1 Introducción a la ingeniería del conocimiento

Ingeniería del conocimiento

A. Rodríguez Rodríguez
J. Hernández Cabrera
A. Plácido Castro

Facultad de Informática
Universidad de Las Palmas de G.C.

Ingeniería del conocimiento

Introducción a la ingeniería del conocimiento
Volumen 1

Abraham Rodríguez Rodríguez
José Juan Hernández Cabrera
Ana María Plácido Castro

GIC S
Ingeniería del Software y del Conocimiento

Fotocopias de los documentos realizadas en el Dpto. de Informática y Sistemas el 28 de Septiembre de 1.994

Dirección: Campus de Tafira, Edificio Departamental de Informática y Sistemas y Matemática Aplicada

Marca: RANK XEROX **Modelo:** 5028
Nº de serie: 2116197190

ISBN: 84-8098-024-9 (Volumen I)
ISBN: 84-8098-023-0 (obra completa)

Índice

1 Evolución de los sistemas expertos

El objetivo de los científicos en IA ha sido siempre desarrollar sistemas dotados de ciertas capacidades de raciocinio, es decir, que fueran capaces de resolver problemas de una manera inteligente. Este es el motivo de la existencia de los sistemas expertos.

En los sesenta, la investigación en IA intentaba simular los procesos del pensamiento mediante métodos generales para resolver tipos genéricos de problemas. Pero el desarrollo de programas de propósito general era demasiado difícil y en último término de escasa eficacia. Por ello, durante los setenta la investigación se centró en desarrollar programas más especializados; se diseñaron técnicas de representación (cómo formular el problema para que fuera fácil de resolver) y de búsqueda (cómo controlar eficientemente la búsqueda de soluciones). Sin embargo, fue a finales de dicha década cuando se puso de manifiesto que la capacidad para resolver problemas de un programa no proviene de los formalismos y esquemas de inferencia que utiliza, sino del conocimiento que posee. Fue este hecho el que condujo al desarrollo de nuevos programas de propósito especial, sistemas que eran expertos en algún aspecto en un dominio específico. Estos programas se denominaron sistemas expertos.

Los Sistemas Expertos (SS.EE.) difieren en algunos aspectos cruciales en relación tanto con las técnicas tradicionales de programación y con otras áreas de Inteligencia Artificial. En contraste con los sistemas tradicionales de procesamiento de texto, las aplicaciones en IA poseen características como son la representación simbólica, la inferencia simbólica, y la búsqueda heurística. Este modelo simbólico es utilizado, al igual que hacen los humanos, para representar los conceptos del problema y aplicar varias estrategias y heurísticas para manipular estos conceptos. Los símbolos pueden combinarse para expresar relaciones entre ellos (estructuras simbólicas). Para resolver un problema, una aplicación en IA manipula estos símbolos en lugar de realizar computaciones matemáticas con ellos. La consecuencia de esta aproximación es que la representación del conocimiento - su elección, forma, interpretación de los símbolos utilizados - es crucial para el éxito del sistema. Pero también los SS.EE difieren de las otras 'ramas' de la IA en múltiples aspectos. Primero, realizan tareas difíciles con niveles de efectividad experta. Pero esto no es suficiente; los expertos humanos no sólo producen buenas soluciones, sino que normalmente las alcanzan rápidamente. Por lo tanto, un sistema experto debe aplicar su conocimiento para producir soluciones eficiente y efectivamente, utilizando los mismos 'trucos' y atajos que utilizan los

humanos para eliminar datos y cálculos inútiles. Para poder imitar con credibilidad a los humanos el sistema experto también debe ser robusto, en el sentido de que debe detectar datos de entrada incorrectos (semánticamente hablando) y reglas incompletas. Segundo, enfatizan las estrategias de resolución de problemas dependientes del dominio sobre los métodos 'débiles' más generales de la IA. Es decir, operan eficientemente en un dominio específico utilizando estrategias específicas, resolviendo problemas difíciles y complejos. Es importante tener en cuenta que los SE operan en dominios del mundo real, más que en dominios que los científicos de IA denominan 'de juguete' (toy-domains). En dominios del mundo real se utilizan datos reales con problemas prácticos y se generan soluciones que son rentables en algún sentido. En un dominio de juguete, el problema es normalmente una gran simplificación o una adaptación poco realista de algún problema complejo del mundo real. Tercero, emplean autoconocimiento para razonar acerca de su propio proceso de inferencia. El Meta-conocimiento posee una serie de ventajas entre las que se encuentran el que los usuarios tiendan a tener más fe en los resultados, más confianza en el sistema; que el desarrollo del sistema sea más rápido al ser más fácil de depurar; que las suposiciones acerca de la operación del sistema se hagan explícitamente en lugar de implícitamente; y que los efectos de un cambio en el funcionamiento del sistema sean más fáciles de predecir y comprobar. Y finalmente, resuelven problemas que caen dentro de algunas de las categorías siguientes: interpretación, diagnosis, predicción, depuración, diseño, planificación, monitorización, reparación, instrucción, y control.

1.1 Introducción Histórica

En 1977 en la conferencia internacional sobre IA, Feigenbaum describió la característica clave de los sistemas expertos: La potencia de un sistema experto deriva del conocimiento que posee, y no de los formalismos de representación y de los esquemas de inferencia que utiliza. Esta frase marca un cambio radical en el pensamiento de muchos investigadores en IA.

Las primeras épocas en IA estaban dominadas por la creencia ingenua de que con unas pocas reglas de razonamiento junto con ordenadores potentes se conseguirían rendimientos expertos y sobrehumanos. Según se ganaba en experiencia, la potencia limitada de las estrategias de propósito general para la resolución de problemas llevaron a la conclusión de que eran demasiado simples para resolver la mayoría de los problemas. Como reacción a las limitaciones que se observaban en las estrategias de propósito general, muchos investigadores empezaron a trabajar sobre problemas de aplicaciones específicas.

Así empezaron a surgir muchos sistemas expertos a mediados de los setenta. Algunos investigadores que se dieron cuenta del papel importante del conocimiento en estos sistemas, empezaron a desarrollar teorías sobre representación del conocimiento y los sistemas asociados de propósito general. En pocos años se hizo evidente el fracaso de dichos esfuerzos por motivos similares a los anteriores. El 'conocimiento' como objetivo del estudio es demasiado amplio y diverso; los intentos por resolver problemas basados en el conocimiento en general fueron prematuros. Por otro lado, algunas aproximaciones de representaciones del conocimiento demostraron ser suficientes en los sistemas expertos que las implementaron. La conclusión que se alcanzó de estas experiencias fue; el conocimiento experto proporciona la clave para el rendimiento del sistema experto, mientras que la representación del conocimiento y el esquema de inferencia proporcionan los mecanismos para utilizarlo. La búsqueda de representaciones del conocimiento generales o potentes, aunque sea aparentemente deseable, no tiene justificación empírica.

La figura 1 muestra algunas de las primeras líneas de investigación que se desarrollaron dentro del campo de los sistemas expertos. Muchos de dichos intentos han marcado el posterior desarrollo de sistemas expertos similares, y hoy día constituyen los ejemplos 'clásicos' dentro de la bibliografía de sistemas expertos. Cada uno de estos sistemas ha aportado alguna característica de las que hoy se consideran básicas en el desarrollo de las aplicaciones expertas, como una representación basada en reglas, o el algoritmo RETE, o una arquitectura Blackboard, etc.. El rastro de estas aplicaciones se pierde a comienzos de los años 80, tal vez sea porque es a partir de esta fecha que los sistemas dejaron de ser simples prototipos de investigación para convertirse en sistemas rentables a todos los niveles (incluido el económico).

El proyecto DENDRAL de la universidad de Stanford estuvo desarrollándose durante más de quince años. Este proyecto produjo dos sistemas, DENDRAL y METADENDRAL. DENDRAL analiza la masa espectrográfica, la resonancia magnética nuclear, y otros datos químicos para inferir posibles estructuras de moléculas desconocidas. Utiliza una variante de 'generar y testar' en la resolución de los problemas. Su generador puede enumerar todas las estructuras orgánicas que satisfagan los requisitos que aparecen en los datos mediante una generación sistemática y parcial de las estructuras moleculares que son consistentes con los datos, elaborándolas entonces de todas las formas posibles. Evita la búsqueda exponencial mediante la eliminación rápida de las subestructuras imposibles. Este sistema no manipula los principios básicos de la química, ni posee facilidades de justificación o explicación del razonamiento que sigue. Cualquier modificación sobre la base de conocimiento implica una reprogramación de todo el sistema.

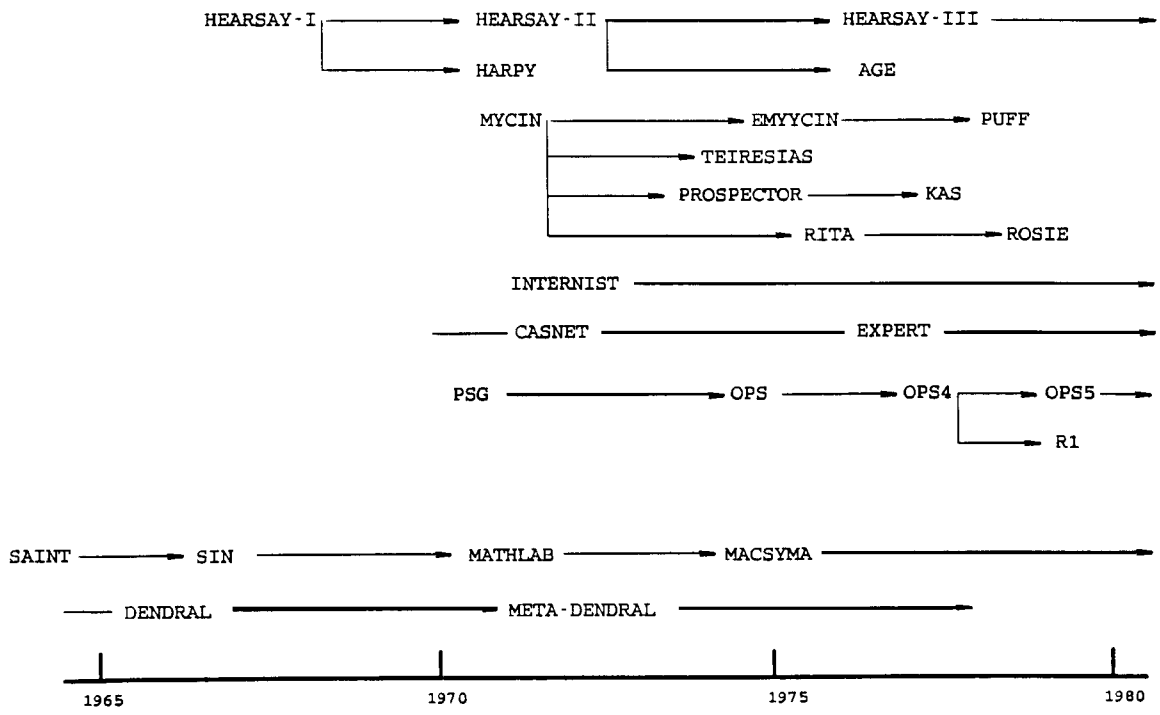

Figura 1

Para facilitar la adquisición de nuevo conocimiento se desarrolló el METADENDRAL. Este añade conocimiento de análisis al DENDRAL, proponiendo y seleccionando reglas de fragmentación para las estructuras orgánicas. DENDRAL supera a todos los humanos en su tarea, y como consecuencia, ha provocado el que se redefinan las reglas de los humanos y las máquinas en la investigación química.

Otra línea de investigación comenzó con SAINT, y culminó con MACSYMA, un sistema experto para matemáticas simbólicas. Al igual que [illegible] MACSYMA supera a la mayoría de los humanos. Realiza cálculo diferencial e integral simbólicamente, simplificando dichas expresiones. MACSYMA no implica búsqueda; identifica el tipo de problema que quiere resolver comparándolo con una tabla de problemas 'tipo', asociándole un método de solución. El SIN fue el primero en introducir un comparador semántico de expresiones (ej.: $(7x+3)*2 = (28x + 12)/2$).

EXPERT es un lenguaje de construcción de sistemas expertos que ha evolucionado de CASNET, un sistema experto para la diagnosis y tratamiento del glaucoma. EXPERT se ha utilizado principalmente en la construcción de sistemas de consulta en oftalmología, endocrinología, y reumatología. Ambos se apoyan en un modelo de datos dirigido por hipótesis, más un sistema causal implementado por una red semántica. También incluyen facilidades de justificación y gestión de la incertidumbre mediante pesos asociados a los

enlaces de las cadenas causales. Estos sistemas se componen de:

- Hipótesis: son conclusiones inferidas con subhipótesis, lo que da lugar a distintos niveles de abstracción. Es decir, una jerarquía de conceptos con niveles de detalles cada vez menores a medida que se baja por la jerarquía.
- Encuentros (findings): son los hechos.
- Reglas de relación: entre FF, FH, y HH.

CADUCEUS (originalmente INTERNIST) de la universidad Carnegie Mellon, y MYCIN de Stanford resuelven distintos tipos de diagnosis médica. CADUCEUS consiste en una gran red semántica de relaciones entre síntomas y enfermedades de medicina interna. En 1982 el sistema tenía unas 100.000 asociaciones que representaban aproximadamente el 85% de todo el conocimiento importante. Razona combinando el razonamiento dirigido por datos y el dirigido por hipótesis. Se utilizan primero los datos del paciente para predecir una hipótesis, y esta se estudia para predecir otras manifestaciones que confirmen o rechacen dicha hipótesis. El sistema se complica por la posible convivencia de distintas enfermedades que pueden enmascarar determinados síntomas. En 1989 se utilizaba con fines educativos.

MYCIN aborda el problema del diagnóstico y el tratamiento de enfermedades infecciosas de la sangre. Su conocimiento se resume en 400 reglas que relacionan las posibles condiciones con sus interpretaciones asociadas. En su estrategia de resolución de problemas, MYCIN comprueba las condiciones de una reglas con los datos disponibles o pide algún dato a los usuarios. Si fuera necesario trata de inferir la certeza o falsedad de las condiciones de otras reglas. El rendimiento de MYCIN ha sido considerado por expertos en el área de trabajo como igual o superior al de los expertos humanos. El sistema se componía de:

- Sistema de consulta: genera un conjunto de hipótesis relacionadas con el organismo en estudio con su correspondiente grado de incertidumbre. A partir de estos recomienda una terapia a seguir. El sistema de consulta esta formado por una serie de estructuras en forma de reglas IF .. THEN, las cuales tienen asociadas un coeficiente de certeza. El control del sistema lo realiza un algoritmo Backward.
- Sistema de justificación de conclusiones (How) y de explicación del razonamiento (Why).
- Sistema de adquisición de reglas: mediante un editor de reglas en la interface de usuario.

TEIRESIAS es un programa evolucionado a partir de MYCIN. Es un sistema que coopera en la construcción de bases de conocimiento muy grandes ayudando a transferir el conocimiento experto desde el experto humano a la base de conocimiento. Los expertos

dialogan con TEIRESIAS en un subconjunto del lenguaje natural. Detecta errores como reglas que no se disparan nunca, que se solapan (generando inconsistencias o redundancias), o que se han quedado obsoletas.

PROSPECTOR utiliza una representación del conocimiento similar a MYCIN (reglas IF-THEN) para especificar el conocimiento de los depósitos de minerales. Utiliza distintas bases de conocimiento para diferentes tipos de depósitos. Al igual que MYCIN determina el diagnóstico más probable calculando el grado de verosimilitud de cada condición antecedente. Este proceso se aplica recursivamente hasta que se han recogido y combinado heurísticamente los datos de todas las condiciones necesarias. En la universidad de Stanford se ha desarrollado una versión independiente del contexto llamada EMYCIN. Esta contiene todo MYCIN excepto el conocimiento acerca de las enfermedades de la sangre. EMYCIN ha facilitado el desarrollo de aplicaciones de diagnóstico similares, como PUFF.

ROSIE proporciona un sistema de programación de propósito general para construir sistemas expertos. ROSIE es una evolución de un programa anterior llamado RITA. Ambos sistemas se apoyaron en el éxito que tuvo la representación del conocimiento orientada a reglas de MYCIN junto con el atractivo que tienen para los usuarios y los diseñadores las facilidades de justificación y programación en un lenguaje similar al inglés. No posee un conocimiento del léxico de las palabras, sino de la lógica de la construcción de sentencias del inglés.

Los primeros trabajos desarrollados en la universidad Carnegie Mellon con el PSG, un sistema de producción para estudiar y modelar el conocimiento humano, llevaron al desarrollo de la serie de lenguajes de sistemas de producción OPS, y al R1 (ahora XCON), un sistema experto para configurar sistemas de computadores DEC VAX. R1 representa la aplicación con más éxito desarrollada en OPS. Contiene más de 3000 reglas e información de más de 5000 componentes. Razona utilizando una estrategia Botton (componentes) - Up (configuración). El OPS5 implementa el RETE, el cual es, quizás, el algoritmo forward más eficiente de los implementados hasta la fecha.

La última línea a comentar la constituye la de los sistemas de comprensión del habla, especialmente el sistema HEARSAY-II. HEARSAY-II, desarrollado en la universidad Carnegie Mellon, fue uno de los dos primeros sistemas capaces de entender un discurso conectado con un vocabulario de 1.000 palabras. Aunque sus rivales en habilidad son los niños de diez años, no se le puede aplicar los criterios de eficiencia de los otros sistemas expertos. La comprensión del habla es una de las tareas más difíciles que se han abordado con cierto éxito en el campo de los sistemas expertos. Las principales características del

HEARSAY-II incluyen múltiples especialistas que cooperan entre sí, una resolución de problemas en diferentes niveles de abstracción, evolucionando desde lo abstracto y añadiendo lo preciso y lo localizado, y un desarrollo incremental de soluciones parciales que aprovecha oportunísticamente datos clave o conocimiento crítico. A partir del HEARSAY-II se han desarrollado dos estructuras de propósito general para ayudar a construir sistemas expertos: AGE en Stanford, y HEARSAY-III en ISI.

2 Elementos de un sistema experto

A medida que los investigadores desarrollaban nuevos esquemas de representación y mecanismos que los controlaran, se iban manifestando las diferencias que existen entre las distintas partes del sistema. El conocimiento específico de la aplicación se organizaba de distinta manera que otro tipo de conocimiento como el de como resolver el problema, interactuar con el usuario, o aprender nuevas heurísticas. Esta organización facilitaba la manipulación del conocimiento del sistema tanto por el ingeniero del conocimiento como por las otras partes de la aplicación.

La estructura resultante estaba compuesta de una base de conocimiento y de un motor de inferencia. La primera organiza todo el conocimiento del sistema mientras que la segunda controla la estrategia a seguir en la resolución de los problemas. La figura 2 muestra un esquema con dicha estructura.

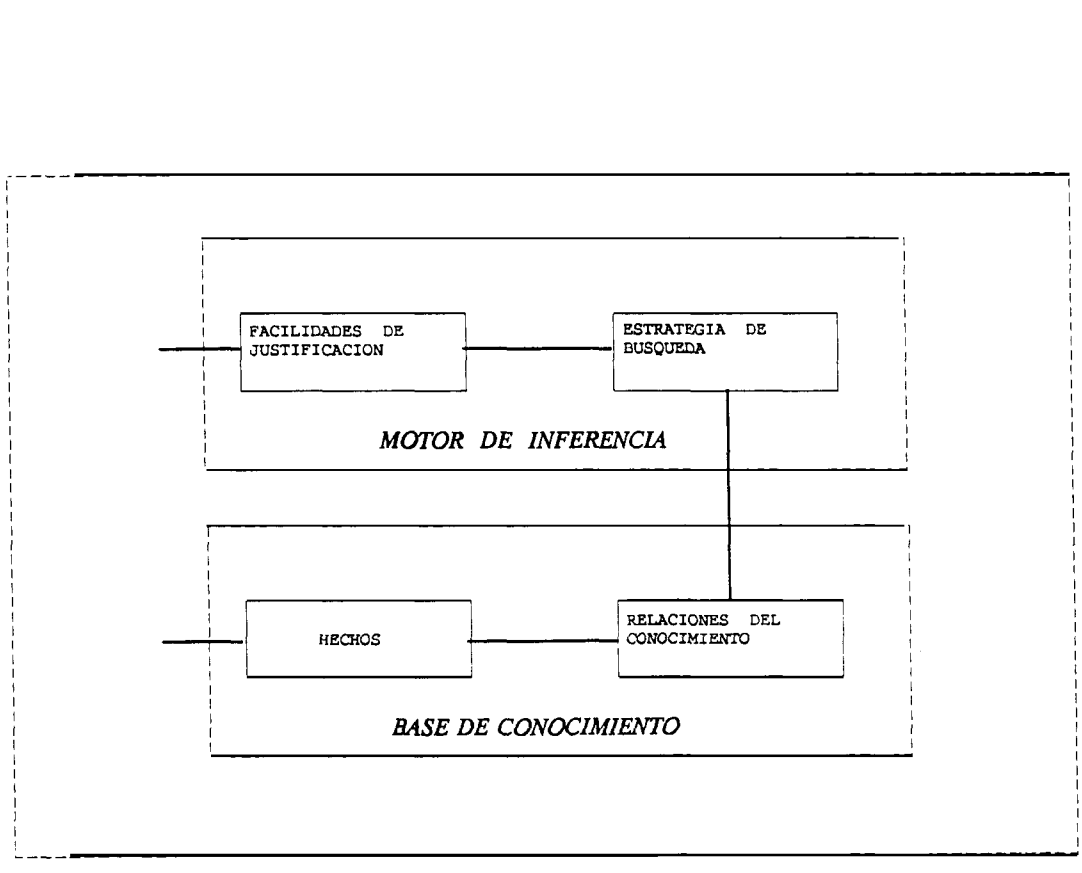

Figura 2

2.1 Base de conocimiento

El 'conocimiento' ha sido representado de múltiples maneras en función de sus particularidades y las del dominio sobre el que trata. Sin embargo, podríamos identificar dos partes que aparecen independientemente de cual sea la estructura representacional que se utilice: una base de hechos y las relaciones entre ellos.

Estas estructuras pueden ser algo tan simple como una red de reglas de la forma IF ..... THEN ...., en las que cada regla implementa una porción de conocimiento heurístico. La potencia de cada una de las reglas es variable, en el sentido que algunas son capaces de resolver completamente una tarea, mientras que otras necesitan agruparse para que puedan generar un resultado con un mismo nivel de abstracción o de contenido semántico.

Las redes causales implementan el conocimiento del dominio de forma similar a las reglas (realmente es otra manera de 'ver' una base de reglas). En esta ocasión los nodos representan las restricciones o las acciones de cada situación, mientras que los enlaces relacionan las condiciones con sus consecuencias.

Otra forma de afrontar el problema es con redes semánticas, las cuales suelen consistir en una implementación de la lógica de primer orden (similar al PROLOG) en la que los nodos son los sujetos y los enlaces son las relaciones que existen entre ellos (verbos).

Los marcos (frames) implementan el conocimiento como un árbol jerárquico de objetos en la que los conceptos más abstractos van en la parte superior mientras que los más específicos se sitúan en los niveles inferiores. De esta manera se pueden representar conceptos complejos mediante la descomposición del mismo en sus partes [illegible] o enumerando todos los subconceptos que pertenezcan al elemento más abstracto. En el primer caso la relación que existe entre el concepto 'padre' y sus 'hijos' es de *parte-de*, y se suele corresponder con los últimos niveles de la jerarquía, o menos abstractos. El resto de los enlaces representan la relación *es-un*. Cada concepto posee una serie de propiedades que lo definen. Estas pueden ser simples valores, o llamadas a procedimientos, demonios que tengan algún efecto sobre otra zona del árbol, o cualquier otro mecanismo que modifique la estrategia de resolución que el sistema utiliza por defecto. Parte de la potencia de los frames radica en que existen reglas de herencia de estas propiedades entre hijos y padres que permiten una estructuración más eficiente del conocimiento.

Los sistemas CBR (razonamiento basado en casos) representan la información simplemente mediante listas de características, dejando el resto del trabajo al sistema de inferencia.

Estas no son las únicas posibilidades de representación del conocimiento. Las representaciones lógicas tienen diversas modalidades como la lógica difusa, la modal, etc.. Muchos sistemas utilizan sistemas híbridos que combinan algunas de las aproximaciones anteriores. En cualquier caso, la selección de un formalismo de representación u otro va a condicionar el método de resolución de problemas a utilizar.

2.2 Motor de inferencia

La base de conocimiento es única para un dominio determinado, pero el motor de inferencia puede ser común a varios tipos de dominios con características similares. Así, un motor de inferencia que inicialmente fue desarrollado para una aplicación en particular puede ser utilizado en otra con sólo sustituir el conocimiento específico de la aplicación en cuestión.

El paradigma de resolución del problema y sus métodos organizan y controlan los pasos a seguir para resolver el problema. Uno de los paradigmas más usados y potentes implica el encadenamiento de reglas para formar una línea de razonamiento. Si la cadena parte desde un conjunto de condiciones y se mueve hacia alguna posible conclusión, el método se denomina encadenamiento forward. Si se conoce la conclusión, pero no el camino que lleva a esa conclusión, el método se denomina encadenamiento backward.

El problema que presenta el encadenamiento forward, cuando no se tiene una heurística adecuada para podar, es que se podría derivar todo lo posible aunque no hiciera falta. El encadenamiento hacia atrás va de metas a submetas. Aquí, el problema es también la falta de control sobre la heurística del problema, lo que puede desembocar en una explosión combinatoria de posibilidades. De todo esto se deduce que es preciso encontrar una heurística apropiada al dominio, así como los esquemas de representación y control adecuados para que se pueda construir un sistema que sea eficiente y efectivo. El conocimiento del dominio de la tarea controla los pasos a seguir en la resolución del problema. A veces el conocimiento es bastante abstracto (por ejemplo un modelo simbólico de "como funcionan las cosas"). La inferencia que parte de las abstracciones del modelo y desciende a sentencias más detalladas (menos abstractas) se denomina inferencia dirigidas por modelos (model-driven). Cada vez que se desciende desde una sentencia simbólica abstracta a una (o varias) sentencia menos abstracta se genera una expectativa, y al comportamiento de la resolución del problema se le denomina 'dirigido por expectativas'. Sin embargo, a menudo se trabaja en sentido inverso; desde detalles o datos específicos del problema hacia niveles mayores de abstracción (dirigido por datos o data-driven). Si el próximo paso a tomar se decide en base a nuevos

datos recibidos o en base al paso anterior, entonces se está respondiendo a eventos, y la actividad se denomina 'event driven'.

Independientemente de la estrategia de control que se quiera utilizar y de la representación que se le quiera dar a los datos, un sistema experto siempre estará compuesto por una base de conocimiento y un motor de inferencia. Una vez los científicos de Inteligencia Artificial se dieron cuenta de este hecho, se preguntaron si la misma estructura (representación y control) que se utilizaba para un sistema experto podría valer en otro sistema experto de características similares. De esta manera tomaron los sistemas expertos desarrollados hasta la hecha y les extrajeron todo el conocimiento dependiente de la aplicación. Así es como surgieron las herramientas para la construcción de sistemas expertos.

3 Herramientas

3.1 Origen

Los diseñadores crearon los lenguajes de programación tradicionales para implementar soluciones basadas en algoritmos claramente definidos; sus estructuras de control, facilidades computacionales, y orientación procedural así lo refleja. Para codificar el conocimiento experto los ingenieros necesitan unos elementos de programación diferentes; nuevos elementos que constituyan un lenguaje de representación del conocimiento que pueda expresar conocimiento que comprenda una gran cantidad de hechos y heurística más que unos algoritmos rígidos.

Esencialmente, los ingenieros del conocimiento deben expresar en el lenguaje de representación formal del conocimiento lo que ellos descubren a través de las entrevistas con los expertos. Dependiendo de la herramienta utilizada, un ingeniero del conocimiento puede desarrollar las habilidades operacionales necesarias para realizar este trabajo en un tiempo de entre seis a ocho semanas. Esto depende de la expresividad del lenguaje; cuanto más conceptos naturales (intuitivos) puedan ser incluidos en el lenguaje de representación del conocimiento, más fácil será la tarea. Las herramientas que poseen unas estructuras ricas de representación mejoran la productividad al liberar a los ingenieros del conocimiento de tener que crear construcciones artificiales cuando simbolizan los conceptos.

Originalmente, los sistemas expertos se desarrollaban en lenguajes de Inteligencia Artificial como Lisp. Sin embargo, los constructores orientaban sus esfuerzos hacia lenguajes de propósito especial sobre Lisp, ayudándoles a controlar más directamente las construcciones del conocimiento y los mecanismos de razonamiento. Las estructuras (frameworks) de sistemas expertos facilitan la tarea de la ingeniería del conocimiento al permitir que la información de alto nivel se pueda codificar de una forma que se asemeja a la manera que tienen los expertos de expresar el conocimiento.

Existen en el mercado una gran diversidad de herramientas para la construcción de SS.EE., siendo algunas más específicas que otras. Difieren entre si en la complejidad y en el rango de problemas que pueden resolver. Las hay desde las que implementan distintas estrategias

de representación y de inferencia, hasta las que sólo implementan un único modelo simbólico. También difieren en las posibilidades de interacción con el entorno, como la capacidad de acceso a bases de datos externas o cualquier otro paquete especializado de software, de control de los sistemas de comunicación, de utilización de placas controladoras de dispositivos, etc.., y en las facilidades de desarrollo y depuración de aplicaciones, como trazas, chequeo de sintaxis, etc...

3.2 Elementos

Independientemente de la aplicación, una herramienta idealmente eficiente debe incluir los siguientes elementos de programación:

- Representación Simbólica
- Puntos de Vista
- Mantenimiento de la Realidad
- Acceso Externo
- Entorno de Desarrollo

3.2.1 Modelo Simbólico

El corazón de cualquier sistema experto es el su modelo simbólico del dominio en el cual razona y de cada instancia de problema que afronta en determinado momento. Un modelo simbólico contiene el estado actual de un sistema experto 'orientado por datos'. A diferencia de la programación procedural convencional, ahora no existe el típico 'contador de instrucción' (program counter) durante la ejecución del programa. Por lo tanto, la riqueza del modelo simbólico determina con qué profundidad un programa entiende tanto su situación actual como su contexto actual de procesamiento. Las buenas representaciones simbólicas soportan un número arbitrario de relaciones n-arias entre objetos especificados por símbolos, y proporciona posibilidades de una organización taxonómica del objeto y sus relaciones. La representación simbólica se complementa con un lenguaje de comparación de patrones (pattern-matching) que permite a las reglas identificar subestructuras dentro del modelo de representación. Por ejemplo, la relación (es-un ?variable partido-político) permitiría la identificación de cualquier concepto que tuviera esa estructura de representación, activándose una vez por cada instancia de partido político que existiera en la base de hechos. Estas (sub)estructuras que definen la situación actual del problema, serán las que recomienden la

acción de resolución a adoptar, como podría ser la realización de una pregunta, o el acceso a una base de datos, o la especificación de otra estructura a identificar, etc ... En definitiva, un modelo simbólico viene caracterizado por unas estructuras de representación más unas estratégias de resolución de problemas.

Los sistemas expertos son datos mas que procedimientos. En lugar de prescribir alguna secuencia de acciones, los ingenieros del conocimiento representan el conocimiento especificando unos hechos y utilizando alguna estructura para definir sus relaciones. El que una estructura se active, sólo depende de los datos que contenga en el momento de la ejecución - los ingenieros del conocimiento no saben ni les preocupa el momento en el que una estructura se activará. Los datos seleccionan al procedimiento adecuado, en lugar de que el procedimiento seleccione el dato apropiado -lo que se opone exactamente a lo que es la programación tradicional. El motor de inferencia accede automáticamente a todo el conocimiento relevante independientemente del estado del procesamiento. Estas estructuras pueden ser reglas, redes semánticas, marcos, casos, etc...

Debe existir algún mecanismo que opere sobre las estructuras de representación definidas en la herramienta. Esta condicionará las posibilidades de alcanzar conclusiones en el sistemas, así como las de justificar el razonamiento que sigue durante la resolución. Los métodos de inferencia más conocidos son los que operan sobre los sistemas de producciones.

Al igual que la lógica inductiva, el encadenamiento forward razona hacia delante a partir de hechos existentes y reglas para derivar hechos adicionales que puedan surgir, mientras se siguen todas las posibilidades sugeridas por los datos. De la misma forma que la lógica deductiva, el encadenamiento backward razona hacia atrás a partir de una meta determinada, buscando en la base de conocimiento hechos o reglas que apoyen dicha meta y permita que se la identifique como verdadera. Ambos métodos funcionan bastante bien. Muchos sistemas expertos incorporan uno u otro. Sin embargo, cuando un motor de inferencia integra ambos métodos sus rendimientos mejoran notablemente. Inicialmente, el encadenamiento backward actúa para dirigir o restringir al encadenamiento forward en un subconjunto de la base de conocimiento determinado por metas conocidas. Entonces, el encadenamiento forward alcanza rápidamente conclusiones o genera nuevas metas para continuar con el proceso. La acción combinada forward-backward permite que los sistemas expertos converjan a las soluciones mucho más rápidamente y eficientemente de lo que lo haría cada método por separado.

La organización del conocimiento es un aspecto vital en los SS.EE., y esto se refleja en las distintas políticas de organización que presentan las herramientas. En la organización del conocimiento hay que considerar dos aspectos: aquel que se refiere a la organización de los

datos en memoria; esto son conceptos y relaciones que puedan existir entre ellos. Y por otro la organización de las funciones que implementa el SE. En lo que respecta a los conceptos, por ejemplo, una organización jerárquica permitiría a los IC organizar el conocimiento como una colección de objetos que comparten algunas propiedades. Debido a que el esquema es jerárquico, los objetos pueden heredar características desde una clase de objeto más general y entonces añadir sus particularidades propias. Al permitir las abstracciones del conocimiento, los esquemas gestionan eficientemente la complejidad. Los ingenieros del conocimiento pueden representar conceptos abstractos directamente como descripciones de propiedades comunes a cada instancia de una clase abstracta. La modularización de las distintas tareas del SE facilita el desarrollo y la depuración del sistema, a la vez que minimiza los posibles perjuicios provocados por la activación de una estructura de conocimiento en un momento inadecuado.

3.2.2 Puntos de Vista

Normalmente, los sistemas expertos deben razonar bajo suposiciones o acerca de situaciones que implican incertidumbre. Los puntos de vista (viewpoints) liberan al sistema de comparar múltiples hipótesis alternativas hasta que pueda seleccionarse una con un grado de certeza. Así, cuando una situación sugiere dos posibles consecuencias o interpretaciones, automáticamente se generan dos puntos de vista para analizar los subsiguientes razonamientos acerca de ambas posibilidades.

Además de permitir que los sistemas expertos sigan múltiples caminos de razonamiento simultáneamente, el mecanismo de puntos de vista mantiene la traza de las suposiciones y de los éxitos a lo largo del razonamiento. Cuando una alternativa conduce a una contradicción, se deja de considerar. Cuanto una alternativa es claramente superior, se selecciona y las alternativas restantes se abandonan. Las reglas codificadas en la base de conocimiento determinan qué alternativas, de entre las que compiten entre sí, será la elegida.

3.2.3 Mantenimiento de la Realidad

Cuando las suposiciones iniciales demuestran ser falsas, las conclusiones que se deducían a partir de ellas se marcan automáticamente como erróneas y se excluyen de los hechos o condiciones posibles. Por lo tanto los sistemas expertos pueden alterar los resultados durante la consulta, modificando las conclusiones en respuesta a nuevos datos. Esto es un ejemplo donde los sistemas expertos están realmente dirigidos por los datos, ya que ellos reaccionan

ante cambios en los datos más que siguen ciegamente el contador del programa con una secuencia procedural.

El mantenimiento de la realidad, también se aplica a los puntos de vista. Cuando las alternativas (puntos de vista) se generan a partir de conclusiones que posteriormente demuestran ser falsas debido a alguna condición que la altera, los puntos de vista se eliminarán automáticamente.

3.2.4 Acceso Externo

Ya han pasado los días en que había que supervisar todo el proceso de funcionamiento del SE en cada consulta. El usuario introducía directamente los datos para que el sistema realizara su tarea. Hoy la mayoría de los SS.EE. son autónomos. Acceden directamente a los dispositivos que necesitan y actúan en consecuencia. La mayoría de las herramientas facilitan que esta tarea se pueda realizar mediante la aportación de los interfaces necesarios para que la aplicación pueda acceder a bases de datos, hojas de cálculo, paquetes de software estadísticos, gráficos, dispositivos exteriores, multimedia, etc..

3.2.5 Entorno de Desarrollo

Y por último el entorno de desarrollo. Cada vez más son mayores las ayudas que proporcionan las herramientas acerca de su modo de funcionamiento, de forma que el IC tenga acceso a los parámetros del sistema y pueda adaptarlos a la aplicación que está desarrollando. Funciones específicas de depuración, visualizadores cruzados (browsers), estadísticas de funcionamiento y de utilización de recursos, etc.. son las características de las que dispone el IC para desarrollar su trabajo más eficientemente.

3.3 Selección

La elección de la herramienta (y de sus características) depende en gran medida de la aplicación que se pretenda construir. Los problemas que hay que afrontar son totalmente distintos para cada aplicación de SE, y ninguna herramienta de las existentes en el mercado cubre todas las posibilidades de implementación. Es necesario realizar un cuidadoso proceso de selección de la herramienta más adecuada para cada aplicación que se pretenda

desarrollar. La selección debe ser lo más formal posible, eliminando cualquier decisión subjetiva mediante tests o pruebas sobre las características de las herramientas en evaluación. Además, en la evaluación debe quedar reflejado de alguna manera las particularidades de la aplicación que se pretende desarrollar con la herramienta. Otros consejos que pueden ser útiles a la hora de seleccionar una herramienta son los siguientes:

- No seleccionar una herramienta con más generalidad de la necesaria.
- Comprobar la herramienta con un pequeño prototipo en las primeras fases del desarrollo.
- Elegir una herramienta que posea un servicio de mantenimiento y actualización por el fabricante.
- Elegir una herramienta con facilidades de justificación e interacción cuando la velocidad de desarrollo sea crítica.
- Utilizar las características del problema para determinar las de la herramienta.

4 Tareas típicas de sistemas expertos; ¿ para que se utilizan?

Los sistemas expertos se construyen para resolver diferentes tipos de problemas, pero sus objetivos básicos pueden agruparse en la categorías que se muestran en la tabla:

Categoría	*Tipo de Problema que Resuelve*
Interpretación	Inferir descripciones de situaciones a partir de los datos adquiridos
Predicción	Inferir consecuencias probables de situaciones dadas
Diagnosis	Inferir el mal funcionamiento de un sistema a partir de observaciones
Diseño	Configurar objetos que verifiquen determinadas restricciones
Planificación	Diseñar acciones
Monitorización	Comparar las observaciones con las salidas esperadas
Depuración	Prescribir remedios contra un funcionamiento erróneo
Reparación	Ejecutar planes para administrar los remedios prescritos
Instrucción	Diagnosis, depuración, y corrección del comportamiento del estudiante
Control	Gobierno de todo el comportamiento del sistema

Estos usos potenciales de sistemas expertos son aplicables a:

- Planificación y control de misiones
- Comunicaciones
- Análisis de señales
- Control
- Análisis de inteligencia
- Selección de objetivos
- Construcción y fabricación
  - diseño, planificación, control
- Educación
  - instrucción, testeo, diagnosis
- Equipamiento
  - diseño, seguimiento, diagnosis, mantenimiento, reparación, operación, instrucción
- Análisis e interpretación de imágenes
- Profesiones (derecho, medicina, ingeniería, económicas...)
  - consultoría, instrucción, interpretación, análisis
- Software
  - especificación, diseño, verificación, mantenimiento, instrucción
- Sistemas de armamento
  identificación de objetivos, guerra electrónica, control adaptativo

Un SE que interpreta utiliza normalmente los datos adquiridos de sensores para inferir descripciones de situaciones. Un ejemplo podría ser la interpretación de las lecturas de los aparatos contadores en una central química para inferir el status del proceso. Los sistemas de interpretación manejan datos reales más que representaciones simbólicas de la situación del problema. Normalmente deben manejar datos que son ruidosos, escasos, incompletos, poco fiables, o erróneos.

Los sistemas de interpretación pueden procesar diferentes tipos de datos. Por ejemplo, tanto los sistemas de visión como los de reconocimiento del habla utilizan entradas naturales. Los sistemas de interpretación química utilizan el espectro de masas para inferir la estructura de los compuestos. La interpretación médica usa las medidas de control del paciente (ej.: ritmo cardíaco, presión sanguínea) para diagnosticar y tratar enfermedades... Los problemas clave en esta categoría podrían ser:

- Los intérpretes deben trabajar con información incompleta.
- En determinados casos los datos pueden ser contradictorios, por lo que el intérprete debe realizar alguna hipótesis de qué datos son creíbles.
- Sería de ayuda que el sistema fuera capaz de explicar en que medida está la interpretación apoyada por la evidencia, ya que normalmente las cadenas de razonamiento son largas y complicadas.

Los SE que realizan predicciones infieren consecuencias probables en determinadas situaciones. Ejemplos son la predicción de los daños a la cosecha por determinados tipos de insectos, la estimación de la demanda global de petróleo a partir de la situación geopolítica actual, y predecir donde va a ocurrir el próximo conflicto bélico a partir de los datos de

inteligencia. Los sistemas de predicción utilizan algunas veces modelos de simulación, programas que reflejan la situación de la actividad real actual, para generar situaciones o escenarios que podrían ocurrir a partir de determinados datos de entrada. Tres problemas clave son:

- La predicción requiere de la integración de información incompleta. (si no, no sería un problema de Inteligencia artificial)
- La predicción debe tener en cuenta varias posibilidades futuras (utilizar el razonamiento hipotético) y debería ser sensible a variaciones de los datos de entrada.
- Puede que sea necesario que la teoría predictiva tenga que ser accidental (eventual); la posibilidad del futuro distante puede que dependa de eventos más cercanos aunque impredecibles.

Los SE que diagnostican utilizan descripciones de situaciones, características del comportamiento, o conocimiento acerca del diseño de componentes para inferir posibles causas de malfuncionamiento del sistema. Ejemplos son determinar las causas de las enfermedades a partir de los síntomas observados en los pacientes y localizar los fallos en circuitos eléctricos. Los sistemas de diagnóstico son a menudo sistemas que no sólo diagnostican el problema, sino que también ayudan en la depuración. Pueden interactuar con el usuario para ayudarle a encontrar los fallos y sugerirle cursos de acción para corregirlos. Las áreas problemáticas son:

- Los fallos pueden en algunos casos ser enmascarados por los síntomas de otros fallos.
- Los fallos pueden ser intermitentes.
- El equipo de diagnóstico también puede fallar.
- Puede que sea muy costoso, o peligroso, o imposible el recuperar determinados datos útiles para el diagnóstico.

Los SE que diseñan desarrollan configuraciones de objetos basados en un conjunto de requisitos a cumplir. Ejemplos son los genes-clonados, el diseño de circuitos integrados y la creación de moléculas orgánicas complejas. Estos sistemas utilizan síntesis para construir diseños parciales y simulaciones para verificar o comprobar sus ideas. Debido a que el diseño está muy relacionado con la planificación, muchos sistemas de diseño proporcionan mecanismos para desarrollar y refinar planes para alcanzar el diseño deseado. Los sistemas de diseño pueden ahorrar gran cantidad de búsqueda innecesaria mediante la generación de planes para producir la configuración deseada y evaluarlos en el contexto de restricciones de problemas. Más problemas clave son:

- Las restricciones de diseño vienen de distintas fuentes, y normalmente no existe una teoría que integre las restricciones con las decisiones de diseño.
- Es fácil olvidar los motivos por los que se tomaron algunas decisiones de diseño y es difícil de evaluar el impacto de un cambio en parte del diseño. Es recomendable que el

diseñador del sistemas mantenga un registro de las justificaciones de las decisiones de diseño y que sea capaz de utilizarlas para explicar las decisiones posteriormente.
- Es importante ser capaz de reconsiderar las posibilidades del diseño cuando se modifica el mismo.
- Muchos problemas de diseño necesitan razonar acerca de relaciones espaciales. Razonar acerca de distancias, formas, y contornos es computacionalmente muy costoso.

Los SE que planifican diseñan acciones; deciden que línea de acción seguir antes de actuar. Un ejemplo podría ser la creación de un ataque aéreo, proyectado para que se prolongue varios días, para reducir una determinada capacidad de las fuerzas enemigas. Los sistemas de planificación deben a menudo volver atrás (backtraking), es decir, rechazar una determinada línea de razonamiento debido a que no se verifica una de las restricciones del problema. La vuelta atrás puede ser costosa, por lo que estos sistemas descomponen la tarea en subproblemas para evitar la replanificación desde los puntos erróneos. Otros problemas clave son:

- Si existen demasiados detalles, el planificador debe ser capaz de centrarse en las consideraciones más importantes.
- En sistemas grandes suelen ocurrir interacciones entre diferentes submetas (subplanes).
- A menudo sólo se conoce aproximadamente el contexto de la planificación, por lo que el planificador debe trabajar con incertidumbre.
- Puede ser necesaria una coordinación (coreografía) cuando el plan deba ser ejecutado por varios actores.

Los SE que monitorizan comparan el comportamiento real del sistema con el esperado. Ejemplos son la monitorización de las lecturas de los instrumentos en una central nuclear para detectar condiciones de accidentes, y asistir a pacientes en una unidad de cuidados intensivos mediante el análisis de los datos del equipo de la UCI. Estos sistemas poseen el 'tiempo' como una de sus variables más importantes y deben realizar una interpretación del comportamiento que observan que sea dependiente tanto del contexto como del tiempo.

Los SE que depuran encuentran remedios cuando existe un malfuncionamiento. Ejemplos son sugerencias de como ajustar un computador para minimizar un tipo particular de problema de rendimiento, o seleccionar el tipo de mantenimiento necesario para corregir daños en los cables telefónicos. Muchos sistemas de depuración se apoyan en simples tablas de asociación entre tipos de malfuncionamiento y remedios específicos, pero el problema de la depuración es bastante difícil y requiere de soluciones de diseño y su evaluación mediante la predicción de su efectividad. Corrientemente estos sistemas incorporan un componente de diagnosis para descubrir la causa del malfuncionamiento.

Los SE que realizan reparaciones siguen un plan para administrar alguna solución prescrita. Se han desarrollado pocos sistemas de reparación hasta la fecha, en parte debido

a que la acción de ejecutar una reparación en el mundo real añade una nueva dimensión de complejidad al problema. Estos sistemas también requieren de diagnosis, depurado, y planificación para poder establecer el entorno de la reparación.

Los SE que instruyen realizan un diagnóstico, un depurado, y una reparación del comportamiento del estudiante. Ejemplos son el adiestramiento de estudiante para reparar circuitos eléctricos, o la educación de estudiantes en medicina en alguna especialidad determinada. Los sistemas de instrucción desarrollan un modelo de lo que el estudiante sabe y cómo se aplica dicho conocimiento para resolver el problema. Diagnostican y depuran las deficiencias de los estudiantes mediante el análisis del modelo y la generación de planes para corregir las deficiencias. Reparan el comportamiento del estudiante mediante la ejecución de dichos planes con una interacción directa con el mismo.

Los SE que realizan un control gobiernan adaptativamente el comportamiento total del sistema. Ejemplos son la dirección de la construcción y distribución de sistemas informáticos y el control de los pacientes en una unidad de vigilancia intensiva. Los sistemas de control deben incluir un componente de monitorización que permita realizar un seguimiento temporal del comportamiento del sistema, pero también necesitan componentes para realizar cualquiera de los otros tipos de tareas anteriormente comentadas: interpretación, predicción, diagnosis, diseño, planificación, depurado, reparación, e instrucción. Una combinación deseable para un sistema de control sería monitorización, diagnosis, depurado, generación de planes, y predicción.

5 Viabilidad de un problema en ingeniería del conocimiento

No fue fácil describir en términos generales, las características que hacen que un problema sea adecuado para resolverlo desarrollando un Sistema Experto, o dicho de otra manera: ¿funcionará eficientemente el enfoque de los sistemas expertos para un problema particular?. Es esta una cuestión trascendental, pues, como señala un estudio realizado por el Carnegie Group, una de las empresas líderes en ingeniería del conocimiento, demuestra que más del 50 por ciento de lo proyectos en Ingeniería del Conocimiento fracasaron por no haber realizado un estudio de viabilidad adecuado.

PORCENTAJES DE ÉXITO/FRACASO EN EL DESARROLLO DE SISTEMAS EXPERTOS

40% Fallan en el análisis.
10% Fallan en el prototipo.
3% Fallan por razones técnicas.
12% Tienen éxito técnico, pero hay cambio en los receptores
35% Tienen Éxito.

La Universidad Politécnica de Madrid ha intentado realizar una formalización[1] del proceso que se debe seguir para dar una respuesta simple a esta cuestión, agrupando y mejorando las aproximaciones que hasta la fecha habían desarrollado otros autores. En este sentido, tal como se muestra en la figura 3, la tecnología de los sistemas expertos debe considerarse sólo si el desarrollo de un sistema experto es posible, está justificado, es apropiado, y va a tener éxito en su construcción. Las tres primeras cuestiones, o premisas son directas; la última, o de éxito, es indirecta o colateral, pero no por ello menos importante.

A continuación se considera cada una de ellas por separado:

[1] La metodología formal se describe en el Anexo 1.

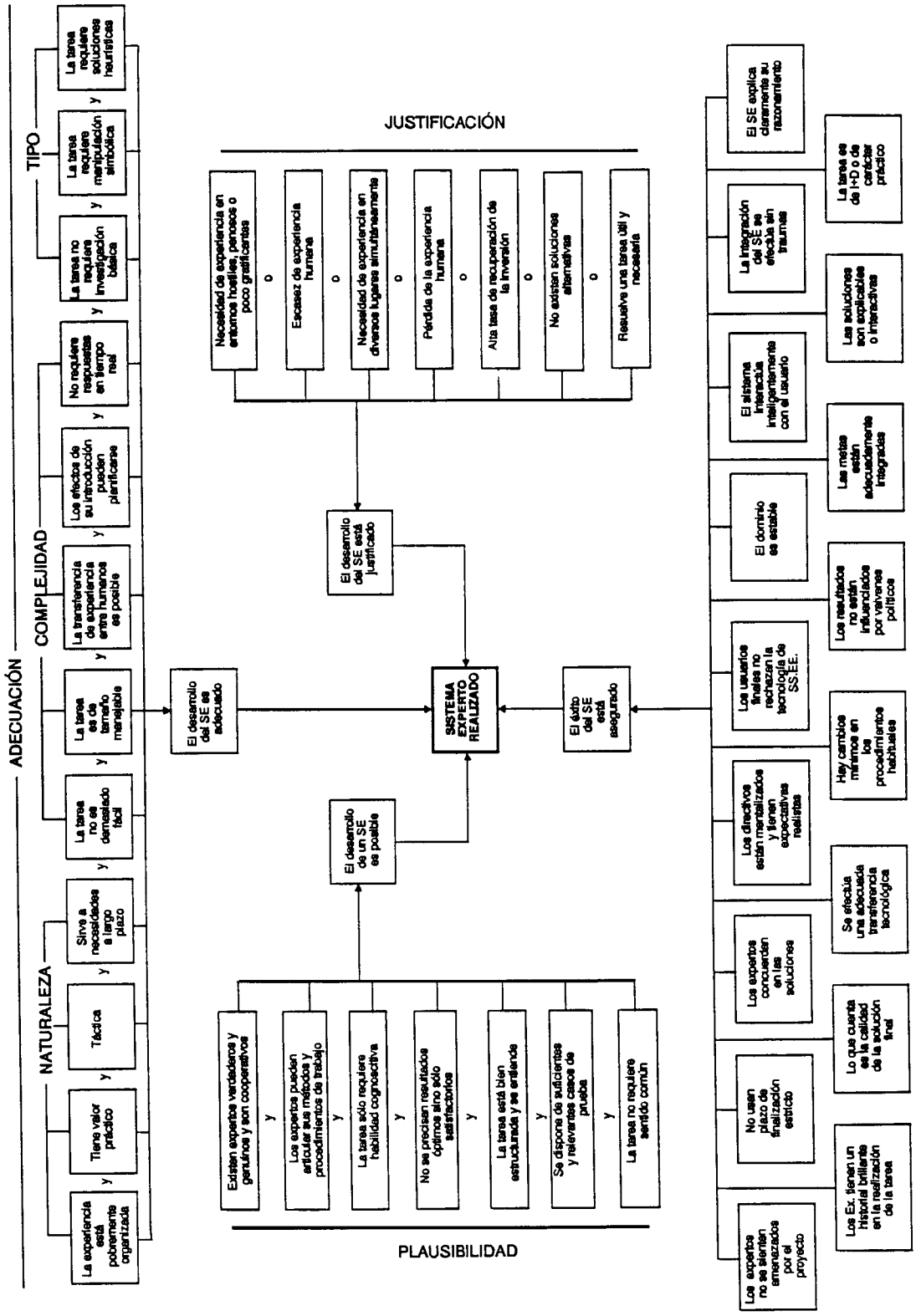

Figura 3: Posibilidad, Justificación, y Adecuación de éxito de un SE.

5.1 Plausibilidad

Uno de los requisitos más importantes, por condición necesaria, es que existan verdaderos expertos en el área del problema. Es decir, personas que son significativamente mejores que los aprendices para resolver los problemas en ese dominio.

Un buen indicador para saber si una tarea es apropiada para su solución usando un SE es que uno o más expertos humanos traten con la tarea como parte de su trabajo habitual, de manera que puedan tener el conocimiento y la experiencia necesarios para entender cuales son realmente los problemas y para elegir los casos de prueba adecuados. Estos expertos deberían estar disponibles para trabajar en el proyecto con una dedicación prioritaria absoluta, deben ser capaces de articular lo que hacen cuando resuelven un problema en el mejor y en el peor de los casos, y colaborar para que con ayuda de un ingeniero del conocimiento esa articulación sea posible.

La elección de los expertos es crítica. Dado que un proyecto de SS.EE. exige dedicación y un compromiso a medio plazo, un experto que esté sólo medianamente interesado por el asunto no es adecuado; antes bien, el experto debe ser uno de los más interesados en obtener una solución. El experto debe entender lo que es el problema y haberlo resuelto con bastante frecuencia. No es suficiente conocer teóricamente como manejar casos similares, o tener ideas brillantes sobre nuevos métodos de hacer las cosas o, incluso, un ávido interés por aprender. Una de las últimas trampas en que pueden caer, y a veces caen, los II.CC. es creer que después de trabajar con lo expertos durante algún tiempo, ellos también se convertirán en experto en el área del problema.

Sin una fuente de vasto y potente conocimiento que extraer, el esfuerzo de desarrollo fracasará produciendo un sistema verdaderamente carente de toda pericia. Sin embargo, no es tarea fácil encontrar un buen experto que esté dispuesto a colaborar en la construcción y el desarrollo de un SE, pues temen que la nueva tecnología elimine su puesto de trabajo.

Una vez que se dispone de un experto genuino es importante que este sea cooperativo, y capaz de articular su conocimiento y modos de razonamiento. Deben ser capaces de explicar de alguna manera los métodos que usan para resolver los problemas.

El experto no debe sentirse forzado al explicar la naturaleza de su experiencia. En las entrevistas iniciales, el IC y el experto deberán decidir si pueden trabajar conjuntamente con garantías de éxito. Ambos tendrán que trabajar junto durante al menos un año, por lo que es importantes establecer una relación cómoda.

Todo ello plantea una cuestión práctica en absoluto trivial: ¿cómo darse cuenta de que uno se encuentra ante un genuino experto?. Para resolver esta cuestión hay que, en primer lugar,

tener una idea clara de la naturaleza de la experiencia. Cuando se habla de un 'experto', se quiere decir un individuo que esta ampliamente reconocido como alguien que es capaz de resolver un tipo de problemas particular que la mayoría de las personas, incluso de su misma profesión, no pueden resolver efectiva y eficientemente. Al enfatizar el acuerdo general acerca de quién es un verdadero experto, uno evita a los falsos expertos, pues, como decía Lincoln, "alguien puede engañar a todos algún tiempo, a algunos todo el tiempo, pero lo que no es posible es engañar a todos todo el tiempo".

En la elección del experto algo debe quedar claro, y es que el SE que modeliza el comportamiento de un experto, nunca puede mejorar a éste cuando está al máximo de sus prestaciones.

Existen diversas técnicas para comprobar la validez del experto. La primera consiste en recabar la opinión de sus compañeros de profesión, porque normalmente todos se conocen entre sí y no es muy difícil establecer quién es quién. La segunda, más formal, consiste en someter al candidato a una prueba de análisis de protocolo para establecer su capacidad de articular sus conocimiento y su voluntad de cooperación. Por último hay que someter al experto a la prueba objetiva de que resuelva el conjunto de casos de prueba seleccionados al efecto. Según sea el porcentaje de éxito durante la resolución de estos problemas, se tendrá una valoración cuantitativa del experto que servirá para establecer a priori una cota superior de la eficacia del SE que se construya. Para poder superar en términos absolutos las prestaciones del experto que se modelizó habría que utilizar otros expertos que incrementen las capacidades del SE. Este valor también sirve para usarlo en la cuantificación de la tarea que se tratará más adelante.

Otro recurso crítico para el desarrollo satisfactorio de un SE, es disponer de un conjunto de casos de prueba de donde extraer el conocimiento de los expertos. Es decir, que permitan observar realmente a los expertos resolver problemas antes de que describan cómo los resuelven, de manera que sea más sencillo entender el proceso real tal como es, así como el conocimiento real que utilizan. Del mismo modo, puede emplearse un juego de casos de prueba como "juego de ensayo" para comprobar las implementaciones a medida que éstas alcanzan los distintos estadios de desarrollo, y, sobre todo, para la validación de los distintos prototipos. El sistema final se validará mediante el uso en paralelo con el trabajo de los expertos.

Un problema en este punto consiste en determinar cuantos casos de prueba y de qué tipo son necesarios para modelizar el comportamiento de un experto. La cuestión sería bastante fácil de resolver si se conociera la distribución estadística de los distintos casos que se pueden presentar. En este supuesto, el problema se reduciría a calcular el tamaño de una muestra conociendo los parámetros poblacionales y fijando el intervalo de confianza de acuerdo con los objetivos a alcanzar. Pero lamentablemente, aún no es posible establecer esas distribuciones de acuerdo con las distintas clases de tareas. Más aún, la comunidad

investigadora no ha conseguido alcanzar un consenso sobre una posible clasificación de las tareas (ej. planificación, diseño, diagnosis....) que se pueden desarrollar con esta tecnología.

Las otras condiciones necesarias para el desarrollo de un SE conciernen a las características del problema que el SE debe resolver. En primer lugar, la tarea debe requerir conocimiento y habilidades no físicas. Si la tarea consiste en manipulaciones físicas que sólo pueden aprenderse a través de la práctica, el enfoque tradicional de los SS.EE no funcionará. Sin embargo, esto no significa que los problemas con una componente física deban ser sistemáticamente desechados. Si la tarea requiere una combinación de habilidades físicas y cognoscitivas, tal como monitorizar y controlar el uso de un brazo robot en una línea de ensamblaje, la parte cognoscitiva puede ser manejada con técnicas de Ingeniería del Conocimiento y la parte física por métodos mas convencionales.

No es conveniente el desarrollo de un sistema experto si la tarea precisa una cantidad relevante o predominante de *sentido común*. En general, problemas que requieran entendimiento de lenguaje natural, modelos espaciales o geométricos complicados, relaciones causales o temporales complejas o entendimiento de las motivaciones o interconexiones humanas, no son, en el momento actual, candidatos adecuados para usar la tecnología de la ingeniería del conocimiento.

5.2 Justificación

El hecho de que sea posible desarrollar una SE para una tarea particular no significa que esté justificado hacerlo. Justificar el esfuerzo de desarrollo de un SE puede hacerse de diversas formas, entre las que se encuentran las que se citan a continuación:

En primer lugar, cuando la toma de decisión del experto debe hacerse en entornos peligrosos u hostiles, tales como plantas nucleares, estaciones espaciales, etc., puede ser muy costoso mantener un experto humano en dichos entornos.

El desarrollo de un SE también está justificado cuando los expertos humanos no pueden usarse por su escasez. A menudo, los expertos humanos escasean; en consecuencia, su demanda es muy alta y su coste elevadísimo. El problema se complica cuando la empresa necesita experiencia similar en distintas ubicaciones, como sucede con el uso de los especialistas en los procedimientos de emergencia de una plataforma petrolífera.

Los SS.EE. están justificados cuando una experiencia relevante y significativa se está perdiendo en una organización debido a cambios del personal: jubilaciones, cambios de empresa, etc. Esta situación puede provocar trastornos e incluso 'estragos' cuando el experto

se va de la empresa llevándose consigo todo 'su saber'. La existencia de un SE puede minimizar o incluso eliminar esos problemas. En estos sentidos, Los SS.EE. son una especie de memoria institucional.

Para la comunidad científica, la razón más importante es la de la formalización y la clarificación del conocimiento que se extrae al hacer explícito el experto humano su razonamiento. Los expertos humanos como físicos, ingenieros, o analistas poseen muchos métodos heurísticos basados en sus experiencias con problemas específicos y difíciles. Raramente abstraen sus métodos de razonamiento y describen sistemáticamente cómo, cuando, y donde los aplican. Cuando se construye un SE hay que hacer ver al experto lo importante que es que organice su conocimiento de forma que sea reproducible y testable por otros expertos.

Otro motivo para la construcción de SS.EE. es la posibilidad de combinar el conocimiento experto de múltiples expertos humanos en una base de conocimiento compartida que pueda ser estudiada en cuanto a su consistencia y a la fiabilidad de sus recomendaciones. De estos estudios puede surgir una síntesis del conocimiento de varios expertos. Pero para que suceda esto, se deben desarrollar métodos de comparación del razonamiento de los expertos.

La forma más importante de justificar el desarrollo de un sistema experto es mediante una alta tasa de recuperación de la inversión. Al elegir cuidadosamente la aplicación, debe considerarse el interés tiene el problema para los directivos. Una manera de confirmarlo es determinando si los directivos están dispuestos a comprometer los recursos materiales y humanos necesarios para desarrollar el proyecto. Esta es una cuestión pragmática, debe utilizarse un método lo más formal posible de forma que se puedan identificar los costes, beneficios, y riesgos.

Finalmente, un SE está justificado cuando no es posible utilizar otras soluciones alternativas distintas de las que proporciona la IA.

5.3 Adecuación

Sería totalmente inadecuado el que porque los SS.EE. puedan ser útiles en determinadas circunstancias, el aplicarles 'si algo es bueno, más de lo mismo es mejor', y usarlos en todo tiempo y lugar. Es decir, no basta con que el desarrollo de un SE sea posible e incluso esté justificado para construirlo, pues además de las dos condiciones anteriores, también el SE debe ser adecuado. Por lo tanto, hay que determinar los factores clave que establecen cuando es apropiado desarrollar un SE. Estos factores se refieren a la naturaleza, y a la complejidad del problema.

En lo que concierne a la naturaleza del problema, el requisito fundamental es que no esté bien estructurado. Un problema debe tener ciertas cualidades intrínsecas. Así, si el conocimiento necesario para llevar a cabo una tarea tiene las características de estable, numérico, y es de fácil agregación, entonces los programas algorítmicos serán la mejor manera de encarar su solución. De hecho, los SS.EE. no suplantan la necesidad de bases de datos, de software estadístico, hojas de cálculo electrónicas, etc.. Bien al contrario, sólo si la productividad de un trabajo depende de un conocimiento que es subjetivo, cambiante, simbólico, dependiente de los juicios particulares de las distintas personas, o es de naturaleza heurística; es decir, requiere reglas de buen juicio para alcanzar soluciones aceptables, entonces es apropiado para desarrollar un SE.

El problema debe ser de naturaleza lo más táctica posible. En cualquier institución se pueden dar dos tipos de problemas: estratégicos y tácticos, cuya distinción no es sencilla, debido a que se basa más en una cuestión de grado que en una diferencia cualitativa, en base a las características siguientes:

a. Teleología: un problema es tanto más estratégico cuanto más implique la determinación de finalidades y fines; es decir, cuanto más orientado a fines esté. En efecto, todos los problemas comprenden la selección de medios para alcanzar los resultados deseados, pero muchos consideran éstos como dados. En la medida en que esto sea así, estos problemas pueden considerarse como tácticos.

b. Rango: un problema es más táctico que otro si el efecto de su solución es de menor duración, o si su solución puede anularse o modificarse fácilmente; es decir, cuando tiene un alto grado de reversibilidad.

c. Alcance: un problema es más estratégico que otro cuanto mayor sea la parte de la organización afectada por él para su solución.

En suma, que los problemas que aquí interesa resolver deben ser orientados a metas, lo más reversible posible y que afecten lo menos posible a la organización.

Otra condición necesaria es que la tarea no sea extremadamente difícil ni demasiado fácil. Si un experto no puede enseñar el proceso a un aprendiz porque la experiencia sólo puede alcanzarse a través de adquirir la pericia con el propio trabajo, el proceso puede ser demasiado difícil de capturar en un SE. O, si cualquier experto tarda días o semanas en vez de horas en resolver el problema, sin considerar el tiempo dedicado a tareas mecánicas como hacer resúmenes o rellenar impresos, hay una cierta evidencia de que esa tarea es demasiado difícil o compleja para un enfoque de Ingeniería del Conocimiento. Sin embargo, si una tarea que requiere días o semanas de esfuerzo concentrado puede descomponerse en subtareas más pequeñas, cortas y relativamente independientes, cada una de estas subtareas puede ser candidata para ser desarrollada en un SE. Por otra parte, si la tarea sólo requiere unos

minutos de dedicación del experto, es probable que pueda resolverse por una tecnología más sencilla, incluso que no necesite mecanización. Debería ser un problema serio en un dominio en el que un ser humano necesite años de estudio o práctica para alcanzar el 'status' de experto. De este modo, problemas como ordenación de elementos, etc., son un casos típicos de poca dificultad que no exige desarrollar un SE para resolverlos. El grado de complejidad, viene dado por la dificultad de resolver el problema. En este caso, son adecuadas tareas que sean fáciles, pero no demasiado fáciles. Los criterios que determinan que una tarea es fácil pueden ser los siguientes: que el experto tarde en realizar la tarea entre 15 minutos y 8 horas, que el dominio del problema esté bien establecido, y que sean improbables cambios importantes durante la vida del proyecto. También que la tarea sea autocontenida, y, finalmente, que la tarea sea definible y precisa.

La dificultad de la tarea también está relacionada en cierta manera con lo bien que los expertos entienden el dominio del problema; es decir, con el grado en el que el conocimiento utilizado para resolver el problema es preciso y está bien estructurado. Si la tarea es tan novedosa, o es pobremente entendida, que requiere investigación básica para encontrar la solución, la utilización de la ingeniería del conocimiento no será conveniente.

Por su parte, entre los criterios que indican que la tarea no es demasiado fácil se incluyen los siguientes: que ejecutar la tarea requiera de verdadera experiencia, de modo que las prestaciones en resolver la tarea se incrementen con la experiencia y no se consigan por simple entrenamiento, que la tarea implique muchos factores, y tengan bastantes interacciones entre sí, y, por último, que los métodos tradicionales sean inadecuados por ineficientes o costosos.

5.4 Éxito

Los SS.EE. no son la panacea para alcanzar lo imposible, ni siquiera lo muy difícil. Una mera disposición para tener un SE no garantiza que pueda construirse. Identificar una necesidad, no es suficiente para determinar una tarea apropiada para realizarla usando un SE, pues no basta con haber determinado qué problema es susceptible de implementarse, para que éste pueda resolverse en la realidad cotidiana. Dicho en otros términos, además de las consideraciones puramente técnicas para la aplicación de un SE en la solución de un problema, existen otras cuestiones que pueden hacer fracasar el proyecto. Entre estas cabe destacar las siguientes:

a. Mentalización de los responsables. La mayoría de las instituciones son reacias a la adopción de nueva tecnología. Esta oposición sólo puede superarse convenciendo a los responsables de que esta tecnología tiene éxito, y que es conveniente.

b. Entrenamiento de los implicados. Es imprescindible la formación de los responsables del proyecto en las fases por las que pasa el desarrollo del sistema, y en el manejo de las herramientas de construcción de SS.EE. Esto reducirá drásticamente los costes y el tiempo de desarrollo de un SE.

c. Aceptación por parte de los usuarios. No basta con que los directivos sean conscientes de la importancia de esta tecnología para que pueda implantarse sin problemas. De hecho, casi tan importante como eso es la aceptacion por los usuarios finales. La resistencia de los usuarios potenciales a emplear SS.EE. en las organizaciones parece basarse en las consideraciones siguientes:

- Los profesionales y los mandos intermedios creen que, con la incorporación de los SS.EE, sus conocimientos ya no serán necesarios y, por consiguiente, sus empleos estarán en el candelero.
- Otros usuarios piensan que la parte interesante de las tareas que realizan le será encomendada a los SS.EE. dejando para ello la parte menos gratificante e intrascendente.

d. Aceptación por los responsables. Para lo cual hay que demostrar la plausibilidad del sistema propuesto y de que las ideas que contiene funcionan. Esta aceptación es relativamente fácil cuando los responsables ya tienen otros SS.EE trabajando bien en su organización, pero es ciertamente difícil cuando se trata de la primera aplicación. En este caso, las siguientes recomendaciones podrían ayudar a que los responsables acepten la aplicación:

- Seleccionar algo sencillo y bien entendido, pero no trivial y que sea lo bastante significativo como para que el SE tenga un valor evidente.
- Elegir algo que los usuarios finales no vean como una amenaza.

e. Concordancia de los expertos en lo que concierne a las soluciones. Además, los expertos, si se usan varios, deben coincidir generalmente acerca de la elección y la exactitud de las soluciones en el dominio del problema. En otro caso, la validación de las prestaciones del SE desarrollado sería poco menos que una misión imposible.

6 Una nueva metodología software

Es evidente que un sistema experto no es otra cosa que un producto software, y como tal debe ajustarse a las metodologías de desarrollo de los productos software. Sin embargo, existen algunas diferencias clave entre los sistemas expertos y los programas de software convencional. Sus desarrollos requieren de aproximaciones de construcción totalmente distintas.

Los programas de software convencional resuelven problemas que tradicionalmente han sido resueltos 'a mano'; es decir, el software convencional está orientado a traducir procedimientos o algoritmos conocidos en código. Durante el proceso de diseño se identifican las posibles situaciones a ser consideradas, los puntos de decisión, y las respuestas. Idealmente, la aproximación para el desarrollo del software top-down minimiza los cambios que posteriormente puedan ser necesarios en el código. La aproximación top-down es crítica para el software convencional porque es muy difícil modificar el diseño del sistema una vez que se ha empezado a codificar. Debido a las muchas interdependencias entre los caminos de control, los valores de los datos, secuencias de estado, etc... el realizar demasiados cambios puede provocar que los programadores pierdan el control del proceso del cambio, provocando retrasos y un aumento considerable del coste del desarrollo.

Normalmente suelen surgir cambios imprevistos o mejoras en el software una vez que el mismo ya ha sido distribuido. El realizar los ajustes oportunos sobre el software una vez que ha sido distribuido, utilizando tecnología convencional, se traduce en un aumento importante del presupuesto del proyecto.

El desarrollo de sistemas expertos difiere de la aproximación top-down del software tradicional. Los sistemas expertos deben codificar sutilmente conocimiento heurístico que incluye símbolos, estrategias, y relaciones. La representación de tales conceptos es extremadamente difícil de realizar con lenguajes de programación convencional como Pascal o C. Pero lo que es más importante, nadie sabe exactamente cómo los expertos humanos construyen sus conclusiones -ni incluso los propios expertos. Por lo tanto, desarrollar un sistema experto no puede ser nunca un proceso lineal.

6.1 Construcción de un Sistema Experto

Los constructores de SS.EE. no tienen una serie de pasos bien definidos que puedan seguir

a la hora de construir un sistema experto. La complejidad inherente del proceso de construcción impide el diseño de todos los pasos por anticipado. Como resultado, los constructores de SE han encontrado que la forma más efectiva de proceder es la de realizar un desarrollo incremental, mejorando paso a paso la organización y la representación del conocimiento del sistema.

El resultado usual del desarrollo incremental es que este llega un punto en el cual la base de conocimiento alcanza un tamaño inmanejable; el control no es efectivo y además es lento, y el sistema parece que está construido a partir de parches y construcciones no integradas. Ante este panorama, es necesario considerar el rediseño y la reimplantación del sistema. El ingeniero del conocimiento y el experto en el dominio deberán reexaminar el problema y rehacer su esquema de representación. Esto podría implicar incluso la selección de una nueva herramienta de construcción de sistemas expertos.

Aunque no es posible especificar los pasos exactos a seguir en la construcción de sistemas expertos, es posible describir los estados del desarrollo del sistema y los tipos de actividades realizados en cada estado.

Tal como se aprecia en la figura 4, el desarrollo de un sistema experto comprende dos fases principales: la primera implica la identificación y la conceptualización del problema. La identificación incluye la selección de un experto, de las fuentes de conocimiento, y de los recursos, y de la definición completa del problema. Durante la conceptualización se ponen de manifiesto cuales van a ser los conceptos clave, y las relaciones que se van a necesitar para caracterizar el problema. La segunda fase abarca la formulación, la implementación, y el testeo de la arquitectura del sistema, lo que incluye una constante reformulación de los conceptos, el rediseño de las representaciones, y el refinamiento del sistema implementado. Las revisiones provienen de las críticas y de las sugerencias que realiza el experto para mejorar el comportamiento y la competencia del sistema.

Las distintas fases no están claramente diferenciadas, aunque sí se pueden identificar distintos aspectos relacionados con cada una durante la construcción del sistema experto. Las siguientes secciones los comentarán en detalle.

6.1.1 Identificación

El primer paso para el desarrollo de un sistema experto es la caracterización de los aspectos importantes del problema. Esto incluye la identificación de los participantes, de las características del problema, de los recursos, y de las metas.

Los participantes deben tener sus papeles muy bien definidos antes de que comience la adquisición del conocimiento. La situación más simple la constituye la interacción entre un

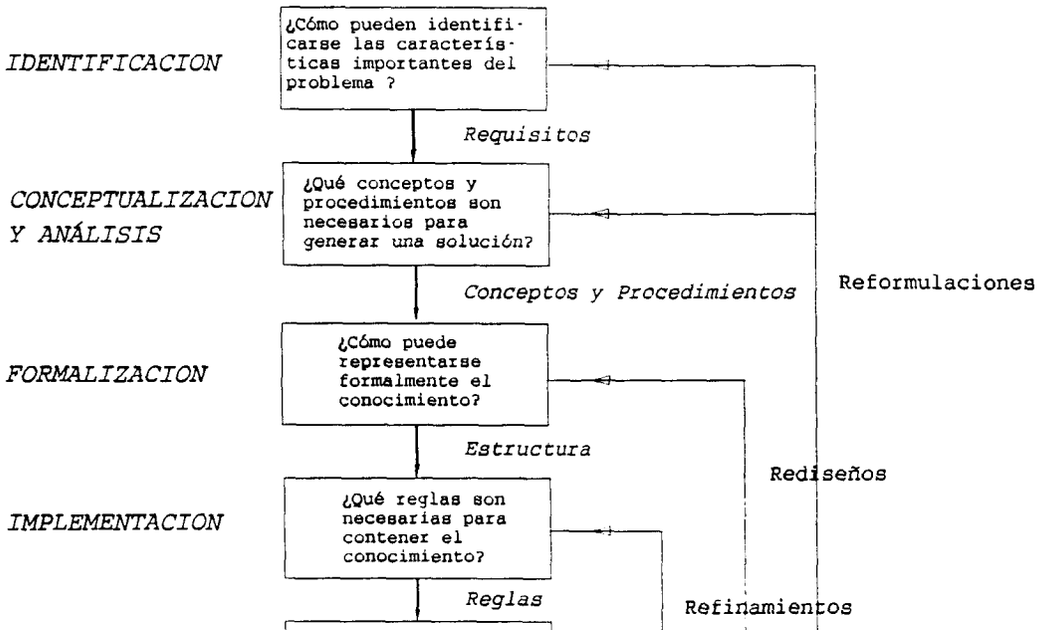

Figura 4

ingeniero del conocimiento y un experto en el dominio. Estos actúan como 'informantes' que cuentan al ingeniero del conocimiento datos sobre su experiencia y conocimiento. Otras situaciones podrían incluir otros participantes. Podrían haber varios expertos en el dominio, varios ingenieros del conocimiento, o incluso expertos interdisciplinarios.

Una vez que se han identificado los participantes, el ingeniero del conocimiento y el experto proceden a identificar el problema en estudio. Esto incluye un intercambio informal de puntos de vista sobre varios aspectos del problema, su definición, características, y subproblemas. El objetivo es caracterizar el problema y las estructuras de conocimiento sobre las que se apoya de forma que pueda empezar el desarrollo de la base de conocimiento. Este proceso necesita de múltiples iteraciones entre el experto y el ingeniero hasta conseguir una definición del problema lo más concreta posible. Algunas preguntas a responder en esta fase podrían ser:

- ¿Qué tipos de problemas que se espera que resuelva el sistema experto?
- ¿Cómo se pueden caracterizar o definir estos problemas?
- ¿Cuales son los subproblemas importantes y cómo se descomponen las tareas?
- ¿Qué son los datos?
- ¿Cuales son los términos importantes y sus interrelaciones?
- ¿Cómo es una solución y qué conceptos se utilizan en ella?
- ¿Qué aspectos del conocimiento experto son esenciales a la hora de resolver dichos

problemas?
- ¿Qué situaciones impiden alcanzar soluciones? ¿En qué se verá afectado el sistema experto?

Una manera de facilitar las respuestas a estas cuestiones consiste en que el experto aporte descripciones narrativas de los problemas 'tipo', explicando cómo los resuelve y el razonamiento que está implícito en las soluciones.

Los recursos son necesarios para adquirir el conocimiento, implementar el sistema, y testarlo. Los recursos típicos son: fuentes de conocimiento, tiempo, facilidades computacionales, y presupuesto.

El ingeniero del conocimiento y el experto utilizan distintas fuentes para obtener el conocimiento importante para la construcción del sistema experto. Para el experto pueden citarse la experiencia en la resolución de problemas, libros de texto, y ejemplos de problemas y soluciones. Para el ingeniero las fuentes incluyen experiencia en situaciones similares, y conocimiento acerca de métodos, representaciones, y herramientas para la construcción de sistemas expertos.

El tiempo, software, hardware, y el presupuesto del que se disponga son críticos por razones obvias.

Es deseable que sea el experto el que identifique las metas o objetivos de la construcción del sistema experto. Sin embargo, sería conveniente separar las metas de las tareas específicas del problema, ya que estas constituyen restricciones adicionales que pueden ayudar a caracterizar lo apropiado o deseable de determinadas aproximaciones. Ejemplos de metas podrían ser liberar al experto de determinadas tareas rutinarias, distribuir el conocimiento, formalizar las metodologías, etc...

6.1.2 Conceptualización y Análisis

En esta fase se concretan los conceptos clave y las relaciones que se introdujeron durante la fase de identificación, así como los procedimientos y análisis que guían la resolución de problemas. Normalmente se utiliza alguna técnica gráfica para representar los conceptos y sus relaciones de forma que se tenga constancia de la base conceptual para el prototipo del sistema. Antes de conceptualizar habría que responder a las siguientes cuestiones:

- ¿De que tipos de datos se dispone?
- ¿Qué datos son de entrada y cuales se infieren?
- ¿Tienen nombre las subtareas?
- ¿Tienen nombre las estrategias?

- Las hipótesis parciales que son utilizadas normalmente, ¿son identificables? ¿Cuáles son?
- ¿Cómo están relacionados los objetos en el dominio?
- ¿Se puede trazar un diagrama con jerarquías, etiquetando las relaciones causales, los conjuntos de inclusiones, las relaciones parte/totalidad, etc?
- ¿Qué procesos intervienen en la solución de un problema?
- ¿Cuales son las restricciones sobre esos procesos?
- ¿Cual es el flujo de información?
- ¿Se puede identificar y separar el conocimiento necesario para resolver un problema del conocimiento utilizado para justificar una solución?

Al igual que la fase anterior, la conceptualización y el análisis implica una serie de iteraciones entre el ingeniero y el experto que son importantes y difíciles y que consumen gran cantidad de tiempo. Suele tenderse a intentar analizar el problema correcta y completamente antes de implementar un sistema de prueba. Sin embargo, la experiencia ha demostrado que lo correcto es identificar los conceptos y procedimientos clave y sus relaciones y después formalizarlos e intentar construir el primer prototipo.

La información adquirida debe comprobarse con ejemplos específicos de la actividad de resolución de problemas, y los conceptos deben modificarse de manera que abarquen y sean consistentes con la actividad.

6.1.3 Formalización

La fase de formalización implica la transformación de los conceptos clave, subproblemas, y características del flujo de información aisladas en la fase anterior, hacia representaciones más formales basadas en las herramientas de construcción de sistemas expertos de las que se disponga. El ingeniero toma un papel más activo, seleccionando la herramienta que mejor se adapte a las condiciones del problema en conjunción con el experto. La salida de esta fase es un conjunto de especificaciones parciales que describen como se puede representar el problema con la estructura de la herramienta seleccionada.

Tres factores importantes durante la formalización son *el espacio de hipótesis, el modelo del proceso, y las características de los datos*. La estructura del espacio de hipótesis viene definida por la formalización de los conceptos y cómo se enlazan para formar las hipótesis. También hay que decidir la granularidad y estructura de los conceptos. Por ejemplo:

¿Son objetos estructurados, o entidades primitivas?
¿Las relaciones entre ellos son causales o espacio-temporales?

Los conceptos proporcionan pistas acerca de la naturaleza del espacio de hipótesis - si es finito o no, si consiste de clases predefinidas o serán generadas a partir de los conceptos, si

es útil considerar una jerarquía de hipótesis, si existirá incertidumbre relacionada con las hipótesis intermedias y finales, y si serían útiles distintos niveles de abstracción.

Conocer el modelo subyacente de los procesos que se utilizan para generar soluciones en el dominio puede ser importante para formalizar el conocimiento. Tipos de modelos podrían ser matemáticos (analíticos o estadísticos) o de comportamiento (simplificado).

La naturaleza de los datos en el dominio del problema aclararán si se pueden explicar directamente en función de determinadas hipótesis, lo que permitiría identificar la naturaleza de esta relación (causal, definicional, correlacional....) y así explicar como las hipótesis que explican directamente a los datos se relacionan con otras hipótesis de mayor nivel, y como estas se relacionan con la estructura de las metas en el proceso de resolución del problema. Otras cuestiones que ayudarían a definir la naturaleza de los datos podrían ser:

- ¿Son los datos escasos e insuficientes o abundantes y redundantes?
- ¿Tienen los datos asociada incertidumbre?
- ¿Depende la interpretación lógica de los datos del orden de ocurrencia en el tiempo?
- ¿Cual es el coste de la adquisición de los datos?
- ¿Cómo se adquieren los datos? ¿Qué tipos de preguntas se necesitan realizar para obtener los datos?
- ¿Cómo se pueden reconocer ciertas características de los datos cuando se extraen de un muestreo o de un flujo continuo de datos?
- ¿Son los datos precisos, fiables, y correctos?
- ¿Son los datos consistentes y completos para los problemas que van a resolver?

El resultado de formalizar el flujo de información conceptual y los elementos de subproblemas es una especificación parcial para la construcción de un prototipo.

6.1.4 Implementación

Implica transformar el conocimiento organizado de las fases anteriores en las estructuras representacionales asociadas con la herramienta seleccionada para el problema. A medida que se construye el programa ejecutable, el conocimiento debe ir ganando consistencia, compatibilidad, y organización, de forma que se defina un determinado control y flujo de información.

Con el conocimiento hecho explícito durante la fase de formalización se especifica el *contenido* de las estructuras de datos, las reglas de inferencia, y las estrategias de control. La herramienta especifica su *forma*.

El objetivo de la construcción del primer prototipo es el de comprobar la idoneidad de la

formalización y de las ideas básicas de diseño.

6.1.5 Testeo

Implica evaluar el prototipo y las formas representacionales utilizadas para implementarlo. Una vez que el prototipo funciona correctamente para dos o tres ejemplos, se debería probar con una variedad mayor para determinar las debilidades de la base de conocimiento y de la estructura de inferencia. Cuatro son los elementos que suelen repercutir en un pobre rendimiento: características de entrada / salida, reglas de inferencia, estrategias de control, y casos de testeo.

Las principales características de entrada/salida son la adquisición de datos y la presentación de conclusiones. El método de adquisición de datos puede ser incompleto o inadecuado debido a que se realizan las preguntas equivocadas o no se consigue de ellas la información suficiente. Por ejemplo, puede que las preguntas sean difíciles de entender o ambiguas para el usuario. Las conclusiones dadas por el programa puede que sean inadecuadas. Pueden ser demasiadas o insuficientes, con pocas o demasiadas hipótesis intermedias especificadas. Puede que no se hayan organizado u ordenado las conclusiones , y la salida puede estar a un nivel de detalle inadecuado.

El lugar más evidente donde encontrar errores en el razonamiento es en el conjunto de reglas de inferencia. Dichas reglas pueden ser incorrectas, inconsistentes, incompletas, o totalmente inútiles.

Los errores también pueden darse en la estrategia de control que sigue el sistema. Se debe sospechar cuando el sistema considera términos en un orden que difiere de el preferido por el experto.

Finalmente, los problemas con el prototipo pueden deberse a una selección inadecuada de los casos ejemplo. Algunos fallos pueden deberse a peculiaridades del caso de prueba que quedaban fuera del ámbito que se pensaba cubrir. Más corrientemente el conjunto de prueba es demasiado homogéneo y falla al comprobar la adecuacidad del programa.

6.1.6 Revisión del prototipo

En el curso de la construcción del sistema experto existe una constante revisión que puede implicar una reformulación de los conceptos, un rediseño de las representaciones, o un refinamiento del sistema implementado. El refinamiento implica un reciclado a través de las fases de implementación y de testeo de forma que se ajusten la reglas y sus estructuras de control hasta que se obtenga el resultado esperado.

Durante la revisión se debe producir una convergencia en el rendimiento. Si esto no ocurriera, el ingeniero del conocimiento debe adoptar medidas más drásticas sobre la arquitectura de la base de conocimiento. Cuando estas medidas conducen al desarrollo de una nueva representación en la fase de formalización, se produce un rediseño del sistema.

Si las dificultades fueran mayores, podría ser necesaria una reformulación de algunos conceptos (objetos, relaciones, o procesos) utilizados en el programa.

Anexo 1:

Método Formal para el Estudio de Viabilidad del Desarrollo de un Sistema Experto

J. Pazos modificó y mejoró un método básico para la cuantificación de la viabilidad de las tareas que había sido desarrollado por Slagel, y que es el que básicamente se va a exponer aquí:

El método consta de las siguientes etapas:

a. Definición de Características

En primer lugar, se consideran las cuatro dimensiones de plausibilidad, justificación, adecuación, y éxito. Sobre cada una de estas dimensiones se establecen tres categorías distintas, como son los directivos/usuarios, los expertos, y la tarea, especificándose para cada categoría las características más importantes que definen esa dimensión.

b. Asignación de Pesos

A cada característica se le asigna un peso de 0 a 10, dependiendo de su importancia relativa. Una característica esencial no tiene porqué tener un peso de 10. Deberá existir por encima de un valor umbral fijado para la aplicación, pero su peso en la evaluación de la tarea puede ser relativamente importante.

c. Evaluación

La evaluación consta de los siguientes pasos:

1.- Asignar un valor a cada característica desde 0 (ausente) hasta 10 (totalmente presente). Si el valor de una característica esencial no alcanza el umbral exigido, su cómputo es cero y la aplicación queda rechazada.

2.- Aplicar la siguiente fórmula para cada dimensión

$$V_{ci} = \frac{\sum_{j=1}^{m} P_{ij}}{\sum_{j=1}^{m} \frac{P_{ij}}{V_{ij}}} \qquad i=1..4$$

donde:

V_{c1} es el valor de plausibilidad
V_{c2} es el valor de justificación
V_{c3} es el valor de adecuación
V_{c4} es el valor de éxito
m: número de características de la dimensión
P: peso de la característica
V: valor de la característica

3.- Calcular el valor global de la aplicación como:

$$V_c = \frac{\sum_{i=1}^{4} V_{ci}}{4}$$

4.- Si el valor resultante es menor que el umbral exigido para la tarea, se rechazará la misma.

La siguientes tablas muestran todas las características que intervienen en la evaluación de la aplicación candidata, agrupadas por dimensiones, y especificando el tipo, la categoría y el peso asociado a cada una de las mismas.

Descripción	*Tipo*	*Categoría*	*Peso*
Existen Expertos	Esencial	Expertos	10
El experto asignado es genuino	Esencial	Expertos	10
El experto es cooperativo	Deseable	Expertos	8
El experto es capaz de articular sus métodos pero no categoriza	Esencial	Expertos	7
Existen suficientes casos de prueba normales, típicos, ejemplares, difíciles, etc..	Deseable	Tarea	10
La tarea no requiere de investigación básica para su solución	Deseable	Tarea	10
La tarea sólo requiere habilidad cognoscitiva	Deseable	Tarea	10
No se precisan resultados óptimos sino sólo satisfactorios	Deseable	Tarea	9
El dominio es lo suficientemente estable	Deseable	Tarea	9
Los directivos están verdaderamente comprometidos en el proyecto	Deseable	Directivos y/o usuarios	7

Dimensión de Plausibilidad

Descripción	*Tipo*	*Categoría*	*Peso*
Hay escasez de experiencia humana	Deseable	Expertos	9
El experto usa básicamente razonamiento simbólico	Deseable	Expertos	10
Existe necesidad de experiencia humana en distintos lugares	Deseable	Tarea	10
Hay necesidad de experiencia en entornos hostiles, penosos y/o poco gratificantes	Deseable	Tarea	8
No existen soluciones alternativas admisibles	Esencial	Tarea	10
Se espera una alta tasa de recuperación de la inversión	Deseable	Directivos y/o usuarios	10
Resuelve una tarea útil y necesaria	Esencial	Directivos y/o usuarios	10

Dimensión de Justificación

Descripción	*Tipo*	*Categoría*	*Peso*
Los expertos no se sienten amenazados por el proyecto, sino todo lo contrario	Esencial	Expertos	8
Los expertos son capaces de sentirse intelectualmente unidos al proyecto	Deseable	Expertos	4
Los expertos tienen un brillante historial en la realización de esa tarea	Deseable	Expertos	6
Entre los expertos hay un acuerdo sobre lo que constituye una buena solución	Deseable	Expertos	3
La única justificación para dar un paso en la solución es la calidad de la solución final	Deseable	Expertos	2
Si se usan varios expertos simultáneamente éstos concuerdan en sus soluciones	Deseable	Expertos	4
Hay cambios mínimos en los procedimientos habituales de resolución de la tarea	Deseable	Tarea	2
Las soluciones son explicables e interactivas	Deseable	Tarea	3
La tarea es de I + D o de carácter práctico	Esencial	Tarea	7
Mentalización de los directivos y usuarios con expectativas realistas en el alcance y en las limitaciones	Deseable	Directivos y/o usuarios	6
Los directivos y usuarios no rechazan de plano esta tecnología	Esencial	Directivos y/o usuarios	7
Se efectúa una adecuada transferencia tecnológica	Esencial	Directivos y/o usuarios	8

Dimensión de Éxito

Descripción	*Tipo*	*Categoría*	*Peso*
Los Expertos están comprometidos durante toda la duración del proyecto	Esencial	Expertos	7
La experiencia usada por el experto está más o menos organizada	Deseable	Expertos	5
La tarea no requiere mucho sentido común	Deseable	Tarea	6
La tarea es básicamente de tipo heurístico	Deseable	Tarea	5
La tarea implica factores subjetivos	Deseable	Tarea	5
La tarea usa, si alguna, poca generación y entendimiento del lenguaje natural	Esencial	Tarea	9
La tarea no está influenciada por vaivenes políticos	Esencial	Tarea	7
No se requieren respuestas en tiempo real inmediato	Esencial	Tarea	9
La tarea no es demasiado difícil o demasiado fácil	Deseable	Tarea	6
Es posible un enfoque incremental y/o una descomposición en subtareas independientes	Deseable	Tarea	6
La tarea está identificada como un problema en el área	Deseable	Tarea	6
La tarea tiene valor práctico	Deseable	Tarea	6
La tarea sirve a necesidades a largo plazo	Esencial	Tarea	6
Existen ya Sistemas Expertos que resuelven esa o parecidas tareas.	Deseable	Tarea	8

Dimensión de Adecuación

Bibliografía

Carrico, Michael. Building Knowledge Systems. ed. McGraw-Hill.1989

Cercone, Nick. The Knowledge Frontier. ed. Springer Verlag 1987

Forsyth, Richard. Expert Systems. Principles and case studies. ed. Chapman and Hall. 2ª edición 1989

Guida, G.. Tasso, C.. Topics in expert system design. ed. North-Holland. 1990

Hayes-Roth, Frederick. Waterman, Donald. Lenat. Building Expert Systems. ed. Addison Wesley 1983

Jackson, Peter. Introduction to expert systems. ed Addison Wesley

Klahr, Phiip. Expert Systems. Techniques, Tools, and Aplications. ed. Addison Wesley

Pazos, J. Material Docente de la Universidad Politécnica de Madrid, 1993

Raeth, Peter G. Expert Systems. A Software Methodology for Modern Applications. ed. IEEE Computer Society Press Reprint Collection 1990

Waterman, Donald. A guide to Expert Systems. ed. Addison Wesley. 1986