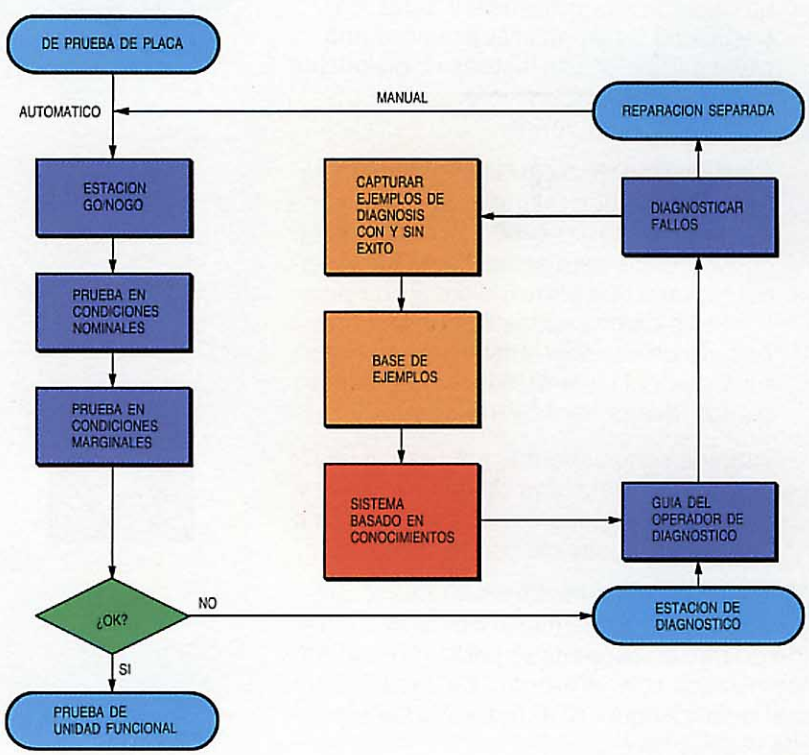


Figura 3
Etapas de prueba en fabricación.

de diagnóstico automáticas (Fig. 3) que difieren sustancialmente en su naturaleza, extensión y complejidad. Por ejemplo, en la mayoría de los casos un mensaje de falta durante una prueba en-circuito indica directamente el defecto, mientras que un mensaje durante la prueba funcional sólo informa sobre las funciones que han fallado y los resultados de la prueba que guardan relación con ellas. En consecuencia, hay un gran número de defectos posibles por cada mensaje de falta. Corresponde al ingeniero probador desarrollar una estrategia de diagnóstico adecuada a partir de un conjunto dado de síntomas, debiendo considerar todos los defectos y métodos de medida posibles para determinar la secuencia de verificaciones conveniente. En otras palabras, el diagnosticador desarrolla primero una hipótesis y después intenta comprobarla haciendo medidas sobre la placa. Este procedimiento se reitera hasta quedar determinado el defecto real.

Además de un profundo conocimiento de los circuitos, sistemas y técnicas de medición, un buen diagnosticador requiere formación, experiencia y "olfato" para localizar los defectos. No obstante, la complejidad de algunas placas combinada con el gran número de posibles causas de defecto hace que el diagnosticador no siempre posea el conocimiento requerido para un problema específico. En tales casos, para poder comenzar la diagnosis hay que obtener información, pero las dos formas usuales de conseguirla, leer la documentación de la placa o preguntar directamente al diseñador, consumen mucho tiempo y son costosas. Los sistemas expertos remedian, en su mayor parte,

Figura 4
Diagnosis de la prueba de procesador ampliada utilizando un sistema experto.



estos inconvenientes y ofrecen las siguientes ventajas sobre los métodos de prueba convencionales:

- La base de conocimientos central asegura constante disponibilidad de los conocimientos necesarios, permitiendo diagnosticar placas en cualquier momento.
- La capacidad de explicación faculta incluso a los menos experimentados para tareas de diagnóstico difíciles.
- El fácil intercambio de conocimientos significa que no hay que partir de cero en cada casa ITT para alcanzar todos los conocimientos esenciales de diagnóstico, pues éstos se transfieren y comparten fácilmente entre compañías.

La figura 4 muestra cómo discurre la EPT con un sistema experto.

Implantación

Para asegurar el éxito de la introducción del nuevo sistema en fábrica, la secuencia de diagnosis (Fig. 5) – que incluye el sistema experto – se ha desarrollado en colaboración con el personal de pruebas que va a utilizarla en los talleres. Todos los datos de diagnóstico pertinentes se registran individualmente junto al número de serie propio de cada placa, y se clasifican en las siguientes categorías:

- Progreso de diagnosis, llegando a la diagnosis final: describe todos los pasos de diagnóstico, diagnosis y todas las sustituciones ya realizadas sobre una placa particular. Las historias individuales de las placas son una información útil para encontrar averías.
- Clasificación de diagnosis: este dato se necesita por dos razones, ayudar a clasificar los casos según su necesidad o conveniencia para actualizar la base de datos, y por otra parte indicar el comportamiento de una aplicación de sistema basado en conocimientos (faltas encontradas por el sistema, faltas no contempladas, diagnósticos erróneos).
- Nuevos conocimientos sobre diagnosis, que son introducidos por los expertos y utilizados después por el ingeniero para actualizar la base de conocimientos.

Se aseguró la aceptación del usuario proporcionando una gama suficiente de modos de operación específicos para la aplicación, combinada con un interfaz de usuario bien definido. La figura 6 muestra una secuencia de diálogo típica.

La constitución de la base de conocimientos se realiza en dos etapas. En la etapa 1 se implanta el conocimiento básico, que incluye los pasos de diagnóstico para las faltas más probables. Estas faltas se determinan analizando el alcance y profundidad de las pruebas de las etapas anteriores. El ingeniero del conocimiento interroga al diseñador de placas, al ingeniero probador y a los diagnosticadores experimentados, e introduce posteriormente en la base de conocimientos del EX-TRAN la información obtenida. En la etapa 2, el nuevo conocimiento conseguido durante la diagnosis real se añade a la base existente, o la modifica. Esta operación se efectúa mediante una herramienta para recogida de datos de diagnosis, diseñada a propósito.

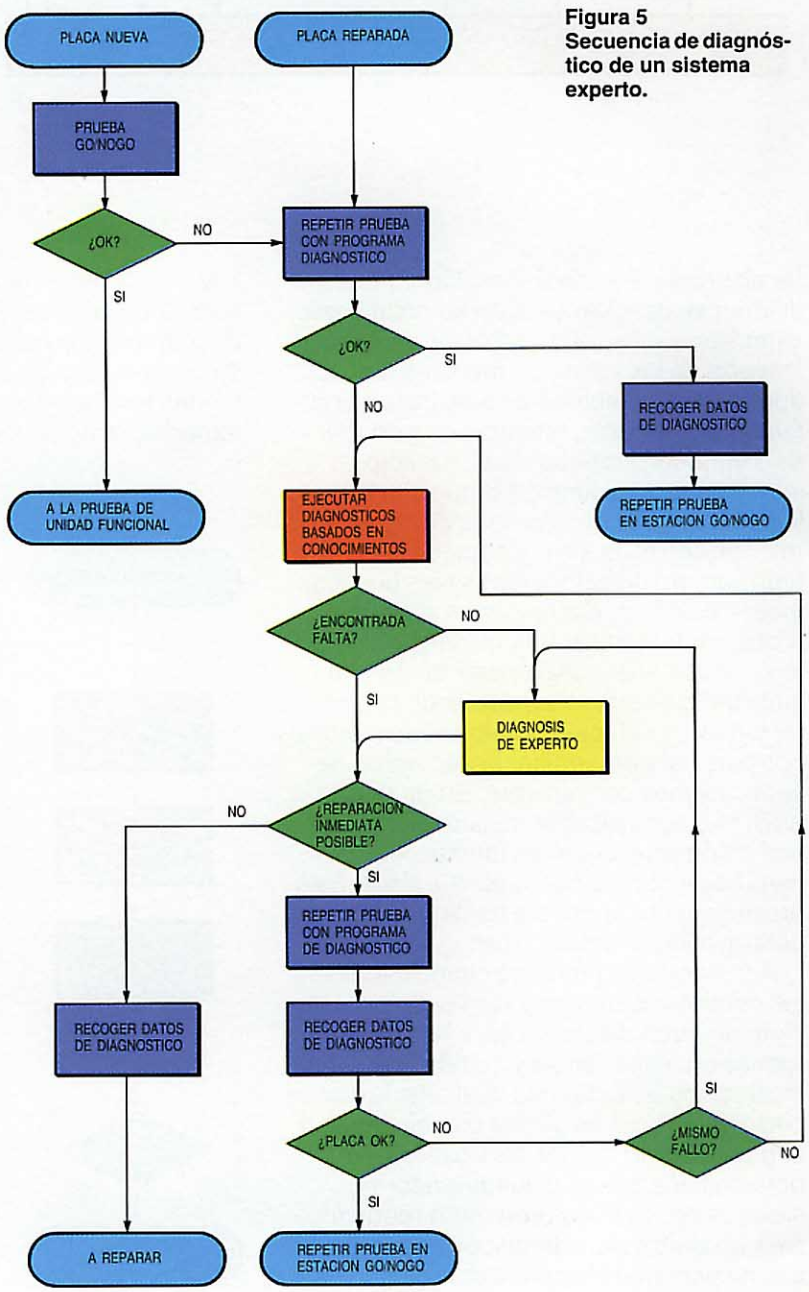


Figura 5
Secuencia de diagnóstico de un sistema experto.

Economía

El rendimiento económico global del EXTRAN no es fácil de evaluar porque ofrece varias ventajas de difícil cuantificación:

- mejora de la flexibilidad dentro del área de pruebas
- intrínseca educación de los usuarios
- base de conocimientos disponible permanentemente
- tiempo de ejecución reducido
- fácil distribución del conocimiento.

Por esta razón sólo se consideran aquí los tiempos de diagnosis, que pueden clasificarse como sigue:

- t_{DK}, tiempo de diagnosis por falta utilizando un sistema experto
- t_{DKM}, tiempo de diagnosis por falta sin utilizar un sistema experto (pero sí el conocimiento heurístico que el técnico ha adquirido utilizando el sistema experto)
- t_{DM}, tiempo de diagnosis por falta con los procedimientos clásicos y toda la documentación necesaria.

Sin ayuda de sistema experto se obtiene un tiempo de diagnosis dado por:

$$T_{DT} = n\,c\,t_{DKM} + n(1-c)\,t_{DM}$$

y con ayuda del sistema basado en conocimientos:

$$T_{Dt} = n\,c\,t_{DK} + n(1-c)\,t_{DM}$$

donde:

- n – número de placas a diagnosticar
- p – volumen de producción
- f – tasa de fallo
- c – proporción de faltas cubiertas por el sistema experto
- x – periodo considerado
- t_{KBi} – tiempo de implantación de la base de conocimientos
- t_{KBm} – tiempo de mantenimiento de la base de conocimientos.

El ahorro en el tiempo de diagnóstico puede ahora ser comparado con el tiempo necesario para la implantación y mantenimiento:

$$p = \frac{t_{KBi} + x\,t_{KBm}}{c\,t_{DKM} - t_{DK}} \cdot \frac{1}{f}$$

$$c\,t_{DKM} - t_{DK} = \frac{t_{KBi} + x\,t_{KBm}}{p\,f}$$

Mediante estas ecuaciones puede obtenerse el volumen de producción necesario para la amortización del sistema, ahorrando además tiempo.

En la aplicación que nos ocupa, los tiempos de diagnosis se redujeron de un promedio de 90 minutos a menos de 40 minutos, y el sistema consiguió cubrir el 80% de las faltas.

Experiencia durante la introducción

Cuando los ingenieros probadores oyeron hablar por primera vez de la idea de introducir un paquete de programación para ayudar al diagnóstico, no se inmutaron. En general, supusieron que se trataba de un ejercicio de simulación más con todas las dificultades e inconvenientes asociados con la modelación de complejos circuitos LSI. Al aclararles que la programación podría representar sus conocimientos, dejaron de tomar en serio el proyecto. No obstante, colaboraron mucho explicando las reglas y dependencias de las diferentes clases de faltas, aunque en algunos casos simplemente no las recordaban, hecho que se repite durante el desarrollo de sistemas expertos.

Una vez obtenidas, preparadas y almacenadas alrededor de 50 reglas, el ordenador podía ya dictar instrucciones específicas de diagnosis mucho más aprisa que los exper-

Figura 6
Ejemplo de diálogo de diagnóstico.

```
SEL EPT Diagnosis Station
*******      *
  *****     ***
    ***   *****
      ********
     ***
   *****                  Information and instructions for every question
*******                   can be displayed by entering ?COM

Please enter PBA serial number: 12345
Any save file or PBA back from repair station (y/n) ? Y
What is the name of the save file (default: 12345.SAV) ?

                    Diagnosis of the Terminal Interface Board

The fault code refers to an error in CRC circuit

Diagnosis of CRC circuit

Change 7D12; is the serial bit pattern '15H' present at 7D12 pin 12 ? ?COM

Change the component at location 7D12. Connect channel 0
of LA with 7E22 pin 2 and channel 1 with 7D12 pin 12.
Adjust the trigger menu to: A+XXXXXXX1.
Timebase: 20 ns. Start LA and CRC macro.
Check the bit pattern '15H' of channel 1
in connection with the 8-bit clock of channel 0.

Change 7D12; is the serial bit pattern '15H' present at 7D12 pin 12 ? Y

   Component at location 7D12 replaced
   Restart of the TERI test program

Please replace unplugged connectors (AA21/AB21 - BA11/BB11)

Please hit return for data collection.
```

tos, quienes comprobaron entonces que el ordenador era capaz de identificar la clase de faltas y sus causas con mucha más rapidez que ellos, en cuanto se le dieran reglas. En este punto, en lugar de aceptar la nueva herramienta, los expertos alegaron que el ordenador les haría la competencia, no pensando que les iba a ahorrar tiempo para acometer nuevas tareas, y por consiguiente se resistieron a dar al ingeniero del conocimiento información adicional, cuestionando incluso la que previamente habían facilitado. En tales circunstancias fue extremadamente difícil convencer a los expertos de que su experiencia seguía siendo necesaria, con lo que el proyecto entró en una fase crítica.

A fin de superar este problema, se introdujo el "aprendizaje por ejemplos" para adquirir y representar el conocimento. Se solicitó a los expertos que identificaran y archivaran sus acciones y consejos, ayudados en lo posible por un soporte informático. Una vez desarrolladas las reglas de inducción, se entregaron éstas a los expertos para un refinamiento adicional. Al transcurrir el tiempo y ascender el número de reglas a varios centenares, el ordenador llegó a responder mucho más aprisa que cualquiera de los expertos. Ello sin embargo sólo ocurría si previamente se habían almacenado la regla y la conclusión que debían ser aplicadas. En esta fase del proyecto, el técnico probador experimentado fue sustituido por una persona en aprendizaje, que fue capaz de diagnosticar casi el 60% de los casos de falta utilizando el conocimiento de expertos representado en el ordenador.

Por fortuna, los expertos volvieron a cambiar de actitud. Comprendieron que todavía se les necesitaba cuando el ordenador fracasaba en un diagnóstico: él no sabía, pero los expertos sí. Entonces se sintieron más necesarios que nunca: en todos los casos que no podía atender el ordenador, nada podía reemplazar al experto humano. De nuevo estaban deseosos de enseñar al ordenador nuevas reglas o de refinar las existentes. Indudablemente, la contribución de los expertos fue un factor vital para el éxito del proyecto.

Conclusiones

La aplicación satisfactoria en varias fábricas ITT de un sistema experto para diagnosis de placas, basado en el EX-TRAN, ha demostrado su eficacia. El sistema puede diagnosticar cerca del 80% de las faltas de placas y ha reducido a menos de la mitad los tiempos de diagnosis. Además, goza de aceptación por los usuarios y su utilización resulta práctica. Por todo ello, se están planificando ahora nuevas aplicaciones, tales como el diseño en función de la posibilidad de prueba, la planificación de pruebas y la programación de fábrica.

El desarrollo de interfaces adecuados para la transferencia bidireccional de datos entre el EX-TRAN y el proceso servido permitirá que los datos de faltas suministrados por el programa de prueba, puedan ser introducidos directamente en el sistema experto como datos de entrada. Las instrucciones de diagnosis resultantes de este sistema podrán utilizarse para ajustar automáticamente los instrumentos de medida (p. ej., un analizador lógico).

Otro campo en desarrollo es el de implantación de sistemas expertos en ordenadores personales, ampliando aún más la gama de posibilidades para las pruebas en fábrica automatizadas.

Rulf Gunhold nació en Bottrop, Alemania, en 1946. Estudió ingeniería eléctrica en la Escuela Técnica Superior de Bochum, donde se graduó MSc en 1969, entrando en Standard Elektrik Lorenz para trabajar en desarrollo de circuitos y equipos. En 1975 fue nombrado jefe de desarrollo de programas en la planta de SEL en Stuttgart, responsable de la introducción de técnicas de programación avanzada para prueba de placas de circuito impreso. En 1983 dirigió la instalación del primer sistema experto de ITT. El Sr. Gunhold es actualmente director de ingeniería de programación y fabricación integrada por ordenador en el departamento de operaciones de SEL.

Joachim Zettel nació en Bayreuth, Alemania Occidental, en 1954. Estudió mecánica de precisión en la Fachhochschule Heilbronn, graduándose en 1978 ingeniero diplomado. Al finalizar sus estudios entró en Standard Elektrik Lorenz para trabajar en desarrollo de programas para pruebas de placas de circuito impreso, pasando a ser miembro del equipo de diseño internacional en el Advanced Technology Center de ITT en EEUU. A su vuelta a SEL en 1981, el Sr. Zettel fue nombrado jefe de desarrollo de programas de prueba de sistemas y unidades funcionales en la planta de Stuttgart y estuvo involucrado en la implantación con éxito de sistemas expertos para diagnóstico de placas. Desde junio de 1986 es director de sistemas de prueba avanzados en el Advanced Manufacturing Technology Center en Bruselas.

Ingeniería de aplicación para circuitos del Sistema 12

La ingeniería de aplicación es un proceso fundado en el conocimiento que transforma los requisitos de una Administración telefónica en una central de configuración adecuada. Se ha desarrollado ahora un sistema experto que aprovecha la experiencia de los ingenieros de aplicaciones para la configuración de centrales.

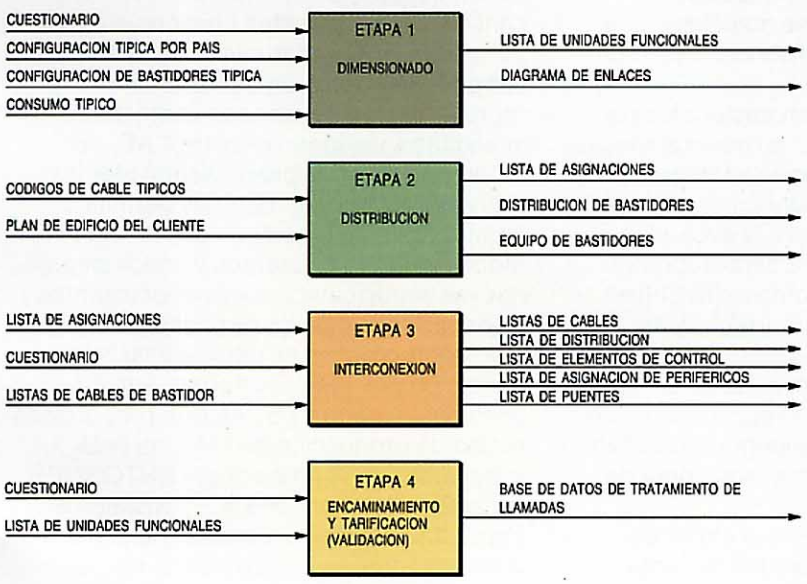
H. Schelfhout
Bell Telephone Manufacturing Company, Amberes, Bélgica

Introducción

El Sistema 12 de conmutación digital de ITT se puede configurar de modo flexible en centrales que cumplan con precisión los requisitos de la Administración, tanto a corto plazo como a lo largo de su vida operativa. En consecuencia, no hay dos centrales exactamente iguales.

La ingeniería de aplicación (CAE) para centrales Sistema 12, transforma los requisitos de una Administración en información detallada del equipo necesario, su distribución e interconexión, sus enlaces con la red telefónica existente, y todos los demás datos necesarios para la instalación de la central (Fig. 1). Una parte de este proceso debe realizarse al ofertar la central, usualmente con datos limitados, y se repetirá el proceso entero cuando se disponga de todos los datos para cada central a instalar.

Figura 1
Proceso CAE.



Datos de entrada CAE

La CAE sólo puede iniciarse cuando la Administración proporcione los datos más significativos, entre ellos el número de líneas de abonado, el tráfico medio por línea y el número de enlaces de entrada y de salida. Tendrán que darse también las condiciones impuestas a las facilidades de abonado, los sistemas de señalización e información de encaminamiento, así como los detalles del edificio y del aire acondicionado.

Los datos se introducen en el proceso CAE mediante un cuestionario. Para los elementos más importantes se presuponen valores que la Administración puede modificar cuando sea oportuno, lo que induce a tener una configuración típica por país con unas cuantas opciones predefinidas para cada clase de central (local, interurbana) de una cierta Administración. De este modo, sin más que conocer la identidad de la configuración normal, se pueden determinar directamente datos que de otra forma habría de suministrar la Administración.

Cuando los diseños existentes no permiten cumplir los requisitos de la Administración, hay que producir otros nuevos mediante el trabajo de ingeniería necesario, llegando quizás a definir nuevas configuraciones típicas.

Una vez elegidas las opciones de diseño, se introduce en el proceso CAE la información relevante, tal como la carga de procesador por llamada.

Definición de proceso CAE

Obtenidos ya todos los datos, puede comenzar la CAE con la tarea de dimensionado (etapa 1, figura 1), que determina el

tamaño inicial o ampliado de una central en cuanto a equipo y funciones basándose en las especificaciones de la Administración, cálculos de tráfico, cálculos de consumo, etc. El resultado de tal labor es una lista de equipo y circuitos funcionales requeridos, pero todavía sin distribuir ni agrupar en bastidores.

La siguiente tarea (etapa 2) determina la posición física de todo el equipo en la central, que comprende la situación de los bastidores, la composición de los mismos y la del repartidor principal.

En la etapa 3 de la referida figura se definen todas las interconexiones físicas y lógicas entre los componentes de la central, incluyendo longitudes y trayectorias de cables.

La última tarea (etapa 4) prepara los valores de datos para el funcionamiento de la central. Todos los datos fijados por la Administración (p. ej., número de dígitos de entrada y salida a la central, tarificación) se traducen a un formato utilizable por la CAE.

Ingeniería de aplicación

La ingeniería de aplicación (CAE) es obra de cualificados ingenieros en telecomunicaciones, expertos en su área y por ello no fáciles de encontrar. El conocimiento de estos ingenieros suele progresar para cubrir la evolución del sistema y las necesidades de las administraciones, sin dejar de mantener sus anteriores conocimientos ya que éstos sirven de base para fabricar las centrales actuales.

El ingeniero de CAE se ayuda de manuales de configuración de centrales en cuyo contexto debe trabajar. Para cada central, sin embargo, hay que modificar y ampliar esta base, añadiendo el apropiado conocimiento según la aplicación y la Administración. Aparecen dificultades cuando las reglas de los manuales se actualizan con menor frecuencia que el conocimiento en sí.

Existen programas convencionales para automatizar la CAE, pero en general resulta difícil su rápida actualización en respuesta a cambios en la base de conocimientos CAE. Estos cambios obedecen a la evolución constante del Sistema 12 para incorporar facilidades avanzadas como la RDSI (red digital de servicios integrados), y a las nuevas exigencias de la Administración. Por todo ello se buscó un método automático que pudiese "captar" el conocimiento de CAE y ofrecerlo a un sistema interactivo, hallando la respuesta en la tecnología de sistemas expertos.

El enfoque de los sistemas expertos tiene varias ventajas sobre las técnicas tradicionales de programación, que requieren codificar y recodificar a menudo el programa para probar o perfeccionar la especificación de la base de conocimientos. La codificación de un programa esencial es tarea delicada, y una pequeña alteración de la base de conocimientos puede obligar a reescribir gran parte del programa o a desecharlo del todo. Especificar una base de conocimientos CAE es demasiado complejo para poderse hacer correctamente a la primera vez: los casos e interacciones a tener en cuenta son muy numerosos, al igual que los cambios de orientación durante los procesos de diseño y realización. Por el contrario, con los sistemas expertos se trata la propia especificación provisional como un programa, necesitando poca o ninguna codificación adicional. Ello permite al diseñador probar y afinar la especificación de la base de conocimientos sin malgastar esfuerzo en versiones intermedias.

Sistemas expertos para CAE

La CAE del Sistema 12 es una aplicación apropiada para un sistema experto, ya que la configuración de una central puede analizarse en dos aspectos: las etapas de formación y los subsistemas que componen la central configurada. Esta facilidad permite también el desarrollo gradual y la rápida realización de prototipos tanto del Sistema 12 como del sistema experto. De ahí que no sea preciso esperar hasta la finalización del diseño para preparar la base de conocimientos, la cual podrá luego servir para evaluar la fabricabilidad de tal diseño y hacer los ajustes oportunos antes de afrontar el costo de fabricación.

El sistema experto desarrollado para la CAE cubre la primera parte del proceso de dimensionado, determinando los tipos y cantidades de unidades funcionales (etapa 1, Fig. 1). Al desarrollar este sistema experto, se puso gran cuidado en que la representación del conocimiento fuese inteligible a los ingenieros de CAE, de modo que en principio pudiesen leer las reglas directamente. Esto les permitiría comprender las funciones de las reglas, modificar las ya existentes y añadir otras nuevas según cambian los conocimientos y se desarrollan nuevos requisitos. Para lograr este objetivo se decidió usar la representación del juego de herramientas para sistemas expertos ESSAI, que proporciona reglas de producción de la forma – SI <*conjunto de condiciones*> ENTONCES <*conjunto de acciones*> –, basadas en lógica de evaluación variable. Se consideró que esta representación podría emular

el modo en que un ingeniero de CAE utiliza sus conocimientos de configuración.

Sea, por ejemplo, la siguiente regla:

Cuando el máximo valor de tráfico medio por tipo de TSU (terminal subscriber unit) es mayor que N1 erlangs, pero menor o igual que N2 erlangs, entonces el número de planos es P.

Utilizando la representación de ESSAI, esto se expresará así:

```
CAE_REGLA_1
[MAXIMO_TRAFICO_MEDIO_TSU > N1]
[MAXIMO_TRAFICO_MEDIO_TSU < = N2]
::>
[NUMERO_DE_PLANOS = P]
```

La sintaxis empleada para esta regla permite su representación natural en el formato de lógica de evaluación variable. Las reglas son puramente declarativas, es decir, no aportan ninguna información de control sobre cómo se debe procesar la regla por el mecanismo de inferencias.

Puede ser condición en las reglas toda expresión susceptible de calificarse verdadera o falsa, por ejemplo, el elemento 1 es mayor que el elemento 2; las acciones evalúan expresiones y asignan las respuestas a elementos de datos (variables). Se puede utilizar disyunción, es decir, conjuntos alternativos de condiciones, cuando haya más de un camino válido para alcanzar la conclusión en una regla.

Las expresiones, ya pertenezcan a condiciones o a acciones, pueden incluir operadores aritméticos normales así como funciones de usuario específicamente codificadas (ej., el cálculo de los circuitos requeridos para un tráfico determinado por la fórmula de Erlang). El ingeniero del conocimiento asigna nemotécnicos a estas funciones de usuario, cuya utilización se aprecia en la siguiente regla que también ilustra el uso de afirmaciones. En tales reglas no existen condiciones (lado izquierdo de la representación), y las acciones del lado derecho se realizan siempre; por ello la palabra clave TRUE (VERDADERO) figura en el lado izquierdo, seguida de las acciones:

El número de MFSR (circuitos emisores-receptores multifrecuencia) necesario para el tráfico requerido está dado por la fórmula de Erlang, utilizando como argumentos el tráfico de MFSR y el grado de servicio requerido.

La declaración completa es una afirmación pues se invoca incondicionalmente (no hay cláusula SI), y la función de usuario es la fórmula de Erlang. Utilizando la representación ESSAI de lógica de evaluación variable, se convierte en:

```
CAE_REGLA_2
TRUE
::>
[NUMERO_DE_MFSR_POR_TRAFICO =
ERLANG(TRAFICO_MFSR,GRADO_DE_SERVICIO_MFSR]
```

Cuando una regla no es autoexplicativa, se puede añadir un texto de comentario no procesable. Las reglas se mantienen mediante un editor de textos.

El ingeniero puede dividir el conocimiento en *grupos de reglas* representativos de una colección lógica de reglas que describan algún subproblema del dominio. Este tipo de división ofrece ciertas ventajas:

- Permite establecer un foco de atención, que favorece a la legibilidad y mantenibilidad de las reglas en el grupo.
- Mejora el rendimiento del sistema al limitarse el proceso del conocimiento a las reglas incluidas en el grupo y no a la base de conocimiento entera.
- Sirve de base para tomar y ejecutar decisiones de control, permitiendo al ingeniero el ajustar las opciones de control con mayor precisión que en la base de conocimientos.
- Alienta un desarrollo progresivo de aplicaciones de sistemas expertos, pues cada subsistema puede ser enteramente desarrollado y puesto en funcionamiento de modo independiente.

El conocimiento para el sistema experto se introdujo a mano, tomado sobre todo de los manuales CAE. Dada la naturaleza prescriptiva de los mismos fue sencillo el convertir el texto en reglas, consultando con expertos en CAE para clarificar, obtener información adicional y resolver cualquier ambigüedad. La base de conocimientos obtenida contiene un grupo de reglas para cada tipo de unidad funcional. Estos grupos de reglas, que contienen hasta 20 reglas, se ejecutan secuencialmente.

La base de conocimientos se divide en dos secciones: en la primera se definen los datos usados dentro de las reglas y en la segunda se definen las reglas en sí.

En los ejemplos expuestos los datos reciben sólo valores numéricos o de carácter. Sin embargo, pueden tomar también la forma de tablas complejas multinivel, susceptibles de proceso en reglas que utilicen cuantificación existencial o universal para acceder a registros individuales de una tabla. La cuantificación universal en una regla hace que cada registro sea seleccionado por orden y sometido a todas las condiciones antes de realizar ninguna acción asociada. Por el contrario, la cuantificación existencial sólo selecciona el registro de la tabla que satisfaga las condiciones

siguientes, también antes de ejecutar cualquier acción relacionada.

El uso de la cuantificación se aclara con un ejemplo que trata de las pruebas de enlaces de salida. En este caso una central se conecta a otras muchas, de forma que sea posible crear una tabla (TABLA_CENTRALES en la figura 2), que dé información sobre todas las centrales a las que está conectada la central en cuestión. Una segunda tabla (TABLA_PRUEBA en figura 3) informa sobre los tiempos y tipos de prueba a realizar en cada central conectada. La cuantificación permite deducir una regla que determine el número de pruebas de una central determinada, examinando el contenido de las tablas. La forma textual de la regla es:

Mira todas las centrales de la tabla-centrales, y para cada una de ellas mira todas las pruebas de la tabla-prueba; si el tipo de prueba es de periféricos y el tipo de central es "central", entonces incrementa el contador de pruebas.

La regla en formato ESSAI se expone en la Tabla 1. Pueden definirse constantes para un diseño dado, si bien sujetas a cambio por modificaciones y mejoras; ejemplo es la carga del procesador por llamada.

El fichero de reglas escrito así se "analiza" con el fin de producir la base de conocimientos para el juego de herramientas ESSAI durante el tiempo de proceso del sistema. Se llama proceso del conocimiento a la tarea realizada por el mecanismo de inferencia en una aplicación operacional de sistema experto. Las reglas son sentencias declarativas del conocimiento del experto y no contienen ninguna información de cómo procesar ese conocimiento. Por lo tanto el mecanismo de inferencia ha de determinar cuál grupo se procesa primero, qué regla se selecciona entre todas las del grupo, qué variable se escoge entre las del lado izquierdo de la regla, cómo se evalúa o determina el valor de veracidad de la regla, y cómo se interpreta esta regla para realizar las acciones del lado derecho. Para que el mecanismo de inferencia sea independiente del conocimiento y de la aplicación se han elaborado métodos generales, incluyendo la concatenación progresiva y la regresiva.

La concatenación progresiva, estrategia gobernada por los datos, es apropiada para las configuraciones, pues aunque los datos de entrada sean finitos hay infinitas configuraciones de central posibles. Esta concatenación funciona del siguiente modo: el mecanismo de inferencia selecciona una regla del grupo de reglas de la base de conocimiento y a su vez una variable de las que forman las condiciones en el lado izquierdo (p. ej., MAXIMO_TRAFICO MEDIO_TSU de la CAE_REGLA_1). Se le pide entonces al usuario el valor de esta variable y de las restantes de la regla. En ese momento el mecanismo de inferencia es capaz de evaluar e interpretar la regla. Un ejemplo de esta acción sería la asignación del valor P a la variable NUMERO_DE PLANOS en la regla CAE_REGLA_1, asignación que típicamente corresponde a una condición en otra regla. Por ello las condiciones son generalmente una mezcla de variables con valores solicitados al usuario y variables con valores resultantes de la acción de otras reglas. Se puede ver cómo las reglas de un grupo de reglas forman una cadena cuyo camino lo marcan los datos que satisfacen las condiciones que figuran en el lado izquierdo.

Como se ha señalado antes, cada subsistema CAE corresponde a un grupo de reglas, y el mecanismo de selección de grupo de reglas procesa un subsistema cada vez. La selección de reglas determina qué datos se pueden deducir de las reglas y cuáles ha de aportar el usuario, bien directamente en el terminal o a partir de un fichero.

Estos mecanismos de proceso de conocimiento son independientes del contenido de las reglas, subrayando la principal diferencia entre sistemas expertos y los programas convencionales, que es la separación

CENTRAL EN PRUEBA	NODO TIPO A	NUMERO TIPO A	NODO TIPO B	NUMERO TIPO B	TABLA DE PRUEBA
ARC 10	TRUNK	0	MED	0	REF TABLE 1
CENTRAL	CENTRAL	0	LAR	0	REF TABLE 2
NORTH	LOCAL	0	SMA	0	REF TABLE 3
WEST	TRUNK	0	MED	0	REF TABLE 4

Figura 2
Ejemplo de TABLA_CENTRALES.

TIPO DE PRUEBA	HORAS DE PRUEBA	TIEMPO DE PRUEBA	PERIODO DE PRUEBA	TIEMPO DISPONIBLE	TIPO DE RUTA
T_DISK	6	1800	0.30	45	SECURE
T_PERIPH	4	1500	0.15	60	OPEN
T_ALARM	1	0600	0.25	15	SECURE

Figura 3
Ejemplo de TABLA_PRUEBA.

Tabla 1 — Regla en formato ESSAI

```
FOR ALL CENTRAL_REC IN TABLA_CENTRALES
   FOR ALL TEST_REC IN TABLA_PRUEBA
   [TIPO_TEST OF TEST_REC = T_PERIPH]
   [NODO_TIPOA OF CENTRAL_REC = CENTRAL]
   ::>
   [NUMERO_TIPOA OF CENTRAL_REC = NUMERO_TIPOA OF CENTRAL_REC+1]
```

entre control y conocimiento. El conocimiento se puede cambiar o mejorar sin afectar al control.

La salida del sistema experto da la cantidad requerida de unidades funcionales, ya sea por pantalla o en un fichero. Se ha comprobado la precisión y corrección del sistema utilizando centrales previamente configuradas por métodos convencionales.

Sistema operacional

El sistema experto desarrollado se ha integrado en las herramientas actuales del Sistema 12. Acepta entradas de procesos anteriores generados por tecnología convencional y produce salidas que sirven para actualizar la base de datos CAE.

El sistema puede funcionar en modo interactivo o por lotes. El modo interactivo se utiliza durante el desarrollo de la base de conocimientos para probar el efecto de los cambios y realizar una depuración interactiva; se escogerá el modo de lotes para el entorno de producción, donde no se requiere entrada de datos interactiva. Cuando se trabaja por lotes, los datos de entrada se toman de un fichero. Por el contrario, en modo interactivo, el sistema pregunta al usuario los valores de los datos de entrada que se necesiten, y de ellos deduce valores de los datos de salida. Junto a cada pregunta pueden visualizarse los valores sustitutivos que el sistema asignaría en caso de no responder el usuario.

El interfaz interactivo con el usuario divide la pantalla de visualización en tres zonas opcionalmente separables por líneas rectas horizontales. Cada una de ellas admite escritura, borrado y desplazamiento vertical del texto de forma independiente. La zona superior consta de una sola línea que indica el título del grupo de reglas; las otras dos son casi de igual anchura.

Toda la actividad se produce en la zona inferior. En ella se visualizan las preguntas del sistema, junto con las posibles respuestas (en varias columnas si es necesario), respondiendo en la misma zona el usuario. Los mensajes de error se dan en la parte inferior de la zona, y también el mensaje "procesando reglas...." que indica que se están procesando o interpretando reglas.

Cuando el usuario da una respuesta válida, el nombre de la variable junto con el valor que se le acaba de asignar se transfieren a la zona central. Al no borrarse esta zona, quedan en ella acumulados los parámetros de entrada y sus valores, a fin de que el usuario los pueda revisar en cualquier momento.

Si el usuario teclea "h" (help), se obtiene información adicional sobre el dato que se está solicitando, o se pueden visualizar las reglas que componen el grupo.

Una vez ejecutados todos los grupos de reglas, el usuario puede visualizar los resultados sobre la pantalla entera y archivarlos después. La sesión se puede recomenzar si así se desea.

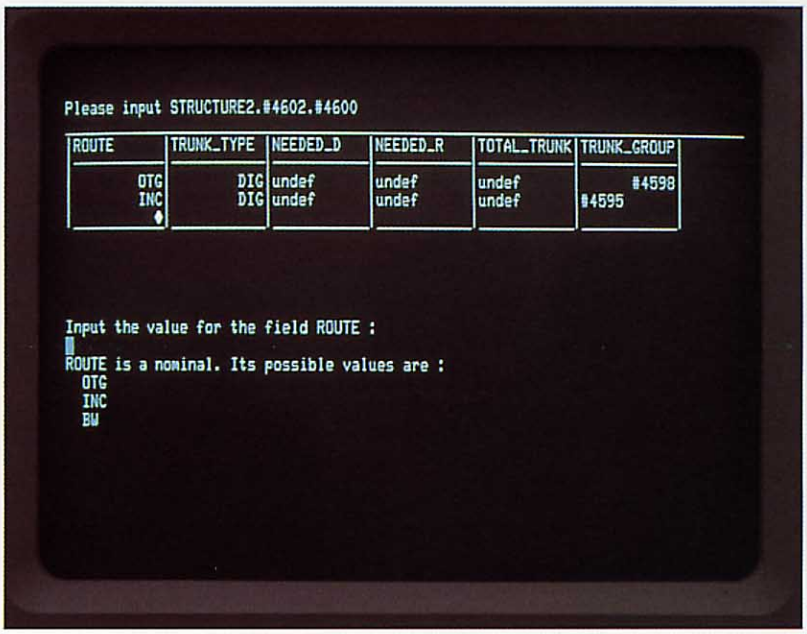
Desarrollos futuros

La aplicación a CAE de la tecnología de sistemas expertos demuestra la facilidad de gestión de la base de conocimientos. La frecuencia de los cambios depende de varios factores, como mejoras de diseño y requisitos de operación locales. Puesto que las reglas afectan al proceso a la vez que lo describen, cualquier cambio en el conocimiento repercute en una variación de las reglas, pudiendo introducirse modificaciones rápida y fácilmente.

El potencial de desarrollo rápido y progresivo que poseen los sistemas expertos permite utilizarlos en procesos donde de otro modo no sería económico desarrollar una solución automatizada. Las ventajas resultantes pueden luego hacerse llegar a toda casa ITT que fabrique centrales Sistema 12. La base de conocimiento es portátil en una forma procesable y dispuesta para actualización "en-línea". Cada compañía ITT podrá reflejar las configuraciones locales y reglas dependientes de su Administración modificando la base genérica.

Se proyecta extender el sistema experto para abarcar todo el proceso CAE; el número estimado de reglas pasa de cinco mil, lo que subraya la importancia del desarrollo progresivo.

Visualización del interfaz típico de usuario.



La mejora del sistema puede seguir dos direcciones. Primero, la parte de entrada se ampliaría de modo que, en principio, pudiera completarse el cuestionario del cliente con ayuda de un sistema experto, aprovechando así las ventajas de un interfaz soporte interactivo con validación inmediata y adecuados valores fijados localmente para el caso de no haber respuesta del usuario. Este sistema podría residir en un ordenador personal normalizado, tal como el ITT XTRA*, a fin de mejorar su portabilidad. Segundo, la parte final podría ampliarse de modo que el sistema experto actualice la base de datos CAE directamente. Un interfaz de sistema experto con base de datos permite que la semántica relacional implícita en dicha base sea captada en forma de reglas en la base de conocimientos. Aprovechando esta semántica el usuario puede tener una visión de los datos en términos de alto nivel familiares, mientras que la base de conocimientos puede mantener la correlación de esta visión de alto nivel con la visión de bajo nivel, mediante la cual se almacenan los datos en dicha base de conocimientos.

A partir del éxito inicial de la aplicación de sistemas expertos para la configuración de centrales, se han descubierto las grandes oportunidades de su futura utilización en toda la programación soporte del Sistema 12.

* Marca registrada del Sistema ITT

H. Schelfhout nació en 1949 en Malinas, Bélgica. Se graduó en ingeniería eléctrica por la Nayer Technical High School de aquella ciudad, en 1973. Tres años después se incorporó al centro de capacitación de BTM en donde trabajó sobre formación informática como Ingeniero Monitor Superior. En 1980, el Sr. Schelfhout pasó a la división de programación soporte, llegando a dirigir el departamento en 1983. Desde 1985 es Jefe de Producto de la ingeniería de aplicación (CAE) del Sistema 12 en ITT.

Generación de datos para el Sistema 12

La producción masiva de soporte lógico para centrales telefónicas requiere un conjunto de programas capaces de poblar la base de datos de cada central. Las técnicas y lenguajes de sistemas expertos se acomodan bien para esta aplicación dado su estilo declarativo y capacidad para expresar dependencias entre relaciones.

J. Yun Cabrera
Centro de Investigación de Standard Eléctrica, S.A., Madrid, España
D. G. Ketels
International Telecommunications Center, Bruselas, Bélgica

Introducción

Uno de los más importantes requisitos de las modernas centrales de telecomunicación controladas por programa almacenado es la posibilidad de producir en serie su programación. Como en otras muchas áreas, la programación se ha convertido en una de las partes más importantes y costosas del sistema, con costes de desarrollo similares a los del equipo físico. En ambos se requiere un proceso de fabricación que minimice los costes de producción.

La necesidad de producir en serie la programación adquirió importancia cuando se utilizaron por primera vez ordenadores en centrales telefónicas, y se ha resuelto, en mayor o menor grado, para los sistemas de conmutación existentes. Sin embargo, el reto persiste para los más complejos sistemas de hoy, cuyo rendimiento económico depende de una acertada solución.

El tamaño de los programas tiene gran importancia. Por ejemplo, una sola central Sistema 12 de 20.000 líneas puede contener más de 20 M-octetos de datos distribuidos entre los diferentes tipos de microprocesadores. Un segundo aspecto es que no hay dos centrales idénticas, y el entorno de cada central es único. Las centrales tienen que comunicarse con otras de diferentes fabricantes que utilizan distintas tecnologías, y cada Administración tiene sus requisitos especiales, de lo que resultan numerosas versiones de centrales. Desafortunadamente, los fabricantes tienen que aceptar estas variaciones, totalmente fuera de su control.

Otra consideración importante es la larga vida y continuo servicio del equipo telefónico. La vida útil normal de una central es de más de 20 años, durante los cuales tiene que trabajar permanentemente, sin interrupción del servicio.

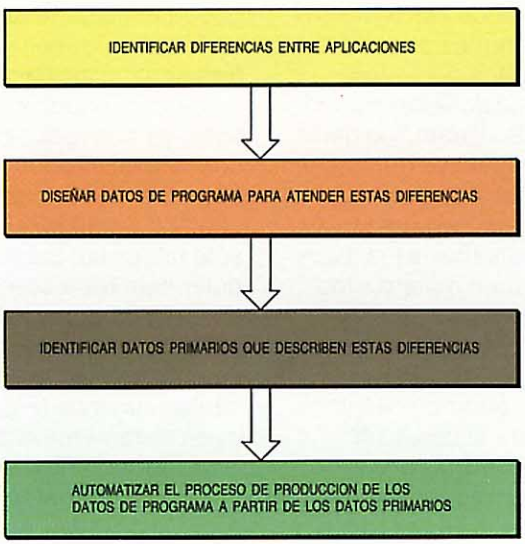


Figura 1
Etapas de la caracterización por datos para la producción masiva de programas.

Caracterización por datos

Desde el inicio del diseño hay que tener presente la necesidad de producción masiva de la programación. El objetivo total es reutilizar la mayor parte posible del trabajo invertido en el desarrollo de la primera central de cada familia. Los métodos, técnicas, sistemas soporte y organizaciones involucrados a lo largo del ciclo de vida tienen que acomodarse a familias de centrales más que a desarrollos individuales. La finalidad última sería producir los programas para nuevas centrales sin necesitar desarrollo adicional.

Para alcanzar este objetivo se sigue una estrategia denominada "caracterización por datos". El primer paso en este proceso

(Fig. 1) es hallar las diferencias entre aplicaciones, o sea, las diferencias en requisitos o facilidades que se traducen en variaciones en la programación a producir. Los diseñadores han de identificar también las partes del programa que podrían ser comunes para todas las versiones, así como aquellas otras sujetas a modificaciones para cumplir exigencias de la Administración. Todas ellas deben tener interfaces claros y bien definidos que permitan una fácil sustitución de las partes variables. Al éxito de esta tarea pueden contribuir conceptos de diseño tales como jerarquía modular y ocultamiento de la información.

El segundo paso consiste en garantizar la selección, atendiendo a valores de los datos, de todas las partes del programa que requiere una aplicación específica, con lo que las diferencias se transfieren a los datos. Los programas se dividen, pues, en dos partes: una con el código ejecutable y la otra con los datos que necesita este código para satisfacer los requisitos de una aplicación dada. Sin embargo, datos y código no pueden concebirse por separado en tanto que son partes integrantes del mismo programa[1]. El código común facilita la prueba de los módulos y reduce el trabajo de mantenimiento por utilizarse la misma copia del código en muchas aplicaciones.

A continuación se han de determinar las características de las centrales que intervienen en la versión del programa preparado para una central concreta. Hay que identificar un conjunto mínimo de datos que describan los requisitos de la Administración (tráfico, número y tipo de abonados, criterios de tasación y tarifas, entorno de la red, etc.); tales "datos primarios", que constituyen la entrada para el proceso de fabricación, caracterizan la central en términos familiares a la Administración.

El coste de producción de los datos del programa puede reducirse al mínimo automatizando su elaboración a partir de los datos primarios. Unos y otros datos describen la central desde diferentes puntos de vista. Mientras que los datos primarios describen la funcionalidad de la central sin considerar los detalles de diseño, los datos del programa incorporan muchas realizaciones concretas de circuitos y programas. Factores como la eficiencia, seguridad, arquitectura, y reglas de diseño de los datos (bases de datos), distinguen mucho a los datos de programa de los datos primarios. Las reglas o algoritmos que guían el proceso de automatización dependen del código del programa. En general, cuanto más conveniente sea para el código la representación de los datos, más dependerán del código las reglas o algoritmos de inicialización de los datos.

Todos los pasos anteriores son esenciales para el diseño de programas que han de producirse a gran escala, y dada su estrecha relación no puede omitirse ninguna de estas etapas compensándola con mayor dedicación a las restantes. Por ejemplo, es imposible automatizar si no se han identificado los datos primarios, y esto a su vez exige haber completado la primera actividad.

Es también importante que todas estas actividades sean realizadas durante la fase de diseño por gente con un profundo conocimiento del equipo. Incluso la última fase, la automatización del proceso, debería encargarse a los diseñadores del programa[1].

El Sistema 12 utiliza esta caracterización por datos, dividiendo la programación de la central en dos partes con interfaces bien definidos[2]. Se agrupan los módulos que dependen de requisitos particulares de la Administración, los datos se separan del código, y las diferencias entre centrales se concentran en la parte de datos a fin de poder reutilizar el código sin modificación. Esta separación favorece a las pruebas y al mantenimiento.

Base de datos relacional embutida

Los datos del programa del Sistema 12 se almacenan en una base de datos relacional embutida[3]. El uso de una base de datos implica la existencia de un modelo especial para representar los datos del sistema; el Sistema 12 utiliza una relacional, en la que los datos se agrupan en tablas denominadas relaciones. Una relación tiene uno o más atributos, perteneciente cada uno a un conjunto de dominios. La relación se puede definir como un subconjunto de todas las combinaciones posibles de estos conjuntos de dominios. Se llama tupla a cada una de tales combinaciones. En otras palabras, se puede considerar que la relación es una tabla donde las columnas son atributos y las filas son tuplas. Representar los datos por un modelo especial tiene importantes ventajas, ya que reduce el código requerido para manejar todos los datos. La lectura, escritura, las medidas de seguridad, el copiado y ciertas tareas de inicialización sólo necesitan hacerse una vez para cualquier dato representado por el modelo de datos común.

Se llama población de datos al proceso de construir la base de datos para una central concreta (Fig. 2), en el cual se distinguen claramente dos etapas. La primera es la producción de datos para cada relación de la base de datos; la segunda es la reunión de todas las relaciones para formar

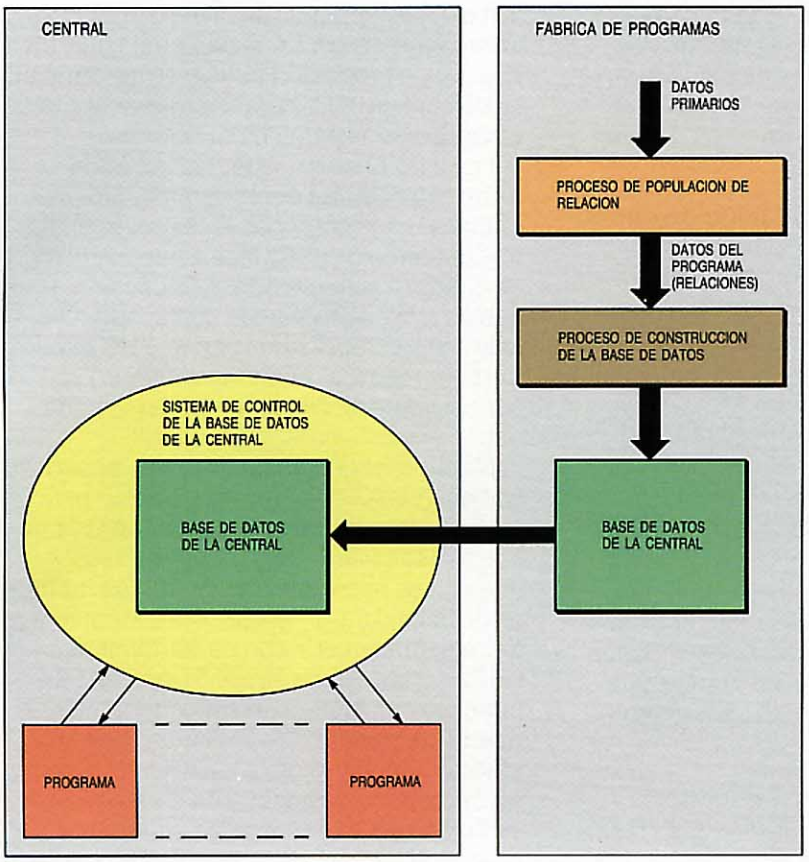


Figura 2
Fabricación de la base de datos de la central.

la base de datos. Mientras que la primera etapa requiere programas para construir cada relación particular, la segunda fase necesita sólo una herramienta que conozca cómo está realizado el modelo de datos y pueda ser utilizada en todas las relaciones de la base de datos. Se puede reducir el coste de la producción de los datos automatizando las reglas o algoritmos aplicados en la primera etapa para deducir los datos del programa a partir de los datos primarios; para esta tarea se han desarrollado programas de apoyo.

Especificar los datos y el algoritmo que han de servir para popular una relación es una tarea especializada, que requiere conocer las características telefónicas del Sistema 12 y los requisitos de las administraciones, así como las bases de datos que contienen información para la central que se configura.

Dicha especificación se prepara en un proceso al que aportan sus respectivas aptitudes el diseñador de programas de centrales, el administrador de la base de datos y el ingeniero de aplicaciones, dando como resultado una descripción informal de los requisitos de población para una relación determinada. Según el proceso automatizado actual (Fig. 3), se entrega esto al diseñador de PPR (programa de población de relaciones), cuya misión es producir una minuciosa especificación de diseño con exactos detalles del algoritmo que servirá para popular la relación cumpliendo los mencionados requisitos. Esta especificación de diseño sigue siendo una descripción informal de la tarea, pero identifica las fuentes de los datos y relaciona éstos con el algoritmo de población.

El programador utiliza la anterior especificación de diseño detallado, una vez aprobada, para preparar un PPR en un lenguaje de programación convencional (en este caso PL/1), y en ese momento la especificación informal se convierte en un objeto ejecutable. Sin embargo, el documento de diseño original ha sufrido dos transformaciones a código fuente PL/1, incurriendo en el riesgo de introducir errores de interpretación y omisión en el PPR que tal vez no se descubran hasta la fase de pruebas.

El PPR codificado a mano se compila y monta para producir el programa objeto ejecutable que podrá luego probarse. Existe una configuración de pruebas completa que permite utilizar datos válidos de centrales telefónicas de un modo controlado para cada PPR. El PPR probado forma parte de la serie de programas con los que se populan todas las relaciones de la central. A su vez,

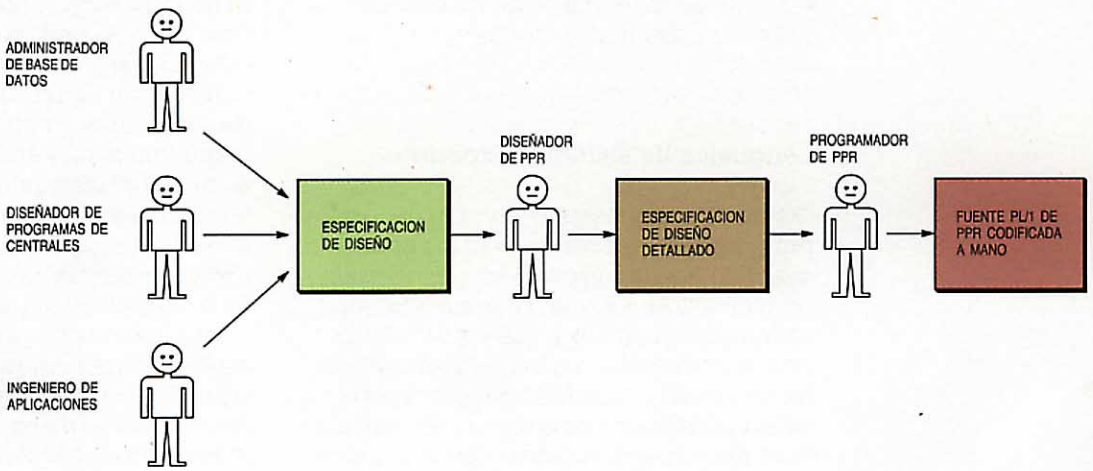


Figura 3
Automatización utilizando un lenguaje de programación convencional.

esta serie pertenece a la cadena completa de herramientas que intervienen en la fabricación automatizada de una central Sistema 12.

Lenguajes para la población de datos

El uso de un lenguaje tal como el PL/1 para la población de datos tiene la desventaja de apartarse mucho de la especificación de requisitos de población. Esto es, al inicio del proceso hay un texto en la forma de especificación de diseño de alto nivel, y al final un código fuente PL/1.

Cualquier lenguaje propuesto como alternativa necesitaría cumplir las siguientes condiciones:

- que pueda comprenderlo el personal encargado de especificar los requisitos de la población, es decir, el diseñador de los programas de la central, el administrador de la base de datos y el ingeniero de aplicaciones;
- que sea capaz de tratar con las relaciones en su integridad, ocultando los detalles de manipulación y almacenamiento para mejorar la legibilidad y el mantenimiento de tales relaciones;
- que ofrezca visibilidad de los datos primarios como si fueran relaciones en la base de datos. Concretamente, los datos primarios deberían estar representados en el modelo relacional de datos;
- que se aproveche de la configuración de pruebas e interfaz con la cadena de herramientas existentes. Su comportamiento en tiempo de ejecución será al menos tan eficiente como el de los lenguajes convencionales.

El uso de lenguajes de sistemas expertos es el elemento esencial de las alternativas examinadas. La especificación de los requisitos de población es una tarea de experto, y por tanto son de directa aplicación las técnicas de representación de conocimientos de los sistemas expertos.

Lenguajes de sistemas expertos

Sistemas expertos es el nombre dado a los programas que captan y usan el conocimiento de expertos para resolver difíciles problemas. Aunque teóricamente puedan construirse sistemas expertos utilizando cualquier lenguaje, es importante el poder de expresión (capacidad de representar cosas reales por modelación y de expresar interrelaciones sin la carga de los detalles de bajo nivel) para la solución de problemas más complicados. En el desarrollo de PPR, el autor de la relación puede verse como el "experto": la única persona que sabe cómo construir datos para sus relaciones.

Los lenguajes de sistemas expertos incorporan control por programa. Sin embargo, las sentencias en estos lenguajes no contienen ninguna indicación explícita en cuanto a la secuencia de ejecución, lo que da a tales programas una apariencia próxima a la representación que los expertos tienen de los problemas, siendo por consiguiente fáciles de entender y mantener.

Muchos lenguajes de sistemas expertos se basan en el concepto de relación para modelar conceptos reales, aunque la terminología sea diferente. La definición de relación es aplicable al elemento de memoria de trabajo del lenguaje OPS[4], al conjunto de registros en el juego de herramientas ESSAI, a las clases en EX-TRAN, etc. En principio, cualquiera de estas herramientas puede servir para desarrollar un PPR, y la elección depende de la legibilidad, el poder de expresión, facilidades de entrada/salida y eficiencia. La eficiencia es importante cuando el número de tuplas de una relación es considerable, como sucede en las centrales telefónicas.

Muchos sistemas expertos incorporan reglas de la forma *IF P THEN Q (SI P ENTONCES Q)* como vehículo para representar el conocimiento de un problema. Estas reglas casan muy bien con el problema de expresar cómo se deducen las relaciones de los datos primarios. Aunque la dependencia entre datos pueda encerrarse en un algoritmo, más a menudo se expresa como regla o conjunto de reglas que se aplican de acuerdo con los valores de los datos primarios.

Lenguaje algorítmico para población de relaciones

El método elegido para una ulterior evaluación y para el desarrollo de un prototipo operacional satisface todas las condiciones señaladas para un lenguaje de población de datos. Este lenguaje, denominado APR (algoritmo para población de relaciones), es un super-conjunto del lenguaje de sistema experto ESSAI, y combina su propiedad de representación del conocimiento de alto nivel con las características de ejecución de los lenguajes convencionales.

Siguiendo esta línea (Fig. 4), el diseñador de PPR utiliza el lenguaje APR para la especificación detallada de los algoritmos de población de datos. A diferencia del texto equivalente, el APR es una especificación

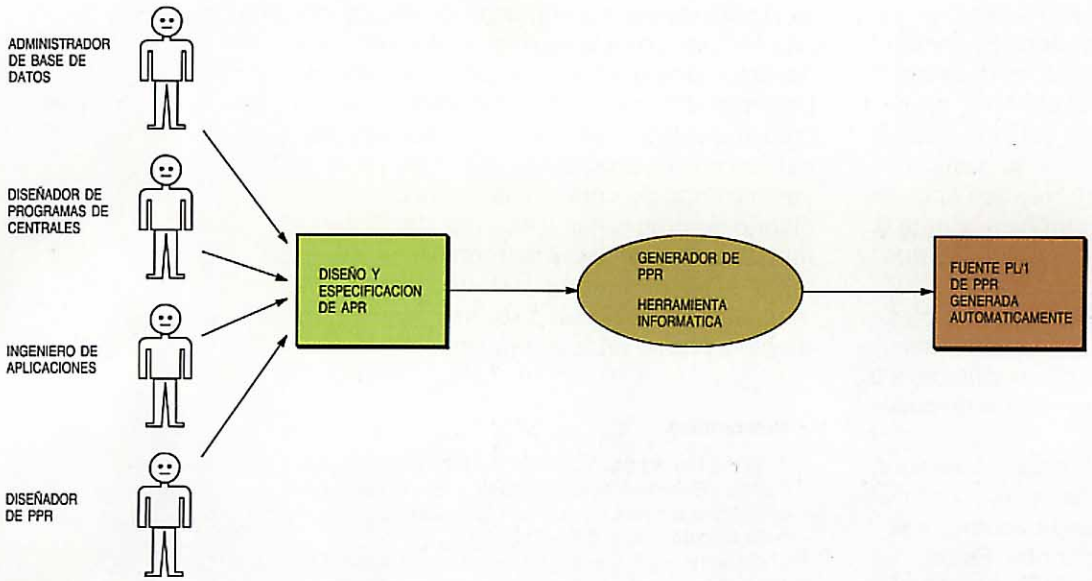


Figura 4
Automatización utilizando el lenguaje de programación APR.

ejecutable que no necesita ninguna intervención manual posterior.

La idea básica del APR es que la especificación de diseño pueda escribirse como un conjunto de reglas a nivel de experto. El lenguaje se ha adaptado a la tarea incluyendo construcciones de lenguaje para la manipulación de relaciones y tablas y las referencias a los dominios de los datos. Utiliza información de las fuentes de los datos primarios empleando directamente convenciones de nombre comunes, sin necesidad de largas y explícitas declaraciones de los datos.

El APR se apoya en un programa que comprueba la sintaxis utilizada y la integridad de las referencias a los datos. Un programa APR válido genera automáticamente un PPR como código fuente formatado en PL/1, que puede entonces ser compilado y probado en la configuración de prueba existente e integrarse en la cadena de herramientas disponible. Un PPR producido de este modo se comporta en ejecución con tanta eficacia como lo haría un programa codificado a mano. La mayor ventaja, sin embargo, es que por su especificación de alto nivel el APR puede directamente servir como medio de comunicación no ambiguo entre todas las partes involucradas en la especificación de los requisitos de la población de datos.

Realización del generador de datos mediante el lenguaje APR

En la figura 5 se muestra un lenguaje APR típico, aunque simplificado; consta de dos partes principales, encabezamiento y cuerpo, que pueden estar separados por las declaraciones de los valores sustitutivos (default) que a falta de otros se asignarán a los dominios de salida.

El encabezamiento (líneas 1 a 3) contiene el nombre que ha de darse al PPR generado (línea 1), seguido por los nombres de las tablas de entrada (línea 2) y las relaciones de salida (línea 3) para cuya población se requiere. Cada tabla consta de una o más tuplas y uno o más dominios; tanto tablas como dominios tienen nombres predefinidos por los que se les conoce comúnmente en las bases de datos. Pueden asignarse valores sustitutivos a un dominio en una relación, haciendo referencia explícita al nombre del dominio (líneas 4 y 5) y citando el valor requerido.

El cuerpo del APR (líneas 6 a 15) consta de una serie de sentencias que definen el algoritmo de población. El concepto clave es el uso de cuantificación, que hace exami-

Figura 5
Un típico algoritmo de población de relación.

```
LINEA   CONTENIDO                                                    
                                                          ENCABEZAMIENTO
 1      PROGRAM NAME (POPTRUNK)                                  APR
 2          INPUT_TABLES (PERIPH)
 3          RELATIONS (R_TRNKGRP) ;

                                                              VALORES
 4      DEFAULT FOR D_TRUNK OF R_TRNKGRP = 0;            SUSTITUTIVOS APR
 5      DEFAULT FOR D_COS OF R_TRNKGRP = 'TONE_VALID;

                                                           CUERPO APR
 6      FOR ALL INPUT_RECS IN PERIPHS
 7          IF TYPE OF INPUT_RECS = 'INCOMING'
 8          AND NBR OF INPUT_RECS = 5
 9          THEN D_TYPE OF R_TRNKGRP = 40 ;
10          D_COS OF R_TRNKGRP = COS OF INPUT_RECS ;
11      ELSE D_TYPE OF R_TRNKGRP = 20 ;
12          D_COS OF R_TRNKGRP = COS_SEC OF INPUT_RECS ;
13      ENDIF ;
14      GENERATE R_TRNKGRP ;
15      END.
```

nar una por una cada tupla de la tabla, y que esa tupla esté sujeta a las series de sentencias comprendidas en el ámbito definido por la cuantificación. En el ejemplo, el ámbito de la cuantificación está marcado al principio por FOR ALL (línea 6) hasta la sentencia END (línea 15). Hay dos tipos de cuantificación: universal como en la línea 6, donde la construcción FOR ALL hace que se seleccione cada tupla de la tabla PERIPHS, y existencial, reconocida por la construcción THERE EXISTS, en la cual sólo se selecciona una tupla, según dicta la condición que sigue a la sentencia de cuantificación.

Pueden utilizarse sentencias condicionales para definir el conjunto de términos que se deben satisfacer antes de acometer el siguiente conjunto de acciones. Estas condiciones siguen el formalismo – basado en reglas – de los sistemas expertos, donde el lado izquierdo representa condiciones y el lado derecho acciones. El uso de una condición dentro de la regla cuantificada queda ilustrado por las líneas 7 y 8, que tienen el efecto de limitar las acciones de las líneas 9 y 10 a registros en la tabla PERIPHS que satisfagan las condiciones. Si estas condiciones no se cumplen, se ejecutan las acciones asociadas a ELSE, expresadas en líneas 11 y 12.

Cada tupla de la relación de salida R_TRNKGRP se construye dominio a dominio desde los datos de la tabla de entrada, como determina el algoritmo de población. La sentencia GENERATE (línea 14) se utiliza dentro del ámbito de la cuantificación para sacar una tupla de la relación para cada registro que entra. La sentencia END (línea 15) termina el conjunto, pudiendo entonces considerar la siguiente tupla de la tabla.

Conclusiones

La naturaleza declarativa de los lenguajes de sistemas expertos, junto al muy elevado nivel de su lenguaje, determina su adecuación para resolver muchos problemas que no caen dentro de las aplicaciones normales de sistemas expertos. Este es el caso de los programas que se están utilizando para crear los datos de una base de datos relacional, los cuales programas encierran un vasto campo de conocimientos relativos al diseño de programas y equipos del Sistema 12. La legibilidad y mantenibilidad se mejoran notablemente haciendo uso de lenguajes de sistemas expertos, sin ningún efecto adverso en la eficiencia.

Referencias

1 A. Pérez Riesco y J. Yun: The Production of Program Families Based on Abstract Data Types: *Proceedings of Conference on Software for Computer Control*, octubre 1982, págs. 405–410.

2 L. Katzschner y F. Van den Brande: Central digital ITT 1240: Conceptos y realización de la programación: *Comunicaciones Eléctricas*, 1981, volumen 56, nº 2/3, págs. 173–183.

3 G. Becker, R. S. Chiapparoli, R. S. Schaaf y C. Vander Straeten: Central digital Sistema 12: Programación: *Comunicaciones Eléctricas*, 1985, volumen 59, nº 1/2, págs. 60–67.

4 C. L. Forgy: OPS83 User's Manual and Report: *Production Systems Technologies Inc*, marzo 1985.

Joaquín Yun Cabrera, natural de Pozoblanco, Córdoba, se graduó ingeniero en la Escuela Técnica Superior de Ingenieros Industriales de Madrid en 1970. En 1973 se incorporó al Centro de Investigación de Standard Eléctrica, donde dirigió el desarrollo de sistemas de soporte lógico y producción para equipos de telecomunicación; participó en el diseño de alto nivel del Sistema 12 y dirigió el desarrollo del System Builder y de GAUDA-12 (herramienta de población de datos para SESA) del Sistema 12. Durante 1985, el Sr. Yun trabajó con el grupo KETM en el ATC-ITT, en Shelton. Pertenece actualmente al grupo de producción de programación avanzada en el citado Centro de Investigación.

Dirk G. Ketels nació en 1951. Se graduó ingeniero en la Universidad de Lovaina en 1977, especializándose en ciencias informáticas. El mismo año entró en GTE ATAE donde partició en el desarrollo de STEP, un paquete de entorno de desarrollo de programación. En 1982 empezó a trabajar en el International Telecommunication Center de ITT en Bruselas, donde es responsable del desarrollo de programas avanzados, y en particular, de la aplicación de la tecnología de sistemas expertos a los sistemas de telecomunicación.

Tecnología de sistemas expertos para sistemas de seguridad crítica en tiempo real

La seguridad de la programación es un problema clave en las aplicaciones de control por ordenador de tareas cuya seguridad es crítica. En el sistema de enclavamiento electrónico ELEKTRA para control de ferrocarriles se logra un funcionamiento a prueba de fallos utilizando una arquitectura informática resistente a faltas, conjugada con un sistema especial de contraseguridad basado en un sistema experto.

N. Theuretzbacher
ITT Austria – Centro de Investigación ELIN, Viena, Austria

Introducción

La mayoría de las administraciones europeas de ferrocarriles atraviesan actualmente un cambio tecnológico, pasando desde los sistemas de enclavamiento por relés a los controlados por ordenador. Mientras que algunas han introducido ya el control informatizado en líneas escogidas (p. ej., Dinamarca y Suecia), otras, como Alemania, lo están experimentando o bien preparando extensas pruebas de campo.

La arquitectura del sistema de enclavamiento electrónico ELEKTRA para control de ferrocarriles es resultado de un estudio realizado en ITT Austria durante los dos últimos años. Este sistema de nueva generación está destinado a sustituir a los actuales equipos de la Compañía basados en relés, ampliamente utilizados en la red de ferrocarriles austriaca.

Los sistemas de enclavamiento por relés, que utilizan relés de señalización especiales con un reducido número de modos de fallo conocidos, ofrecen un grado de seguridad y fiabilidad difícilmente alcanzable con los sistemas informatizados. En el ELEKTRA, se consigue la fiabilidad necesaria por medio del sistema de ordenador VOTRICS*, un sistema modular de votación ternaria resistente a los fallos[1] basado en el ordenador de control de procesos ITT-16-Plus*, que combina la redundancia modular triple con los algoritmos de votación distribuidos.

Mientras que la fiabilidad necesaria puede conseguirse mediante un sistema de ordenador resistente a los fallos, la seguridad de la programación continúa siendo una área problemática clave cuando el ordenador se aplica a tareas cuya seguridad es crítica.

Son fundamentales en este contexto las definiciones siguientes:

- *Seguridad* es la probabilidad de que un sistema funcione en condiciones establecidas, por un periodo de tiempo dado, sin ocasionar ningún accidente.
- Un sistema está *a prueba de fallos* si se le puede pasar a un estado seguro al detectar un error potencialmente peligroso.

El enclavamiento electrónico es un ejemplo clásico de aplicación en tiempo real a prueba de fallos. Siempre que se detecte una situación peligrosa, puede ponerse la estación de ferrocarril en un estado seguro cambiando a rojo las señales de tráfico e inhibiendo cualquier orden de salida a la periferia (agujas, señales, etc.).

El sistema de contraseguridad propuesto para la arquitectura del ELEKTRA[2] es un subsistema autónomo que vigila continuamente la seguridad de la estación ferroviaria para garantizar que el enclavamiento opera de un modo inmune a los fallos, y se implantará como sistema experto en tiempo real que utiliza un lenguaje de inteligencia artificial a base de reglas.

Seguridad en la programación

Las faltas en programación son siempre faltas de diseño. Desgraciadamente, en los sistemas de seguridad crítica estas faltas encierran un peligro, y, si no se detectan durante el proceso de desarrollo o cuando el sistema se pone en servicio, pueden amenazar la seguridad del sistema. La tecnología de programación más avanzada no ofrece metodologías para diseñar programas sin ninguna falta.

* Marca registrada del Sistema ITT

El método de programación elegido para el ELEKTRA combina la prevención de faltas en la fase de diseño con la detección de faltas a lo largo del funcionamiento, a fin de reducir al máximo las faltas de diseño residuales potencialmente peligrosas. La prevención de faltas se ha conseguido al unir el potente lenguaje de programación CHILL[3] del CCITT con la metodología de diseño HCDM (método jerárquico de diseño en CHILL[4]). El HCDM y sus herramientas asociadas llevan en desarrollo desde 1983 en ITT Austria, y aunque no se hayan utilizado anteriormente en un proyecto comparable, se estima que reducirán los errores residuales entre tres y cinco veces.

Aunque las técnicas de prevención de faltas ayuden a minimizar los errores residuales de programación, en el ELEKTRA se utiliza detección de faltas suplementaria para tener máxima seguridad en la programación. En el pasado se han propuesto diversos métodos para conseguir programas resistentes a faltas, entre los que figuran el NVP (programación en versión N[5]), el plan de bloque de recuperación[6], y las afirmaciones de seguridad[7]. Sin embargo, hasta ahora sólo se ha utilizado el NVP en un sistema de enclavamiento por ordenador[8]. La idea básica del NVP es ejecutar más de una realización de la misma especificación, bien en paralelo o bien en serie, y decidir por mayoría (votar) sobre los resultados.

La ejecución en serie de programas redundantes en el mismo ordenador, adolece de una posible correlación entre faltas. Por ejemplo, el fallo de una versión del programa podría confundir al sistema operativo. La ejecución en paralelo en ordenadores separados reduce la carga y desacopla las reacciones ante faltas.

Se ha demostrado que un sistema que ejecuta dos versiones de programa es menos fiable pero más seguro que un sistema de tres versiones[9]. Aunque el NVP-2 ofrezca ciertamente mayor seguridad que un programa único, hay cierta probabilidad de faltas de diseño comunes y peligrosas (Fig. 1). La fuente principal de tales faltas está en la especificación común, a partir de la cual se producen las versiones del programa. La diversidad de las dos implantaciones tiene también una influencia importante en la probabilidad de faltas de diseño comunes. Un serio inconveniente del NVP es el coste de los múltiples diseños e implantaciones.

Un objetivo primordial en el diseño de la programación de un sistema de seguridad crítica debe ser el evitar faltas graves. Las faltas de diseño que afecten *solamente* al funcionamiento normal crean problemas, pero no son cruciales. Por consiguiente la

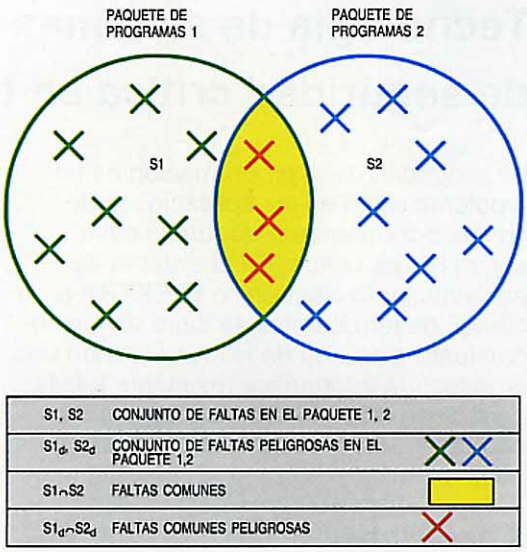


Figura 1
Distribución de faltas en un sistema de programación en dos versiones.

alternativa consiste en limitar el uso de la diversidad de programación a las funciones cuya seguridad es crítica (p. ej., el cambio de una señal de rojo a verde). Siguiendo esta orientación, la programación de una aplicación en tiempo real inmune a fallos puede comprender un sistema de programación con plena funcionalidad y un segundo sistema – el de contraseguridad – que continuamente compruebe la seguridad del sistema de control y su periferia. El uso de la contraseguridad tiene varias ventajas sobre la programación convencional en N versiones:

- diversidad, tanto en especificación como en diseño
- menos complejidad, por lo tanto aún menor número de faltas residuales
- coste de desarrollo más reducido
- mayor seguridad.

En virtud de lo expuesto, ITT Austria ha adoptado esta técnica en el sistema ELEKTRA.

Técnica del sistema de contraseguridad

Esta técnica significa un nuevo enfoque en el área de programación de seguridad crítica. Se basa en un sistema informático autónomo que comprueba constantemente el estado de seguridad del sistema de control y su periferia: todas las acciones y reacciones del sistema se contrastan por segunda vez con las reglas de seguridad. En el sistema de contraseguridad se distinguen dos partes:

Parte pasiva: todas las acciones críticas para la seguridad, resultado de la programación de control, se contrastan con reglas de

seguridad antes de transferirlas a la periferia (p. ej., antes de ordenar el paso de rojo a verde de una señal), inhibiendo así todas las acciones peligrosas.

Parte activa: la contraseguridad garantiza que se producen a tiempo todas las reacciones críticas necesarias para la seguridad, como el cambio de una señal de verde a rojo cuando pasa un tren. Si la programación de control omitiera una de tales reacciones, la contraseguridad aseguraría su ejecución.

Para poder analizar el estado de seguridad del sistema entero, el sistema de contraseguridad debe mantener en tiempo real una información completa de la topología y configuración del sistema en su base de datos, y por tanto ha de recibir todos los mensajes relativos a cambios del estado de la periferia.

El objetivo del sistema en cuestión – vigilancia de la seguridad – difiere mucho del de la programación de control (operación de la estación ferroviaria). Al tener, pues, los dos paquetes especificaciones diferentes, habrá muy poca probabilidad de modos comunes de fallo.

Además de la diferencia entre especificaciones, se consideró importante tener máxima diversidad de diseño entre la programación de control del enclavamiento y la de la contraseguridad. El modo convencional de introducir diversidad en los diseños sería el utilizar distintos lenguajes de programación (p. ej., CHILL y ADA), pero se puede conseguir una diversidad mucho mayor mediante el uso de diferentes métodos de programación en cada paquete.

Dado que la referida programación de control realiza principalmente las clásicas tareas de control de procesos, resulta muy adecuado elegir el CHILL y los métodos de programación por procedimientos. Por otra parte, la principal tarea del sistema de contraseguridad consiste en la continua vigilancia del estado de la estación ferroviaria en lo que atañe a normas de seguridad. Estas pueden expresarse idealmente en un lenguaje de programación basado en reglas, y por ello se decidió realizar la contraseguridad en forma de sistema de producción, un sistema experto que utiliza reglas del tipo IF (SI) <*condiciones*> THEN (ENTONCES) <*acciones*>.

Uso de un sistema de producción en condiciones de tiempo real

Los procesos de ordenador de los sistemas de producción son de diferente estilo de los realizados por programas escritos en un lenguaje de procedimientos tal como el CHILL. Una de las diferencias principales consiste en que los sistemas de producción toman como unidad básica de sus procesos unas reglas no ordenadas (llamadas *producciones*) sensibles a los datos, en vez de una secuencia de instrucciones[10]. Un sistema de producción suele constar de tres bloques principales (Fig. 2): la memoria de trabajo que sirve como almacén de datos globales, la memoria de producción o base de reglas que contiene las reglas, y la máquina de inferencia que controla la ejecución de las reglas en el ciclo denominado *reconocimiento-acción*.

El ciclo reconocimiento-acción consiste básicamente en una operación de tres etapas:

- Concordancia, que evalúa los lados izquierdos de las reglas para determinar cuáles de ellos son verdaderos y por tanto ejecutables, denominándoles *conjunto en conflicto*.
- Selección, la cual determina qué reglas han de ejecutarse a continuación.
- Ejecución, que realiza la parte de acción de la regla o reglas seleccionadas.

La máquina de inferencia se para normalmente cuando el conjunto en conflicto está vacío (objetivo resuelto) o se ha alcanzado un número máximo de ciclos (no hay resolución).

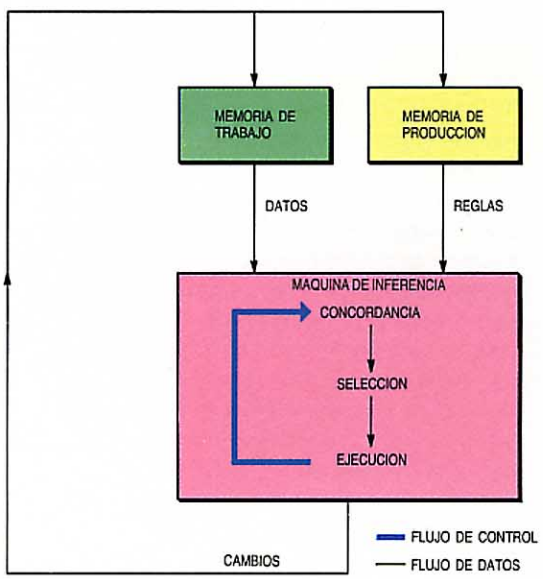


Figura 2
Arquitectura básica de un sistema de producción.

La mayoría de los sistemas expertos son sistemas autónomos para ordenadores personales y centrales, y se comunican con el usuario mediante un diálogo de consola. Su aplicación en el control de procesos origina los siguientes requisitos nuevos:

- restricciones de tiempo
- cambio continuo de datos

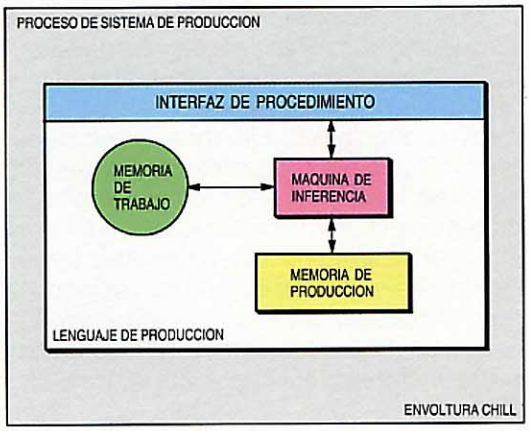


Figura 3
Sistema experto con su envoltura CHILL.

– sincronización

– comunicación.

Las restricciones de tiempo sólo pueden resolverse mediante prestaciones adecuadas y una cuidadosa codificación de las reglas. El funcionamiento con datos que cambian continuamente exige sincronización y comunicación con las fuentes de datos (bases de datos en tiempo real, otros procesos). Ni una ni otra cosa la ofrecen los actuales lenguajes de programación de inteligencia artificial basados en reglas.

Para el sistema de contraseguridad ELEKTRA se ha elegido el desarrollo, a partir de los bien conocidos conceptos de lenguaje de la familia OPS (OPS5[10], OPS83[11]), de un lenguaje de programación basado en reglas que satisfaga los mencionados requisitos de sincronización y comunicación. Es esencial que tal lenguaje sustente no sólo los métodos de programación por reglas de las anteriores versiones OPS, sino también la programación por procedimientos, como en el CHILL. Se resolverán así muchos problemas de manera más sencilla y eficaz, y además se capacitará al usuario para programar el ciclo reconocimiento-acción de la máquina de inferencia, optimizando la estrategia de control para una aplicación dada.

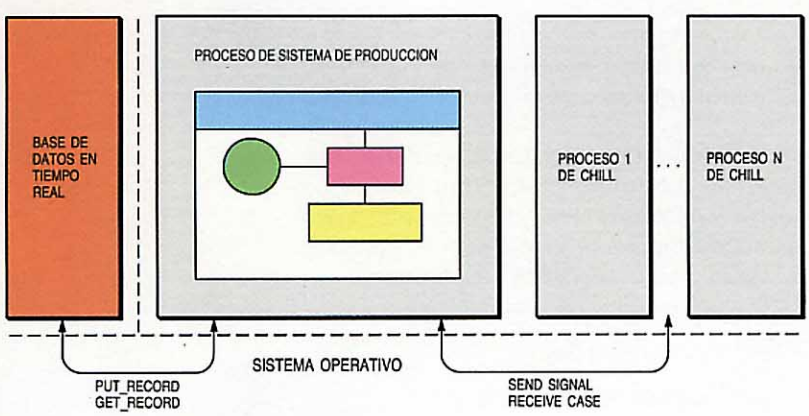


Figura 4
Sistema experto en un entorno de procesamiento en tiempo real.

Se ha elegido un lenguaje basado en reglas de producción por su alta velocidad de ejecución. Debido a su base de conocimientos compilada y a un algoritmo muy eficiente para contrastar configuraciones (el llamado "RETE"[12]), el sistema experto puede ejecutar una media de hasta 100 reglas por segundo. Antes de utilizarse en ELEKTRA, tenía que funcionar en el ordenador de control de procesos ITT-16-Plus que ejecuta bajo el control del VOTRICS[1].

Los problemas de sincronización y comunicación se resolvieron poniendo el sistema experto completo (base de conocimientos, área de trabajo de datos y máquina de inferencia) bajo envoltura CHILL (Fig. 3), de manera que tenga la apariencia de un proceso CHILL ordinario, capaz de acceder a la base de datos en tiempo real y comunicar con otros procesos.

Un interfaz de procedimiento da acceso a todos los servicios necesarios del sistema operativo. En vez de comunicar por interfaz de diálogo con el usuario, el sistema dentro de su envoltura CHILL comunica mediante primitivas de este lenguaje con otros procesos CHILL y con la base de datos en tiempo real (Fig. 4).

Implantación del sistema de contraseguridad

El uso de la tecnología de sistemas expertos permite la mayor diversidad de diseño posible para la contraseguridad, pues se utilizan tanto la programación declarativa (por reglas) como la programación por procedimientos (CHILL). Otro factor positivo es la facilidad para expresar reglas de seguridad; la representación por reglas puede verse casi como una especificación formal directamente ejecutable (sin codificación suplementaria). Sin duda, la codificación en CHILL de las reglas de seguridad requeriría mucho más trabajo y aumentaría el riesgo de errores adicionales de codificación.

La ejecución por el sistema de una revisión de seguridad comporta tres fases. La primera fase, o preevaluación, consta de dos partes: definición de la orden y recuperación de los datos. La definición de la orden traduce la solicitud de orden (normalmente un mensaje recibido desde un proceso de aplicación) en un objetivo a alcanzar. La recuperación de datos carga los objetos de datos que representan los estados actual y requerido del sistema, obteniendo dichos objetos de la base de datos en tiempo real por medio de las primitivas de CHILL GET_RECORD y PUT_RECORD, y transformándolos en elementos "objetivo" y de "estado" que luego se almacenan en la memoria de trabajo. Después de la preevaluación viene la evaluación de segu-

```
type    SWITCH = element
         (NUMBER          : integer;    -- number of the switch
         POSITION         : symbol;     -- left, right, undefined
         OCCUPANCY        : symbol;     -- free, occupied, undefined
         MANUAL_MOVE      : symbol;     -- allowed, not_allowed
         LOCKED           : symbol;     -- yes, no
         CLOSED           : symbol;     -- not_used, used_in_main_route,
                                        -- used_in_shunting_route
         ROUTE_NR         : symbol;     -- the number of the route in which
                                        -- the switch is located
        );
```

Figura 5
Declaración tipo de un objeto de datos que describe el estado de un conmutador de agujas.

```
type   GOAL = element
        ( STATUS          : symbol;     -- safety_check, check_ok,
                                        -- danger
         COMMAND          : symbol;     -- move_switch, ...
         COMMAND_ATTR     : symbol;     -- single_move, move_in_route,
                                        -- manual_move, ...
         OBJECT           : integer;    -- number of switch, signal, etc.

         ) ;
```

Figura 6
Declaración tipo de un elemento objetivo para el sistema de contraseguridad.

ridad, en la cual se ejecutan las reglas de seguridad. Finalmente, en la fase de post-evaluación se envía el resultado positivo de la evaluación al proceso adecuado o se inicia una operación de urgencia (p. ej., inhibir una orden peligrosa y señalizar la emergencia al operador de la estación).

Los siguientes ejemplos simplificados muestran cómo podrían presentarse las definiciones de tipos de datos, objetivos y reglas del sistema de contraseguridad. La figura 5 expone una declaración tipo de un objeto de datos que describe el estado de seguridad de un conmutador de agujas ferroviario. El sistema de contraseguridad contiene declaraciones tipo por cada clase de elemento periférico (señales diversas, vías, etc.).

Cuando de recibe una solicitud de orden (p. ej., un cambio de agujas), el sistema recoge la información apropiada de la base de datos en tiempo real externa (fase de preevaluación) y la almacena en una edición particular – una copia lógica – de un objeto de datos del tipo correspondiente (estos objetos se almacenan en la memoria de trabajo). Durante la revisión de seguridad se contrastan tales objetos con las condiciones expresadas en las reglas de seguridad.

Los elementos objetivo (Fig. 6) sirven para controlar la ejecución de las reglas de contraseguridad. Cada solicitud de orden tiene que traducirse en una edición del correspondiente elemento objetivo que se almacena en la memoria de trabajo. La máquina de inferencia contrasta el elemento objetivo con las condiciones objetivo del lado izquierdo de las reglas y selecciona las reglas que concuerdan. Normalmente una regla contiene un elemento objetivo y algunas condiciones adicionales de cumplimiento de seguridad.

El ejemplo de regla dado en la figura 7 contiene todas las pruebas de seguridad que han de efectuarse para una orden de cambio de agujas durante el establecimiento de una ruta. El lado izquierdo de la regla tiene dos condiciones de concordancia: el elemento objetivo selecciona el tipo de operación que ha de realizar el sistema

```
rule MOVE_SWITCH_IN_ROUTE
{                                                    -- LHS (match conditions)
     &1    (GOAL                                     -- GOAL element controls
                                                     -- operation of the
                                                     -- expert system
             STATUS              = safety_check;     -- perform safety check of
             COMMAND             = move_switch;      -- switch changeover command
             COMMAND_ATTR        = move_in_route;    -- during route setup
            );
     &2    (SWITCH                                   -- SWITCH data object
             NUMBER              = &1.OBJECT;        -- describes the status of
             POSITION            <>undefined;        -- the switch to be changed
             OCCUPANCY           = free;             -- over
            MANUAL_MOVE          = not_allowed;
             LOCKED              = no;
             CLOSED              = no;
            );
     -->                                             -- RHS (action part)
                                                     -- signal "safety_check_ok"
                                                     -- if the safety conditions
                                                     -- are met

  modify &1 (STATUS = check_ok);
}
```

Figura 7
Ejemplo de regla de seguridad para un cambio de agujas del ferrocarril. Esta regla contiene todas las pruebas de seguridad necesarias antes de mover el cambio de agujas cuando se establece una ruta.
LHS- lado izquierdo
RHS- lado derecho

experto, y el elemento conmutador mantiene información completa sobre el estado de seguridad del cambio de agujas. La regla ha de ejecutarse para todas las órdenes de cambio de agujas que exige el establecimiento de una ruta a través de una estación de ferrocarril. El conmutador de agujas sólo podrá moverse si se cumplen las condiciones de seguridad de esta regla.

El sistema experto tiene que trabajar sobre datos almacenados en una base de datos en tiempo real (fuera del sistema experto), la cual se mantiene mediante un conjunto de procesos CHILL que responden a los mensajes de eventos procedentes de la periferia. La sincronización y comunicación del sistema experto de seguridad con otros procesos (CHILL) de la contraseguridad solamente es posible a través de mecanismos de sincronización CHILL.

Arquitectura de la programación ELEKTRA

La arquitectura de ELEKTRA consta de seis funciones (Fig. 8). Verticalmente el sistema se divide en tres capas funcionales:

- interfaz hombre-máquina del operador
- procesamiento de enclavamiento y de seguridad
- control de periféricos.

En sentido horizontal, se distinguen el subsistema de enclavamiento y el de seguridad. El primero ofrece todas las funciones necesarias para controlar una estación ferroviaria. La introducción de órdenes del operador se hace con un lápiz de luz en un interfaz hombre-máquina con gráficos de colores, dirigido por menús. Tras un procesamiento previo por el procesador maestro de vídeo, dichas órdenes se envían al procesador de enclavamiento.

Este último controla el estado de todo el sistema y, de acuerdo con las reglas de seguridad, transforma las órdenes provenientes del procesador maestro de vídeo en un conjunto de órdenes de control de periféricos que han de ser enviadas al procesador de control de elementos A. Por otra parte, el procesador de enclavamiento actualiza continuamente su base de datos con el estado del sistema, conforme a los mensajes de eventos que recibe del referido procesador A. Se le exige además reaccionar a tiempo ante ciertos eventos, como el ya citado de cambio de señal verde a roja al paso del tren. El procesador maestro de vídeo recibe del procesador de enclavamiento información sobre el estado.

Después de recibir una orden de control desde el procesador de enclavamiento, el

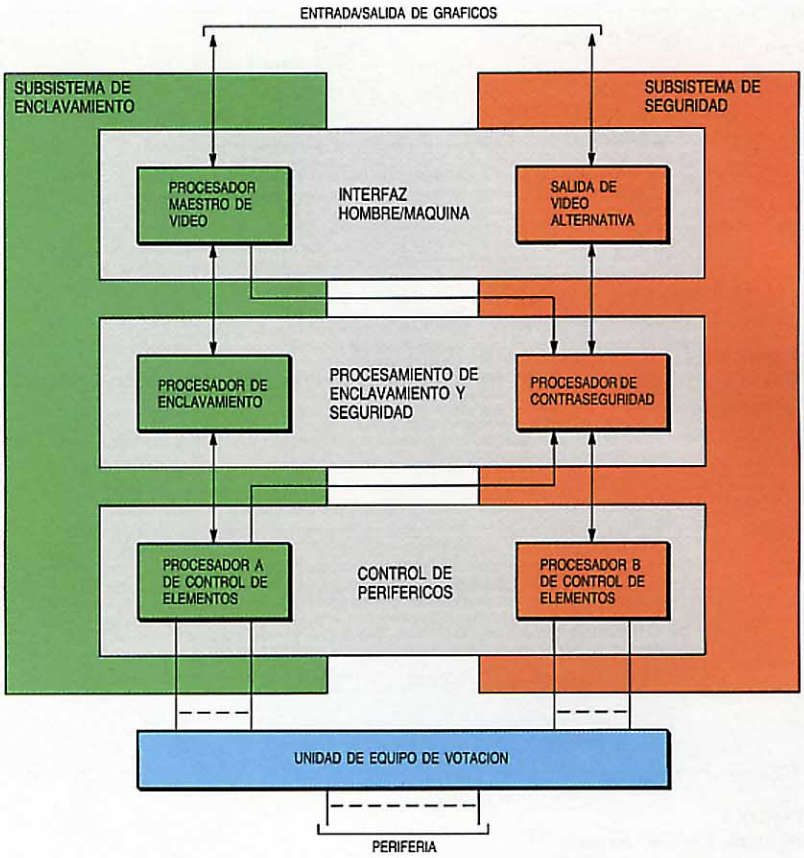


Figura 8
Modelo de arquitectura básica de ELEKTRA (las cajas representan bloques funcionales, y las flechas canales de comunicación).

procesador A antes mencionado almacena dicha orden y la retransmite al procesador de contraseguridad, el cual comprueba si tal orden pudiera violar la seguridad del sistema. Si la orden supera todas las pruebas, el procesador de contraseguridad la envía al procesador de control de elementos B; por el contrario, cuando se advierte que puede crear una situación peligrosa, se la inhibe y se manda al operador una señal de emergencia. Por razones de sincronización y para que las pruebas de seguridad sean más completas, el procesador de contraseguridad recibe también todas las órdenes que el procesador maestro de vídeo transmite al de enclavamiento.

Una vez que el procesador B ha recibido una solicitud de orden desde el procesador de contraseguridad, reexpide esta petición al procesador A y emite una orden de control a la periferia. Al recibir dicha solicitud de orden, el procesador A la compara con la correspondiente orden recibida del procesador de enclavamiento y, si son idénticas, el procesador A emite también una orden de control a la periferia. Las señales eléctricas de ambas órdenes se vuelven a comparar por una unidad de equipo de votación, y sólo en caso de ser iguales se transfieren al interfaz del equipo físico. Los dos procesadores de control de elementos están constantemente vigilando el estado de la periferia y comunican todos los cambios a los

procesadores de enclavamiento y del sistema de contraseguridad.

Se necesita el equipo de votación adicional por razones de seguridad; en efecto, aunque se tomen decisiones por votación lógica en cada operación de salida, cualquier fallo en la producción de las señales eléctricas de salida correctas podría dar lugar a que los periféricos de la estación actuaran de modo peligroso (p. ej., un cambio de agujas no deseado). Para impedir que se introduzca ningún punto de fallo aislado hay una unidad de equipo de votación por cada señal de salida.

El procesador de contraseguridad recibe todos los mensajes de estado de los periféricos desde el procesador de control de elementos B, y actualiza continuamente su propia base de datos en tiempo real con el estado del sistema. Con objeto de reaccionar ante eventos de la periferia, el procesador de contraseguridad emite de forma autónoma al procesador B las solicitudes de órdenes que sean necesarias.

Para ciertas órdenes de operador críticas (*órdenes registradas*) ELEKTRA proporciona un canal adicional de salida de gráficos, el módulo de salida de vídeo. Este camino alternativo se utiliza para reflejar y reconocer órdenes manuales de operador potencialmente peligrosas, tales como el establecimiento de una *señal de sustitución* especial – equivalente a una luz verde – que permita a un tren entrar en una ruta aunque, a causa de la avería de un sensor, una parte de la vía libre se dé por ocupada.

Conclusiones

ITT Austria ha desarrollado un nuevo método para optimizar la seguridad de la programación de las aplicaciones de control de procesos inmunes a fallos. Se incluye en la arquitectura del sistema un sistema de contraseguridad; este subsistema autónomo comprueba continuamente el estado de seguridad del sistema completo, que comprende control y periferia. Para la implantación del sistema de contraseguridad se usa un lenguaje de inteligencia artificial, basado en reglas, que garantiza una máxima diversidad de diseño. La primera aplicación de la técnica de contraseguridad será el sistema de enclavamiento electrónico ELEKTRA, que está desarrollando ITT Austria y cuyo prototipo se espera terminar en 1988/89.

Referencias

1 N. Theuretzbacher: VOTRICS: Voting Triple Modular Computing System: *Proceedings of FTCS-16*, Viena, junio 1986.

2 N. Theuretzbacher: ELEKTRA: A System Architecture that Applies New Principles to Electronic Interlocking: *Proceedings IFAC Conference on Control in Transportation Systems*, 1986, Viena.

3 Recomendación Z.200 del CCITT, Lenguaje de alto nivel del CCITT (CHILL), *Libro amarillo*, 1981, volumen VI, fasc. VI.8, Ginebra.

4 N. Theuretzbacher: HCDM: A Hierarchical Design Method for CHILL-Based Systems: *3rd International CHILL Conference*, 1984.

5 A. Avizienis y L. Chen: N-Version Programming: A Fault-Tolerance Approach to Reliability of Software Operation: *8th International Conference on Fault-Tolerant Computing*, Toulouse, junio 1978, págs. 3–9.

6 J. J. Horning, H. C. Lauer, P. M. Melliar-Smith y B. Randell: A program Structure for Error Detection and Recovery: *Proceedings of the Conference on Operating Systems*, Theoretical and Practical Aspects, INRIA, abril 1974, págs. 177–193.

7 N. Leveson: Safety Assertions for Process-Control Systems: *Proceedings of FTCS-13*, 1983, págs. 236–240.

8 A. A. Jonasen y N. Siggard: Microcomputers Take Over the Interlocking Function: *Railway Gazette International*, diciembre 1981.

9 M. Mulazzani: Reliability Versus Safety: *Proceedings of SAFECOMP*, 1985.

10 L. Brownston y otros: Programming Expert Systems in OPS5 – An Introduction to Rule-Based Programming: *Addison Wesley*, 1985.

11 C. L. Forgy: The OPS83 Report, System Version 2.1: *Production Systems Technologies Inc*, octubre 1984.

12 C. L. Forgy: RETE: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem: *Artificial Intelligence*, 1982, 19.

Norbert Theuretzbacher nació en 1952. Estudió telecomunicación en la Universidad Técnica de Viena, donde se le concedió el grado de Diplomado en Ingeniería. Ingresó en ITT Austria en 1975 y al año siguiente asumió la responsabilidad del desarrollo de tecnologías de programación en tiempo real. En 1983 recibió el "ITT Programming Recognition Award" por la mejor cadena de herramientas de programación. El Sr. Theuretzbacher es actualmente director del Centro de Investigación ITT ELIN, constituido recientemente en Viena.

Sistemas expertos aplicados a centrales TXE4A

Uno de los mayores problemas de los operadores de centrales telefónicas grandes y complejas es asegurar que funcionan sin fallos. La tecnología de sistemas expertos se ha aplicado con éxito a la puesta en servicio y mantenimiento de centrales telefónicas TXE4A, consiguiendo una espectacular disminución del número de mensajes de fallo de las centrales.

M. Thandasseri
STC Telecommunications Ltd, Londres, Inglaterra

Introducción

Las centrales TXE4A cubren alrededor del 30% de la red de British Telecom, y dan servicio a más de 8 millones de abonados. El sistema está basado en una matriz de puntos de cruce de relés reed con retención eléctrica, conmutados bajo el control de programa almacenado. Las áreas de control y de conmutación, separadas por seguridad, se componen de bloques funcionales normalizados, obteniendo así un sistema conceptualmente sencillo, versátil, tolerante a los fallos y altamente seguro.

La figura 1 muestra las cuatro partes principales de una central TXE4A: red de conmutación; registradores, interrogadores (que interrogan la vía de conmutación para identificar un camino libre) y marcadores; exploradores de líneas y memorias de datos; unidades de procesadores de control. La red modular de conmutación acepta grandes variaciones en las tasas de llamadas y en las relaciones de tráfico de abonados a tráfico de enlaces. Al estar normalizadas las unidades de conmutación, se garantiza una vía de transmisión equilibrada de calidad mucho mayor que la ofrecida por los sistemas electromecánicos.

Los registradores actúan como interfaces con los procesadores principales de control, y para aumentar la seguridad su número está en función del tráfico de la central. Los interrogadores y los marcadores están asociados a las unidades de conmutación; en respuesta a peticiones, identifican el estado de la red y marcan vías a su través.

Los exploradores de líneas y las memorias de datos tienen un alto grado de seguridad y corresponden a bloques discretos de capacidad de líneas de la central. Esto minimiza el tamaño del módulo de ampliación, que es grande en muchos sistemas electrónicos de conmutación con control común.

Los procesadores principales de control proporcionan las funciones propias de una "operadora", tales como identificación del abonado que llama, determinación de las conexiones necesarias y selección de las rutas apropiadas a través de la red. El control por programa separa las funciones operacionales de los equipos físicos, lo que permite cambiar las facilidades de la central con mínimos trastornos al equipo en funcionamiento.

Diagnosis de fallos en TXE4A

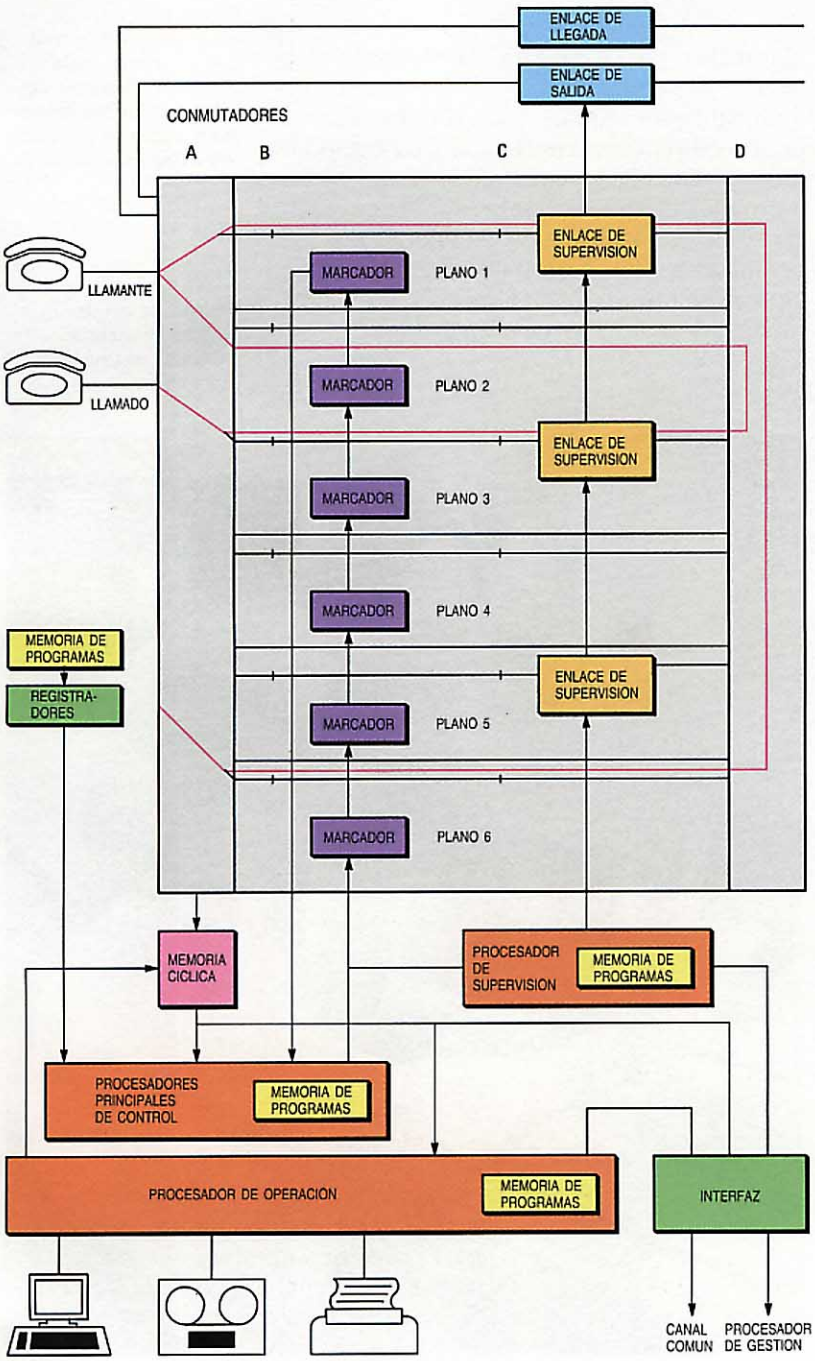
Las centrales TXE4A incorporan sistemas de diagnóstico que producen alarmas e informes de fallo. El ingeniero de mantenimiento o de puesta en servicio está rodeado, casi literalmente, de información para ayudarle a identificar y localizar los fallos, la cual adopta formas muy variadas: notas para cursos de capacitación, manuales, diagramas de ingeniería, esquemáticos y mensajes de fallo. En consecuencia, tendrá que dedicar un tiempo considerable a comparar referencias y actualizar sus conocimientos personales, y quizás sus propios apuntes (por ejemplo, para formación, análisis de fallos, localización y reparación de fallos). Por ser difícil manipular enormes cantidades de información sobre diagnósticos y fallos, se le puede escapar el fallo que estaba causando el problema, e incluso ya localizado y corregido éste, habrá de trabajar bastante para idear una prueba que verifique la reanudación del correcto funcionamiento de la central.

La situación exige cambios y plantea un conflicto de prioridades. Para entender un sistema concreto, interpretar los informes de fallos y aprovechar plenamente las herramientas de prueba disponibles, se necesita un elevado nivel de especialización, y por

tanto una prolongada capacitación y una selección cuidadosa de técnicos que sean capaces de asimilar y analizar grandes cantidades de información.

Aunque STC tenga unos 150 ingenieros de mantenimiento, no todos poseen la formación y experiencia suficiente para ser considerados expertos. En consecuencia, el reto era que incluso los menos experimentados alcanzaran la eficacia de diagnóstico tradicionalmente reservada a los expertos. STC pensó que una herramienta que pusiera la experiencia de los mejores ingenieros a disposición del grupo entero mejoraría notablemente la eficacia global de mantenimiento.

Figura 1
Diagrama de bloques de una central TXE4A.



Desarrollo del sistema experto

La tecnología de sistemas expertos, relativamente nueva, ofrece la oportunidad de desarrollar una herramienta "inteligente" de diagnosis de fallos para mantener complejos equipos de comunicaciones. En 1982, STC lanzó una propuesta de sistema experto de diagnóstico que se utilizaría para poner en servicio y mantener centrales TXE4A. El desarrollo de este sistema comenzó en 1983 con un plazo de terminación fijado en nueve meses, y se realizó en un miniordenador que utilizaba la "envoltura" de sistemas expertos Sage, herramienta para construir sistemas expertos esencialmente compuesta de una máquina de inferencia y una base de conocimientos vacía que puede llenarse con informaciones de ese dominio. Como el sistema experto debía diagnosticar fallos con suficiente profundidad de análisis, se utilizó una versión con memoria de 1 M-octeto.

El miniordenador resultó ser un buen medio para desarrollar y probar el modelo, ya que fue rápido y ofreció herramientas suficientes para la envoltura de sistemas expertos elegida. No obstante, por el elevado coste del miniordenador no se podía instalar uno en cada central, y se necesitaba una alternativa más económica. El tamaño del modelo y el tener que utilizar unos cuantos procedimientos de usuario (para análisis de los mensajes de fallo, por ejemplo), indicaban que la máquina debía ser potente pero rentable. Se seleccionó un microordenador de 16 bitios con sistema operativo UNIX, y en él se cargó el sistema experto paralelamente a su desarrollo, con el fin de garantizar la disponibilidad en abril de 1984 (fecha tope señalada) de un sistema de sobremesa capaz de instalarse en una central telefónica. La versión con microordenador ofrece exactamente las mismas funciones que la versión desarrollada y es también capaz de realizar un análisis automático de fallos.

El equipo de desarrollo estaba formado por ingenieros del conocimiento, expertos en diagnosis de fallos del TXE4A, y especialistas en ordenadores. Se recogieron conocimientos de los expertos y se condujeron, a través de un ingeniero del conocimiento, a la envoltura del sistema experto, la cual consta de un interfaz de usuario, una máquina de inferencia y la base de conocimientos que contiene reglas y consejos (Fig. 2). En la versión actual, dicha base de conocimientos está formada por unas 1.350 reglas, 2.200 preguntas, 2.400 afirmaciones y 2.300 consejos, y está organizada en 660 áreas con indicaciones de fallos específicos.

La tabla 1 muestra una regla típica, mientras que la figura 3 esquematiza la base de conocimientos: los bloques representan las principales áreas de conocimientos almacenados en el sistema, y las líneas, las vías de acceso que las relacionan. La mayoría de las áreas mostradas en la figura se subdividen en el sistema en grupos de áreas más pequeñas. En la figura se aprecia no sólo la estructura de la base de conocimientos,

Tabla 1 — Una regla típica

Fallo de las pruebas del controlador del disco

Las pruebas del controlador del disco fallan

SI:

1) El conector de borde del controlador del disco está sucio o dañado
O 2) Los puentes del controlador del disco no son correctos
O 3) Los puentes del disco Winchester no son correctos
O 4) El cable cinta de 50 hilos que conecta el disco Winchester con el controlador tiene defecto o está mal insertado
O 5) El disco Winchester no está formatado
O 6) La información de "bloque malo" en el disco Winchester es correcta
O 7) El disco Winchester tiene defecto
O 8) El controlador del disco tiene defecto
O 9) La placa hija del controlador del disco flexible está defectuosa

sino también el alcance del sistema; se ve, en efecto, que el sistema tratará los tres tipos principales de indicación de fallo — fallos de la unidad procesadora de operación, mensajes de fallo y alarmas — y se indica la cobertura dentro de estas áreas. El procesador de operación está destinado al mantenimiento y administración de las centrales TXE4A, pudiendo actualizar información en las memorias de datos, controlar las pruebas rutinarias, e investigar las condiciones específicas de cada línea.

En líneas generales, el sistema experto se dirige a localizar fallos en una central TXE4A, usualmente hasta el nivel de unidad enchufable, y en casos concretos a nivel de componentes de la red de conmutación.

Sistema experto de operación

El sistema experto de operación, al que se denomina AMF (*Advanced Maintenance Facility*, dispositivo avanzado de mantenimiento), se ejecuta en un microordenador de 16 bitios conectado a la central, y puede utilizarse tanto en modo manual como

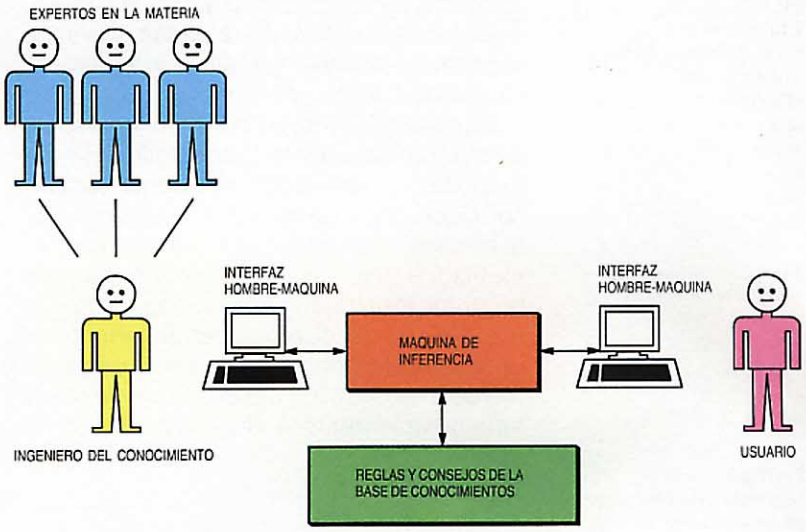
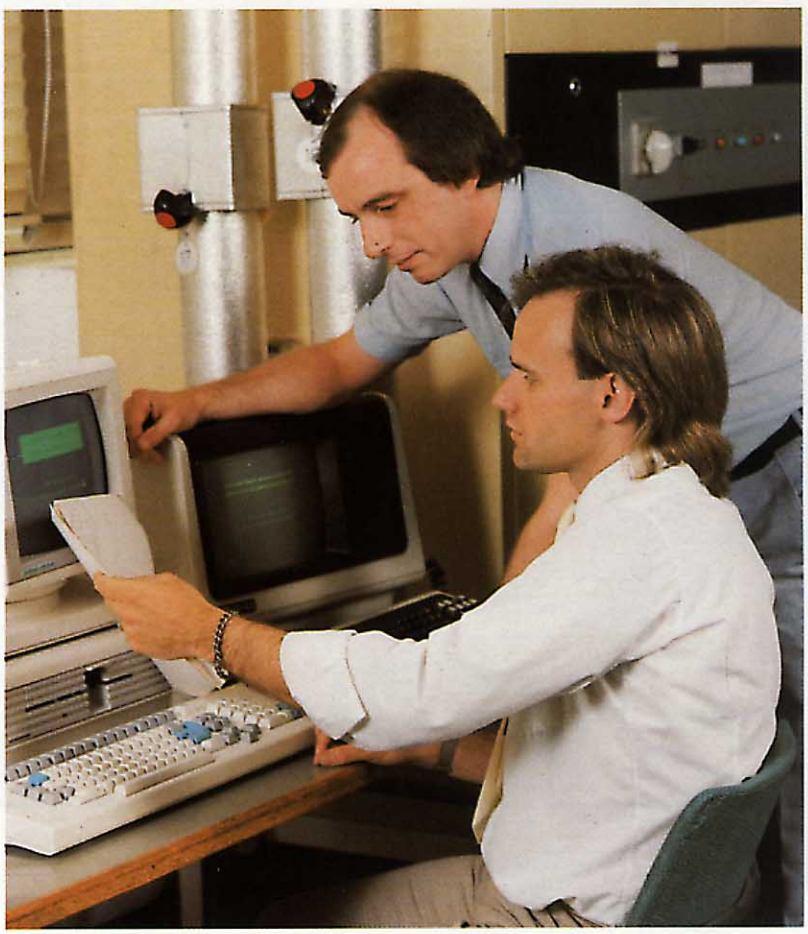


Figura 2
Ingeniería del conocimiento, mostrando cómo se introduce en la base de conocimientos el saber de ingenieros experimentados .

automático para localizar e identificar fallos. Si para el diagnóstico se utiliza el modo automático, el sistema debe conectarse al puerto de impresora del bastidor de equipos electrónicos misceláneos de la central. En este caso, el AMF recoge las informaciones de fallos directamente de la central y las consultas se basan en la información almacenada en el ordenador. Casi el 70% de los mensajes de fallo se refieren a fallos al

Equipo de mantenimiento avanzado utilizado en una central TXE4A.



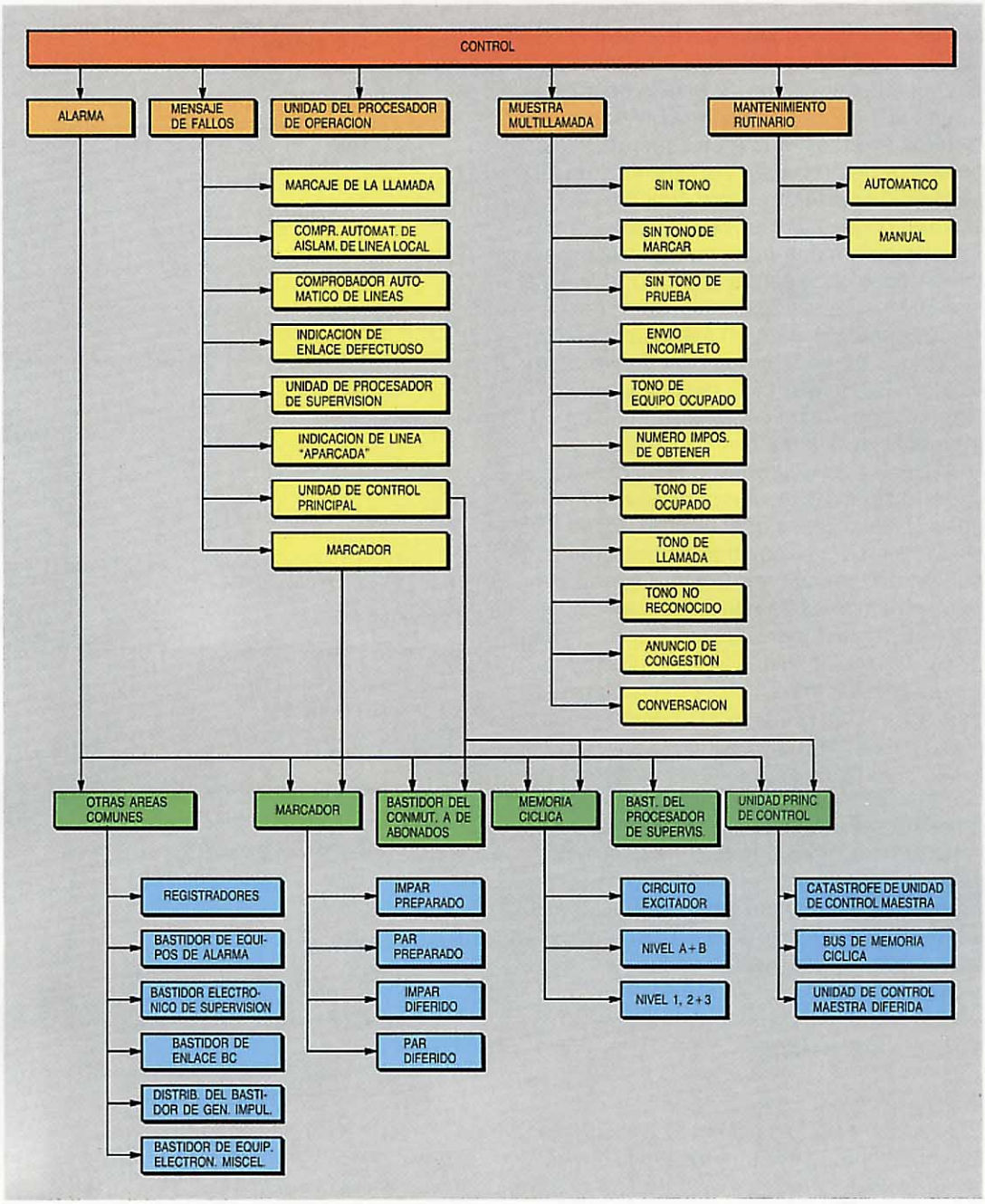


Figura 3
Esquema de la base de conocimientos del sistema experto de diagnóstico de centrales TXE4A. Los bloques representan los conocimientos almacenados en el sistema, mientras que las líneas muestran las vías de acceso que unen estas áreas de conocimiento.

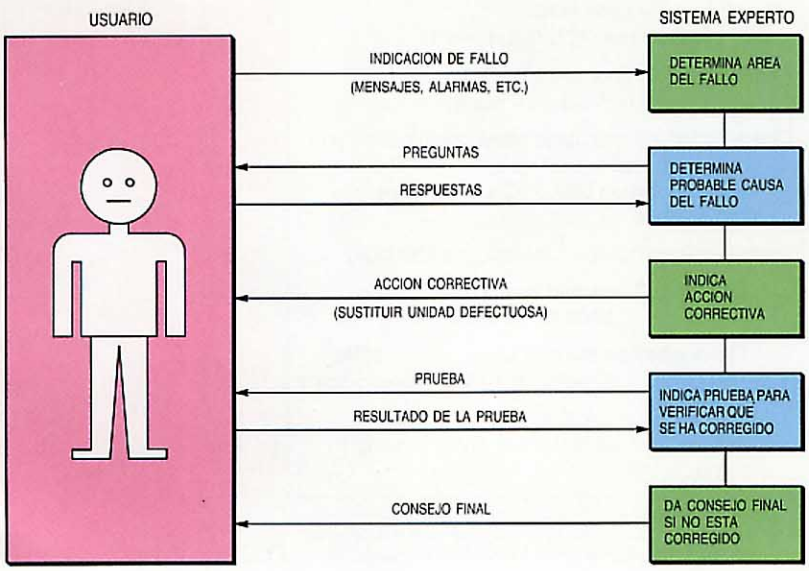


Figura 4 (abajo)
Procedimiento de consulta que indica cómo guía al usuario el sistema experto a través de la secuencia de diagnóstico.

primer intento, y alrededor del 80% de éstos pueden eliminarse utilizando el modo automático.

No es preciso ser operador de ordenadores para usar el sistema AMF, ya que todas las interacciones están controladas por el sistema experto. No obstante, un usuario más experimentado puede tomar el control y concentrarse directamente en áreas específicas de investigación. El usuario puede cuestionar el razonamiento del sistema y obtener una explicación de por qué hace una pregunta concreta o por qué ha llegado a una conclusión determinada. La figura 4 muestra una consulta típica que comienza solicitando al usuario que introduzca la indicación de fallo de la central, y

que continúa con una serie de preguntas y respuestas hasta conducirle a la localización del fallo, normalmente a nivel de unidad enchufable específica. Si el fallo no aparenta estar en una unidad enchufable, el sistema aconseja qué área concreta de la central debe probarse: por ejemplo, una sección de cableado. Normalmente, las preguntas que hace el sistema obligan al ingeniero de mantenimiento a realizar ciertas pruebas en la central y a introducir en el sistema los resultados, o bien datos específicos contenidos en un mensaje de fallo. La tabla 2 muestra una secuencia típica de diálogo. Existe una facilidad de ayuda en línea que puede utilizarse en cualquier momento de la consulta.

Al llegar a un diagnóstico final, el sistema experto señala la acción correctiva y recomienda las pruebas que confirmarán el diagnóstico. Para registrar la operación se puede utilizar una impresora, y los resultados sirven como base de datos para actualizar el sistema experto o para el análisis de fallos. Corregido el fallo, los resultados de las pruebas se introducen en el sistema, que indica "fallo eliminado" o sugiere más acciones en caso necesario.

De esta manera y mediante este sencillo interfaz de usuario, toda la información que en el método tradicional era tan desconcertante se presenta ahora de modo eficiente y controlado, casi sin necesidad de papeles. Además, el funcionamiento manual ofrece un excelente método para adiestrar a los ingenieros en diagnóstico de fallos.

Eficacia del sistema

El desarrollo del sistema experto se terminó antes de plazo y dentro de lo presupuestado, y fue introducido para la puesta en servicio de centrales en mayo de 1984. Aunque el objetivo marcado para el sistema era localizar y eliminar el 70% de los fallos de la central, el producto final es capaz, de hecho, de corregir el 85% de los fallos.

En el modo de funcionamiento automático, la mayoría de los fallos de conmutación se identifican a nivel de componente, en cuestión de minutos, frente a las varias horas que a menudo necesitaba un ingeniero sin experiencia con los métodos tradicionales. Por ejemplo, en la tabla 3 se compara la eficacia en el caso de un punto de cruce defectuoso para el que se han dado 150 mensajes de fallo. Mientras que el método convencional requiere un ingeniero experimentado, el AMF puede ser empleado con eficacia por un ingeniero con menos experiencia. Utilizando sistemas clásicos de diagnóstico, un ingeniero de esas características puede tardar hasta tres

Tabla 2 — Ejemplo de consulta

TXE4A Fault Diagnosis
Consultation started at 10:29:44 on 17 Feb 86

Information is available on a number of facilities in a help file. These facilities may be listed by typing 'HELP INDEX'

What is the indication of the fault?

1. Fault printout
2. Alarms
3. Operational processing unit
4. intermittent fault

Make a choice: 1

Which type of fault printout would you like to consider?

1 MCU
2 MKR
3 RTR
4 SPU
5 PLI
6 FJI
7 LLIR
8 CT
9 End consultation.

Please reply: 1

Considering MCU fault printouts.

The following is a list of fault codes and the occurrences of each:
FC 10 == 2: FC 11 == 23: FC 13 == 21:
FC 17 == 2: FC 81 == 2: FC 82 == 2: FC 99 == 2:

Which one of the above FCs do you wish to consider? FC 11

Select the type of analysis:

1. Automatic
2. Manual

Make a choice: 1

Fault code 11 ... Path fail first attempt.
Failure to set a chosen free path by the markers.
Chosen switching network detail is given in the FPO.
MEV digit gives the STS paths being set.
(See 32QMP00001AAR Sht 10.1)

Checking for a common EN.
No common EN found.

Checking for a common MUC.
Common MUC found == 121

Checking for common POC.
Common POC found == 1

Checking for common C switch.
Common C switch found == 1

Checking for common path, plane combination, and link type
Common path, P/C, and link type found == 1, 1, 2

Analysis shows that the C switch could be faulty.

The faulty C switch is located on MKR 12 1/2 Rack.

The position on the rack is: Shelf C, SIU 4AAB Appearance 1

Change the above C switch.
Refer to AMF User Manual, Appendices E, F and G.

Se indican en letra negrilla los lugares en los que el usuario ha de elegir. Las salidas del sistema AMF identificando el tipo de fallo y su situación se muestran en letra cursiva más intensa.

Tabla 3 — Comparación de los tiempos de eliminación del fallo para un punto de cruce defectuoso en un conmutador C

Sistema experto de diagnóstico (ingeniero sin experiencia)		Sistema convencional de diagnóstico (ingeniero experimentado)	
Tareas	Tiempo (min.)	Tareas	Tiempo (min.)
Tiempo total de consulta, incluyendo análisis automático del mensaje de fallo y diagnosis del fallo a nivel de unidad	5,5	Análisis del mensaje de fallo	15
Localización y cambio de la unidad defectuosa (posición del bastidor dada por el AMF)	10	Análisis de mensajes marcados para localizar el bastidor y la unidad sospechosos (incluyendo el uso de diagramas)	10
Prueba de la unidad y localización del punto de cruce defectuoso (posición dada por el AMF)	5	Localización y cambio de la unidad defectuosa	10
		Prueba de la unidad y localización del punto de cruce defectuoso (incluyendo uso de diagramas)	10
TOTAL	20,5	TOTAL	45

Tabla 4 — Comparación de los tiempos de eliminación del fallo para una alarma diferida de marcador

Sistema experto de diagnóstico (ingeniero sin experiencia; la localización de los puntos de prueba y de los puntos-U la da el sistema experto)		Sistema convencional de diagnóstico (ingeniero experimentado; la localización de los puntos de prueba y de los puntos-U debe deducirse de diagramas)	
Tareas	Tiempo (min.)	Tareas	Tiempo (min.)
Tiempo de consulta sin prueba física	4,5	Comprobación de la unidad de alarmas	8
Comprobación de la unidad de alarmas (se dan los puntos de prueba)	5	Comprobación de la frequencia de alarma del marcador	3
Comprobación de la frecuencia de alarma del marcador	3	Localización del diagrama necesario	5
Comprobaciones de los puntos de prueba y de los puntos-U de la unidad 2BEV (posiciones dadas por el sistema experto)	5	Comprobaciones de las señales y de los puntos-U de la unidad 2BEV, utilizando diagramas	10
Comprobaciones de los puntos-U de la unidad 2BEY	5	Comprobaciones de las señales y de los puntos-U de la unidad 2BEY, utilizando diagramas	10
Comprobaciones de los puntos-U de la unidad 2BEF	5	Comprobaciones de las señales y de los puntos-U de la unidad 2BEF, utilizando diagramas	10
Cambio de la unidad y prueba	5	Análisis de la localización del fallo	10
		Cambio de la unidad y prueba	5
TOTAL	32,5	TOTAL	61

horas en localizar el fallo. La tabla 4 ofrece una comparación similar para el caso de una alarma diferida del marcador dentro de la central. En este caso se aplican observaciones análogas.

La utilización de este sistema experto para el diagnóstico de fallos entraña una serie de ventajas:

- el tiempo medio necesario para eliminar los fallos y reponer en servicio el equipo se reduce en la tercera parte, teniendo en cuenta los fallos no cubiertos por el equipo AMF y los que deben localizarse utilizando el modo interactivo
- la cantidad de documentación necesaria en cada central puede descender aproximadamente en el 75%
- se necesita menos capacitación sistemática, puesto que el sistema experto ofrece una formación constante "durante el trabajo"
- pueden reducirse los costes de operación
- se reduce el tiempo medio de reparación
- aumenta el tiempo medio entre fallos, ya que el equipo de la central sufre menos perturbaciones al estar la eliminación de fallos dirigida con más exactitud y con menos tanteos
- se mejora la disponibilidad, volviendo a generar beneficios el equipo con la mayor rapidez posible
- la localización de errores ya no tiene carácter de tentativa, sino que utiliza siempre la solución óptima.

Para comprobar si se conseguían en la realidad tales ventajas, el sistema fue sometido a extensas pruebas de campo en dos centrales TXE4A.

Pruebas de campo

Dos sistemas expertos de diagnóstico se probaron en campo durante tres meses, a partir de abril de 1985, en centrales TXE4A situadas en Londres y en Kent. Ambas centrales, elegidas por el British Telecom, estaban en servicio y habían estado cursando tráfico real durante varios meses. Las figuras 5 a 7 muestran los resultados medidos de la mejora en calidad.

Al comienzo de la prueba, los mensajes de fallo se producían a razón de unos 2.500 por día, cifra típica durante el periodo anterior. (Debe hacerse notar que cada vez que se accede a una pieza de equipo defectuoso se produce un informe de fallo, de forma que un solo fallo sin corregir puede producir

Figura 5
Reducción del número de mensajes de fallo en una central de prueba como resultado de introducir el AMF.

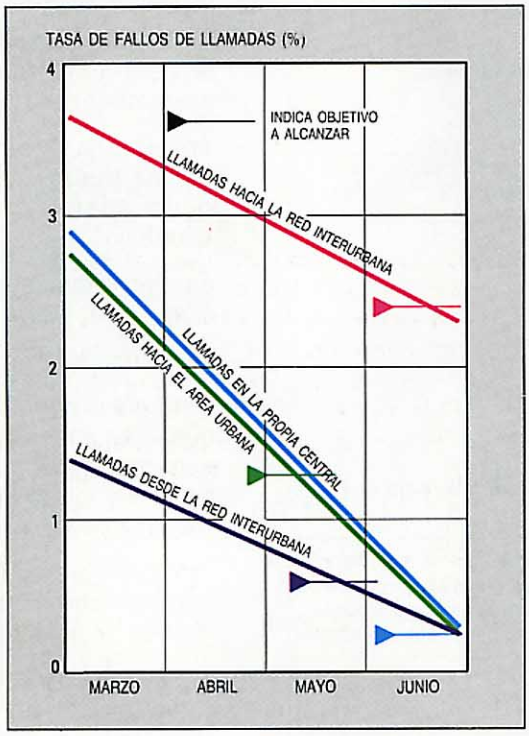


Figura 6
Reducción en la tasa de fallos de llamadas en la central de pruebas de Londres tras la introducción del sistema experto de diagnóstico. Se muestran también los valores fijados como objetivo.

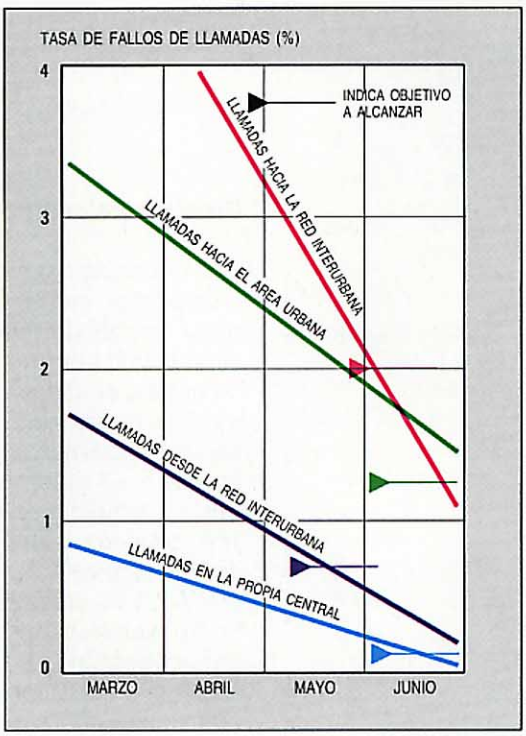


Figura 7
Reducción en la tasa de fallos de llamadas en la central de pruebas de Kent, tras la introducción del sistema experto de diagnóstico. Se indican también los valores tomados como objetivo.

centenares de informes al día). Aunque esta proporción de mensajes de fallo no sea característica de una central TXE4A, sin embargo no es raro encontrarla entre las 550 centrales TXE4A ya instaladas. Durante el periodo de prueba de tres meses, la tasa de mensajes diarios descendió hasta 100, una importante reducción en la proporción 25 a 1 que fue notada por el sistema de detección del propio equipo. Debe resaltarse que la figura 5 no está representada en escala logarítmica ni con el eje de abscisas desplazado. El resultado es realmente tan bueno como parece.

Como comprobación adicional de la calidad del servicio, se utilizó el sistema MAC (*Measurements and Analysis Center*, centro de medidas y análisis) para generar y observar llamadas de prueba durante la experiencia de campo, tal y como presenta la tabla 5. El sistema MAC proporciona una medida significativa de la calidad, tanto desde el punto de vista de la Administración como de los usuarios, ya que se basa en llamadas de prueba desde circuitos de línea de reserva a los números de prueba a través de la red pública conmutada. De esta manera, las llamadas no completadas por línea ocupada, número no obtenible y otras causas, dan una medida real del grado en que el usuario está satisfecho del funcionamiento del sistema de conmutación.

La figura 6 muestra las secuencias de medición (MS) 1 a 4, con valores reales comparados a los objetivos, de la central de Londres. MS1, que es el porcentaje de fallos de llamadas hacia la propia central, era de 2,9 al comienzo de la prueba y al cabo de tres meses se había reducido a 0,31, muy próximo al objetivo de 0,25. MS2, la tasa de fallos de llamadas hacia el área urbana, cayó del 2,76 al 0,23, aunque el objetivo era sólo 1,3. MS3, que es el porcentaje de fallos de llamadas automáticas hacia la red interurbana, bajó del 3,65 al 2,3, también por debajo del objetivo 2,4. MS4, el fallo porcentual de llamadas automáticas procedentes de la red interurbana, mejoró también desde el 1,4 hasta el 0,23, muy por debajo del objetivo 0,6. La figura 7 muestra mejoras similares en la eficacia de la central de Kent.

En octubre de 1985 se ofreció una demostración adicional de las posibilidades del sistema AMF a los jefes de distrito de British Telecom, en una central que se suponía "limpia". Durante las cuatro semanas, se eliminaron 13 fallos con ayuda del AMF, 8 de los cuales habían pasado totalmente desapercibidos, incluyendo varios cortocircuitos y componentes defectuosos. Tras estas pruebas, se perfeccionó el sistema AMF merced a informaciones recibidas de British Telecom.

Tabla 5 — Secuencias de medición

Número de la secuencia	Descripción de la secuencia		Llamadas por mes
1	Propia central	Desde los circuitos de líneas de reserva de British Telecom a los números de prueba dentro del múltiple de la misma central	1000
2	Area urbana	Desde los circuitos de líneas de reserva de British Telecom a los números de prueba en el múltiple de centrales carentes de acceso a la red automática interurbana	480
3	Red interurbana	Desde los circuitos de líneas de reserva de British Telecom a los números de prueba de la central distante conseguidos a través de la red automática interurbana	280
4	Procedentes de la red interurbana	Desde los circuitos de líneas de reserva de British Telecom a los números de prueba en el centro de conmutación de grupo de la central local	480

Conclusiones

El sistema experto de diagnóstico para centrales TXE4A ha demostrado ya su capacidad de reducir en un tercio el tiempo normal de puesta en servicio. Ha quedado comprobado que pueden bajar notablemente los tiempos necesarios para el mantenimiento mediante instalaciones basadas en este sistema experto de diagnóstico, cuya centralización puede plantearse.

British Telecom y STC han acometido un programa conjunto de mejoras de la central TXE4A que hacia 1989 permitirá el acceso a la red digital de las centrales TXE4A actualmente existentes. La versión potenciada proporcionará una amplia gama de nuevas facilidades, tales como encaminamiento alternativo automático y registro de datos de llamadas, que mejorarán las prestaciones tanto para la Administración como para los abonados. Se está probando ya un prototipo de sistema experto de diagnóstico para potenciar centrales TXE4A, dentro de un primer contrato de fabricación.

Mo Thandasseri nació en Kerala, India, en 1941. Se graduó ingeniero electrónico y eléctrico en 1963 por el Trichur Engineering College de la Universidad de Kerala, trabajando después como ingeniero electrónico en la Universidad de Londres, en el desarrollo de la primera máquina artificial británica de diálisis en casa. En 1968 ingresó en GEC Telecommunications para trabajar en las centrales TXE2, antes de ingresar en STC en 1970. El Sr. Thandasseri es actualmente jefe de sistemas aplicados en la División de Transmisión Local de STC Telecommunications Ltd. Posee un diploma en contabilidad y finanzas, es ingeniero colegiado, y miembro del Institute of Electrical Engineering del Reino Unido. En 1985 recibió el Premio Corfield a la innovación por el desarrollo del sistema experto para las centrales TXE4A. Es también el representante del IEE en STC Telecommunications.

Encaminamiento por inteligencia artificial en redes de paquetes vía radio

Se pueden emplear técnicas de inteligencia artificial para realizar una red radio de paquetes con control de encaminamiento distribuido. Cada nodo extrae los datos para establecer conexiones de los encabezamientos de mensaje, y puede así construir una base de conocimientos de la cual se determine el encaminamiento óptimo para cada mensaje, y al mismo tiempo se reduzca la probabilidad de ser detectado el nodo.

P. Benson
ITT Defense Communications Division, San Diego, California, Estados Unidos de América

Introducción

Los sistemas de comunicaciones tácticas prestan servicios similares a los de los sistemas comerciales, pero en un ambiente mucho más hostil: los nodos se mueven, alterándose los enlaces con otros nodos; los enlaces se degradan por sobrecarga de tráfico y perturbaciones deliberadas; se pueden perder nodos a consecuencia de acciones del enemigo. Todos estos factores reclaman sistemas de comunicaciones capaces de modificar con rapidez sus conexiones.

En un entorno de este género, la comunicación ha de ser eficiente y capaz de sobrevivir a la pérdida de nodos, entorpeciendo además la localización por el enemigo de la posición del emisor. Las comunicaciones militares utilizan enlaces de radio con conmutación de paquetes para el envío de información en operaciones tácticas, ya que este método se acomoda bien a la transmisión en ráfagas típica de estas comunicaciones. Los enlaces de radio facilitan la movilidad de la red y la rápida respuesta a cambios de situación táctica.

La figura 1 ilustra varias formas de encaminar paquetes en una red de radio móvil hacia sus destinos respectivos, a través de nodos llamados repetidores. Se utilizan como "exploradores" paquetes especiales — los datagramas — que determinan el encaminamiento en el instante de la transmisión. Cuando no se conoce encaminamiento de punto a punto, se envía una emisión generalizada (difusión) que inunde la red con datagramas, y los acuses de recibo a los mismos en su recorrido van estableciendo circuitos virtuales por los cuales pueden circular los paquetes posteriores. No obstante, estas difusiones para determinar la vía ocupan un ancho de banda considerable y aumentan el riesgo de ser descubierto el emisor.

La movilidad de nodos en una red radio de paquetes dificulta el establecimiento de una ruta, pues la topología de la red cambia constantemente. La congestión de tráfico en nodos clave o la interferencia deliberada en situaciones tácticas pueden también alterar de manera dinámica dicha topología. La gestión de tales cambios requiere potentes algoritmos que permitan un encaminamiento adaptativo. En aplicaciones militares es esencial que las funciones de gestión no ocupen una gran parte de la banda de transporte de mensajes; por ello este tipo de encaminamiento debe restringir mucho la clase y cantidad de información enviada para mantener las bases de datos locales sobre posibilidades de conexión. Aplicando a este problema sistemas expertos — una rama de la inteligencia artificial — se puede reducir el tráfico administrativo así como la probabilidad de detección, mediante inferencia de las rutas existentes a partir de información actualmente no utilizada.

Las redes radio de paquetes permiten conexiones entre nodos alejados por retransmisión de mensajes llegados de nodos vecinos. La información de encaminamiento (nodos conectados) se establece al constituir la red. Se pueden tratar de varias maneras los cambios en la topología de red: por medio de "estaciones", por algoritmos convencionales de encaminamiento, o aplicando un sistema experto.

El primer modo de estudiar la accesibilidad en la red se basa en un conjunto de uno

o más nodos de control denominados estaciones[1]. En sistemas normales de este tipo, la accesibilidad y el encaminamiento se determinan mediante programación convencional residente en dichas estaciones. Estas "etiquetan" a los nodos situados en su entorno inmediato, es decir, les facilitan los caminos necesarios para dirigir un mensaje que será retransmitido a través de la red. La pérdida o el fallo de una estación provoca pérdida del servicio hasta que otra estación pueda etiquetar los nodos afectados y se sustituya la función de la estación original. Como último recurso, si no hay estaciones disponibles, se adopta en ese segmento de red un modo de funcionar sin estaciones, en el cual cada radio trabaja en difusión para establecer una ruta hacia un destino concreto. Esto se denomina búsqueda por inundación.

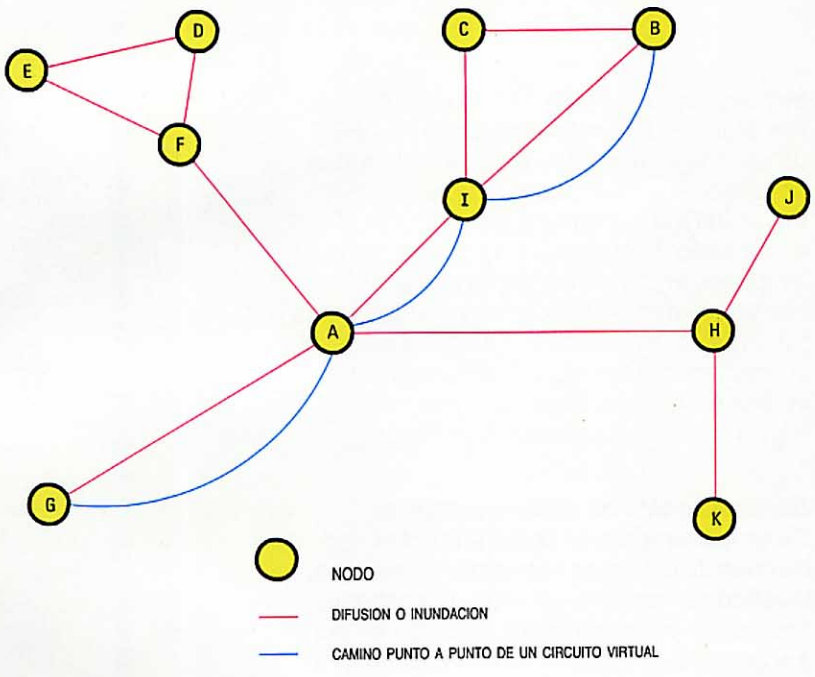


Figura 1
Encaminamiento de paquetes en una red de radio móvil.

Un protocolo de encaminamiento más moderno se basa en que las estaciones mantengan cierta información de la red local[2], obtenida mediante mensajes ROP (paquetes de radio activada) difundidos a los vecinos a intervalos de 5 a 30 s. Sin embargo, en ambientes hostiles estos ROP pueden provocar la localización de nodos ocultos.

ITT DCD (División de Comunicaciones para la Defensa de ITT) ha desarrollado un sistema experto capaz de reemplazar las funciones de la estación por un control local, consiguiendo así un control de red distribuido. El sistema experto deduce la información sobre la red y para la selección de caminos, y en esto aventaja a las técnicas de programación convencionales con algoritmos fijos, pues aprovecha la flexibilidad y potencia de la inteligencia artificial para integrar la información y para inferir procedimientos de búsqueda heurísticos (reglas prácticas). El sistema de control de red distribuido se llama NCAI (*Network Control using Artificial Intelligence*, control de red por inteligencia artificial).

Diseño del sistema

Los mensajes de la red radio de paquetes se componen de encabezamiento, que contiene la información de encaminamiento del paquete, y de cuerpo. En dicho encabezamiento se puede indicar a través de qué nodos ha pasado el mensaje, así como la información de direccionamiento a través de los nodos sucesivos. Cuando un nodo recibe un paquete, lee su encabezamiento y lo retransmite al siguiente nodo apropiado. Las conexiones entre nodos se llaman enlaces; un camino comprende una secuencia de enlaces desde el origen al destino.

En el encabezamiento del paquete se inserta la lista de los nodos utilizados como repetidores junto con la calidad de los enlaces entre tales nodos, lo cual puede considerarse como la historia de la conexión. Se indican así tanto los enlaces que unen un subconjunto de nodos como su calidad, mesurable como relación señal-ruido, tasa de errores de bitios o potencia recibida. Si esta información se recopila en una base de datos local, podrán deducirse de ella las posibles conexiones en la red y el encaminamiento de paquetes se reducirá a una sencilla búsqueda[3]. Para establecer y ordenar un conjunto de caminos se aplican ciertos criterios, tales como la mínima tasa de error acumulada por enlace, la menor longitud del camino, o el mínimo retardo de propagación sobre el camino completo, dependiendo de los objetivos marcados por el diseñador de la red.

Valoración basada en reglas de las posibilidades de encaminamiento

ITT DCD ha diseñado un sistema basado en reglas para implantar estos conceptos básicos, según el cual se reciben los encabezamientos y de ellos se extrae la información. Los caminos y enlaces entre nodos se almacenan como *elementos de memoria de trabajo* dentro de un programa OPS5[4]. Se aplican reglas para elegir caminos, buscar caminos nuevos e integrar nueva información sobre enlaces en la base de conocimientos existente. Cuando se reciben o

retransmiten mensajes en un nodo, éste es capaz de evaluar la calidad del enlace por la relación señal-ruido o por la tasa de errores de la señal recibida. Se desechan los enlaces defectuosos, se actualizan los enlaces variables (p. ej., los que unen nodos desplazados, con carga de tráfico alterada o con nuevas interferencias), y la calidad de los caminos se reevalúa en base a las historias de los mismos recibidas en los mensajes. Como siempre, los criterios para juzgar la calidad del enlace dependen de los objetivos del diseñador de la red.

Para tales fines hay varios algoritmos aceptables, basados en buscar el camino más corto; se implantó uno de ellos, de búsqueda transversal con limitaciones dinámicas en programación, mediante reglas y una cola de espera formada por elementos de la memoria de trabajo. En esta cola se colocan todos los nodos al alcance de un solo tramo. Cuando uno de ellos es el nodo de destino, ahí termina la búsqueda; en los demás casos el primer nodo de la cola se expansiona (es decir, se sitúan al final de la cola todos los nodos vecinos a distancia de un tramo, con la restricción de que no se repita ningún nodo). Después de tal expansión, se elimina el nodo de la cola y se le pasa a una lista de nodos utilizados, prosiguiéndose el proceso de expansión hasta llegar al destino o vaciarse la cola. Este procedimiento es muy operativo, ya que los enlaces cuya información posee el nodo suelen estructurarse aproximadamente en estrella, con el nodo en el centro (Fig. 2). Si se inicia una búsqueda de camino en el nodo A para llegar al B, el factor de ramificación para la búsqueda con prioridad transversal produce una explosión combinatoria. Sin embargo, si se comienza en el nodo B, el referido factor es menor y la búsqueda termina rápidamente por estar los nodos distantes en la periferia y tener el nodo A poca información sobre ellos.

Este procedimiento de búsqueda no necesita transcribirse en reglas; la programación convencional será también eficaz y rápida. Sin embargo, el uso de reglas ofrece las grandes ventajas de facilidad de cambio y modularidad del conocimiento. Se añaden con facilidad reglas sin nuevos diseños ni codificaciones sustanciales. Así, si un diseñador de redes selecciona un nuevo criterio de propagación, como sería uno de los métodos de ordenación de caminos ya mencionados, éste podrá incorporarse directamente. Tal flexibilidad facilita una rápida modelación del nuevo módulo de encaminamiento. La modularidad del conocimiento es también importante; mediante sistemas expertos puede representarse sucintamente el conocimiento en unas pocas reglas, en lugar de desparramarse por todo el módulo. La localización del conocimiento hace más comprensible el programa y facilita su mantenimiento.

Otro aspecto de la solución por reglas es que obedece a los datos: las reglas se aplican en un orden que depende de los datos vigentes. Los datos se representan como elementos complejos en una estructura de programación OPS5 denominada memoria de trabajo. Si bien existen varias

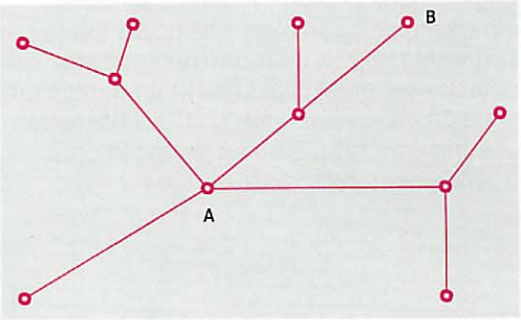


Figura 2
Ejemplo de enlaces conocidos por el nodo A.

estrategias para determinar qué regla deba ejecutarse en un momento dado, la OPS5 utiliza una táctica basada en la antigüedad de creación o modificación de los elementos, unida a la concreción de las reglas. (Por el contrario, la estructura de control de los lenguajes *imperativos*, como el Pascal, es fija, y su orden de aplicación es mucho más rígido). Esta flexibilidad estructural permite al diseñador crear reglas válidas en condiciones generales, dejando al programa que elija las más adecuadas a la situación.

Bases de datos de encaminamiento

Conceptualmente se consideran dos tipos de base de datos: de caminos y de enlaces. Cuando se trata una petición de encaminamiento, se investiga entre los caminos para encontrar el óptimo. Si no se halla ninguno adecuado, se busca en la base de datos de enlaces una secuencia de ellos que conecte el origen y el destino. Si tampoco de esto puede deducirse un camino, el nodo puede pedir a los nodos vecinos que lo busquen, bien de forma activa mediante petición directa, o pasiva solicitando una búsqueda por inundación.

Un nodo vecino que reciba un paquete difundido, puede redirigirlo añadiendo información de su propia base de datos. El nodo origen se enterará del camino seguido por este paquete, pues casi con seguridad estará todavía dentro del alcance radio del nodo repetidor y podrá escuchar sus transmisiones. De modo más directo, el originante puede enviar a otro nodo una petición concreta de información, por ejemplo, las

posibilidades de encaminamiento en una región. La información recibida del vecino podrá ser entonces introducida en la base de datos local del nodo de origen para establecer un nuevo plan de encaminamiento. Con este procedimiento un nuevo nodo incorporado a una red puede crear rápidamente su base de datos local. Como último recurso, cabría la búsqueda por difusión o inundación, necesaria en una red convencional de paquetes por radio carente de estaciones.

Sensibilidad a las condiciones de red

El camino óptimo en cada caso varía con las condiciones de la red (tráfico, interferencias), que también influyen en las reglas de encaminamiento por NCAI. Al observar el tráfico, se evalúan las condiciones actuales de la red, y se modifican de acuerdo con ellas las reglas de selección de camino y enlace. En la simulación de red de paquetes via radio, estas estimaciones se hicieron sobre una base apropiada; cuando el número de mensajes cursado por un nodo superaba un límite establecido, se disparaba un indicador. Está proyectado evaluar las condiciones de red, así como preparar medidas adecuadas de respuesta. Tales mejoras servirán de base para simular cargas de tráfico y condiciones variables del entorno.

Simulación

Se decidió la simulación de la red de paquetes por radio con el fin de demostrar la posibilidad de utilizar técnicas de sistemas expertos para su control. Para ello se utilizó el FranzLisp, simulando una red de radios capaces de enviar, recibir y retransmitir mensajes de un modo probabilista: es decir, la intensidad de la señal varía en función de la distancia del enlace y de su calidad, también variable. El encabezamiento de un mensaje contiene su historia de encaminamiento y las calidades de los enlaces utilizados. Cada nodo puede encaminar mensajes empleando un sistema OPS5 basado en reglas, que ejecuta dentro del FranzLisp, y en él también se extrae información de los encabezamientos recibidos, que se integra en las bases de conocimiento. Se encuentran nuevos caminos, ya sea por búsqueda simple en la base de conocimientos o mediante una búsqueda con prioridad transversal entre los enlaces. Otras técnicas de búsqueda basadas en heurística son fácilmente realizables en este esquema. Como el proyecto pretende ante todo probar la utilización de sistemas expertos en el control de red, es importante el poder variar las técnicas de búsqueda o realizar una heurística determinada, ya que ello permite la prueba rápida de nuevos conceptos de encaminamiento.

Se creó una red experimental de 29 radios simuladas. Durante la iniciación de la simulación, se emitieron series de mensajes de búsqueda por inundación desde puntos aleatorios de la red. Al cabo de 10 de estas series, se observó que los nodos centrales habían recibido información del 90% de los nodos de la red.

La figura 3 muestra las relaciones entre la simulación de red y el módulo de encaminamiento. El controlador de mensajes de cada nodo registra la información de cada encabezamiento de mensaje. Un generador estocástico ordena la transmisión de mensajes de un nodo a otro, lo cual requiere una asignación de camino. La entrada y salida de información del módulo de encaminamiento, escrito en OPS5, se hace por medio del controlador de simulación Lisp.

La porción OPS5 del programa está separada de la simulación en sí, con la que se comunica mediante llamadas de entrada y salida normales. El tipo exacto de cada llamada depende de que el módulo se esté ejecutando como proceso separado en un miniordenador o como programa independiente en un ITT XTRA*, ordenador perso-

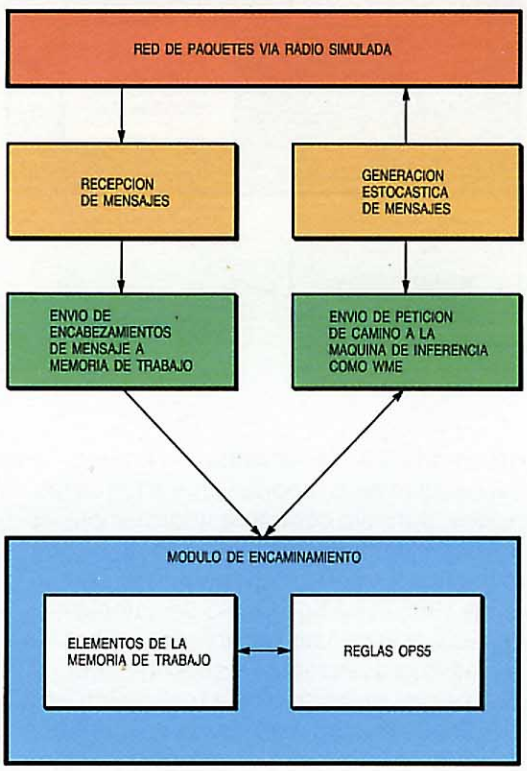


Figura 3
Diagrama de bloques de una simulación de red de paquetes vía radio, módulo de encaminamiento por reglas y funciones de entrada salida.
WME - elemento de memoria de trabajo.

* Marca registrada del Sistema ITT

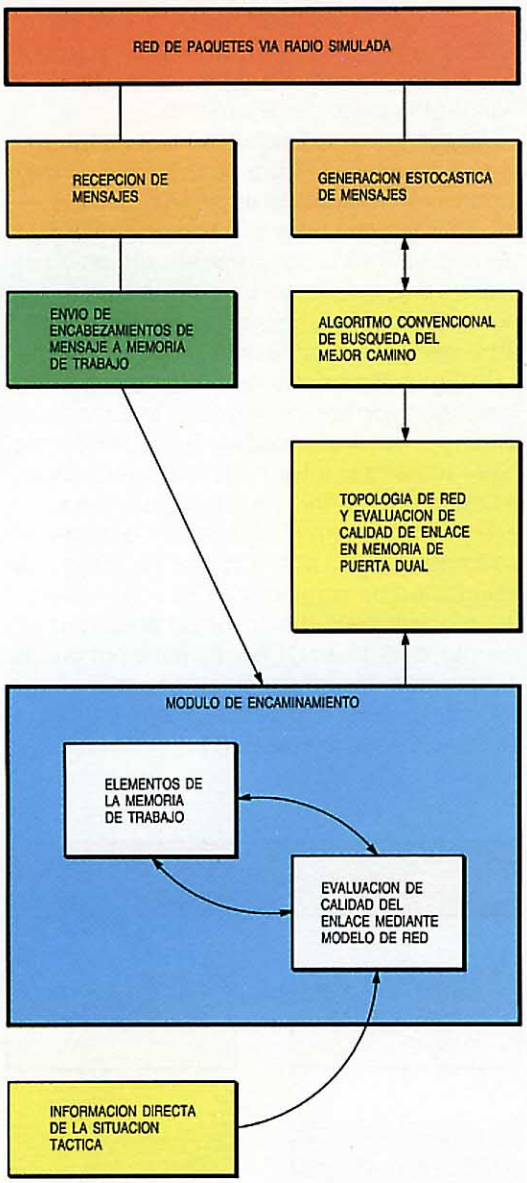


Figura 4
Diagrama de bloques de una simulación de red de paquetes vía radio empleando un sistema híbrido de algoritmos convencionales y una estimación de enlaces mediante reglas.

nal estándar en la industria. En ambos casos, el mismo módulo sirve a todos los nodos, pero sin compartir información. Así, el encaminamiento se realiza como si todos los nodos fuesen independientes.

En 1985 el código OPS5 se cambió a OPS83, que constituye un sistema por reglas más avanzado. Pese a no haber diferencias en principio, la realización en OPS83 es mucho más rápida ya que puede codificar los efectos de las reglas (la cláusula "then") como una llamada a procedimiento, de manera similar a los lenguajes PASCAL y C.

En principio, el sistema se insertó en una radio de paquetes a través de una memoria de puerta dual (Fig. 4). Dicha radio almacena su tabla de encaminamiento en una zona de la memoria accesible desde la misma radio y desde el NCAI. La nueva información obtenida de los encabezamientos de mensaje se sitúa también en esta memoria.

Trabajando en modo asíncrono y en paralelo, el NCAI deduce nuevos caminos y encaminamientos de la información antes referida, los cuales actualizados se inscribirán en la memoria de puerta dual de manera transparente para la radio propiamente dicha.

Operación del sistema

La secuencia operacional en una red de paquetes controlada por NCAI comienza con un sistema de radios que crece gradualmente a partir de un número pequeño, ubicadas en el campo dentro de una situación táctica. Por este concepto, todas las radios de paquetes con encaminamiento asistido por inteligencia artificial (PRA, *packet radios with AI-assisted routing*) se las supone emplazadas sobre vehículos móviles, sin estaciones (o nodos de control). Así, en el momento inicial, en la base de conocimientos de una PRA habrá muy poca información de encaminamiento. Pese a no estar previsto en la secuencia operacional, se puede contactar con instalaciones fijas empleando una PRA.

Las comunicaciones iniciales consistirán en una difusión de paquetes a los vecinos cuyas PRA estén al alcance de un solo tramo. A medida que se transmiten los paquetes, comienza a formarse la base de datos, y al añadirse más PRA a la red, entra en operación el modo de retransmisión. El sistema NCAI de cada PRA extrae información de los encabezamientos del tráfico que retransmite, a fin de construir su propia base de conocimientos a partir de las diversas secuencias de encaminamiento. Cuando la red alcance su total operatividad, estas bases de conocimientos individuales podrán encaminar tráfico a casi todos los destinos posibles de la red. Los resultados de la simulación indican que una red activa llega a adquirir buenas secuencias de encaminamiento en la base de conocimientos de cada PRA, tras recibir sólo un pequeño número de mensajes: como antes se ha indicado, bastan 10 mensajes para que un nodo en posición central acumule rutas hacia el 90% de los nodos restantes.

El sistema NCAI en el lado de recepción continúa escuchando, extrayendo información cada vez que se retransmite un paquete (o se oye a un vecino) para mantener la base de conocimientos en tiempo real y presentar secuencias de encaminamiento utilizables a su PRA. Esta concepción permite el encaminamiento distribuido como se expone a continuación.

Si la base de conocimientos dispone de los datos de encaminamiento necesarios, la PRA puede originar un paquete completo

cuyo encabezamiento indique la secuencia sugerida hacia el destino, y que se desplace de PRA a PRA sin necesidad de búsqueda general. Además, cada PRA puede modificar el encaminamiento a medida que el paquete avanza y llega a una PRA con mejor información de encaminamiento hacia el destino.

Si la PRA de origen no posee información suficiente para suministrar una secuencia de encaminamento completa hacia el destino, se puede transmitir una secuencia incompleta, que posteriores PRA van completando conforme el paquete se desplaza sobre el camino. Se asegura así un encaminamiento realmente distribuido a través de la red. Puede además efectuarse un segundo o tercer intento de encaminamiento, si el primero falla a causa de haber empeorado el enlace hasta el punto de no admitir los paquetes.

Conclusiones

El esquema de encaminamiento por inteligencia artificial desarrollado por ITT DCD permite eliminar todos los nodos de control, admitiendo el modo de operar en la red sin estaciones como forma normal de trabajo, y no como un modo degradado de funcionamiento. La solución de búsqueda por difusión se mantiene como último recurso. El sistema experto utilizado en el NCAI permite que cada nodo de la red radio determine su mejor estimación de la vía a seguir, haciendo para ello uso de la información de su base de conocimientos que ha adquirido, desarrollado, y mantenido en tiempo real después de haberse incorporado a la red. El modo de difusión raramente se utiliza para obtener datos de encaminamiento; en su lugar, se forma una base de conocimientos obteniendo indicadores e información de encaminamiento durante la transmisión normal de paquetes en la red.

Reconocimientos

William Lee ha aportado la mayor parte de conceptos fundamentales de radio en paquetes y ha desarrollado la idea de aplicar la inteligencia artificial al encaminamiento en este tipo de redes. Janet Haake ayudó a incorporar las partes basadas en reglas a un ordenador personal ITT XTRA.

Referencias

1 R. E. Kahn, S. A. Gronemeyer, J. Burchfield y R.Kunzelman: Advances in Packet Radio Technology: *Proceedings of the Institute of Electrical and Electronics Engineers*, noviembre 1978, volumen 66, nº 11, págs. 1468–1496.

2 J. Westcott y J. Jubin: A Distributed Routing Design for a Broadcast Environment: *Institute of Electrical and Electronics Engineers Conference on Military Communications, ILCOM'82*, octubre 1982, volumen 3, págs. 104.1–104.5, Boston, Massachusetts.

3 A. Girade y S. Hurtubise: Dynamic Routing and Call Repacking in Circuit-Switched Networks: *Institute of Electrical and Electronics Engineers Transaction on Communications*, diciembre 1983,volumen COM-31, nº 12, págs. 1290–1294; INRS Telecommun, PQ.

4 C. L. Forgy: OPS5 User's Manual, *Carnegie-Mellon University*, 1981.

Peter Benson se graduó BA en psicología y artes liberales por la Universidad John Hopkins en 1969. Después de completar un PhD en lingüística en la Universidad de California en 1974, obtuvo una beca postdoctoral en neurociencias, y luego fue adjunto de cátedra en el departamento de ciencias de la comunicación de la Universidad de Michigan. El Dr. Benson se incorporó a ITT en 1981, donde se ha dedicado a investigación en reconocimiento automático de voz, inteligencia artificial y procesamiento de lenguaje natural. Actualmente es miembro del departamento técnico de la División de Comunicaciones para la Defensa de ITT en San Diego.

Sistemas expertos aplicados a la guerra electrónica

ITT Avionics ha construido un sistema experto que realiza las funciones de identificación y asignación de recursos de una unidad de control de vigilancia para un sistema de contramedidas electrónicas. El prototipo demuestra la potencial idoneidad de las técnicas de inteligencia artificial en complejas aplicaciones de guerra electrónica.

E. Gaudry
ITT Avionics, Nutley, Nueva Jersey,
Estados Unidos de América

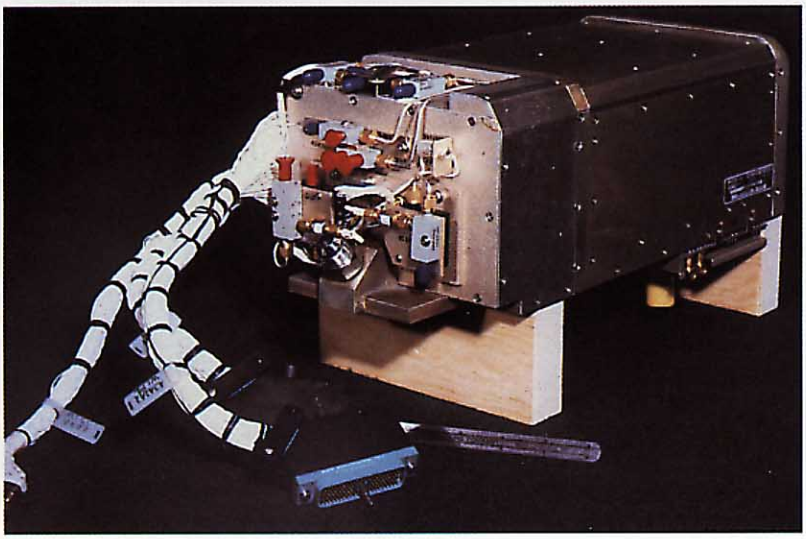
Introducción

En aviónica se entiende por ECM (contramedidas electrónicas) el uso de la electrónica para contrarrestar cualquier radar que amenace a una aeronave. Se utilizan radares ofensivos para localizar una aeronave y así poder destruirla dirigiendo armas contra ella; a esos radares tiene que interferir en tiempo real el sistema ECM de a bordo.

Diversos factores complican esta tarea. Primero, el ECM debe identificar rápidamente el radar ofensivo en un entorno de gran densidad de señales, tal vez a partir de datos incompletos o con ruido. Segundo, el entorno cambia continuamente, pues el avión se acerca a nuevos radares y se aleja de otros. Los propios radares pueden cambiar el modo de funcionamiento si comienzan a seguir a la aeronave y tal vez le disparan un misil. Incluso varían los recursos en equipo del propio sistema ECM si parte de ellos se inutilizan por fallo o ataque del enemigo. Tercero, el sistema está sujeto a estrictas limitaciones de tiempo; cada tarea debe realizarse en breves milisegundos, ya que para interferir un radar sólo se dispone de ese tiempo total de respuesta. Algunas funciones, como el seguimiento, requieren responder en microsegundos. La mayoría de los nuevos ECM funcionan automáticamente sólo por no haber tiempo suficiente para la intervención del piloto. A causa de tales limitaciones inherentes al trabajo en tiempo real y de las restricciones de tamaño propias de los sistemas en aviones, los programas para ECM de a bordo suelen implantarse en un sistema con microprocesadores "embutidos", utilizando lenguaje ensamblador.

Cada vez se exige más a los sistemas de guerra electrónica, por lo que deben examinarse nuevas tecnologías de programación para probar y satisfacer las nuevas demandas de los usuarios. Las técnicas de inteligencia artificial (IA), y en particular los sistemas expertos, ofrecen una solución innovadora para optimizar algunas aplicaciones complejas de guerra electrónica. Por ejemplo, un ECM basado en sistemas expertos sería capaz de elegir de modo "inteligente" la mejor contramedida contra un radar no identificable. En 1985, la Ingeniería de Programación de Electrónica de Defensa de ITT Avionics emprendió un proyecto de investigación y desarrollo para adquirir experiencia en herramientas de sistemas expertos y evaluar su utilidad para escribir programas de sistemas de guerra electrónica. El resultado fue un sistema experto piloto capaz de identificar radares ofensivos y asignar los limitados recursos de equipo a interferir los más peligrosos.

Vista exterior de la unidad de gestión de potencia de un sistema típico de contramedidas electrónicas.

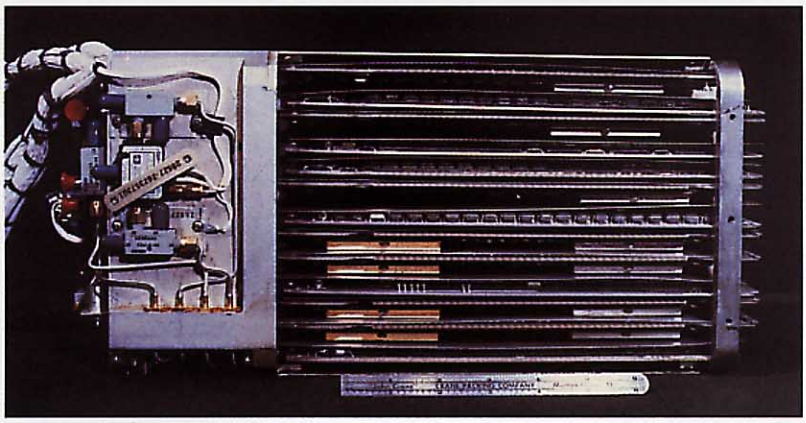


Sistema experto para control de vigilancia

Identificar la amenaza. determinar la prioridad y asignar recursos son funciones del control de vigilancia, como lo es la observación continua del entorno. El sistema experto diseñado por ITT Avionics corresponde a esta parte del sistema ECM, y actúa como simulador, respondiendo dinámicamente a los sucesos del escenario ofensivo.

La tarea

La tarea seleccionada fue desarrollar un sistema experto basado en reglas que realizara las funciones de control de vigilancia típicas de un sistema ECM, muy adecuadas para el tratamiento por reglas. Por su orientación al objeto, el control de vigilancia no es un gran "devorador de datos" como otras funciones de ECM (por ejemplo, la captación de señales), ni tampoco es tan crítico en tiempo como esta última función, pues los tiempos de ejecución de las tareas se miden aquí en milisegundos, y no en microsegundos.



Vista interna de la unidad de gestion de potencia. El control de vigilancia reside en un microprocesador "embutido" en una de las placas. El pequeño tamaño de la unidad limita severamente la memoria destinada a los programas.

El control de vigilancia es difícil de realizar y de depurar en código ensamblador, pues se trata de una función lógica formada por árboles de decisión cuyo seguimiento es arduo por sus complejas interacciones. Las técnicas de sistemas expertos son adecuadas para este tipo de problemas.

Un combate de guerra electrónica se estimula externamente por entradas/salidas de señales eléctricas y depende del entorno. El flujo de control obedece a interrupciones generadas por los mensajes procedentes del equipo físico. Esta característica de control en tiempo real, típica de un sistema ECM, contrasta con la mayoría de los sistemas expertos actuales utilizados en entornos estáticos[1].

Herramientas

ITT Avionics evaluó diversas herramientas para desarrollo de sistemas expertos. Como ordenador se eligió un ITT XTRA* con 640 k-octetos de memoria RAM, pues el menor coste de los programas en un ordenador personal permite estudiar más herramientas de programación que si se utilizara un ordenador mayor. Además, el equipo se aproxima mucho más a lo que suele haber en un sistema de guerra electrónica.

Entre las herramientas de programación examinadas había versiones del LISP y envolturas de sistemas expertos. Finalmente se seleccionó el Prolog por su especial aptitud para construir prototipos de sistemas expertos de tamaño medio[2] y por ofrecer representaciones del conocimiento, tanto lógicas como por procedimientos, que posibilitan ejercer cierto control sobre la ejecución del programa. Los conocimientos se codifican en reglas de producción que expresan inferencias lógicas en formato *IF* (*SI*) ... *THEN* (*ENTONCES*) ... Estas reglas difieren de las sentencias *IF... THEN...* de la programación convencional en que no están unidas secuencialmente, sino que su activación depende de la reunión de sucesos en la base de conocimientos. La máquina de inferencias que controla cómo se utilizan estas reglas de producción opera esencialmente en concatenación regresiva, si bien puede crearse un efecto de concatenación progresiva que permita al usuario dirigir preguntas, así como un control dirigido por el objetivo. La facultad de Prolog para adaptarse a la configuración lo hace ideal para manejar una gran base de datos con sucesos, como la que se acumularía durante la simulación de un episodio de ECM.

Se escogió una versión denominada MProlog, que posee una rica biblioteca de predicados incorporados y un entorno de desarrollo de programa provisto de un editor interactivo con facilidades de rastreo y comprobación de sintaxis. El MProlog admite un tratamiento de excepciones a múltiples niveles definido por el usuario y rutinas de entrada/salida de cierto refinamiento.

Adquisición de los conocimientos

La base de conocimientos fue desarrollada conjuntamente por un experto en el tema y un ingeniero del conocimiento. El experto, que había ayudado a diseñar el control de vigilancia ECM, aportó información sobre el sistema y el entorno ofensivo que luego utilizó el ingeniero para formular las reglas de producción y traducirlas a Prolog.

* Marca registrada del Sistema ITT

Inicialmente el experto describió las prestaciones que debía dar el sistema y las reglas heurísticas necesarias para conseguir ese comportamiento. Codificadas tales reglas por el ingeniero del conocimiento en un sistema prototipo, se comprobó su formulación interpretándolas con la máquina de inferencia Prolog. Seguidamente el experto precisó más las exigencias del sistema, sobre todo en cuanto al comportamiento sustitutivo de acciones del usuario y excepciones a las reglas generales, tras lo cual el ingeniero añadió más reglas para codificar estos nuevos conocimientos. Se volvieron a revisar las reglas hasta que el experto juzgó que el sistema se comportaba correctamente. La naturaleza del lenguaje Prolog y el que se utilizara una versión interpretada hizo relativamente fácil esta modificación iterativa de las reglas.

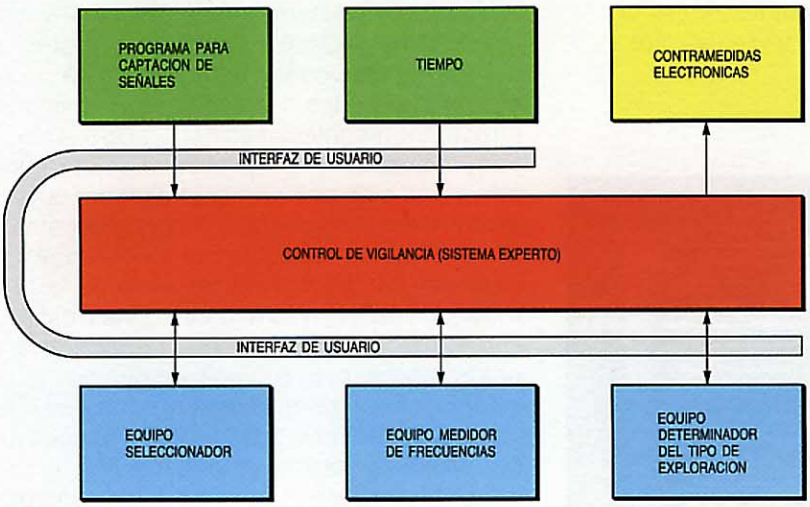


Figura 1
Diagrama de bloques del sistema de control de vigilancia que fue simulado por el sistema experto.

Diseño del sistema

El sistema experto realiza todas las funciones del microprocesador de control de vigilancia. La figura 1 muestra el sistema ECM a controlar. En un entorno ofensivo dinámico, el control de vigilancia recibe datos de entrada procedentes de los mensajes – tanto del equipo como de la programación – para la captación de señales, mientras que el tiempo entra por interrupciones procedentes de un reloj de tiempo real. Una vez que el control de vigilancia ha hecho una identificación preliminar de la señal, asigna unos equipos al seguimiento del radar y a medir su frecuencia y tipo de exploración, valiéndose de los datos que aporten tales equipos para afinar su identificación inicial y decidir si interfiere o no la señal.

En la realización del sistema experto, el usuario introduce los datos de partida interactivamente por medio del teclado. Esta entrada simula los datos que normalmente se recibirían de otros programas y equipos. El usuario se pone en lugar del reloj, incrementando el tiempo cuando lo considere apropiado, y sustituye también al proceso de captación de señales introduciendo datos sobre el número de señales identificadas y sus parámetros. A partir de estos datos, el sistema experto utiliza las reglas de su base de conocimientos para asignar una identidad y una prioridad preliminar a cada señal. No manda ningún dato al programa de captación de señales, pero sí planifica esta captación conforme a las reglas establecidas.

Los equipos del sistema ECM (seleccionador, medidor de frecuencia y determinador del tipo de exploración) envían datos al sistema experto, y los reciben del mismo, a través del interfaz de usuario. Una vez que el sistema experto ha identificado una señal enemiga, le asigna un seleccionador, el cual toma como entrada los parámetros de dicha señal e intenta seguirla. Si tiene éxito, envía un mensaje al sistema experto a través del interfaz de usuario, indicando que el seleccionador está "enclavado". A continuación el sistema experto asigna a la señal equipos medidor de frecuencia y determinador del tipo de exploración. El usuario puede introducir en el sistema experto datos sobre la frecuencia y el tipo de exploración, y éste los utiliza para afinar la identificación inicial de la señal.

El sistema experto simula también el control de interferencias. La base de conocimientos restringe el ECM a las amenazas que superen un nivel de prioridad fijado. Además, el sistema experto puede desasignar el ECM a una señal si, al ser ésta reidentificada, queda rebajada su prioridad.

Utilizando sus propias reglas, el sistema experto hace y deshace asignaciones de recursos de equipo a cada señal ofensiva. También incorpora reglas para determinar si cada elemento de equipo es operacional o no.

Interfaz de usuario

El Prolog ayudó a crear un interfaz agradable al usuario permitiendo un diálogo en inglés casi natural y admitiendo reglas para validar los datos de entrada. Las entradas del usuario son para el Prolog un objetivo que debe alcanzar la máquina de inferencia. El mecanismo de concatenación regresiva obliga al sistema experto a solicitar datos adicionales si son necesarios. Por ejemplo, si el usuario declara que se ha recibido un mensaje de medida de frecuencia, el sistema pregunta el origen del mensaje y la frecuencia (Fig. 2).

Figura 2
Ejemplo de diálogo entre un sistema experto de control de vigilancia (fondo azul) y el usuario (fondo gris).

```
Time now is 1205 msecs
Type in the next input to the simulation

freq_measurement_msg

Which frequency measurement hardware is this message from ?

3

What is the frequency (in MHz) ?

10500

ID of this threat has changed to Type C – acquisition mode
Priority of this threat has changed to 4

Time is now 1205 msecs
Type in the next input to the simulation
```

Figura 3 (derecha)
Regla de alto nivel del sistema experto para control de vigilancia, y uno de sus objetivos de segundo nivel. El objetivo es simular mientras existan datos disponibles o hasta que el usuario teclee "stop".

```
simulate :-
    initialize,
    repeat,
        do_scheduled_tasks,
        print_time,
        print_input message,
        getword(INPUT),
          (done(INPUT);
          bad_input(INPUT), fail;
          good_input(INPUT), fail).

done(INPUT) :-
    INPUT=stop,
    write("Simulation is done"),
    clear_dynamic_knowledge_base.
```

Figura 4 (abajo)
Regla para decidir si se dispone de un seleccionador para una señal ofensiva nueva. Se le puede tomar de una lista de seleccionadores no asignados, o bien reasignar uno actualmente asignado a una señal de inferior prioridad.

```
available_sorter(SORTER_NUM,PRIORITY) :-
    unassigned_sorter_list(SORTER_LIST),
    get_one(SORTER_NUM,SORTER_LIST,REMAINING_LIST),
    del_matching_statement(unassigned_sorter_list(SORTER_LIST) ),
    add_statement(unassigned_sorter_list(REMAINING_LIST) ).
available_sorter(SORTER_NUM,PRIORITY) :-
    get_priorities(PRIORITIES_LIST),
    sort(PRIORITIES_LIST,SORTED_PRIORITIES),
    get_one(LEAST_LETHAL,SORTED_PRIORITIES,LEFTOVER),
    !,
    PRIORITY<LEAST_LETHAL,
    sorter(SORTER_NUMBER,THREAT_NUMBER,LEAST_LETHAL),
    add_statement_ordered(no_sorter_avail(THREAT_NUMBER,LEAST_LETHAL),2),
    deassign(SORTER_NUMBER,THREAT_NUMBER,LEAST_LETHAL).
```

Formulación de reglas

El reto mayor fue traducir conocimientos previamente estructurados en forma de algoritmos a un conjunto de reglas de producción no basadas en procedimientos. El problema inicial entrañaba la formulación de una regla de máximo nivel que permitiera continuar la simulación mientras hubiera datos disponibles. Esa regla tenía que estar controlada por los datos, pese a ser el Prolog de concatenación regresiva; las figuras 3 y 4 muestran ejemplos de reglas Prolog. La representación de los datos en Prolog planteaba también dificultades, ya que no se permite incrementar el valor de una variable y por ello carece de significado la convencional sentencia de asignación "*tiempo = tiempo* + 1". El problema se resolvió tratando el valor del tiempo como un suceso que podía insertarse o eliminarse de la base de conocimientos siempre que el usuario solicitara un incremento de tiempo. A lo largo del desarrollo se explotaron las ventajas del Prolog, y se evitó escribir "código PASCAL" en lenguaje Prolog.

Evaluación

El concepto de sistema experto demostró su idoneidad para el control de vigilancia. Resultó fácil especificar y modificar las reglas, e incluso funciones complejas como la reidentificación de señales ofensivas pudieron realizarse con unas pocas reglas concisas: en total, menos de 100 reglas. Asimismo hubo que incluir relativamente pocos sucesos en el programa, pues la mayoría de los sucesos de la base de conocimientos eran observaciones del entorno que el sistema experto fue "aprendiendo" en su operación. Esta dinámica realización difiere de la mayoría de sistemas expertos, donde se utiliza una proporción mucho mayor de sucesos "a priori".

La versión actual del sistema experto puede llegar a conclusiones correctas sobre el entorno. Si varían los requisitos, es sencillo modificar las reglas para obtener nuevas conclusiones. Esto contrasta totalmente con la ingente tarea de modificar algoritmos y/o estructuras de datos en los lenguajes convencionales, y hace aún más rentable la utilización de técnicas basadas en reglas en una aplicación como la defensa electrónica, en la que se prevén muchas modificaciones. Queda por investigar si un sistema experto proporciona o no conclusiones correctas en un entorno mayor y más complejo. Tal vez haya un límite en el número de reglas y sucesos que puede manejar un sistema, y el tiempo necesario para buscar en la base de conocimientos quizá llegue a ser excesivo en programas más complejos. Con esta salvedad, el

sistema experto se comporta bien en su realización actual .

A la facilidad de escritura y de modificación de las reglas en Prolog debe contraponerse la dificultad inicial de aprender a pensar sin vincularse a procedimientos. Las técnicas y lenguajes de los sistemas expertos difieren totalmente en su concepto de los tradicionales lenguajes de alto nivel y del lenguaje ensamblador, lo que hace que incluso un experimentado ingeniero de programación necesite un largo aprendizaje para trabajar en ellos de modo productivo. No obstante, los programas en Prolog son mucho más cortos que los escritos en otros lenguajes para las mismas funciones: el sistema experto necesitó justamente la cuarta parte de líneas que el código original en lenguaje ensamblador para el sistema objeto[3]. Por consiguiente, hay una notable mejora en la productividad final de los programadores.

Como en el ordenador personal no había compilador para MProlog, todo el desarrollo y las pruebas tuvieron que realizarse a través de un interpretador, dificultando la medida de las características en tiempo real del sistema experto. Los terminales de entrada/salida añadieron además un retardo que desfiguraba cualquier medida en tiempo real. Solamente fue posible advertir que había un retardo visible de 1 ó 2 segundos desde que se tecleaba una entrada. A veces el interpretador se detenía también durante 6 ó 7 segundos para ejecutar una rutina de eliminar datos inútiles. Aunque esto no sería aceptable en un sistema ECM de a bordo, fue en cambio satisfactorio para evaluar el conjunto de reglas durante el desarrollo.

Por diversas características, el uso de sistemas expertos está muy indicado en el ámbito de ECM. En primer lugar, tales sistemas están concebidos para problemas donde hay interacciones complejas entre las posibles soluciones, cual sucede cuando hay que reaccionar en un entorno denso en señales ofensivas de múltiples modos y variables en el tiempo. Los lenguajes convencionales necesitan muchas líneas de código para cubrir todas las decisiones de control posibles; en cambio los sistemas expertos están formados por reglas generalizadas que interaccionan entre sí para controlar el entorno de amenazas electrónicas a medida que va evolucionando.

En segundo lugar, los sistemas expertos son adecuados en situaciones donde los datos son ambiguos o incompletos. Los programas de ECM a menudo se encuentran ante datos parciales o muy afectados por ruido, lo cual dificulta el identificar y dar prioridad con precisión a las distintas amenazas. Las técnicas de sistemas expertos facultan al sistema para tomar decisiones preliminares y después afinarlas o cambiarlas enteramente cuando se disponga de más datos.

Una tercera propiedad es que un sistema experto puede modificarse fácilmente por adiciones a la base de conocimientos en lugar de rediseñar algoritmos. Esto reduce los elevados costes del ciclo de vida y los largos tiempos de anticipación típicos de los sistemas con microprocesadores "embutidos".

Finalmente, las técnicas de sistemas expertos ofrecen a los sistemas ECM alternativas cada vez más "inteligentes". Las actuales contramedidas electrónicas se caracterizan por dar una respuesta relativamente rígida a un entorno ofensivo, mientras que los sistemas expertos prometen soluciones más innovadoras. Su utilización en la gestión de recursos mejorará el comportamiento de los sistemas al asignar los escasos medios disponibles de forma inteligente a las amenazas más mortíferas.

Por desgracia, ciertas limitaciones a la actual tecnología de sistemas expertos restringen su aplicabilidad a la guerra electrónica[4]. Todavía no existen procesadores de IA (inteligencia artificial) específicos para sistemas expertos, tales como las máquinas LISP, suficientemente pequeños para utilizarse en aviónica. Un procesador de uso general que ejecute en un lenguaje compilado de IA es demasiado lento para satisfacer las exigencias de tiempo real, y probablemente requeriría mucha memoria. No son adecuados los lenguajes interpretados de IA porque su necesidad de eliminar periódicamente datos inútiles es una sobrecarga inaceptable si se trabaja en tiempo real. En consecuencia, la tecnología de equipo de los sistemas expertos necesita más elaboración y madurez para poderse aplicar al control en tiempo real en la guerra electrónica.

Conclusiones

El desarrollo de un sistema experto para control de vigilancia ha demostrado poder resolver con éxito los problemas de las contramedidas. El aumento de la productividad de los programadores y la facilidad de modificación de programas revelan que los sistemas expertos ofrecen la capacidad de reducir notablemente los costes y los largos tiempos de preparación asociados con el desarrollo de la programación convencional. No obstante, para que los programas de IA tengan valor práctico en las contramedidas de a bordo, habrá que superar las dificultades que conlleva el comportamiento

en tiempo real y la obtención de memoria adecuada merced a perfeccionamientos del equipo físico.

Mientras tanto, la experiencia adquirida con el sistema antes descrito podría aprovecharse en aplicaciones en tiempo no-real de guerra electrónica. Ejemplo es el uso de un sistema experto de diagnóstico para codificar la maestría que requiere la depuración de un sistema ECM. Podría ser útil una herramienta de este tipo durante la producción, cuando los programas "embutidos" son correctos en términos básicos y el ingeniero de programación se dedica a ayudar a diagnosticar problemas en los equipos. Se necesita una pericia considerable para poder prestar un eficaz soporte de programación en este tipo de entorno. Es frecuente que las personas dotadas de los conocimientos necesarios deban cambiarse a otras funciones, e interesaría mucho guardar su experiencia en un sistema experto.

Otra aplicación consiste en servir como herramienta para formar con rapidez un prototipo de sistema ECM durante el análisis de los requisitos. La especificación de las reglas de actuación del sistema es, en sí misma, un proceso instructivo. En esencia, la base de conocimientos de un sistema experto no es sino una especificación del comportamiento que se pediría a un experto. Al ser interpretada esta especificación preliminar por la máquina de inferencia, se la puede perfeccionar con anterioridad a cualquier realización[5].

Por la facilidad y rapidez con que se escriben y modifican las reglas, un sistema que opere con reglas será una herramienta de desarrollo ideal. Las especificaciones de ECM pueden codificarse en un sistema experto dentro del proceso de análisis de requisitos, siendo entonces factible hacer modificaciones y revisar la especificación antes de iniciar el costoso y largo trabajo del diseño y la codificación convencionales.

Para llevar a la práctica este objetivo se requiere que el sistema ECM entero, con toda su complejidad, sea una aplicación adecuada para un sistema experto. El prototipo de ITT Avionics sólo probó una parte de tal sistema; hacen falta más estudios para determinar el tamaño de la base de reglas necesaria y analizar cómo afectaría al comportamiento en tiempo real el aumento de dicha base. Si se comprueba su valor práctico, esta aplicación de los sistemas expertos al análisis de requisitos podría resultar ser su más útil contribución al desarrollo del soporte informático para la guerra electrónica.

Referencias

1 F. Hayes-Roth, D. A. Waterman y D. B. Lenat (editores): *Building Expert Systems*, Addison-Wesley, Reading, Massachusetts.

2 P. A. Subrahmanyam: The Software Engineering of Expert Systems: Is Prolog Appropriate?: *Institute of Electrical and Electronics Engineers Transactions on Software Engineering*, noviembre 1985, volumen SE-11, nº 11, págs. 1391–1400.

3 J. Cohen: Describing Prolog by its Interpretation and Compilation: *Communications of the Association of Computing Machinery*, diciembre 1985, volumen 28, nº 12, pág. 1322.

4 M. L. Wright, M. W. Green, G. Fiegl y P. F. Cross: An Expert System for Real Time Control: *Institute of Electrical and Electronics Engineers: Software*: marzo 1986, volumen 3, nº 2, págs. 16–24.

5 J. Doyle: Expert Systems and the Myth of Symbolic Reasoning: *Institute of Electrical and Electronics Engineers: Transactions on Software Engineering*: noviembre 1985, volumen SE-11, nº 11, págs. 1386–1387.

Ersilia Gaudry nació en Nueva Jersey, en 1955. Estudió en la Universidad de Princeton, en la que obtuvo el título AB, y después en la Fairleigh Dickinson University, donde se graduó MS en ciencias informáticas. En 1982, la Sra. Gaudry ingresó en ITT Avionics como ingeniera de programación y se ha dedicado al desarrollo de programas para microprocesadores "embutidos" en sistemas de contramedidas electrónicas en tiempo real. Desde 1985 está trabajando en un proyecto de investigación y desarrollo cuyo fin es evaluar las técnicas de sistemas expertos en aplicaciones de contramedidas electrónicas.

Arquitectura de sistema de diagnóstico

Los sistemas expertos ofrecen una técnica potente para diagnosticar y reparar fallos en equipos complejos. Se ha desarrollado una arquitectura que incorpora los métodos utilizados por expertos humanos en la resolución de problemas de diagnóstico.

M. E. Atwood
E. R. Radlinski
ITT Advanced Technology Center, Shelton, Connecticut, Estados Unidos de América

Introducción

En los pasados treinta años, la investigación psicológica y educacional se ha planteado dos cuestiones fundamentales: cómo diagnosticar los fallos en los sistemas y cómo mejorar su rendimiento. En un principio, el aumento de la complejidad del equipo militar fue el acicate del interés por estas cuestiones, pero más recientemente lo ha sido la introducción de sistemas electrónicos a prueba de fallos, como el Sistema 12 de conmutación digital. Estos sistemas raramente fallan, y es muy raro que un técnico en reparaciones vea aparecer dos veces el mismo fallo. Por otra parte, las reparaciones han de realizarse con el sistema en servicio.

El interés en mejorar el rendimiento del diagnóstico está ahora desplazándose hacia el trabajo en sistemas expertos. Normalmente siempre habrá alguien capaz de reparar un dispositivo en fallo, bien sea el diseñador, el realizador o uno con aptitudes generales para localizar averías. Capturando esta experiencia y haciéndola accesible a todo el personal de reparaciones, los sistemas expertos potencian la capacidad de reparar equipos.

Diagnosis de fallos

Concepto de la diagnosis
Un sistema necesita ser diagnosticado y reparado cuando presenta un comportamiento *anormal*, que se reconoce por unos *síntomas observables:* el sistema no funciona correctamente, los indicadores de error están encendidos, no hay respuesta a las señales de entrada, etc. Dichos síntomas tienen *causas internas*, asociadas con partes del sistema que se podrían reparar o sustituir para corregir los referidos síntomas. Por ejemplo, en un sistema de equipo físico, las causas internas pueden situarse en las placas, el alambrado, u otros elementos. Diagnosis es la tarea de identificar las *causas internas* que producen los *síntomas observables*, y corregirlas reparando o sustituyendo componentes del sistema.

Dificultades de la diagnosis
Aparentemente debería ser fácil la diagnosis: bastaría con enumerar todos los síntomas observables, todas las causas internas y todas las relaciones entre ellos, obteniendo así una gran tabla que relaciona síntomas y causas. Sin embargo, a no ser en casos muy sencillos, esto no es factible, pues hay demasiados síntomas y causas para que se puedan enumerar, y las relaciones entre unos y otras no son muy comprensibles. Seguidamente se citan diversas razones de dificultad.

Manifestaciones nuevas: síntomas que no se hayan observado nunca. Esto ocurre siempre en los sistemas nuevos, y a veces incluso en un sistema conocido. Es fácil diagnosticar cuando los síntomas han aparecido anteriormente y se sabe que una determinada reparación es adecuada, pero si ello no es así el diagnóstico puede ser mucho más difícil.

Manifestaciones con varias causas posibles: si cada síntoma se pudiera corregir mediante una reparación única, la diagnosis sería bastante más sencilla. Sin embargo, a menudo se necesitan varias reparaciones para corregir un solo síntoma, siendo importante el orden en que se realizan.

Síntomas que no se conocen todos inicialmente: la diagnosis comienza con la *queja primaria*. Por ejemplo, una inflamación de garganta que se manifiesta al médico. Es poco probable que se le pueda informar de una ligera fiebre, y mucho menos de tener bajo el número de leucocitos, que ni siquiera se nota. Sin embargo, estos síntomas son importantes y deben ser descubiertos durante la diagnosis. De forma similar, algunos síntomas de fallo son

obvios, mientras que otros se han de encontrar en el proceso de diagnóstico.

Falta de relación directa entre causas y síntomas: a un síntoma (o conjunto de ellos) no le corresponde un conjunto, claramente definido, de causas susceptibles de corrección. Diferentes fallos pueden originar el mismo conjunto de síntomas, y un mismo fallo puede presentarse en varias formas diferentes.

Nuevos problemas creados por los procedimientos de reparación: hay mucho de verdad en la máxima "si no está roto, no lo arregles". Durante la diagnosis, sin embargo, no se ve claro dónde está el fallo. Los procedimientos de reparación pueden resolver todos los problemas conocidos, o bien solamente algunos de ellos, o también introducir otros o sustituir los problemas antiguos por otros nuevos. La diagnosis se integra con reparaciones que cambiarán el estado del sistema que se diagnostica.

Necesidad de reparaciones económicas: sería muy sencillo y eficaz sustituir la totalidad del sistema a reparar por un sistema nuevo, aunque ello resultaría muy costoso. Una buena solución debe implicar el número mínimo de reparaciones, lo más económicas posible.

Comportamientos en la diagnosis

Actitudes humanas en el diagnóstico de sistemas

Los sistemas expertos de diagnóstico son sistemas interactivos que ayudan al personal de reparaciones. Para que tales sistemas sean eficaces es necesario comprender las posibilidades y las limitaciones de las personas que diagnostican, de forma que el diseño del sistema experto se aproveche de unas y haga más tolerables las otras.

Utilización de teorías híbridas: los técnicos de diagnóstico aprenden teoría en algún momento, pero los técnicos prácticos no utilizan teoría solamente. Sus conocimientos teóricos han sido más bien "compilados" en el procedimiento con el que diagnostican los sistemas. Esto es, los técnicos diagnostican correctamente a partir de la teoría, pero sustituyendo la parte puramente teórica de su experiencia por el conocimiento teórico de "ese sistema concreto". Así, un sistema de diagnóstico no puede describir la teoría sin relacionarla con el dispositivo diagnosticado.

Utilización ineficaz de los datos empíricos: a menudo se olvidan los intentos anteriores y se hacen pruebas y reparaciones sin relación con el estado actual del sistema. Tampoco puede preverse la forma en que las acciones afectarán al sistema. Para evitar esto, un sistema de diagnóstico debe almacenar la historia de su comportamiento. Siempre que el sistema solicite una prueba o reparación, o siempre que el usuario lo desee, se visualizarán los efectos de cada una de tales acciones.

Actuación anterior al razonamiento: como los técnicos tienden a "hacer cosas" cuando examinan un sistema, un sistema interactivo de diagnóstico debería permitir tal comportamiento, al menos por dos razones. Primera: un sistema que se interfiera con los procesos usuales para localizar averías no se considerará útil y probablemente no se le aplicará. Segunda: al poseer el técnico cierta experiencia sobre el dispositivo diagnosticado, puede saber cómo reparar el fallo sin tener que realizar todas las acciones que el sistema de diagnóstico recomienda. Por consiguiente, estos sistemas deberán aceptar resultados de pruebas procedentes del usuario en cualquier fase del diagnóstico.

Comportamiento poco lógico de los técnicos: los técnicos no razonan bien cuando han de considerarse simultáneamente varios aspectos. Tienden a admitir toda evidencia que apoya una hipótesis particular y a rechazar las que puedan negarla. Por lo tanto, un sistema de diagnóstico debe ser capaz de mostrar las hipótesis actuales sobre fallos en el sistema, las bases en que se sustentan dichas hipótesis y los eventos que pudieran provocar su rechazo. Debe proporcionar información para saber cuándo desechar una hipótesis o cómo aceptarla, insistiendo en lo referente al rechazo de hipótesis, que los usuarios suelen minusvalorar. Un sistema de diagnóstico ha de poder mostrar cómo afecta cada hecho real a la idea actual de la operación del sistema.

Utilización del razonamiento empírico y causal: los técnicos pueden seguir cualquiera de estos dos razonamientos, y pasar del uno al otro durante un diagnóstico; por ejemplo, el técnico tratará de aislar un fallo razonando de modo empírico, y cuando fracase pasará al razonamiento causal, volviendo al empírico cuando aparezca un hecho particular. En consecuencia, un sistema de diagnóstico debe admitir ambos procesos lógicos, ser capaz de integrarlos y de alternar su utilización.

Razonamiento a partir de modelos de faltas: un sistema de diagnóstico no está necesariamente destinado a la enseñanza. Es decir, no tiene que determinar si un modelo causal de usuario es incorrecto y desarrollar

entonces un procedimiento para que el usuario lo modifique. Sin embargo, el sistema debe presentar claramente al usuario sus modelos causales.

La experiencia es difícil de comunicar: la comunicación entre expertos y principiantes es difícil; entre sistemas expertos y técnicos principiantes es aún más difícil. Por lo tanto, el interfaz usuario-sistema para sistemas expertos requiere un considerable trabajo de investigación.

La arquitectura

El objetivo es diseñar una arquitectura que tenga en cuenta los factores anteriores. Esta arquitectura se describe aquí con respecto a tramas de conocimiento y mecanismos de inferencia.

Tramas de conocimiento

Muchas estructuras de datos se han utilizado o podrían utilizarse en sistemas de diagnóstico. Sin embargo, hay que sujetarse a dos limitaciones básicas para satisfacer los criterios anteriormente expuestos. En primer lugar, al estar involucrados diversos tipos de conocimientos, se requiere un formalismo común. En segundo lugar, a medida que el técnico observa síntomas visibles y sustituye o repara componentes, hay que indexar la estructura elegida según estos comportamientos y componentes reparables.

La diagnosis implica conocimientos empíricos y causales. Mientras que el conocimiento empírico suele representarse según reglas de producción, no existe representación convencional para el conocimiento causal, ni tampoco una que sea común a ambos. Además, no debe suponerse que los tipos de conocimiento señalados como relevantes sean los únicos que se necesitarán en el futuro; un formalismo de representación debe estar abierto al crecimiento.

Se definen aquí cuatro clases de tramas de datos, ajustadas todas a los principios ya expuestos, las cuales tienen zonas o "casilleros" predefinidos para información, y especifican cómo se relaciona la información de un casillero con la de otro. En el presente artículo, estas tramas se definen dentro del contexto de diagnosticar un sistema estéreo de componentes.

Los cuatro tipos de tramas son *tramas de hipótesis*, *tramas de prueba*, *tramas de componentes* y *tramas históricas*. Las tramas de hipótesis describen la relación entre los síntomas observables por el usuario y las posibles condiciones de fallo. Las tramas de prueba proporcionan una descripción de los procedimientos de prueba y reparación utilizables durante los diagnósticos. Las de componentes describen la composición física del sistema en pruebas, y las históricas ofrecen una perspectiva en el tiempo de la eficacia de cualquier procedimiento de diagnóstico dado.

Tramas de hipótesis: describen la relación entre el conocido comportamiento de los sistemas y sus posibles causas (Fig. 1). La trama contiene el conjunto de síntomas observables conocidos que apoyan la hipótesis de fallo en el circuito preamplificador de audio: hay dos síntomas en la lista. Si el diagnosticador notifica uno de ellos o ambos, se anota esta hipótesis como activa durante la generación de hipótesis. Dentro de esta trama se especifican también los procedimientos de prueba, que indican el nombre de la prueba (o pruebas) que apoyan o rechazan la hipótesis de fallo en el preamplificador.

name:	faulty audio pre-amp circuit
status:	(yes, no, unknown, active)
symptom list:	AM/FM tuner has power No sound from speakers . . . . . .
tests:	audio pre-amp circuit test pre-amp current test . . . .

Figura 1
Trama de hipótesis.

```
test_name:          Audio pre-amp circuit test
component name:  pre-amp circuit #A42
test_description:  'text.....'
action_list:
   check for preconditions;
   if preconditions=ok,
      then use component pre-amp circuit #A42;
           probe_input_test_point
           for input signal
    if input=ok
         then probe output_test_point
              .....
    if output=ok
         return (status=ok)
    else if output <> ok
         return (status=broken)

   else return (status=unknown)
              . . .
              . . .
precondition list:
   Turn off connected components
   Disconnect connected components
   Turn off power to receiver
   . . . .
```

Figura 2
Trama de pruebas para circuito preamplificador.

Tramas de prueba: las tramas de este tipo contienen esquemas que describen cómo generar información sobre el estado del sistema en diagnóstico, pudiendo hacer referencia a otras tramas de prueba. En la figura 2 se muestra un guión para la comprobación de fallo en un preamplificador de audio. Las condiciones previas de prueba especifican los requisitos a cumplir antes de poder acometer la prueba. Por ejemplo, no puede probarse el estado de un circuito si el sistema no tiene fuente de alimentación.

Tramas de componentes: se utilizan para describir los atributos físicos de cada elemento del sistema (Fig. 3). Cualquier trama de componentes puede hacer referencia a subcomponentes, admitiendo una estructura jerárquica cuando sea necesario. Los atributos descritos en tales tramas caracterizan a la entrada y salida de los componentes. La descripción de la entrada incluye la fuente de la señal, su tipo, y el punto de prueba para la entrada. Es análoga la descripción de los atributos de la señal de salida. Esta información de señal puede ser utilizada por múltiples tramas de prueba, cuando se requiera, y ser referenciada por ellas. Se especifican procedimientos de sustitución para los elementos que pueda cambiar un técnico.

Tramas históricas: este tipo de trama aporta al mecanismo de inferencia del sistema de diagnóstico un registro de las pasadas reparaciones, agrupadas por síntomas. Contiene una lista de las hipótesis formuladas para los síntomas dados, las pruebas realizadas para su verificación, el coste por cada prueba, y un registro de la frecuencia de los sucesos pasados (Fig. 4). Estos atributos históricos sirven de ayuda al mecanismo de inferencia para seleccionar las mejores pruebas a la hora de evaluar hipótesis.

Inferencia: hipótesis y pruebas

Los expertos en diagnóstico confían en un procedimiento secuencial de hipótesis y prueba incorporado al sistema. Así, en efecto, recogen síntomas, formulan hipótesis, seleccionan pruebas, las realizan, y comienzan de nuevo registrando los síntomas que permanecen tras la realización de una prueba.

Recogida de síntomas: se supone que el conjunto de síntomas es incompleto, representando únicamente aquéllos que son observables. Por ejemplo, un síntoma inicial podría ser "LED de alimentación no operativos", pero en cambio no es probable que sea "impulsos espurios en la entrada de alimentación".

```
component_frame
        component_name=pre-amp circuit # A42
        function=amplify audio signals
        status=unknown
        replaceable=true
        replace_procedure name=Pre-amp disassembly
        component_supported=TX27 tuner/receiver
        sub_component list:   Pre-amp switch # 2
                              Low level amp circuit
                              Pre-amp switch # 3
                              High level amp circuit
        input_source=AM/FM tuner
        input_test_point=L1, R1
        input_description=100 pwatts
        output_test=Audio power amplifier
        output_test_point=L2, R2
        output_description=0.10 mwatts
```

Figura 3
Trama de componentes para circuito preamplificador.

History-frame
Symptom=system has power, no speaker output
Source=user
Belief=high

HYPOTHESIS	TEST	COST	WEIGHT
bad connections	empirical-c1	low	8
blown speaker fuse	empirical-f8	low	6
tuner circuit fault	causal-t32	high	3
pre-amp circuit fault	causal-pa32	high	3
power amp fault	causal-pa34	high	3

Figura 4
Trama histórica para ausencia de salida de altavoz.

Formulación de hipótesis: el sistema utiliza *diagnosis diferencial* para formular hipótesis que corresponden, bien a acciones de reparación posibles (por ejemplo, ha saltado un fusible), o bien a estados del sistema que aportan información (p. ej., se ha encendido el LED de alimentación). Cualquier hipótesis relacionada con los síntomas observados pertenece al conjunto de hipótesis activas. Es de destacar que este conjunto puede incluir hipótesis relativas a todos los síntomas conocidos, algunas que correspondan a síntomas aislados, y otras que se excluyan entre sí.

Selección de prueba: el sistema trata muchas hipótesis simultáneamente, y por ello se realizan pruebas de donde extraer información utilizable para reducir estas hipótesis al conjunto mínimo que explique los síntomas observados. En la selección de las pruebas influyen dos factores: la cantidad de información obtenible y el coste de la prueba. El volumen de información actualmente se determina en función del número de hipótesis sobre las que da información la prueba. En cambio, el coste puede tener diversos valores según el

estado del sistema y el emplazamiento del diagnosticador; así, por ejemplo, una biopsia de estómago dirigida por un médico tiene un elevado coste inicial, pero si coincide con la extirpación de apéndice del paciente, ya su coste es mucho menor. También pueden utilizarse otros métodos de selección de pruebas. Una versión posterior del sistema de diagnóstico incorporará mecanismos de selección de pruebas definidos por el usuario.

Ejecución de pruebas: los diagnosticadores ejecutan las pruebas, cuyos resultados se comunican al sistema experto. Actualmente los sistemas no están concebidos para obtener información sin intervención humana, aunque ello es claramente posible.

Iteración: los diagnósticos y los intentos de reparación guardan estrecha relación, informándose entre sí. Este procedimiento secuencial de hipótesis y pruebas continúa hasta que se hayan corregido tantos síntomas como sea posible.

Localización de fallos en un sistema estéreo

La estructura escogida se ilustra tomando como ejemplo la diagnosis de un sistema estéreo en fallo (Fig. 5).

Recogida de síntomas: la diagnosis comienza cuando el usuario notifica síntomas de fallo. En este ejemplo, el sintonizador estéreo tiene alimentación (como indican los LED del panel frontal), pero no hay salida de audio desde ningún altavoz.

Formulación de hipótesis: el sistema de diagnóstico comienza a generar hipótesis, para lo cual empareja los síntomas conocidos con diversas tramas de hipótesis. En este caso, se activan todas las hipótesis que contienen los síntomas de *sistema con alimentación* y/o *ausencia de salida de altavoces*.

Selección de prueba: el programa de diagnóstico utiliza luego la trama histórica (Fig. 4) como ayuda para determinar las hipótesis que deben explorarse primero. En el ejemplo, suelen ser las conexiones flojas o defectuosas las que producen tal síntoma, y por ello el sistema pide al usuario que compruebe las conexiones entre los componentes. Si el fallo no está en las conexiones, como en el presente caso, se descarta esa hipótesis. Esta tarea no afecta a otras hipótesis ni genera otras nuevas.

Iteración: continúa el proceso de generar información y modificar el conjunto de hipótesis activas. El sistema de diagnóstico solicita una comprobación del fusible de protección del altavoz, pero también éste demuestra funcionar correctamente. Al llegar a este punto se han agotado todas las relaciones empíricas, y faltan solamente las pruebas más complicadas, orientadas hacia la comprobación de componentes individuales.

Quedan tres hipótesis primarias (Fig. 4 y 5): fallo en el sintonizador, en el preamplificador, o en el amplificador de potencia. El sistema de diagnóstico elige probar el preamplificador, ya que esto reducirá la magnitud de la prueba de hipótesis, sea cual fuere el resultado.

La trama de hipótesis de la figura 1 encamina el programa de diagnóstico a la trama de prueba del preamplificador de audio (Fig. 2), la cual prescribe la verificación de las características de entrada y salida del circuito para determinar su estado. Las pruebas interactivas permiten al usuario introducir información a lo largo de toda la prueba. Sin embargo, se revisan las condiciones previas de la prueba antes de iniciar ésta. Al comprobar la primera condición previa (desactivar los componentes conec-

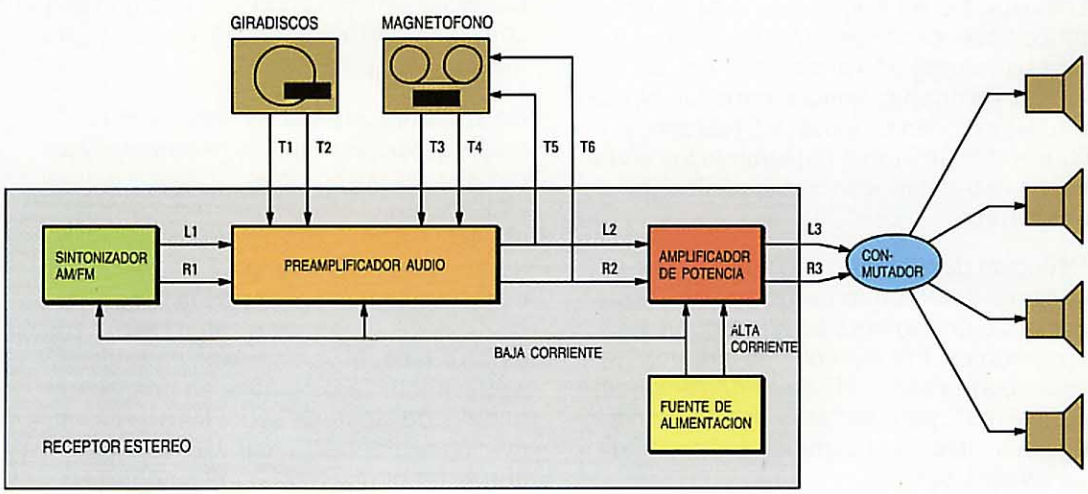


Figura 5
Esquema del receptor estéreo utilizado como ejemplo para la diagnosis de fallo por el sistema experto.

tados, en este caso un magnetófono) los altavoces funcionan correctamente, por lo que el usuario comunica los resultados al sistema de diagnóstico, y éste elimina todas las hipótesis relativas a *ausencia de salida de audio* e introduce el nuevo síntoma *ausencia de salida de receptor con el magnetófono activado*. Tras haber resuelto el problema inicial de no haber salida de audio, el sistema de diagnóstico pasa ahora a tratar el problema asociado a este nuevo síntoma. La diagnosis se desplaza desde el sistema estéreo completo a un componente, avanzando de forma similar a la descrita anteriormente.

Conclusiones

El ITT Advanced Technology Center está desarrollando un sistema experto cuyas estructuras de datos y mecanismos de inferencia se basan en los métodos aplicados por diagnosticadores expertos. La diagnosis, especialmente de los dispositivos a prueba de fallos, es una tarea difícil e involucra a varias fuentes de conocimiento. La experiencia humana constituye un modelo apropiado para el desarrollo de una arquitectura que aglutine estas diversas fuentes de conocimiento en un sistema único.

Para construir sistemas tales como el descrito se requiere información que inicialmente proviene del diseño y se suplementa con información empírica. La mayoría de las tramas iniciales de hipótesis y de prueba, así como las de componentes, proceden del diseño del sistema. Las tramas de hipótesis y prueba pueden enriquecerse o modificarse a medida que crece el conocimiento empírico, e igual ocurre con las tramas históricas. Esto permite al sistema de diagnóstico experto adaptarse al dispositivo al que se aplica.

Aunque el desarrollo de la arquitectura se oriente inicialmente hacia los sistemas de diagnóstico, también será adecuada para fines explicativos, sistemas de enseñanza y ayudas al diseño, ya que en ellos se utiliza el mismo tipo de conocimientos.

Michael E. Atwood obtuvo un BA en psicología y matemáticas en la Universidad de Kansas y un PhD en psicología cuantitativa en la Universidad de Colorado. El Dr. Atwood trabajó en varios campos, incluyendo psicología del conocimiento, factores humanos y sistemas expertos, hasta ingresar, en 1983, en el ITT Advanced Technology Center, donde trabaja como investigador. Actualmente se ocupa de la aplicación de la inteligencia artificial a los sistemas de diagnóstico y reparación.

E. R. Radlinski se graduó BS en matemáticas por la Universidad de San Buenaventura en 1970, y MS en informática en el Rensselaer Polytechnic Institute, en 1975. Ha trabajado en el desarrollo de paquetes integrados de herramientas de programación y en el diseño de arquitecturas de sistemas avanzados para aplicaciones financieras y comerciales. Desde su ingreso en el ITT Advanced Technology Center, en 1980, el Sr. Radlinski ha investigado diseños alternativos para un interfaz adaptable de usuario y actualmente diseña sistemas expertos de diagnóstico.

Modelos causales: la nueva generación de sistemas expertos

La diagnosis de faltas en sistemas de conmutación telefónicos puede mejorar considerablemente utilizando programas de diagnóstico capaces de "razonar" de manera similar a un ingeniero de mantenimiento. Estos sistemas, denominados expertos, pueden trabajar a dos niveles: los modelos superficiales, que utilizan datos empíricos previamente observados para detectar las faltas, y los modelos causales más potentes, que analizan cómo trabaja el sistema.

M. E. Atwood
R. Brooks
E. R. Radlinski
ITT Advanced Technology Center, Shelton, Connecticut, Estados Unidos de América

Introducción

Los expertos en mantenimiento pueden seguir diversos caminos para aislar una falta; tomando como ejemplo que un ordenador personal no se cargue correctamente, un ingeniero avezado a este tipo de ordenadores debería sopesar los síntomas, quitar la cubierta superior y apretar en el lugar correcto de una de las ROM con un destornillador. Muchas veces con ello se resolvería el problema.

En el otro extremo, un ingeniero experimentado examinaría los síntomas y, al ser éstos nuevos, tendría que reflexionar más, considerando la constitución del ordenador, el papel de cada pieza, y qué ocurriría si fallara un determinado componente.

Los dos métodos valen para resolver el problema, si bien con diferente estilo de razonamiento, basado uno en anteriores experiencias con ese mismo sistema y el otro en conocer su funcionamiento. Uno u otro en las circunstancias adecuadas crean un comportamiento "experto".

Los sistemas expertos, así llamados por su semejanza al modo de razonar de los expertos humanos, son programas de ordenador que dan consejos o toman decisiones basadas en la manipulación simbólica de información, en contraste con otras herramientas fundamentalmente de naturaleza estadística o numérica. Estos sistemas se estudian actualmente en ITT como ayuda en muy diversas operaciones, y especialmente en el diagnóstico de fallos en sistemas de conmutación.

Sistemas expertos en el diagnóstico de fallos

Las primeras ayudas para el diagnóstico de fallos solían basarse en tablas de decisión, bien fueran matrices, diagramas de flujo o máquinas de estados finitos, que relacionaban los síntomas observados y las acciones correspondientes. El personal de mantenimiento localizaba en la tabla los síntomas en cuestión y ejecutaba la acción allí indicada.

La operación de los sistemas expertos es similar, mas con diferencias importantes. Primero, el surtido de representaciones de información es mucho más rico, admitiendo por ejemplo fácilmente disposiciones jerárquicas en las que un conjunto de decisiones anida en otro conjunto, sin que la tabla sufra una expansión combinatoria. Es normal, además, que la gama de pruebas posibles sea mucho más vasta y flexible: en algunos casos, la prueba puede consistir en un programa entero que calcule la decisión a tomar.

Segundo, los sistemas expertos suelen tratar varias alternativas a la vez, escogiendo un número reducido de pruebas que sean comunes al máximo número de tales caminos a seguir. Este método puede dar los mismos resultados que una tabla de decisión, pero con mucho mejor rendimiento económico.

Tercero, los sistemas expertos pueden manejar información dudosa. En un sistema de tabla de decisión convencional, los resultados de las pruebas deben ser verda-

deros o falsos, y en consecuencia se aceptan o no. No hay modo conveniente de expresar que una prueba sea cierta sólo en determinados casos, o que un resultado deba preferirse a otro si se cumplen condiciones dadas.

Durante los cinco últimos años, se han construido muchos de estos sistemas, con aplicación comercial a diagnósticos médicos, cables de teléfonos y configuraciones de ordenadores. Asimismo, han servido para indicar al personal de mantenimiento la mejor manera de identificar y reparar los fallos de equipos electrónicos. Reduciendo el tiempo de aprendizaje, asegurando los diagnósticos y optimizando el proceso de reparación, los sistemas expertos encierran un gran potencial para mejorar la rentabilidad y aumentar la fiabilidad de los sistemas. Por ejemplo, el ITT Engineering Support Centre en Harlow. Inglaterra, ha ayudado a los ingenieros de Standard Elektrik Lorenz a desarrollar un sistema para probar y reparar en fábrica las placas de procesador del Sistema 12. El uso generalizado de estos sistemas para diagnosis y reparación en ITT podría aportar una mejora de calidad adicional.

Arquitecturas para los sistemas expertos

Los sistemas expertos de diagnosis por reglas pueden concebirse en cualquiera de las dos arquitecturas primarias, asimilables a la conducta de un técnico capacitado. La primera de ellas, el modelo "superficial", se fundamenta en asociaciones de índole empírica, como la relación entre un fallo en la carga y apretar una ROM que está floja con un destornillador. Los sistemas que trabajan sobre estas bases se componen con relativa facilidad y pueden operar bastante aprisa, constituyendo casi el total de sistemas expertos actualmente en uso.

Un segundo método, el modelo causal, emula un comportamiento racional, incorporando la pericia de un ingeniero que razona sobre la operación del sistema con vistas a hacer un diagnóstico y a repararlo. Aunque en algunos casos esta diagnosis sea más lenta que con una tabla de decisión o un modelo superficial, presenta la ventaja de poder tratar con síntomas antes no observados. Saber cómo trabaja un sistema es crucial para su diagnóstico.

Sistemas de modelos superficiales

Mientras dura su proceso de activación, un ordenador personal tiene que completar con éxito una prueba del sistema de encendido, lo cual requiere verificar que el equipo y el programa son correctos. El problema considerado aquí, es que la prueba se ha iniciado pero no puede llegar a su término. Los ejemplos que siguen ilustran posibles reglas para determinar la causa de tal fallo, indicándose en la figura 1 las partes de equipo y programas implicadas.

Un sistema de modelo superficial tiene reglas válidas en cualquier momento, y otras reglas encadenadas de tal modo que hay que aplicarlas en una cierta secuencia. La primera regla, por ejemplo, asigna medidas de probabilidad a las posibles causas de no completarse la prueba del sistema de encendido (ver tabla 1), sin declarar la causa real sino sólo las opciones que hay y cuál es su orden de verosimilitud. La segunda regla usa una prueba de diagnóstico para comprobar la configuración de los conmutadores y localizar la causa del problema en el equipo; supuesto que resida ahí el fallo, existen reglas que alteran la probabilidad de su origen. Otros criterios adicionales combinan los resultados de procesos diagnósticos con causas plausibles, para ir así limitando la lista de causas al grupo correcto.

Si bien los sistemas de modelo superficial admiten diversas realizaciones, todos ellos se diferencian por el hecho de estar basados en datos empíricos. Un modelo superficial describe qué síntomas se han observado anteriormente y cuál reparación es la adecuada en cada caso. Estas asociaciones empíricas están contenidas en reglas del tipo "Si (IF) se observan estos síntomas, entonces (THEN) ejecutar esas acciones". Por ejemplo, en el caso de la prueba de encendido tales síntomas incluyen: fracaso en completar la prueba de encendido, configuración de conmutadores, encaje de las placas impresas del ordenador y de la ROM, y programación de la ROM.

Un sistema de modelo superficial supone que no hay otras formas de resolver el

Figura 1
Caso de prueba de encendido.

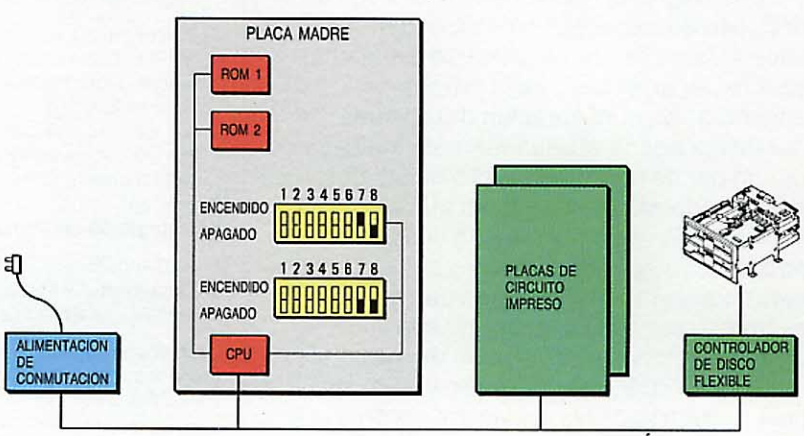


Tabla 1 — Sistema de modelo superficial

Regla 1
Si la prueba de encendido no se completa,
entonces deducir que la configuración de conmutadores es incorrecta ($p = 0,8$)
la configuración de conmutadores es correcta ($p = 0,2$).

Regla 2
Si la configuración de conmutadores es correcta,
entonces debe estar defectuoso el equipo ($p = 0,9$).

Regla 3
Si la configuración de conmutadores no está correctamente puesta,
entonces deducir que el problema está en su mal posicionamiento y terminar.

Regla 4
Si el equipo está defectuoso y
las placas no están bien colocadas
entonces deducir que las placas son el problema y terminar.

Regla 5
Si el equipo está defectuoso y
las ROM no están bien insertadas
entonces deducir que el problema está en las ROM y terminar.

p = probabilidad

problema detectado en esta prueba. Sin embargo, quizá el fallo provenga de causas no consideradas en las reglas formuladas: por ejemplo, un bitio mal en una ROM correctamente programada e insertada. Si bien los modelos superficiales pueden ser eficaces en resolver un problema que aparece por segunda vez, esto no sucede en su primera aparición. Por supuesto, se podría modificar el sistema para tratar con un nuevo caso después de haberse observado y tratado éste por otros medios, mas no con el modelo superficial. A diferencia de los sistemas electrónicos automáticos que suelen tratar defectos comunes en el equipo, el personal de mantenimiento generalmente se enfrenta con fallos nuevos, y por ello les convendría más un sistema experto de diagnosis basado en un modelo causal.

Puesto que las relaciones asociativas se deducen de experiencias, los modelos superficiales tienden a aplicarse a sistemas concretos; en el caso de la prueba de encendido, la configuración del sistema puede escogerla el usuario y está indicada por un par de conmutadores basculantes. Pero puede haber un sistema afín cuya configuración se almacene en cualquier otro sitio, ya sea en un disco o en ROM, y para tratar esta variante habría que modificar el modelo superficial, sustituyendo la prueba de "revisar la posición de los conmutadores" por otra diferente, lo cual afectaría a muchas de las reglas del sistema.

Sistemas de modelos causales

Los sistemas de modelos causales buscan superar las limitaciones de los modelos superficiales, y tienen por tanto una arquitectura diferente. Las reglas en estos modelos se estructuran en tres módulos: un modelo funcional, un modelo de faltas y una

Tabla 2 — Modelo causal para el problema de la prueba de encendido del microordenador

Modelos de dispositivo

Si hay alimentación y la placa madre está trabajando bien,
entonces la CPU lee los diagnósticos en la ROM 1.

Si la CPU ha leído los diagnósticos de la ROM 1,
entonces la CPU ejecuta los diagnósticos.

Si la CPU está ejecutando los diagnósticos,
entonces la CPU leerá los conmutadores 1 y 2 para obtener la configuración del sistema.

Si la CPU está ejecutando los diagnósticos y ha leído los conmutadores,
entonces la CPU comprobará la configuración.

Si la comprobación de la configuración es positiva,
entonces la CPU leerá el cargador en la ROM 2.

Si la CPU ha leído el cargador de la ROM 2,
entonces la CPU cargará el sistema operativo del disco.

Descripción de conexiones

La descripción de conexiones se muestra en la figura 1. Por añadidura el modelo causal sabe lo que significa para los dispositivos el estar conectados. Esto puede ser representado por una regla de la siguiente forma:

Si ambos dispositivos están correctamente colocados y
se puede transmitir una señal adecuada entre los componentes,
entonces los dos dispositivos están conectados.

Modelos de faltas

Si una ROM falla,
entonces la ROM está floja en su zócalo ($p = 0,8$)
la ROM no puede ser leída ($p = 0,2$).

Si una placa madre falla,
entonces deducir que la placa está mal insertada ($p = 0,8$)
que la placa está defectuosa ($p = 0,1$)
que los componentes están mal ($p = 0,1$).

Si los conmutadores no dan información correcta sobre la configuración del sistema,
entonces deducir que los conmutadores están mal colocados ($p = 0,9$)
que los conmutadores están abiertos ($p = 0,05$)
que los conmutadores no se están leyendo correctamente ($p = 0,05$).

Estrategia de diagnóstico

Si un dispositivo no está funcionando,
entonces ver si los dispositivos que le suministran las señales de entrada están averiados.

Si se sospecha que un dispositivo está defectuoso,
entonces ver si han fallado los dispositivos que usan las salidas de él.

estrategia de diagnosis. La tabla 2 presenta ejemplos de cada uno de ellos.

El modelo funcional describe cómo trabaja el sistema en circunstancias normales. Típicamente se compone de unidades (que a su vez pueden ser modelos funcionales a un nivel más detallado) y conexiones, que describen cómo fluye la información entre las unidades o dispositivos.

Los modelos de dispositivo describen las unidades que forman el sistema a diagnosticar, detallando el flujo de señales de entrada y salida de cada unidad y las transiciones de estado que pueden ocurrir en función de dichas señales. Como ejemplo consideremos un procesador (CPU) durante una prueba de encendido. Cuando se aplica la alimentación a la CPU, ésta trata primero de leer en la ROM 1 el programa de prueba de encendido, y después intenta ejecutarlo. Al comenzar tal ejecución, la CPU toma los datos sobre la configuración del sistema de un par de conmutadores basculantes y los compara con la configuración real. Si el resultado es aceptable, se lee el programa cargador en la ROM 2 y el sistema operativo se carga desde el disco.

Las descripciones de conexión son, en principio, similares a los diagramas esquemáticos de sistemas electrónicos. En el presente trabajo sobre construcción de modelos causales, por ejemplo, la descripción de conexión inicial venía definida por un diagrama esquemático o circuito. En la figura 1 se representa una de tales descripciones, que determina cómo están conectados los diversos componentes.

Los modelos de faltas describen la manera en que pueden fallar los componentes. Combinados con el modelo funcional, sirven para generar posibles comportamientos de fallos susceptibles de contrastarse con las reacciones observadas. Así, un modelo de faltas para una ROM indicaría que el funcionamiento es incorrecto cuando la ROM está floja en su zócalo o su contenido no es legible por otro dispositivo. Un modelo de faltas señala qué factores podrían causar los síntomas observados, aunque deban recogerse más datos para determinar la verdadera causa del fallo.

Por último, la estrategia de diagnosis dicta los pasos a seguir cuando se razona sobre el estado actual del sistema: cómo utilizar los modelos de dispositivo y de faltas, cuándo pedir más información a un técnico, qué procedimientos de prueba deberían emplearse, y qué información se necesita en una situación dada. Una postura básica podría consistir en sospechar de todos los componentes que anteceden funcionalmente a aquél donde se ha detectado una anomalía.

Los modelos causales se basan en descripciones lógicas sobre el modo de diseño y de operación del sistema, y no – como en los modelos superficiales – en asociaciones obtenidas de la experiencia operativa en dicho sistema. Tienen la ventaja de poder tratar problemas nuevos y ser construidos en paralelo con los sistemas a diagnosticar, aplicándose con ligeros cambios a otros afines. Dado que un dispositivo es utilizable en varios sistemas, los modelos causales pueden sacar partido de las partes comunes de tales sistemas, En el ejemplo escogido, las ROM podrían aparecer en ordenadores similares, realizando la misma función. Aunque el paso a un sistema afín implicara grandes alteraciones en una descripción de conexión, las variaciones en modelos de faltas serían mucho más leves, y los de dispositivo quizá no necesitarían cambio alguno.

Desarrollo futuro

El grupo dedicado a sistemas expertos en el ITT Advanced Technology Center está actualmente desarrollando la tecnología necesaria para el uso generalizado de los modelos causales. El foco inicial de esta actividad ha sido la diagnosis de faltas y reparaciones para los sistemas de conmutación de ITT. Las áreas de investigación son: control de profundidad de razonamiento, razonamiento cualitativo, y representación de estados y procesos.

En los modelos causales para ordenadores personales (Tabla 2), los componentes se representan a nivel de bloques funcionales, aunque es posible descender a niveles más detallados. Cada bloque funcional puede descomponerse en otros más pequeños, que a su vez podrían ser modelados a nivel lógico y, por debajo de ello, a nivel de circuito. Con esto se pretende *el control de la profundidad de razonamiento;* se pueden construir modelos causales para cada uno de estos niveles. Por ejemplo, un programa de simulación de circuitos valdría para construir modelos de dispositivo a nivel de circuito, sirviendo luego el sistema experto para insertar en ellos faltas y vectores de prueba y así obtener un modelo causal a ese mismo nivel.

Cuanto mayor es la profundidad a que se realiza el modelo causal, más exacta es la representación de lo que acontece en el dispositivo. Sin embargo, la cantidad de cálculo requerida crece con el número de niveles. Un modo evidente de salvar este obstáculo es iniciar el proceso de diagnóstico a un nivel de representación alto (menos detallado) y acudir a los niveles de

mayor detalle sólo cuando ellos proporcionen información necesaria. Por ejemplo, si los niveles altos permiten aislar en una unidad funcional una falta del sistema, los más detallados pueden servir luego para localizar el fallo con más precisión. O también, si en los niveles superiores no puede distinguirse entre dos faltas, los niveles bajos pueden aportar la información necesaria para diferenciarlas.

Decidir en qué punto se debe invocar un nivel de representación diferente, es un tema abierto para un desarrollo ulterior de los modelos causales. Actualmente en el ATC se trabaja con un modelo de dos niveles: en el superior, los bloques funcionales están al nivel de elementos reemplazables, mientras que en el inferior el sistema se representa por bloques funcionales comprendidos dentro de un elemento reemplazable. Siguiendo una táctica eliminatoria se identificarán los fallos obvios en el nivel alto, antes de razonar en el nivel inferior.

Hasta ahora el trabajo se ha concentrado en fenómenos discretos, en los que existe una señal con un corto número de estados posibles predefinidos (razonamiento cuantitativo pero no *cualitativo*). En algunas clases de diagnosis, surgen situaciones en las que se necesita razonar sobre cantidades continuas. Por ejemplo, un sistema para diagnosticar problemas de red debe ser capaz de razonar acerca de niveles de tráfico en diferentes partes de la red. Los razonamientos necesarios no son, sin embargo, rigurosos cálculos matemáticos del nivel cuya ejecución y posterior resumen no sería viable por su lentitud y por carecer de las medidas requeridas. Antes bien, lo que hace falta son métodos de razonamiento cualitativo, que trabajen con valores simbólicos tales como nivel "alto" o "bajo", "creciente" o "decreciente".

Los modelos causales aquí examinados tratan sobre conexiones entre los componentes y los tipos de señales que se propagan por estas conexiones; no describen el comportamiento en el tiempo de tales componentes, o sea la *representación de estados y procesos*. Esto restringe fuertemente el razonamiento que puede realizar el sistema. Un primer paso para suprimir tal restricción será incorporar la capacidad de representar dispositivos en determinados estados, lo cual equivale a un esquemático de circuito en condiciones estables de señales de entrada y con los conmutadores en posiciones fijas. Tarea mucho más difícil, aunque necesaria para diagnosticar fallos intermitentes y que dependen de su secuencia, es la de potenciar al sistema para que razone sobre el tiempo y sobre los procesos que se desarrollan en el tiempo.

Conclusiones

Los sistemas expertos para diagnosis y reparación de fallos son eficientes y económicos. Las aplicaciones actuales se apoyan en el uso de modelos superficiales, es decir, en sistemas expertos que captan las relaciones externas entre síntomas observados y componentes defectuosos. Puede lograrse un mejor resultado mediante los modelos causales, en los que el sistema experto utiliza modelos de la arquitectura y función del sistema a diagnosticar. El trabajo en curso se dirige a desarrollar y ampliar las posibilidades de los sistemas de modelos causales para facilitar su uso en productos ITT.

Michael E. Atwood obtuvo un grado BA en psicología y matemáticas por la Universidad de Kansas, y un PhD en psicología cuantitativa por la Universidad de Colorado. Trabajó en varios campos, incluyendo psicología del conocimiento, factores humanos y sistemas expertos, antes de entrar en el ITT Advanced Technology Center en 1983. El Dr. Atwood es un científico de la división de investigación de sistemas, donde trabaja sobre las aplicaciones de la inteligencia artificial a la diagnosis y reparación de sistemas.

Ruven Brooks se graduó BA en psicología por la Universidad de Michigan y MS y PhD por la Universidad de Carnegie Mellon, también en psicología. Ha trabajado en sistemas expertos durante más de 10 años, incluyendo sistemas para generar programas de ordenador, recomendaciones de terapias médicas y diagnósticos electrónicos de faltas, y ha sido uno de los que diseñó el programa educativo sobre inteligencia artifical y sistemas expertos de mayor éxito. El Dr. Brooks entró en el ITT ATC en 1982 como científico de la división de investigación de sistemas. Actualmente desarrolla nuevas arquitecturas de equipos y programas para inteligencia artificial y sistemas expertos.

E. R. Radlinski se graduó BS en matemáticas por la Universidad de San Buenaventura en 1970 y MS en informática por el Rensselaer Polytechnic Institute, en 1975. Ha trabajado en el desarrollo de paquetes integrados de herramientas de programación, y en el diseño de arquitecturas de sistemas avanzados para finanzas. Desde su entrada en el ITT ATC en 1980, el Sr. Radlinski ha investigado diseños alternativos para un interfaz de usuario adaptable, y en este momento está investigando sobre prototipos de sistemas expertos.

Consideraciones tecnológicas sobre el uso industrial de los sistemas expertos

Los sistemas expertos van a aumentar el abanico de aplicaciones industriales de los ordenadores, pero sus soluciones, prácticas y rentables, tendrán que coexistir con los equipos y sistemas actuales.

D. Neiman
ITT Advanced Technology Center, Shelton, Connecticut, Estados Unidos de América

Introducción

Hay un interés creciente en utilizar la tecnología de los sistemas expertos en aquellos problemas industriales que no han podido resolverse con los métodos convencionales de programación, así como en reducir el esfuerzo necesario para el desarrollo y mantenimiento de los programas. Sin embargo, muchas de las actuales herramientas de programación de IA (inteligencia artificial) se diseñaron en un entorno académico con el principal fin de obtener modelos del conocimiento, por lo que no son necesariamente eficaces en aplicaciones industriales, en las que el conocimiento se almacena para conseguir un trabajo útil del sistema.

Dado el origen académico de la mayoría de las herramientas y lenguajes, el diseñador de sistemas expertos deberá asegurarse de que el lenguaje y la herramienta que escoja sean adecuados para captar el conocimiento de la aplicación correspondiente. Además, el ordenador que utilice deberá disponer de dichos herramienta y lenguaje, así como el ordenador en que vaya finalmente a ejecutarse el sistema experto, una vez terminado. Por ejemplo, es frecuente desarrollar el sistema en un miniordenador, por su velocidad y disponibilidad de herramientas, y transportarlo después a un microordenador, que es más económico y permite la instalación en muchos lugares. Es, pues, necesario que el lenguaje o la herramienta puedan ejecutarse en ambas máquinas.

Aunque la tecnología de sistemas expertos es relativamente nueva, las metodologías para extraer el conocimiento de los expertos y codificarlo en forma de reglas están ya bien definidas. En consecuencia, el éxito de un proyecto de sistemas expertos es muy probable que dependa de aspectos como la selección de métodos y herramientas, del interfaz entre herramientas y aparatos y de la elección de equipos y programas.

El diseño de un sistema experto

Al estudiar la posible aplicación de los sistemas expertos a un determinado problema, debe primero decidirse si esa tecnología conviene al citado problema. Muchos casos pueden resolverse con técnicas más simples, mientras que otros resultarán demasiado complejos para las herramientas disponibles. En el proyecto deberá participar una persona experta, dispuesta a ofrecer sus conocimientos sobre el tema, que se le extraerán y codificarán después en lenguaje representacional. Seguidamente se escribirá el programa que aplique el conocimiento extraído al problema en cuestión.

Como en cualquier proyecto, el diseñador de sistemas expertos deberá plantearse ciertas preguntas:

- ¿Qué lenguaje de programación deberá utilizarse?
- ¿Cuál será el sistema de desarrollo (en equipo y soporte lógico)?
- ¿En qué ordenador se ejecutará el sistema definitivo?

Para llegar a resultados satisfactorios, hay que resolver las cuestiones anteriores en una fase temprana del proyecto.

Lenguajes

Una de las tareas más comprometidas para un diseñador de sistemas expertos es la selección del lenguaje. Se consideran, a grandes rasgos, tres niveles de lenguajes: los de IA de aplicación general, como el LISP, los que sustentan un método concreto de solución de problemas, como el OPS5 y Prolog, y los juegos de herramientas propios de los sistemas expertos.

El lenguaje de programación más utilizado en IA es el LISP, lenguaje de uso general, interactivo e interpretativo, con estructuras de datos potentes y flexibles.

Sin embargo, se le achaca con razón su ineficacia y el no ser estándar. Aunque la introducción de entornos de programación más potentes y la creciente popularidad del Common LISP hayan restado fuerza a estas objeciones, todavía mantienen su validez en aplicaciones en las que deba coexistir con sistemas convencionales de programación. En el campo de los sistemas expertos, el LISP presenta el serio inconveniente de que, aun siendo un lenguaje de IA, no es *per se* una herramienta de sistemas expertos, ya que carece de esquemas internos de representación del conocimiento y de estructuras de control.

Todos los lenguajes de sistemas expertos deben poseer capacidad de representar el conocimiento del experto de manera explícita. Un *sistema de producción* trata de encapsular el conocimiento en *reglas* y *memoria de trabajo*. En los *recuperadores deductivos*, se almacena en forma de hechos, que se combinan para crear otros nuevos o probar afirmaciones. En ambos casos, el conocimiento se codifica manualmente en la base de datos de los programas y no hay ningún aprendizaje. En cambio, en los sistemas *inductivos*, es el mismo programa el que crea las reglas a partir de ejemplos que se le suministran. Algunos conocimientos, en particular los relacionados con la estructura física y la causalidad, no tienen fácil expresión en reglas, y requieren un lenguaje de *representación del conocimiento*.

Sistemas de producción: OPS5

El OPS5 es uno de los lenguajes de sistemas de producción más conocidos. Un programa OPS5 consta de reglas que actúan sobre un conjunto de hechos almacenados en la *memoria de trabajo*. El formalismo del sistema de producción utilizado por el OPS5 procede de la investigación sobre la ciencia cognitiva. Lo que se pretende es que el conocimiento de un experto sobre un área determinada pueda ser extraído y almacenado como reglas que generalmente adoptan la forma SI <*condición*> ENTONCES <*acción*>.

El mecanismo subyacente en el OPS5 es la confrontación de patrones. Cada regla está diseñada para ejecutarse al cumplirse la condición que expresa su lado izquierdo. Por esta razón, el programador de OPS5 ha de enumerar un gran número de estados posibles del problema que se está resolviendo. Los lenguajes de los sistemas de producción por confrontación de patrones son muy útiles en las tareas de concatenación progresiva, como las de configuración o las que requieren reglas de accionamiento asíncrono. Actualmente se dispone del OPS5 en casi todos los ordenadores, desde los microordenadores hasta las unidades centrales.

Recuperadores deductivos: Prolog

El Prolog es el lenguaje de programación lógica más popular en la actualidad. Un programa Prolog almacena el conocimiento como afirmaciones y reglas, siendo el mecanismo subyacente la prueba de teoremas. Cuando se hace una afirmación en Prolog, el programa intenta probarla combinando hechos en la base de datos de acuerdo con el conjunto de reglas. Los recuperadores deductivos son generalmente útiles para tareas de concatenación regresiva, como el diagnóstico, que requieren formular múltiples hipótesis. Igual que con el OPS5, se dispone del Prolog en una amplia gama de máquinas, aunque sus características varían según el equipo utilizado.

Lenguajes híbridos de IA

Por lo general, tanto el Prolog como el OPS5 adolecen de un gran inconveniente: fueron concebidos para implantar estrategias concretas de solución de problemas y no como lenguajes de aplicación general. En gran parte, cualquier proyecto de programación además de resolver problemas incluye la codificación de los interfaces de usuario, rutinas de entrada/salida e inicialización de datos. Por lo general, estas tareas se codifican por algoritmos, y son difíciles de codificar en OPS5 o Prolog puros.

Está en desarrollo una nueva generación de lenguajes "híbridos" que combinan las propiedades de resolver problemas que poseen los lenguajes de sistemas expertos con las capacidades de programación de los lenguajes convencionales. En esta categoría se incluyen las herramientas OPS83 y Poplog.

El OPS83, basado en el lenguaje C, es el último y más potente miembro de la familia de programas OPS; además de aportar la metodología de un sistema de producción, permite al usuario programar en un lenguaje algorítmico semejante al C y al Pascal. El OPS83 es transportable, y puede ejecutarse en miniordenadores, terminales de usuario y ordenadores personales como el ITT XTRA*. El que un sistema experto implantado en OPS83 pueda transportarse y ejecutarse en un ordenador personal es

* Marca registrada del Sistema ITT

muy valioso, ya que el sistema puede instalarse de modo económico en todo el "mundo ITT".

La sintaxis del OPS83 se asemeja a la del C y del Pascal en grado suficiente para resultar familiar a los programadores expertos, aunque extrañen la sintaxis de las reglas de producción. El tiempo de aprendizaje es, por lo tanto, mínimo. Además, como el OPS83 está realizado en C, los programas OPS83 admiten fácil interfaz con los programas en C y en Fortran.

Lenguajes de representación del conocimiento

La mayoría de los sistemas expertos actuales almacenan el conocimiento en forma de reglas. Este método esencialmente empírico oculta, sin embargo, las relaciones entre causa y efecto. Hay una serie de *lenguajes de representación del conocimiento* – KLONE, FRL, FRAIL y MRS – que intentan captar la estructura subyacente del conocimiento que se está representando, y si bien todavía estan en fase de investigación, algunos de ellos empiezan ya a utilizarse en aplicaciones industriales.

Sistemas comerciales

Se dispone en el mercado de algunos juegos de herramientas para sistemas expertos, tales como el ART, el KEE, el S.1 y el Knowledge Craft. Aunque existen diferencias fundamentales en sus estrategias de implantación y en los formalismos subyacentes, todos ellos ofrecen un contexto muy eficaz para codificar y depurar las reglas. La mayoría de estos sistemas permiten elegir entre concatenación progresiva y regresiva, incluyen editores y rastreadores con modificación interactiva de reglas, permiten que el sistema se ejecute interactivamente e incorporan medios para explicación e interfaces gráficos. El funcionamiento suele ser muy bueno, siempre que se implanten en equipos potentes, pero son costosos y los programadores necesitan un largo periodo de capacitación.

El vehículo primario para el desarrollo y entrega de estas herramientas es la máquina LISP, procesador individual optimizado para ejecutar código LISP. Su entorno típico es un sistema operativo escrito parcial o totalmente en LISP, un intérprete y un compilador de LISP, y una colección de herramientas. Generalmente cuenta con un potente interfaz de gráficos, que ofrece al usuario un entorno de programación altamente interactivo.

El inconveniente de las máquinas LISP es que son terminales individuales costosos, cuyo uso eficaz requiere una amplia formación y experiencia del usuario. Los prototipos desarrollados en el LISP son difíciles de entregar al cliente, salvo que éste se disponga a comprar la máquina LISP y a formar al personal necesario.

Sistemas expertos basados en microordenadores

El mercado de microordenadores ha sido recientemente inundado de lenguajes para sistemas expertos, acompañado cada uno de ellos por extravagantes pretensiones. Dado que la mayoría de herramientas de IA requiere un gran volumen de procesamiento, es difícil tomar en serio estos mensajes publicitarios. Las herramientas basadas en ordenadores personales tienden a presentar problemas de ejecución, y por carecer de memoria virtual están limitadas al tamaño de la memoria física de la máquina.

Otra desventaja es su falta de transportabilidad. La mayor parte de los sistemas expertos basados en ordenadores personales solamente se ejecutan con el sistema operativo MS-DOS, lo que impide emprender los desarrollos en equipos más potentes.

Estos sistemas expertos tienen sus principales aplicaciones en la creación de prototipos y la formación de programadores. Hay muchas implantaciones de LISP y Prolog que pueden utilizarse para capacitar a los programadores. Además, siempre que el sistema operativo lo permita, puede utilizarse un ordenador personal para ejecutar sistemas expertos desarrollados en superminiordenadores o en los terminales.

Creación de un lenguaje de IA propio

Como alternativa razonable a la compra de un lenguaje de IA comercial cabe el escribir uno propio. Un lenguaje de IA propio permite especificar exactamente el diseño y la función del lenguaje, modificarlo a gusto de cada uno y distribuirlo libremente. Además, el sistema puede ajustarse con precisión a las necesidades de la aplicación. La desventaja de esta alternativa es que el tiempo de desarrollo es largo y exige un considerable trabajo de programación, documentación y soporte. Además, el coste puede exceder sustancialmente al de los productos existentes en el mercado, a no ser que la intención sea poner a la venta el lenguaje desarrollado, o que éste vaya a ser utilizado por muchas casas ITT.

Sin embargo, el desarrollo propio ofrece ventajas notables que pueden muy bien compensar su mayor coste. La primera

viene de que los lenguajes comerciales de IA no dan facilidades para la investigación, ya que no suele disponerse del código fuente. La segunda es que los algoritmos subyacentes del lenguaje pueden modificarse como se quiera. La tercera, que se pueden añadir módulos experimentales al sistema para implantar inducción o explicación. Por último, el proceso de creación de una herramienta de IA aumenta la experiencia del grupo de desarrollo, y ello beneficia a los proyectos futuros.

ITT ha elegido esta vía y ha producido un juego de herramientas que abrevia el desarrollo de una extensa gama de aplicaciones. Dicho conjunto de herramientas, denominado ESSAI, se está desarrollando en el ITT Europe Engineering Support Centre de Harlow, Inglaterra.

Experiencia en el desarrollo

El ATC (Centro de Tecnología Avanzada de ITT), ha producido varios prototipos de sistemas expertos para las Divisiones de ITT. La experiencia inicial con los proyectos DECIDES y SPOT aportó una valiosa profundización en las técnicas de adquisición del conocimiento y su formalización en un conjunto de reglas, pero puso de manifiesto la falta de idoneidad de los lenguajes y herramientas disponibles en aquel momento. Los proyectos recientes, como el Ingrid y la reimplantación del SPOT, se han basado en herramientas de mayor potencia y versatilidad. Combinando la experiencia de los primeros sistemas con el uso de las modernas herramientas se ha demostrado la posibilidad de desarrollar sistemas expertos para una amplia gama de aplicaciones industriales a un coste rentable.

DECIDES/OPS5

El proyecto DECIDES fue un intento inicial de crear un sistema experto para configurar la base de datos de la central digital Sistema 12 de ITT, y para su desarrollo se utilizó un miniordenador con el OPS5. Aunque el sistema realizaba la tarea de configuración adecuadamente, era lento y requería demasiada memoria.

SPOT/DUCK

El proyecto SPOT fue un estudio de la arquitectura de un sistema experto de diagnóstico que se aplicó a una especificación de LAN (red de área local) como ejemplo suficientemente complejo para ser interesante. La herramienta de desarrollo elegida fue el recuperador deductivo DUCK, implantado en el dialecto T del LISP, uno y otro desarrollados en Yale. Aunque muy potente, el DUCK resultó ser una herramienta de desarrollo pobre , ya que necesitaba enormes recursos de procesamiento y memoria, se ejecutaba muy despacio y carecía casi por completo de soporte. Además, el lenguaje de implantación subyacente no estaba suficientemente documentado y tenía un soporte escaso. La conclusión obtenida de este proyecto fue que las herramientas de investigación no suelen resultar eficaces en un entorno industrial.

Reimplantación del SPOT

La experiencia en el ATC con los sistemas expertos puso de manifiesto que las herramientas basadas en el LISP no son todavía prácticas en aquellas aplicaciones que no dispongan de estaciones de trabajo autónomas. Por consiguiente, la búsqueda de lenguajes adecuados se ha concentrado en herramientas que incorporen las metodologías de IA implantadas en lenguajes compilados. Uno de los requisitos era que los lenguajes fueran eficaces, con un buen soporte, económicos y compatibles con la mayor parte de los equipos utilizados por ITT. El OPS83 cumple estas exigencias básicas.

En el ATC, se ha utilizado el OPS83 para implantar dos sistemas. El primero es una nueva realización del programa SPOT. La conversión del DUCK al OPS83 exigió la identificación de reglas empíricas implícitas en el prototipo y la modelacion de la arquitectura LAN en las estructuras de la memoria de trabajo OPS83. Se comprobó que era posible realizar una correlación casi individualizada entre las reglas del DUCK y las del OPS83. Se necesitó un número adicional de elementos de memoria de trabajo para posibilitar la concatenación regresiva y las funciones de registro requeridas para el diagnóstico del sistema.

La conversión tan sólo ocupó cuatro semanas a un ingeniero, que estaba aprendiendo OPS83 al mismo tiempo. El desarrollo se facilitó notablemente por la posibilidad de utilizar una interacción de alto nivel para el OPS83, desarrollada en el ATC, que permitió al diseñador examinar la memoria de trabajo y el contenido del conjunto de reglas en conflicto. La versión final OPS83 del SPOT contenía unas 50 reglas y algunos centenares de líneas de código imperativo; esta versión se ejecutaba a dos órdenes de magnitud más deprisa que la versión DUCK y requería un orden de magnitud menos de memoria.

Ingrid

El segundo sistema implantado en OPS83 fue el Ingrid, un programa de configuración de paneles de control desarrollado para estaciones de bombeo fabricadas por la Flygt Corporation. Los controles existentes

o necesarios en una instalación de bombeo dependen de muchos factores, tales como el número y tamaño de las bombas, la tensión eléctrica y las necesidades del usuario. Muchas de las opciones que existen para el panel de control se excluyen mutuamente o presentan interdependencias con otras opciones. No siendo habitualmente el usuario final ni el vendedor expertos en la configuración de los paneles de control, la Flygt debe estudiar detalladamente cada especificación de pedido para detectar posibles incoherencias.

El sistema experto Ingrid se diseñó para que los vendedores pudieran generar la configuración de los paneles con un ordenador personal. Aunque era posible hacer un sistema similar con una programación convencional, se comprendió que un lenguaje de sistema experto como el OPS83 ofrecía muchas ventajas. Así, en efecto, las interdependencias entre opciones pueden expresarse con sencillez en forma de reglas y modificarse fácilmente cuando se producen cambios en el diseño; los elementos de la memoria de trabajo proporcionan una estructura interna de datos para almacenar las opciones de que dispone el usuario, y la arquitectura del sistema de producción aísla los componentes del sistema que dependen de los datos del problema, permitiendo la introducción de cambios en la base de datos sin alterar la estructura de control subyacente.

Para asegurar su facilidad de uso, el Ingrid incorpora un interfaz donde se utilizan abundantemente menús generados por un paquete comercial de gráficos. El desarrollo del interfaz entre el sistema experto y el del menú se simplificó merced a la capacidad del OPS83 de llamar a rutinas escritas en otros lenguajes de programación y admitir técnicas de códigos algorítmicos.

La Flygt está evaluando actualmente el sistema Ingrid, como paso previo para acometer las pruebas de campo.

Conclusiones

Existe una clara tendencia de abandono de los lenguajes interpretados y de avance hacia las herramientas de IA híbridas, que son más eficaces. Esto no significa que la industria deba olvidar el LISP, el Prolog, los lenguajes interpretados y las máquinas LISP, ya que la investigación sobre IA a gran escala tiene también su papel en el entorno actual. Sin embargo, para mantener esta investigación, las técnicas de IA no deben desligarse de los problemas de la "vida real".

El requisito de que los sistemas expertos deban coexistir con los equipos y sistemas operativos actuales, limita seriamente la elección de lenguajes y herramientas. Los productos deben implantarse con lenguajes híbridos, compilados, rentables y en los que primen la velocidad, la eficacia y la transportabilidad. Actualmente, pocas herramientas cumplen tales condiciones. Sin embargo, cada año se van introduciendo muchos lenguajes nuevos, que tendrán que ser evaluados según estos criterios. Las casas de ITT involucradas en la investigación y desarrollo de sistemas expertos analizan continuamente los nuevos lenguajes que están apareciendo, y difunden sus recomendaciones al resto de compañías asociadas en todo el mundo para garantizar que en cada aplicación de sistemas expertos se utiliza la herramienta más adecuada.

Daniel Neiman se graduó MS en ciencias informáticas en la Universidad de Connecticut en 1982. El mismo año, comenzó a trabajar en el Systems Research Division del ITT Advanced Technology Center, Shelton. El Sr. Neiman perteneció al grupo de sistemas expertos desde que se formó en 1984. El trabajo de este grupo consiste en investigar métodos y herramientas para la creación de sistemas de programación basados en el conocimiento.

En este número

Harvey, J. J.
Introducción a los sistemas expertos
Comunicaciones Eléctricas (1986), volumen 60, n° 2, págs. 100–108

La tecnología de los sistemas expertos es una rama de la inteligencia artificial que trata de captar la competencia de reconocidos expertos en un campo particular y poner ese conocimiento a disposición de otros. Pese a estar en su infancia tal tecnología, ITT la está utilizando con eficacia para resolver problemas que se resisten a las técnicas de programación convencionales. El autor presenta la tecnología y examina los elementos principales de un sistema experto: el interfaz de usuario, el mecanismo de inferencia y la base de conocimientos. Finalmente, examina la dirección probable de investigación en sistemas expertos, incluyendo el uso de modelos causales y sistemas que comparten la iniciativa con el usuario.

Harvey, J. J.
ESSAI, juego de herramientas para sistemas expertos
Comunicaciones Eléctricas (1986), volumen 60, n° 2, págs. 109–114

El desarrollo de sistemas expertos a menudo ha exigido al ingeniero del conocimiento la elaboración de unas técnicas de representación y control adecuadas. Igual que en la tecnología convencional de programación, el diseño de un sistema experto se acelera mucho disponiendo de herramientas apropiadas de ayuda al diseñador. El autor describe las características principales del juego de herramientas ESSAI que ofrece una completa gama de facilidades a los ingenieros de esta especialidad para construir sistemas expertos de forma rápida y eficaz con una gran variedad de aplicaciones.

Newstead, M. A.; Pettipher, R.
Adquisición de conocimiento para sistemas expertos
Comunicaciones Eléctricas (1986), volumen 60, n° 2, págs. 115–121

Como parte de la ingeniería del conocimiento, la adquisición de conocimiento se ocupa de captar las especiales aptitudes de resolución de problemas, a fin de utilizarlas en un sistema experto. Es un proceso a la vez complejo y lento, sobre todo porque los expertos tienen dificultad en expresar el conocimiento acumulado en muchos años de práctica y experiencia. Se usan principalmente dos técnicas en la adquisición de conocimiento: la artesana, donde el ingeniero de conocimiento prepara las reglas a mano, y la inductiva informatizada, con reglas inducidas automáticamente a partir de ejemplos. Los autores exponen estas técnicas con detalle e ilustran su uso en el diseño de un sistema experto para la diagnosis de fallos en placas de circuito impreso.

Jones, G.; Nuttall, R.; Stone, K.
Integración de múltiples esquemas de control
Comunicaciones Eléctricas (1986), volumen 60, n° 2, págs. 122–127

Los sistemas expertos son un buen método para resolver muchos tipos de problemas. Si se elige un esquema de control adecuado se pueden automatizar problemas complejos para los que se desconocen métodos algorítmicos o éstos son imposibles de calcular. Los sistemas expertos son también muy apropiados para problemas más simples que, aun teniendo métodos de solución algorítmicos, pueda cambiar su definición frecuentemente. Los autores describen las ventajas de tener un extenso conjunto de esquemas de control independientes del dominio, de modo que el ingeniero del conocimiento pueda seleccionar el más adecuado para la aplicación, sin tener que desarrollar uno nuevo cada vez. Un ejemplo es el ESSAI, juego de herramientas para construir sistemas expertos que se está desarrollando en el ITT Europe Engineering Support Centre.

Gunhold, R.; Zettel, J.
Pruebas en fábrica del Sistema 12
Comunicaciones Eléctricas (1986), volumen 60, n° 2, págs. 128–134

Los sistemas integrados de pruebas y diagnosis basados en conocimientos son hoy un componente importante del concepto de fabricación integrada por ordenador. Los autores describen un sistema experto en uso en las factorías de ITT en Stuttgart y Gunzenhausen para diagnosis de faltas en placas de circuito impreso. Se examina el proceso de diagnóstico para pruebas funcionales, la implantación y su estrategia, los resultados operativos y la rentabilidad. El artículo concluye con una ojeada a las aplicaciones futuras de los sistemas expertos en este área.

Schelfhout, H.
Ingeniería de aplicación para circuitos del Sistema 12
Comunicaciones Eléctricas (1986), volumen 60, n° 2, págs. 135–140

La ingeniería de aplicación es un proceso basado en el conocimiento que transforma los requisitos de una Administración telefónica en una central del Sistema 12 apropiadamente configurada. El autor describe un sistema experto diseñado para captar el conocimiento y experiencia de los ingenieros de aplicaciones, pudiendo utilizar y mantener la representación de dicho conocimiento los mismos ingenieros que participan en el desarrollo del sistema. Una gran ventaja de esta aplicación es la facilidad y rapidez con que se actualiza la base de conocimientos, siguiendo la evolución del diseño y los nuevos requisitos de la Administración.

Yun Cabrera, J.; Ketels, D. G.
Generación de datos para el Sistema 12
Comunicaciones Eléctricas (1986), volumen 60, n° 2, págs. 141–146

La capacidad de producción masiva de programas para centrales telefónicas como las del Sistema 12 de ITT contribuye apreciablemente a su rendimiento económico. Se tiende a que los programas sean reutilizables, ya que no es rentable tener que desarrollar un nuevo soporte lógico para cada nueva central. Este objetivo puede lograrse restringiendo las diferencias a la parte de datos de los programas y utilizando bases de datos embutidas para almacenar los datos de la central. Los autores describen cómo se crean los datos para cada central utilizando un conjunto de programas para poblar la base de datos y, en particular, el uso de lenguajes de sistemas expertos, muy adecuados para producir estos programas de población de datos a la vista de su estilo declarativo y capacidad de expresar dependencias entre relaciones.

Theuretzbacher, N.
Tecnología de sistemas expertos para sistemas de seguridad crítica en tiempo real
Comunicaciones Eléctricas (1986), volumen 60, n° 2, págs. 147–153

La tecnología de sistemas expertos se está utilizando para aumentar la seguridad en el sistema de enclavamiento electrónico para control de ferrocarriles ELEKTRA, de ITT Austria. Se incluye un sistema de contraseguridad que vigila continuamente la seguridad de la estación de ferrocarriles. El autor describe la realización del mismo como sistema de producción que utiliza un lenguaje de inteligencia artificial, y se ejecuta en el ordenador ITT-16-Plus bajo control del sistema operativo VOTRICS* resistente a fallos.

---

* Marca registrada del Sistema ITT

Thandasseri, M.
Sistemas expertos aplicados a centrales TXE4A
Comunicaciones Eléctricas (1986), volumen 60, nº 2, págs. 154–161

La tecnología de sistemas expertos ha alcanzado un nivel en el que es posible desarrollar sistemas para una amplia gama de aplicaciones. Un ejemplo reciente ha sido un sistema para puesta en servicio y mantenimiento de centrales TXE4A, descrito en este artículo. Tal sistema es capaz de diagnosticar fallos y el funcionamiento incorrecto de la central, y el autor indica las ventajas que ofrece en comparación con los sistemas convencionales de diagnóstico.

Benson, P.
Encaminamiento por inteligencia artificial en redes de paquetes vía radio
Comunicaciones Eléctricas (1986), volumen 60, nº 2, págs. 162–167

Las técnicas y conceptos de inteligencia artificial constituyen la base de un sistema autoadaptativo para establecer comunicaciones en redes tácticas de paquetes por radio sin nodos de control. El autor describe un sistema experto que ofrece una alternativa a las técnicas de encaminamiento mediante algoritmos prefijados. En esta solución, el sistema experto reúne conocimiento de la red extraído de la red local. Los caminos óptimos entre nodos se determinan por medio de algoritmos heurísticos de búsqueda. Las ventajas de este enfoque para las redes de paquetes por radio se fundamentan en la capacidad de aumentar el rendimiento, la mejor supervivencia en ambientes hostiles y una menor detectabilidad al necesitarse menos actividad radio para establecer las comunicaciones.

Gaudry, E.
Sistemas expertos aplicados a la guerra electrónica
Comunicaciones Eléctricas (1986), volumen 60, nº 2, págs. 168–173

ITT Avionics produce sistemas en tiempo real con microprocesadores "embutidos", para contramedidas electrónicas de a bordo. Para investigar la aplicabilidad a este entorno de las técnicas de sistemas expertos, se construyó un prototipo, escrito en Prolog, que realiza las funciones de control de vigilancia típicas de un sistema de contramedidas electrónicas. La autora describe su realización, incluida la selección de tareas y herramientas, la metodología del desarrollo, el interfaz de usuario y la formulación de reglas. Evalúa asimismo la idoneidad y las limitaciones del uso de las técnicas de sistemas expertos en la guerra electrónica, y sugiere áreas en las que esta experiencia prototipo podría aplicarse para el desarrollo de la programación.

Atwood, M. E.; Radlinski, E. R.
Arquitectura de sistema de diagnóstico
Comunicaciones Eléctricas (1986), volumen 60, nº 2, págs. 174–179

El desarrollo de sistemas de diagnóstico basados en la experiencia entraña un enorme trabajo, sobre todo por la mal estructurada naturaleza de la propia tarea. El diagnóstico requiere múltiples niveles de conocimientos generales y específicos del sistema, diversas estrategias, y está basado en modelos de razonamiento no muy bien definidos. La tarea se complica aún más al desconocerse las relaciones entre estos atributos y no haber un formalismo consolidado para expresarlas. Los autores comentan sus estudios de arquitecturas de este tipo de sistemas de diagnóstico. En especial describen el problema de diagnosis de fallos, presentan una arquitectura de sistema de diagnóstico experto y ofrecen un ejemplo del funcionamiento del mismo.

Atwood, M. E.; Brooks, R.; Radlinski, E. R.
Modelos causales: la nueva generación de sistemas expertos
Comunicaciones Eléctricas (1986), volumen 60, nº 2, págs. 180–184

Se llama sistemas expertos a aquellos programas que pueden "razonar" sobre un problema y que no se limitan, por ejemplo, a seleccionar una respuesta de una tabla de posibilidades. Los autores describen cómo está ITT desarrollando tales sistemas para ayudar al diagnóstico de faltas en sistemas de conmutación telefónicos; el propósito es detectar un porcentaje de errores mucho mayor del que permiten los equipos actuales. Se están estudiando dos alternativas: sistemas de modelos superficiales que usan reglas basadas en datos empíricos, previamente observados, para localizar las faltas y, en contraste, los modelos causales más potentes que basan su diagnóstico en "cómo trabaja el sistema", pudiendo así detectar errores no advertidos anteriormente.

Neiman, D.
Consideraciones tecnológicas sobre el uso industrial de los sistemas expertos
Comunicaciones Eléctricas (1986), volumen 60, nº 2, págs. 185–189

Las técnicas y herramientas derivadas de la investigación sobre los sistemas expertos son muy prometedoras en aquellas aplicaciones industriales que no se prestan a los métodos de programación convencionales. El diseñador debe definir la forma en que ha de representarse el conocimiento de un experto, los problemas que deben resolverse con las técnicas convencionales y la metodología para la resolución de los problemas. El autor analiza los diferentes aspectos que aparecen en la creación y entrega satisfactoria de un sistema experto.

Oficinas Editoriales

La correspondencia relacionada con las diferentes versiones de Electrical Communication debe dirigirse al editor correspondiente:

Rod Hazell
Electrical Communication
Great Eastern House
Edinburgh Way
Harlow, Essex
England

Wolfgang Schmid
Elektrisches Nachrichtenwesen
Lorenzstrasse 10
7000 Stuttgart 40
Bundesrepublik Deutschland

Antonio Soto
Comunicaciones Eléctricas
Ramírez de Prado, 5
28045 Madrid
España

Jean-Pierre Dartois
Revue des Télécommunications
30 Avenue George V
75008 París
France