INTEGRACIÓN DE TÉCNICAS DE ANÁLISIS Y CLASIFICACIÓN MEDIANTE UN SISTEMA BASADO EN EL CONOCIMIENTO PARA PROBLEMAS DE DIAGNÓSTICO

J.F. Sigut, J.D. Piñeiro, R.L. Marichal, L. Moreno
Dpto. de Física Fund. y Exp., Electrónica y Sistemas, Universidad de La Laguna. Tenerife,
sigut@cyc.dfis.ull.es

Resumen

El presente trabajo pretende mostrar el diseño de un sistema inteligente para resolver una clase general de problemas de diagnóstico. Se trata de plantear el diseño e implementación de un Sistema Basado en el Conocimiento en el que se integran diversas herramientas y técnicas de clasificación y de reconocimiento de patrones. Se hace especial hincapié en la metodología de desarrollo CommonKADS, que permite hacer explícito el conocimiento incorporado. Los dominios en los que se ha trabajado hasta el momento son: el diagnóstico de patologías en señales cerebrales y el de la dislexia.

Palabras Clave: Diagnóstico, Reconocimiento de Patrones, Sistema Basado en el Conocimiento, Metodología CommonKADS

1 INTRODUCCIÓN

En este trabajo, se presenta una de las líneas de investigación del grupo de Computadoras y Control de la Universidad de La Laguna orientada a automatizar la resolución de una clase de problemas de diagnóstico o clasificación. Hasta ahora el dominio de aplicación se centraba en problemas en Biomedicina, pero se pretende comprobar la capacidad del sistema en otros dominios en los que la falta de conocimiento específico sobre el problema haga útil un sistema con las características del sistema propuesto.

Una tarea de diagnóstico tiene como objetivo encontrar la avería o el defecto que explique el mal funcionamiento de un sistema. Abusando del lenguaje, esta definición es aplicable a dominios tan diferentes como la medicina, la psicología o la ingeniería industrial, por citar algunos ejemplos representativos. Habitualmente, la tarea de diagnóstico se reduce a una tarea de clasificación en la que los síntomas observados se asocian con determinados fallos en el sistema.

La tarea de clasificación supervisada podría definirse de las siguiente manera: a partir de un conjunto de casos previamente clasificados de acuerdo con una serie de categorías predeterminadas, se trata de diseñar un sistema clasificador capaz de asignar un nuevo caso a la categoría que le corresponda.

En muy pocos dominios, se conocen reglas específicas que puedan ser usadas para diseñar un clasificador. En la gran mayoría de aplicaciones, no se asume nada acerca de la estructura del mismo, quedando ésta completamente determinada por los datos de los que se dispone (conjunto de casos de entrenamiento). Entre las cuestiones fundamentales a tener en cuenta en el diseño, podemos citar las siguientes:

- Relación entre dimensionalidad y número de datos disponibles (normalmente escasos).
- Estructura probabilística del problema. Desviaciones de la hipótesis de normalidad.
- Elección de un tipo de clasificador: paramétrico, no paramétrico, red neuronal, árboles de clasificación, ...
- Entrenamiento y validación del clasificador seleccionado.
- Poder explicativo del clasificador.

La gran interdependencia que existe entre todos estos factores, convierte el diseño de un clasificador en un proceso complejo, no existiendo una solución ideal válida para cualquier problema sino que hay

que tratar de encontrar una solución apropiada para cada caso particular considerado.

Se presenta en este trabajo una aproximación al problema del diseño del clasificador consistente en la integración de técnicas de análisis de datos y clasificación mediante un Sistema Basado en el Conocimiento (SBC). Lo que se pretende con este enfoque es automatizar el proceso, extrayendo información útil de los datos de entrada que sirva de orientación en la elección del tipo de clasificador más adecuado para el problema en cuestión. Con este fin, se ha incorporado en el sistema conocimiento propio del dominio del reconocimiento de patrones en forma de heurísticas y algoritmos [1,3,4,5].

En general, las dificultades en el análisis, diseño e implementación de los Sistemas Basados en Conocimiento son enormes, mucho mayores que las de cualquier sistema software convencional. El primer escollo está en el proceso de adquisición de conocimiento, que trata de recoger y formalizar ('representar') la información para elaborar un sistema que la manipule adecuadamente. Tradicionalmente, el diseño de SBCs ha descansado directamente sobre mecanismos particulares de implementación, tales como redes semánticas, reglas, etc. Esto implica que las estructuras del razonamiento y el conocimiento sobre el que se aplican están indisolublemente unidos. Como consecuencia, se hace imposible la reutilización de las estrategias de inferencia en otros dominios distintos de aquel para el que se han creado inicialmente. Con el fin de solucionar estas dificultades y simplificar el problema de la adquisición de conocimiento surgen en la última década diversas metodologías como *Protégé-II*, *Generic Tasks*, *Components of Expertise* o CommonKADS. Esta última es la que se ha utilizado en este trabajo.

Uno de los dominios en los que se ha trabajado son las patologías observables en las señales cerebrales. En este sentido, existe una colaboración con el Departamento de Neurofisiología del Hospital Nuestra Señora de la Candelaria de Tenerife, que nos permite disponer de datos de diferentes patologías, tales como la enfermedad de Alzheimer o demencias vasculares. Otro de nuestros problemas de interés es el diagnóstico de la dislexia. En este caso, los datos provienen de una colaboración más reciente con miembros del Departamento de Psicología Educativa, Evolutiva y Psicobiología de la Universidad de La Laguna.

En las secciones siguientes, se describirá en primer lugar la estrategia general que seguirá el sistema en el diseño de los clasificadores. A continuación, se reseñan las características de la metodología CommonKADS, prestando posteriormente mayor atención a los modelos más relevantes de entre los prescritos por la metodología: el Modelo de Conocimiento y el Modelo de Diseño. Se continúa con la descripción de la arquitectura de la implementación deasarrollada y finalmente se establecen las conclusiones del trabajo.

2 DISEÑO DEL CLASIFICADOR

El diseño del clasificador se ha planteado como una tarea de diseño general, que según Chandrasekaran [2] es una actividad compleja que implica cierto número de subtareas y métodos potencialmente disponibles para acometer estas subtareas. Así, el problema del diseño se puede definir formalmente como una búsqueda en una espacio amplio de objetos (el espacio de posibles diseños), que satisfacen ciertas restricciones. Estas restricciones se establecen de forma explícita como propias del diseño o se derivarán del comportamiento que se espera del mismo. El número de soluciones o diseños válidos es normalmente muy pequeño si lo comparamos con el gran número de posibles candidatos. Esto lleva a la utilización de algún tipo de conocimiento que acote la búsqueda de forma significativa. Con este fin, se ha seguido una estrategia del tipo "Proponer-Criticar-Modificar" cuya estructura básica es:

1. Proponer un diseño de entre la lista de posibles candidatos.
2. Verificar dicho diseño. Si satisface las restricciones, añadir a la lista de diseños válidos y volver al paso 1. Si no, continuar con el paso 3.
3. Criticar el diseño y generar una lista ordenada de posibles acciones a llevar a cabo para mejorarlo.
4. Seleccionar una de las acciones y modificar el diseño hasta que la verificación sea positiva.
5. Volver al paso 1.

3 METODOLOGÍA COMMONKADS

La metodología CommonKADS [6] se apoya en ciertos principios empíricos obtenidos en la práctica de la Ingeniería del Conocimiento a través de los

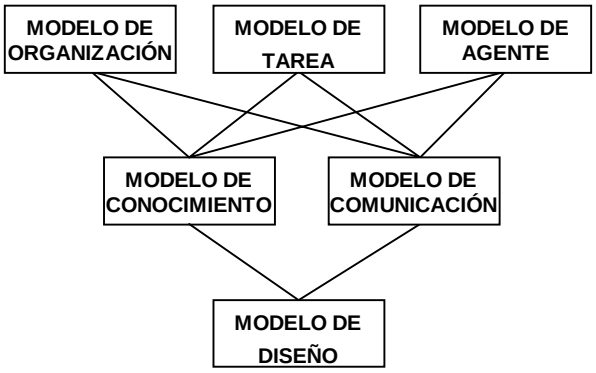

Figura 1: Modelos de la metodología CommonKADS

años por un importante grupo de expertos. Estos principios son:

- La Ingeniería de Conocimiento consiste en construir modelos de diferentes aspectos del conocimiento.
- Principio del Nivel de Conocimiento (*Knowledge-level Hypothesis*): en el modelado del conocimiento, hay que concentrarse primero en la estructura conceptual y dejar los detalles y mecanismos de implementación para más adelante.
- El Conocimiento tiene una estructura interna estable que es analizable distinguiendo tipos específicos de conocimiento y los papeles que en cada momento pueden jugar en el proceso de razonamiento.
- Un proyecto de Ingeniería de Conocimiento debe ser desarrollado en un proceso "espiral" cíclico, es decir, dada la tarea tan poco estructurada a llevar a cabo, los objetivos de cada paso del proceso deben ser flexibles y estar sujetos a replanificación en cada ciclo.

En la Figura 1, se muestran los modelos que se definen en la metodología. No todos los modelos deben ser elaborados en una aplicación concreta. Algunos de ellos pueden ser triviales según el caso. Hay que destacar la importancia de los modelos de Conocimiento y de Diseño, que se describirán a continuación.

4 MODELO DE CONOCIMIENTO

El modelo de conocimiento es una especificación de los datos y estructuras de conocimiento requeridos en una aplicación. Se trata de una descripción inteligible del papel que los diferentes componentes del conocimiento juegan en la resolución del problema. Otra cuestión a destacar es que esta descripción es independiente de la implementación posterior que se realice, quedando estos detalles para el Modelo de Diseño. El Modelo de Conocimiento tiene una estructura que es similar en esencia a los modelos tradicionales de análisis pertenecientes a la *ingeniería del software*. La tarea de razonamiento se describe a través de una descomposición jerárquica de funciones o procesos.

El Modelo de Conocimiento tiene tres componentes, cada uno de los cuales captura un grupo particular de estructuras de conocimiento. De esta manera, se distingue entre Conocimiento de Dominio, Conocimiento de Inferencia y Conocimiento de Tarea.

La primera categoría, Conocimiento del Dominio, describe el conocimiento específico del dominio y tipos de información acerca de los que queremos hablar en la aplicación. Por ejemplo, el Conocimiento del Dominio de una aplicación en torno al diagnóstico médico podría contener definiciones de diagnósticos relevantes, síntomas, y tests o pruebas clínicas, además de relaciones entre estos tipos. La descripción del conocimiento del dominio es algo comparable a un modelo de datos o un modelo de objetos dentro de la ingeniería del software. Para representar este conocimiento se usan diagramas similares a los que se emplean en la programación orientada a objetos. En la Figura 2 se muestra un diagrama de este tipo con algunos de los conceptos necesarios en el dominio del diseño del clasificador. Las flechas del diagrama indican una relación de especialización entre conceptos.

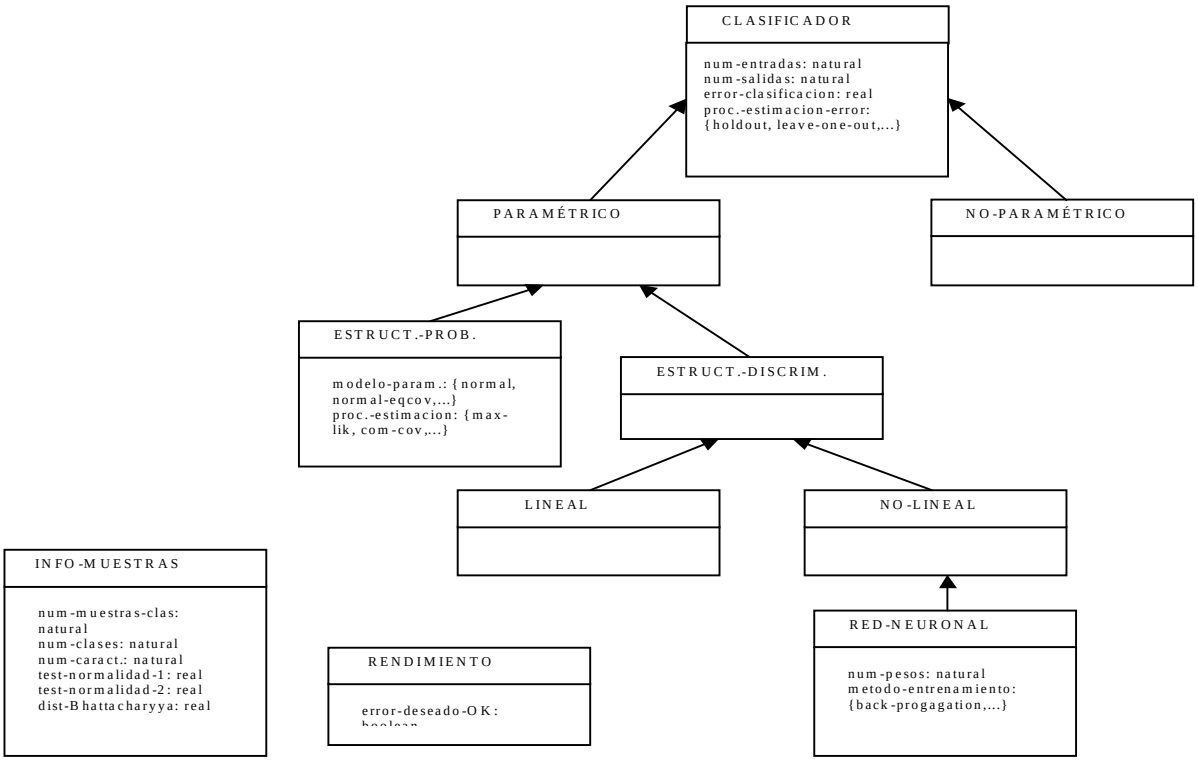

Figura 2: Algunos conceptos pertenecientes al dominio del diseño de clasificadores

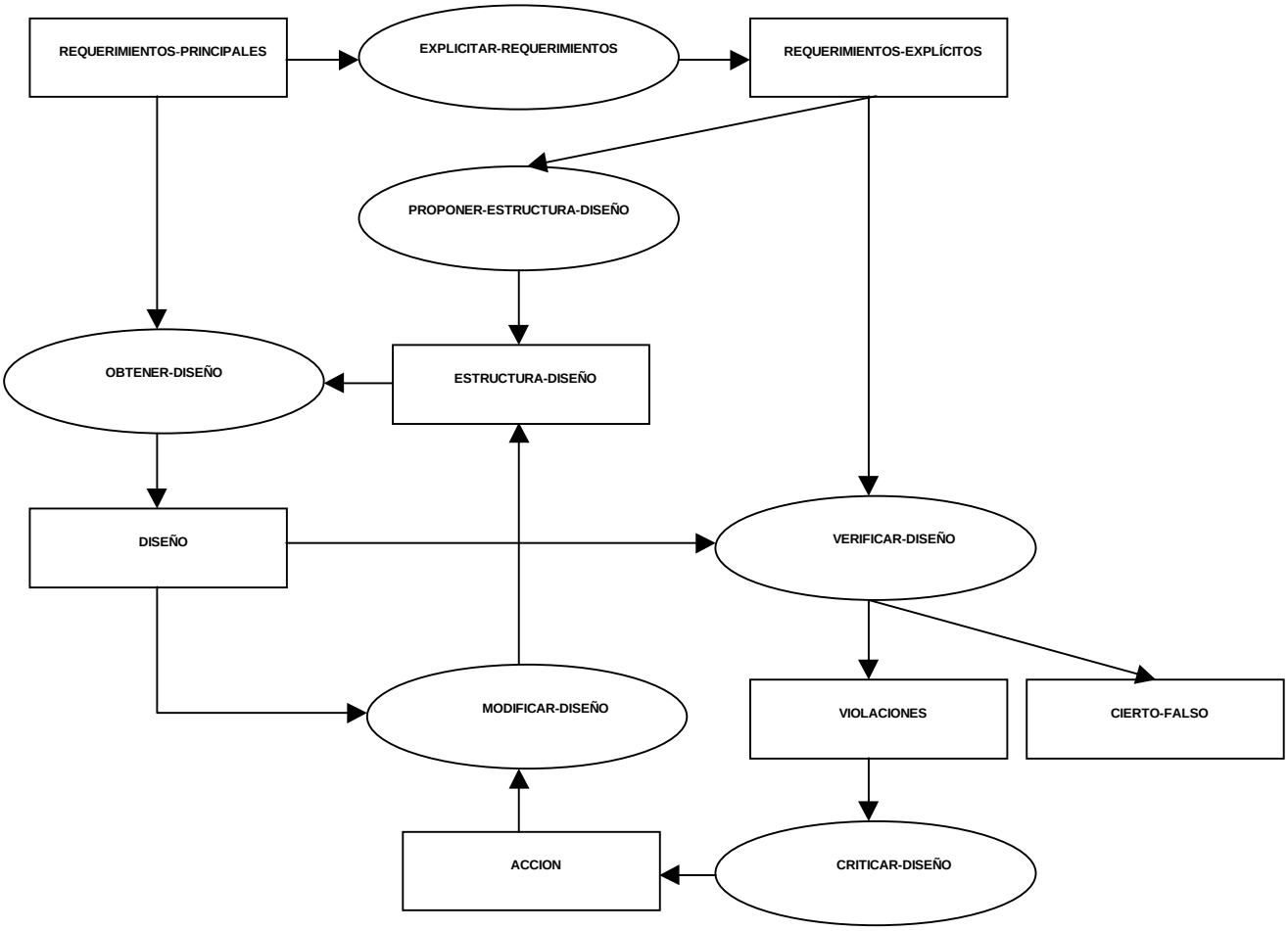

Figura 3: Estructura de inferencias para el problema del diagnóstico

La segunda categoría del Modelo de Conocimiento contiene el Conocimiento de Inferencia. El Conocimiento de Inferencia describe los pasos de inferencia básicos que queremos realizar usando el conocimiento del dominio. Las inferencias se ven mejor como los bloques de construcción de la máquina de razonamiento. Una inferencia tiene como entradas elementos del dominio y obtiene a su salida nuevos elementos que podrán ser usados por otras inferencias. En términos de ingeniería del software las inferencias representan el nivel más bajo de descomposición funcional. Dos ejemplos de inferencias en una aplicación de diagnóstico médico podrían ser una inferencia *generar-hipótesis*, que asocia síntomas con posibles diagnósticos, y una inferencia *verificar* que se refiere a tests que pueden ser usados para confirmar que cierto diagnóstico corresponde con los síntomas encontrados. La Figura 3 representa gráficamente una estructura de inferencia para el problema del diseño del clasificador. Se aprecian en este diagrama las relaciones Entrada/Salida entre las distintas inferencias básicas (óvalos).

La tercera categoría de conocimiento es el Conocimiento de Tarea. El Conocimiento de Tarea describe qué objetivo o conjunto de ellos persigue la aplicación, y cómo estos objetivos pueden ser realizados a través de una descomposición de tareas y (en último término) inferencias. Se incluye una descripción del comportamiento dinámico de las tareas, por ejemplo, su control interno. La aplicación médica podría tener *diagnosis* como su tarea de más alto nivel, y definir que puede ser realizada a través de una secuencia repetida de invocaciones de las inferencias generar-*hipótesis* y *verificar*. El conocimiento de tarea es similar a los niveles superiores de descomposición funcional de ingeniería del software, pero además incluye control sobre las funciones involucradas.

5 MODELO DE DISEÑO

Nos centraremos ahora en el problema de convertir los requerimientos especificados en el modelo de conocimiento, descrito anteriormente, en un sistema *software*. Basado en estos requerimientos, el modelo de diseño del CommonKADS describe la estructura que deberá tener dicho sistema software en términos de los subsistemas, módulos, mecanismos computacionales, y construcciones necesarios. Todos estos elementos se crean a partir de principios generales que están definidos en lo que se conoce como una arquitectura software. En el Modelo de Diseño se recomienda hacer uso del principio fundamental de preservar la identidad de los objetos definidos en los modelos previos del análisis. Esto permite establecer una correspondencia entre los productos del análisis y los obtenidos en la implementación. Otra elección importante es la plataforma de implementación a utilizar.

Con la preservación de la estructura en el diseño se consiguen satisfacer ciertos criterios:

- Reusabilidad del código: con la conservación de la estructura, se hacen explícitos el propósito y el papel que juegan los fragmentos de código, permitiendo que sean reusados en la implementación de otros sistemas basados en conocimiento. Estos fragmentos pueden ser de varios tipos y tamaños, desde implementaciones de inferencias por separado a implementaciones más complejas de tareas compuestas por varias inferencias.
- Mantenibilidad y adaptabilidad: el mantenimiento del sistema se simplifica notablemente ya que la estructura existente en la implementación permite identificar más fácilmente posibles fuentes de errores o inconsistencias y relacionarlos con una parte específica del modelo. Además, resulta más sencillo añadir mejoras con el fin de aumentar su funcionalidad.
- Poder explicativo: la necesidad de explicar por qué se ha seguido un determinado proceso de razonamiento es una característica típica de los sistemas basados en conocimiento. Con la preservación de la estructura, se posibilita la explicación de dicho proceso en el vocabulario del modelo de conocimiento, siendo posible contestar a preguntas del tipo:
  - En qué pasos elementales de la resolución del problema se usa una determinada pieza de conocimiento y qué papel juega en la inferencia que lo utiliza.
  - Cuándo y por qué es usada para resolver un problema particular (conocimiento de inferencias y tareas).

6 IMPLEMENTACIÓN

En la Figura 4 se muestran los elementos principales de la arquitectura software de nuestra implementación del modelo de diseño, que se han organizado en una estructura formada por tres niveles. La plataforma de implementación elegida ha sido el CLIPS (*C Language Integrated Production System*), herramienta de desarrollo para sistemas basados en conocimiento (nivel inferior). Hemos

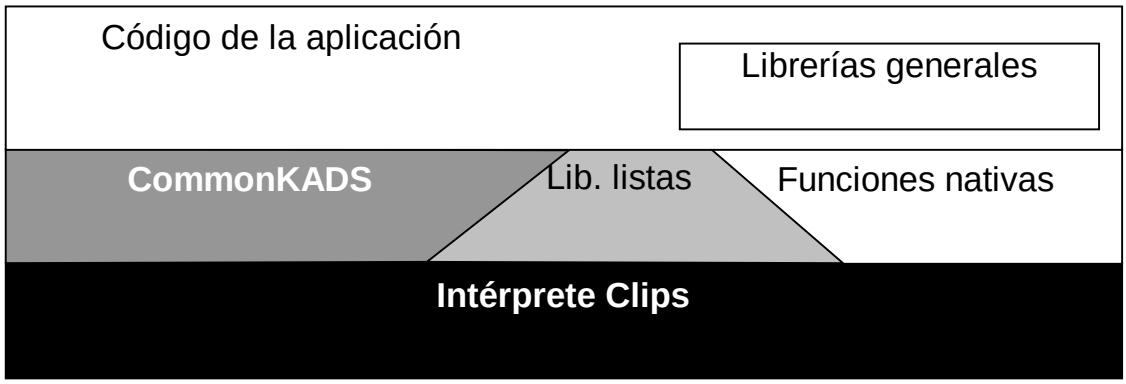

Figura 4: Arquitectura Software de la implementación realizada

optado por esta herramienta por su adecuación y versatilidad. Por encima del CLIPS encontramos un nivel intermedio que implementa de forma genérica el "*mapping*" de los objetos del CommonKADS a la arquitectura CLIPS. La última capa constituye la implementación de la aplicación particular considerada. Esta estructura permite un nivel adicional de reusabilidad del código CLIPS. Se define el nivel intermedio como una capa genérica que cualquier SBC realizado con la metodología CommonKADS empleará sin modificación, reservándose el nivel superior para las especificidades de la aplicación concreta.

El producto final que se obtiene es un conjunto de librerías CLIPS que dan soporte a la implementación de los objetos definidos en los modelos de análisis del CommonKADS. Para automatizar en gran parte este proceso, se ha elaborado también una herramienta que permite la generación semiautomática del código que será finalmente ejecutado por el intérprete CLIPS a partir de los modelos CommonKADS.

Debido a la gran cantidad de algoritmos que se debían de integrar en el SBC de diseño de clasificadores se consideró oportuno integrar al sistema un lenguaje orientado al análisis y visualización de datos, el lenguaje R. Este es una versión compatible con el conocido lenguaje S-PLUS [7]. De esta forma, se aprovechan las facilidades de este lenguaje para la manipulación de datos y las numerosas funciones con que cuentan sus librerías, evitando el desarrollo de código adicional en CLIPS.

7 CONCLUSIONES

Se ha presentado un sistema inteligente para elaborar clasificadores a partir de datos previamente clasificados. A partir de las características de los datos, el sistema diseña un método de clasificación apropiado al problema. Se pretende que, en dominios donde no exista un conocimiento profundo sobre el problema a tratar, se obtenga como resultado un sistema clasificador adecuado. Se ha hecho énfasis en el uso de la metodología, ya que la complejidad del sistema hace necesario que se hagan explícitas las estrategias de razonamiento para facilitar su mantenimiento.

Referencias

[1] Bishop, C. M., (1995) Neural Networks for Pattern Recognition. Oxford University Press

[2] Chandrasekaran, B. (1990) "Design Problem Solving: A Task Analysis", *AI Magazine*

[3] Duda, R. y Hart, P., (1973) Pattern Classification and Scene Analysis, John Wiley & Sons

[4] Fukunaga, K., (1990) Statistical Pattern Recognition, Academic Press

[5] Ripley, B. D., (1996) Pattern Recognition and Neural Networks. Cambridge University Press

[6] Schreiber, G., (1999) Knowledge Engineering and Management: The CommonKADS Methodology, MIT Press

[7] Venables, W.N. y Ripley, B.D., (1999). Modern Applied Statistics with S-PLUS. pringer-Verlag