# *Nuevas perspectivas en resolución de problemas de matemáticas mediante el uso de modelos semánticos de representación del conocimiento*[*]

**Fernando Díez, Rafael Gil**

Departamento de Ingeniería Informática
Universidad Autónoma de Madrid
Francisco Tomás y Valiente 11, 28046 Madrid (Spain)
fernando.diez@uam.es, rafael.gil@estudiante.ii.uam.es

**Resumen**: El trabajo que se presenta está relacionado con el desarrollo de herramientas que permitan la interacción con el alumno en sistemas de resolución de ejercicios de Matemáticas. Existe un déficit importante de aplicaciones que fomenten y supervisen el trabajo deductivo en Matemáticas y que permitan explorar el nivel de aprendizaje alcanzado por un alumno en términos de razonamiento. Con el fin de mejorar la capacidad de supervisión, estamos desarrollando una ontología así como un motor de razonamiento para la representación del conocimiento en Matemáticas, con el fin de hacer uso de la misma mediante aplicaciones específicas (por ejemplo para realizar demostraciones sobre geometría). Se presenta el prototipo del sistema RamSys el cual integra la información semántica necesaria para la resolución de problemas de Matemáticas.
**Palabras clave**: Representación de Conocimiento, Ontologías, OWL, Entornos Educativos Inteligentes, Matemáticas.

**Abstract**: In this paper we present the development of tools that permit interaction with the student in problem solving systems of Mathematical exercises. Currently there exists an important deficit of applications which promote and supervise deductive work in Mathematics and that allow exploration of the learning level reached by a student in reasoning terms. With the aim of improving supervisory capacity, we are developing an ontology for the representation of knowledge in Mathematics, with the purpose of making use of the same by means of specific applications (for example carrying out geometric demonstrations). We present the RamSys system prototype which involves semantic information needed in order to Mathematical problem solving
**Key words:** Knowledge Representation, Ontologies, OWL, Intelligent Educational Systems, Mathematics.

## 1. Introducción

La enseñanza de las Matemáticas, en la pizarra y con lápiz y papel, se está viendo complementada, en estos últimos años, por la aparición de aplicaciones informáticas específicas para el aprendizaje en diferentes contextos. Además de la actividad docente desarrollada por el profesor, con frecuencia los alumnos acuden a los laboratorios de informática para hacer uso de programas específicos de aprendizaje. La implantación de las nuevas tecnologías está influyendo en la metodología de la enseñanza. Las Matemáticas constituyen una disciplina con una notable dificultad en la definición de contenidos. Esto

[*] *Artículo seleccionado de SIIE 2005, VII Simposio Internacional de Tecnología Educativa (Leiria, Portugal 2005), extendido y revisado para su publicación en IE Comunicaciones.*

 

contribuye a orientar la creación de IES (*Intelligent Educational Systems*) hacia aplicaciones que fomenten el aprendizaje puramente algorítmico, dejando de lado el aprendizaje deductivo basado en conceptos. Creemos pues necesario investigar en la creación de aplicaciones que permitan expresar al alumno sus conocimientos sin dejar de lado la supervisión conceptual de su actividad. Para llevar a cabo la investigación hemos escogido como contexto de trabajo la Geometría Afín Plana. En el trabajo, además de los autores, han colaborado profesores del Departamento de Matemática y Didáctica de la Universidad de Aveiro (Portugal). Dicho contexto se adapta adecuadamente a la representación de conceptos como por ejemplo: *punto*, *recta*, *vector*, *coordenada*, *segmento*, y relaciones como *pertenece*, *perpendicular*, *paralela*, etc.

En el artículo que presentamos vamos a exponer la concepción de un sistema que pretende dar respuestas a las cuestiones que acabamos de plantear. Para poner de manifiesto el tipo de actividad del alumno que deseamos supervisar, vamos a comenzar planteando un ejemplo de una hipotética situación de aprendizaje. Este ejemplo nos va permitir definir, como veremos en el siguiente epígrafe, los requisitos básicos de un sistema de las características del que estamos desarrollando. Una vez que el lector tenga formada una idea relativamente concreta del objetivo será el momento de poder abordar la descripción del sistema desde el punto de vista relativo a su arquitectura y al entorno de comunicación de mensajes entre los distintos módulos. Como veremos este es uno de los aspectos fundamentales en la concepción del sistema, dado el elevado número de tecnologías que incorpora. Para concluir el artículo plantearemos algunas cuestiones abiertas que aún están por desarrollar.

### 1.1. La necesidad de nuevas herramientas

La duda es inherente al proceso de enseñanza-aprendizaje. Todos nos hemos encontrado, en alguna ocasión, con esta situación. La presencia de la duda en el alumno y la necesidad de que el profesor la resuelva es parte del proceso. Es más, es una parte esencial, dado que la aparición de la duda pone de manifiesto la existencia de un trabajo previo del alumno. Sería deseable, además, que dicho trabajo previo fuera acompañado de una reflexión sobre lo que se está haciendo. Finalmente, de la respuesta que obtiene el alumno, se deriva su aprendizaje.

En Matemáticas esta circunstancia es, si cabe, más frecuente que en otras materias por distintas razones: su carácter abstracto, su axiomatización, la posibilidad de representación de datos, objetos y proposiciones de formas muy diferentes según el contexto, etc. Estos motivos, unidos a la eterna pregunta de *para qué sirven* (mencionemos aquí al hombre anumérico de Allen Paulos [Paulos 00]) o la cada vez más escasa formalización de conceptos que se da entre los escolares, hacen que las Matemáticas no gocen de un elevado prestigio social. Es más, se da la paradoja de que la creciente tecnificación de nuestra sociedad nos aleja aún más de la materia, cuando seguramente debería ser al contrario. En una sociedad moderna como en la que vivimos, el progreso ha venido de la mano de una nueva disciplina, la Informática. La nueva revolución social que está teniendo lugar ya está produciendo sus primeras consecuencias como pueden ser la brecha digital o el desencuentro definitivo entre la Sociedad y las Matemáticas.

En este contexto los escolares, sean del nivel que sean, son conscientes de la situación en la que viven. Lógicamente intentan sacar partido de ello y, como acabamos de decir, tratan de trivializar en la medida de lo posible su relación con la materia. Así, con el fin de jugar en el terreno preferido por los alumnos, debemos tratar de aprovechar las circunstancias tecnológicas que vivimos. La mejora en la calidad de la docencia debe venir, necesariamente, desde dos vías: de una parte el trabajo de formación del profesorado en Didáctica de las Matemáticas, debemos romper con la tradición de enseñar como nos enseñaron; de otra parte, creemos que debe venir de la integración de la tecnología en el aula.

Esta segunda vía motiva, en gran medida, el contenido de nuestro trabajo. Como ya hemos indicado más arriba, la dificultad en la representación del conocimiento es una de las trabas más importantes a la hora de diseñar y desarrollar sistemas que traten de utilizar este conocimiento para comunicarse tanto con el profesor como con el alumno. Existen iniciativas en esta dirección como por ejemplo *Calculus Machina* [Quinney 02], sistema

tutor para el aprendizaje de Cálculo; *Wiris* [Xambó et al. 02], sistema de cálculo numérico y simbólico a través de Internet, o *MathEdu* [Díez et al. 04], herramienta de autor orientada al diseño y resolución de problemas de cálculo simbólico. Sin embargo, en palabras de Mizoguchi [Mizoguchi et al. 97], aun sigue existiendo una notable distancia entre las herramientas de autor y estos sistemas. En consecuencia, deseamos desarrollar herramientas que posibiliten su interacción con los alumnos, siendo capaces de supervisar su actividad, las expresiones que escriben y su validez lógica y semántica.

### 1.2. La activad del alumno

Con el fin de hacernos idea de la dificultad del tipo de actividades que deseamos analizar vamos a plantear un ejemplo relacionado con el tipo de actividades del usuario, generalmente un estudiante, que deseamos supervisar.

Consideremos el enunciado siguiente:

> *Obtén la ecuación de la recta que pasa por el punto P(1,2) y es paralela a la recta* $s:\begin{cases} x = 2 + t \\ y = -1 - t \end{cases}$

Este ejercicio es bastante sencillo. Sin embargo, un análisis detallado va a poner de manifiesto las dificultades que pueden surgir a la hora de controlar las actividades de un alumno durante la resolución. Para ilustrar el análisis vamos a desarrollar sendas resoluciones del ejercicio.

*Resolución 1*: La resolución más evidente de este ejercicio pasa por observar que el vector director de la recta pedida es el mismo que el de la recta dada, esto es $\vec{v} = (1,-1)$. Dado que tenemos un punto, *P*, y el vector director, podemos construir la recta solución de forma inmediata

$$r:\begin{cases} x = 1 + m \\ y = 2 - m \end{cases};\quad \forall m \in \mathrm{R}$$

Finalmente el alumno debe concluir que *r* es paralela a *s*.

*Resolución 2*: Algo más rebuscada pero igualmente válida resulta la resolución según el siguiente planteamiento. Como tenemos un punto, *P*, por el que debe pasar la recta, definimos el haz de rectas que pasan por dicho punto. Dicho haz lo podemos expresar como

$$\frac{x-1}{u_1} = \frac{y-2}{u_2}$$

Ahora, como sabemos que la recta pedida debe ser paralela a la recta *s* de ecuación dada, sus vectores directores deben coincidir o ser equivalentes. Podemos por tanto efectuar la identificación

$$\vec{u} = \vec{v} = (1,-1)$$

Sustituyendo en la ecuación anterior obtenemos la siguiente forma continua de la recta pedida que, aun siendo válida, no suele ser la forma habitual en que se expresa la respuesta

$$r: \frac{x-1}{1} = \frac{y-2}{-1}$$

Finalmente el alumno debería expresar esta ecuación en su forma completa, con la que daría por concluido el ejercicio

$$r: x + y - 3 = 0$$

Como podemos comprobar, el resultado obtenido en la segunda resolución difiere del obtenido en la primera. Sin embargo ambas alternativas son equivalentes e igualmente válidas. Si analizamos con detalle las diferencias existentes entre ambas resoluciones desde el punto de vista del tipo de objetos utilizados, vemos que, en el primer caso, el alumno ha hecho uso de tres conceptos: vector, punto, recta paramétrica y la relación paralela. En el segundo caso ha utilizado: haz de rectas, vector, punto, recta continua, recta completa y la relación paralela.

Existe coincidencia entre algunos de los conceptos y relaciones utilizados. El análisis de ejercicios relativos a la Geometría Afín nos muestra que mediante una cantidad pequeña de objetos conceptuales y relaciones podemos abordar la resolución de numerosos ejercicios. Esto nos llevó a la idea de que siendo capaces de representar a través de una ontología el tipo de conceptos que se manipulan así como las relaciones entre los mismos, podríamos tratar de supervisar la actividad del alumno. En definitiva, el problema planteado es el de representar la información semántica necesaria que

está contenida en la definición de un problema en un contexto particular, para poder plantear ejercicios cuya fuente de información proceda, tanto de los modelos definidos por el profesor como de las taxonomías de problemas que componen los mismos. Además, un valor añadido es la garantía de que los ejercicios generados sólo pueden ser coincidentes en los objetivos instruccionales; los datos son aleatorios y generados en cada caso en tiempo real. Hemos comprobamos con agrado que la tarea de definir ontologías que describen objetos geométricos ya se ha acometido anteriormente, lo cual nos ha permitido un avance más rápido en la concepción y desarrollo de nuestro prototipo.

Las necesidades de supervisión que surgen a la hora de monitorizar la actividad del alumno son de dos tipos. Por un lado es necesario verificar la corrección, desde un punto de vista semántico, de los objetos que el alumno define, evitando la posibilidad de que genere inconsistencias en la base de conocimiento del problema. Esto sucede cuando por ejemplo intenta definir, de forma genérica, un objeto geométrico. Por ejemplo supongamos que el alumno declara que *P* es un punto. El sistema lo instancia como tal, aunque todavía no conozca cuál es la definición del mismo en términos de sus coordenadas. En su razonamiento puede suceder, por ejemplo, que el alumno trate de identificar a *P* con otro tipo de objeto, o trate de usarlo en una relación que no admita dicho objeto en sus argumentos (por ejemplo si afirma que *P* es perpendicular a *r*). Veremos más adelante que existe un componente en *RamSys* dedicado específicamente a mantener en todo momento la consistencia de los objetos definidos.

Un segundo tipo de supervisión es el que se requiere cuando el alumno escribe expresiones matemáticas. En este caso es necesario comprobar, generalmente, la equivalencia entre dos expresiones o si se cumple una determinada relación. Tal caso se presenta cuando el alumno ha definido una recta, por ejemplo $r : x + y - 3 = 0$, y afirma, en otra parte de su construcción, que dicha recta es paralela a la recta de ecuación dada $s : \begin{cases} x = 2 + t \\ y = -1 - t \end{cases}$. En esta situación se debe producir, en primer lugar, una verificación semántica de la actividad, comprobando que la relación de pertenencia es aplicable a los dos objetos geométricos utilizados. Al ser ambos objetos rectas la comprobación semántica es correcta. Queda una segunda parte, la comprobación matemática de que, en efecto, *r* y *s* son paralelas. En RamSys hemos incorporado otro componente que hace uso del programa de cálculo simbólico *Mathematica*, y se encarga de efectuar este tipo de comprobaciones.

RamSys incorpora, adicionalmente, otra forma de supervisión que permite valorar cuándo el alumno ha conseguido los objetivos instruccionales definidos para el tipo de ejercicio que está resolviendo. El significado de estos es equivalente al de un conjunto de predicados. Cuando el profesor define un tipo de ejercicio debe especificar, entre otros elementos, cuáles son los objetivos de ese ejercicio. Para cada tarea que realiza el alumno el sistema comprueba si se cumplen. Mientras esto no suceda el sistema no da por concluida la resolución. En el caso del ejercicio del ejemplo anterior el objetivo definido es

```
And[Parallel[Line[$l1$], Line[$l2$]],
    Incident[Point[$p1$], Line[$l2$]]]
```

El objetivo definido requiere que se verifiquen simultáneamente las dos condiciones exigidas en el enunciado del ejercicio. Por un lado que la recta que defina el alumno (etiquetada como $l2$) sea paralela a la dada. Por otro lado, la recta que obtenga el alumno debe ser, a su vez, incidente con el punto que se proporciona en el enunciado (etiquetado como $p1$). Como ya hemos indicado más arriba, mientras no se cumpla este objetivo no se considera concluido el ejercicio.

En resumen, hemos analizado las necesidades de supervisión que debemos tener presentes a la hora de desarrollar un sistema capaz de controlar la actividad que realiza un alumno para la resolución de un ejercicio de Matemáticas. Probablemente se puedan definir otras formas de supervisión que incidan más en aspectos, por el momento no considerados, tales como el propio proceso de razonamiento en sí mismo. Sin embargo, abordar este tipo de supervisión plantea unas dificultades indudablemente mayores.

### 1.3. Requisitos del sistema

En Geometría Afín es habitual efectuar demostraciones a partir de un conjunto de axiomas

básicos (postulados), como por ejemplo "por dos puntos distintos *P* y *Q* pasa una única recta *r*". El alumno puede explorar con relativa sencillez tanto este tipo de enunciados básicos como otros de mayor complejidad conceptual haciendo uso de herramientas de geometría dinámica o DGT (Dynamic Geometry Tools), como *Cabri-geometry* [Cabri], *Cinderella* [Cinderella] o *Geometer's Sketchpad* [Geometer]. El alumno que hace uso de estas herramientas generalmente manifiesta su sorpresa y satisfacción fundada principalmente en los dos motivos [Breda et al. 2003] siguientes:

- la posibilidad de visualizar la abstracción teórica que está explorando,
- la sencillez de uso y lo grato que resulta hacer matemáticas sin "padecer" lo ingrato de su abstracción

Ahora bien, a nuestro juicio, sería deseable proporcionar a este tipo de sistemas de geometría dinámica otras funcionalidades que permitan al alumno realizar de forma libre construcciones basadas en los conceptos teóricos como el que mostrábamos en el ejemplo de más arriba. Por tanto, desde un punto de vista computacional, parece necesario definir un marco de representación de conocimiento sobre el que poder desarrollar mecanismos de demostración. Dicho marco teórico debe contener, por una parte, los conceptos con los que vayamos a trabajar. Por otro lado es necesario representar las relaciones entre ellos. Así, por ejemplo, en el postulado anterior hemos de representar los conceptos *punto* y *recta* y las relaciones *pertenece* y *paralela*. En este sentido, nos planteamos un sistema con dos objetivos diferenciados: i) definición de ejercicios por parte del profesor y ii) resolución razonada de ejercicios por parte del alumno. A continuación detallamos los requisitos básicos de ambos objetivos:

- Para el profesor: una herramienta de autor que le permita plantear problemas de forma sencilla:
  - Definir los textos de los enunciados de los problemas, los cuales pueden incorporar fórmulas matemáticas. Se admite, además, la presencia de elementos aleatorios en los enunciados.
  - Identificar los elementos del enunciado que el alumno debe interpretar.
  - Definir los datos iniciales del problema. El profesor puede interpretar parte del enunciado incorporando información al sistema (instancias, relaciones entre objetos, etc.). Esta información estará a disposición del alumno al comenzar la resolución del problema, como parte de la base de conocimiento del mismo. El alumno podrá hacer uso de los conceptos y relaciones que desee por el mero hecho de ser parte del enunciado.
  - Objetivos a alcanzar por el alumno. El profesor debe especificar objetivos que el alumno debe cumplir. Los objetivos pueden requerir una secuencia deductiva (grafo de objetivos) por parte del alumno o no. Cuando el alumno considere que el problema está resuelto el sistema verificará que los objetivos definidos por el profesor se han satisfecho.

Al respecto de la secuencia de objetivos que acabamos de comentar, la idea que se persigue es que el profesor tenga la capacidad de definir un cierto orden en la consecución de los mismos o, por el contrario y si así lo desea, dejar libertad en la obtención de los resultados pedidos. Así hemos previsto la existencia de la propiedad *hasPreviousObjective* que pueden tener algunos problemas la cual, usada en combinación con *hasObjective*, permite definir un grafo de objetivos y subobjetivos (figura 1).

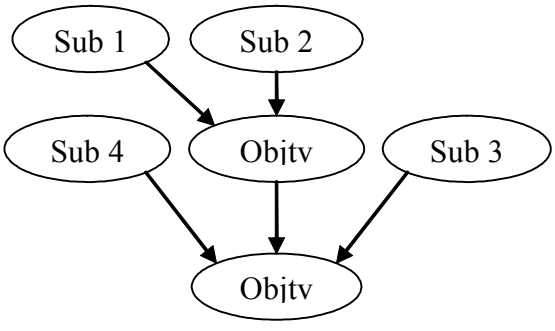


**Figura 1.** Grafo de objetivos para un problema

La alternativa a este esquema es la definición de un conjunto (no ordenado) de objetivos (se debe cumplir

la conjunción de todos ellos independientemente del orden).

- Para el alumno: un entorno dinámico, de interacción sencilla (uso de paletas, botones, explicaciones, etc.), y personalizada. El alumno debe interpretar el enunciado y aplicar los mecanismos de resolución que considere adecuados para alcanzar los objetivos propuestos por el profesor. Debe disponer de una notable libertad en la resolución de ejercicios, asociada con su nivel de razonamiento. Esta circunstancia requiere que el sistema analice cada acción realizada por el alumno, verificando su coherencia desde un punto de vista lógico.

En el siguiente apartado, vamos a realizar una descripción del sistema al nivel de sus componentes fundamentales y de los mecanismos de comunicación e integración de los mismos.

## 2. Descripción del sistema

### 2.1. Arquitectura

La arquitectura distribuida del sistema se compone de una serie de módulos que funcionan en máquinas virtuales diferentes. El sistema se estructura en dos subsistemas principales:

- un servidor Web que se encarga de la comunicación con los usuarios (profesores o alumnos) mediante los correspondientes editores que funcionan sobre un navegador estándar.
- un sistema de conocimiento, RAMSYS (Reasoning And Managing SYStem), que se encarga de servir de base de datos de problemas, dar soporte de cálculo simbólico y, la parte fundamental del mismo, un módulo de representación del conocimiento.

La comunicación entre ambos módulos utiliza la librería RIACA OM Library (ROML) [Riem 04]. Dicha librería está orientada a la comunicación entre aplicaciones mediante OpenMath, un lenguaje estándar y de fácil extensión para representar conceptos matemáticos desde un punto de vista semántico. Por otro lado, el módulo que alberga la parte de representación de conocimiento (las clases de objetos matemáticos y sus relaciones), la verificación lógica de las acciones del alumno, está basado en el lenguaje OWL (*Ontology Web Language*) [Baader 03], [Gómez-Pérez et al. 04]. Para que el lector pueda hacerse una idea de cómo es la definición de las clases de objetos y relaciones definidos en la ontología, vamos a incluir a continuación la estructura de clases correspondientes a la definición de un problema.

#### 2.1.1. Definición de la Ontología

Para la definición de las clases de la ontología hemos utilizado una herramienta denominada *Protégé* [Protégé], desarrollada en el centro *Stanford Medical Informatics* de la Escuela de Medicina de la Universidad de Stanford (EE.UU.). *Protégé* es un editor de ontologías así como un entorno de Bases de Conocimiento. En concreto nosotros hemos hecho uso de una extensión de la herramienta que soporta el lenguaje OWL. No es este el momento apropiado para introducir las características y riqueza que ofrece tanto la herramienta como el lenguaje utilizado. Para ello remitimos al lector a la abundante y buena información existente en la página Web de *Protégé* o bien al texto [Baader 03]. Pero para que pueda formarse una idea del tipo de información que manipulamos vamos a dar algunos detalles de la implementación de la ontología. Así, en la figura 2 se aprecia, por ejemplo, la definición que presentan las *propiedades* de la clase definida *RamSysProblem*, una de las utilizadas para la representación de los problemas.

A modo de explicación precisaremos que las propiedades etiquetadas como D representan propiedades de tipo de tipo de datos (enteros, reales, strings, etc.), mientras que las etiquetadas como O representan propiedades de los objetos propiamente dichos. Así, por ejemplo, todo problema en RamSys debe tener una descripción (un literal) y un enunciado. Además debe existir un objetivo para el problema (representado por el cuantificador existencial de la propiedad). Además se aprecia que la propiedad *hasObjective* está definida como *múltiple* dando idea de que puede haber distintos objetivos en la resolución de un problema.

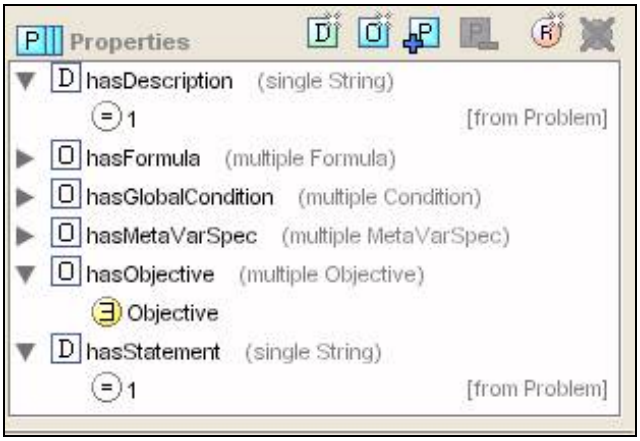


**Figura 2.** Propiedades de un problema en RamSys

Siguiendo con la descripción de los problemas en *RamSys*, estos se definen mediante una serie de clases. A modo de ejemplo explicaremos cómo se define la clase *Formula* (figura 3). Junto a cada aspecto de su definición explicamos, a continuación, el significado de la misma:

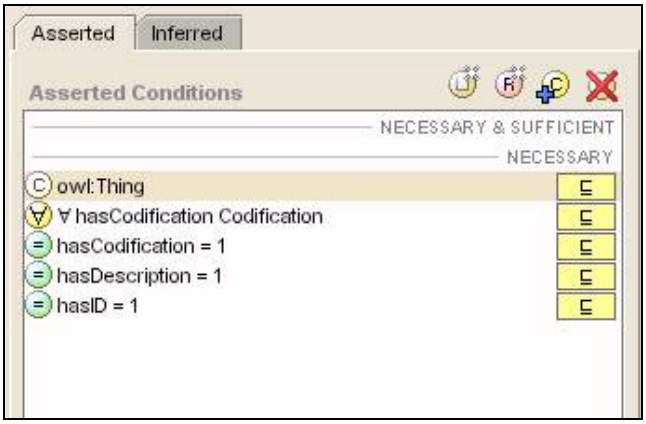


**Figura 3.** Condiciones de la clase Fórmula en RamSys

Clase: Fórmula

Descripción: Una fórmula o patrón de fórmula se caracteriza por un identificador (ID) para determinar en qué partes del enunciado aparece dicha fórmula, así como una descripción en función de distintas metavariables.

Definición: (condiciones necesarias)

- *owl:Thing*: indica que la clase *Formula* es subclase de *owl:Thing*, la clase más general de OWL.
- ∀ *hasCodification Codification*: indica que todas las codificaciones son de la clase *Codification*, la cual toma uno de los siguientes posibles valores: {OpenMath, Mathematica, MathML, Jome}
- *hasCodification = 1*: indica que toda fórmula tiene exactamente una codificación.
- *hasDescription = 1*: indica que toda fórmula tiene exactamente una descripción.
- *hasID = 1*: indica que toda fórmula tiene exactamente un identificador.

A su vez, la clase fórmula posee una serie de subclases que permiten definir condiciones y metavariables para las mismas. Para una descripción más detallada del tipo de datos empleados en la definición de problemas puede consultarse [Díez 02], dado que hemos utilizado la descripción de los problemas que se utiliza en la declaración de problemas en *MathEdu*.

### 2.1.2. El valor añadido del uso de ontologías

Visto lo anterior, más allá de las posibilidades semánticas de representación que nos ofrece OpenMath, mediante el uso una ontología nos es posible verificar, desde un punto de vista lógico, los objetos y las relaciones entre los mismos que define el alumno. Es posible que durante su actividad con el sistema el alumno manipule objetos de forma incorrecta o incompleta. El sistema ha de ser capaz de procesar dicha información para advertir al alumno de su incorrección. En tales situaciones OpenMath no es capaz de realizar las comprobaciones semánticas del tipo que vamos a analizar en el ejemplo siguiente. Consideremos el siguiente problema, pequeña variación del visto anteriormente:

> *Determinar la familia de rectas que pasan por* $P=(x_0,y_0)$ *y son paralelas a la recta de vector director* $\vec{u}=(u_0,u_1)$

Supongamos que un alumno realiza las siguientes tareas de definición de objetos del dominio a partir de los datos contenidos en el enunciado. Para realizar estas tareas dispone de un editor de ecuaciones accesible vía Web:

- Define el objeto *recta* identificado como *r*. Se añade la instancia *line*(*r*) en la base de conocimiento.
- Define el objeto *recta* identificado como *s*. Se añade la instancia *line*(*s*) en la base de conocimiento.
- Establece el paralelismo entre ambas rectas. Se añade la instancia *parallel*(*r*,*s*) en la base de conocimiento.

Mediante las dos primeras acciones el alumno ha definido dos objetos: las dos rectas identificadas como *r* y *s*. Con la tercera los ha relacionado mediante la propiedad de paralelismo. En OpenMath no existe una forma de verificar la corrección semántica de esta última sentencia, pese a ser correcta matemáticamente, sin tener que volver a especificar que los objetos *r* y *s* son rectas, incluyendo sus tipos, para comprobar así que pueden relacionarse mediante el predicado de paralelismo.

Sin embargo mediante el uso de la ontología, en los dos primeros pasos se instancian dos nuevos objetos de tipo *line*. Así el sistema conoce qué propiedades y qué relaciones pueden aplicarse sobre los objetos que define el alumno y no es necesario volver a especificar los tipos. Por ejemplo, si la tercera acción hubiera sido $r \in s$ (representado por *in*(*r*,*s*)), se hubiera detectado una inconsistencia entre la relación de pertenencia de los objetos *r* y *s* instanciados en la ontología. El sistema detectaría el error puesto que dos rectas no pueden relacionarse mediante la relación de pertenencia.

Como acabamos de ver, en el ejemplo sobre la ontología se incorporan dinámicamente las instancias y relaciones asociadas a las acciones que realiza el alumno. Para mantener la consistencia de la base de conocimiento es necesario analizar la coherencia de las nuevas acciones con respecto a las anteriores mediante el uso de un sistema razonador. Por ejemplo, si el alumno ha definido *A* como punto y posteriormente trata de utilizarlo o redefinirlo como si fuera una recta; o el caso en que utiliza un objeto aún no definido en una relación, ¿qué debe hacer el sistema en estas circunstancias? Analizaremos más adelante las necesidades de consistencia.

Volviendo sobre la arquitectura del sistema, en la figura 4 mostramos un esquema de la misma. Como se puede apreciar la hemos dividido en dos grandes módulos: la parte correspondiente al servidor Web, que establece la comunicación del sistema tanto con el profesor como con el alumno y la parte del módulo de razonamiento, que almacena las definiciones de problemas y los elementos de codificación, verificación, etc. En el ejemplo anterior acabamos de describir una situación típica de interacción del alumno con el sistema en la que es necesario comprobar las acciones de aquel. Por tanto es necesario efectuar una traducción entre los diferentes lenguajes de representación que soporta cada componente. La comunicación entre *RamSys* y los usuarios se realiza a través de un editor de expresiones matemáticas alojado en un servidor Web, el cual se encarga de distribuir el sistema a través de Internet (puede accederse en la dirección http://150.244.57.127:8084/RamSys/index.jsp). Para la edición de fórmulas a que nos referimos hacemos uso de JOME (*Java OpenMath Editor*) [Dirat et al. 99], que proporciona una interfaz para la edición y comunicación de expresiones matemáticas mediante el estándar OpenMath. La interfaz que proporciona JOME ha resultado suficiente para los propósitos del prototipo desarrollado, aunque pensando en un uso más continuado de la herramienta sería necesario mejorar dicha interfaz.

En el caso del profesor, éste dispone de un editor de problemas (actualmente sobre *Protégé*) para definir los enunciados y sus características, los cuales se almacenan en una base de datos de problemas. En el momento de la resolución, existe un módulo generador de problemas que es el encargado de acceder a la misma para obtener un modelo de problema. En caso de ser necesario, el generador obtiene del sistema de cálculo simbólico *Mathematica* [Wolfram 99] los datos necesarios para generar los problemas que se proponen al alumno para su resolución.

En el caso del alumno, cada una de sus acciones de resolución se codifican en OpenMath. El código resultante se procesa a través del módulo al que hemos denominado BATMAN (*Bivalent AnnoTation Manager*). El proceso involucra dos etapas. En la primera se obtiene su traducción a código OWL para poder efectuar las anotaciones y verificaciones correspondientes en la base de conocimiento. Todas las anotaciones se efectúan dinámicamente mediante

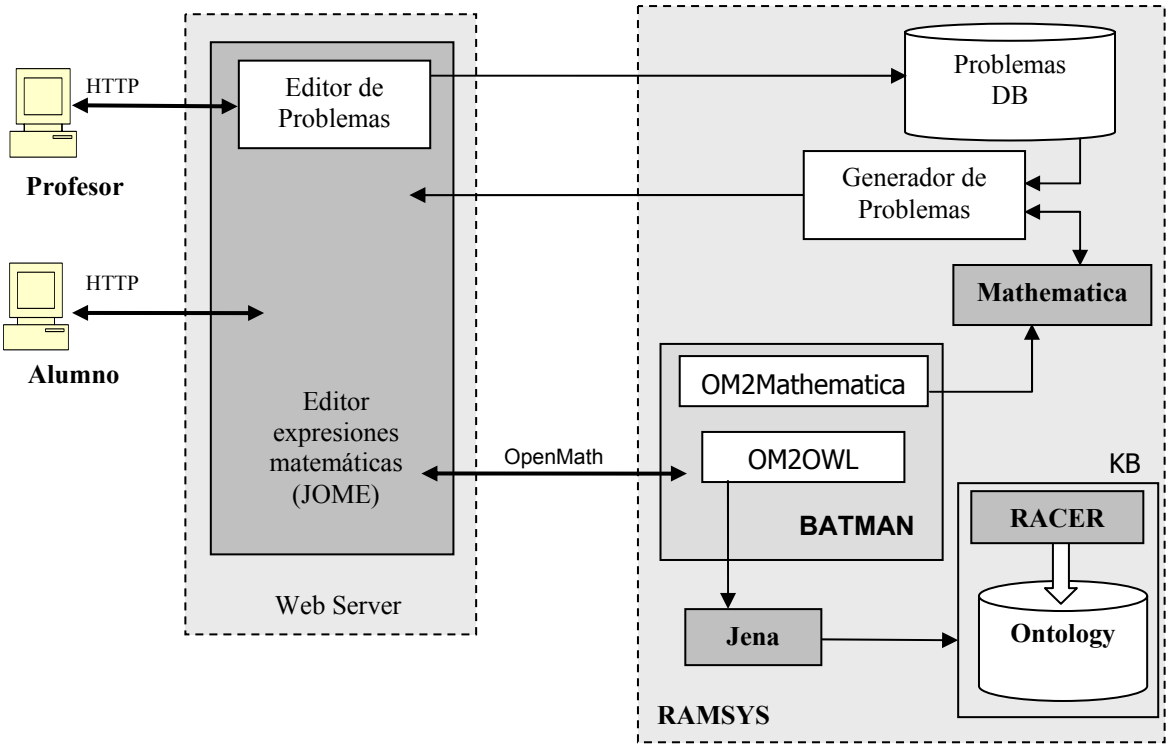


**Figura 4.** Arquitectura del sistema

la API *Jena Semantic Web Framework* [Jena]. Cada anotación requiere comprobar la integridad de la base de conocimiento al objeto de evitar posibles inconsistencias (como decíamos, por ejemplo, identificar un punto con una recta). Para llevar a cabo esta verificación semántica hacemos uso del razonador RACER [Haarslev et al. 01], el cual no sólo preserva la integridad sino que se encarga de derivar nuevos hechos en la base de conocimiento cuando es necesario. En la segunda etapa y en el caso de que la verificación de las acciones del alumno en la ontología sea positiva, BATMAN se encarga de traducir el código OpenMath a código *Mathematica*, con objeto de crear en el núcleo de dicho sistema los objetos definidos por el alumno.

Como hemos indicado, en RAMSYS se encuentran todos los procesos matemáticos encargados de la supervisión y seguimiento del alumno. Uno de los módulos relevantes del sistema lo constituye la base de conocimiento del mismo. La misma consta de dos partes bien diferenciadas:

- Ontología: La ontología utiliza la taxonomía de símbolos definidos en OpenMath obtenida del proyecto MONET [MONET] así como la ontología que hemos desarrollado para representar los problemas y sus componentes. La ontología de MONET está orientada a la descripción de servicios matemáticos en la Web. Sobre ella se instancian los objetos y las relaciones entre éstos que va definiendo el alumno. Por tanto, en cada instante de la resolución, la base de conocimiento dispone de todos elementos (objetos y relaciones) implicados en la misma.
- Razonador: RACER asegura la coherencia de los conceptos definidos y sus relaciones además de derivar nuevos hechos a partir de la información aportada por el alumno.

Finalmente, hemos de referirnos al sistema de cálculo simbólico *Mathematica*. Ya hemos indicado el uso que se hace del mismo en la generación de las partes aleatorias de los ejercicios que se proponen al alumno. También hemos descrito su uso para la ejecución de los posibles cálculos que requiera el alumno así como en la verificación de la corrección sintáctica de las fórmulas que introduzca éste. Por

ejemplo, para definir la ecuación de una recta que pasa por el punto

$$P = (2,3)$$

y con la dirección

$$\vec{v} = (1,-1)$$

el alumno debe efectuar los cálculos oportunos e introducir (haciendo uso de JOME) la ecuación de la recta resultante

$$x + y - 5 = 0$$

El sistema, con los mismos datos, se encarga de verificar que dicha ecuación es correcta, advirtiendo el error en caso contrario. Gracias a su sistema de comparación de patrones, *Mathematica* está capacitado para admitir como correctas tanto la expresión anterior como cualquier otra equivalente, como por ejemplo

$$x + y = 5 \quad o \quad y = -x + 5$$

Vistas las cuestiones relativas a los beneficios que proporciona el uso de información semántica integrada en una ontología y las diversas herramientas que la acompañan, vamos a detallar, en el siguiente epígrafe, algunas consideraciones relacionadas con el escenario de comunicación entre los distintos componentes que integran el sistema.

### 2.1.3. Comunicación entre componentes

La resolución de cualquier ejercicio, supuesto que este ya ha sido modelado por el profesor, está precedida de una fase de generación aleatoria del mismo que se efectúa de forma automática y transparente para el alumno. Adicionalmente hay una fase de interacción alumno-sistema cuando aquél inicia la resolución del problema.

En la primera fase interviene de forma determinante la estructura interna del problema, el cual está codificado como una clase más de la ontología y para cuya representación se ha hecho uso de la taxonomía desarrollada para el sistema *MathEdu* [Díez 02], con una nueva clase *Objetivo* incorporada a la misma. Cada problema definido en la ontología es susceptible de ser escogido para su resolución. Para generar un enunciado concreto se debe procesar el problema. Para ello se establece un enlace con *Mathematica*, llamando a la función que genera las fórmulas presentes en el enunciado. Dicha función manipula tres listas de objetos. La primera lista contiene las *fórmulas*. La segunda contiene las *metavariables* y la tercera contiene las *condiciones* para que el enunciado generado sea válido. Mientras no se satisfagan las condiciones el enunciado no se muestra al alumno. Una vez se han generado las fórmulas se construye el enunciado en HTML + MathML exportando las fórmulas de *Mathematica* en formato MathML. En este punto el enunciado está listo para presentarlo al alumno a través del servidor Web.

En la segunda fase, resolución del ejercicio, el sistema espera a que el alumno introduzca una expresión matemática a través del editor JOME. La expresión introducida se traduce a OpenMath para que sea interpretada en el módulo BATMAN. Para que se pueda realizar la interpretación BATMAN debe comunicar con RACER a través de la API Jena. En este paso ha sido necesaria la trascripción de OpenMath a *statements* de OWL para incorporar la nueva información a la ontología. La traducción de OpenMath a OWL está basada en la librería ROML, la cual ofrece clases para manejar objetos OpenMath con facilidad. La codificación de OWL utilizada es *N3-Triple*, basada en ternas con el formato (*sujeto*, *relación*, *predicado*), el objeto *sujeto* se relaciona por medio de la relación *relacion* con el objeto *predicado*. Aquí se intenta integrar la nueva información. Caso de que se encuentren inconsistencias se indica el error mediante una excepción. Para detectar las inconsistencias Jena incorpora un mecanismo por el cual, una vez introducidos los nuevos *statements*, se valida la ontología resultante que retorna un *ValidityReport*. Este informe contiene todas las inconsistencias detectadas. Si el informe está vacío quiere decir que no se han detectado inconsistencias y el sistema continúa con las verificaciones matemáticas. Para realizarlas se efectúa una nueva traducción en BATMAN. En este caso se traduce el objeto OpenMath a código Mathematica y se evalúa su contenido. Si es necesario realizar algún cálculo se hace en ese momento.

Finalmente, llevadas a cabo las etapas de verificación lógica (consistencia en la ontología) y matemática

(comprobación de que los objetivos matemáticos del problema se verifican), el sistema responde al alumno a través de un espacio de Log en la interfaz del sistema, figura 5, indicando tanto las acciones efectuadas como el resultado de las mismas.

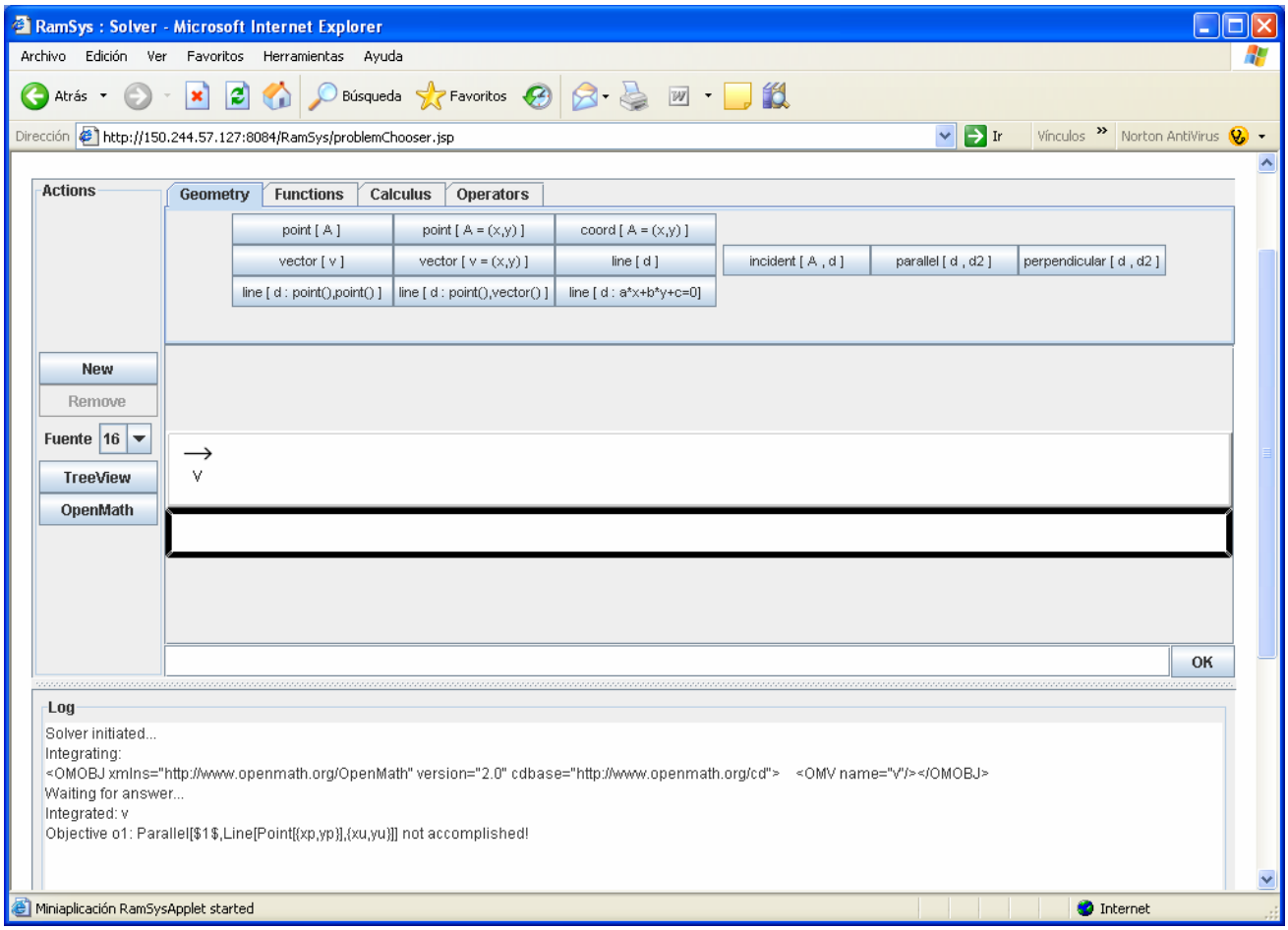


**Figura 5.** Interfaz de RamSys

## 3. Conclusión

Hemos presentado en el artículo tanto la motivación como los aspectos básicos y más novedosos de la arquitectura de *RamSys*, un sistema basado en ontologías para la resolución de problemas de Matemáticas. *RamSys* incorpora una base de conocimiento de conceptos matemáticos definida mediante una ontología; como parte de las distintas tecnologías empleadas en su diseño. La definición de conceptos matemáticos constituye una de las cuestiones clave a la hora de desarrollar sistemas que sean capaces de tratar con la resolución interactiva de problemas, sea esta guiada o libre. Debido a dicha dificultad la tendencia hasta el presente ha sido la de orientar la creación de IES hacia aplicaciones que fomentan el aprendizaje puramente algorítmico, dejando de lado el aprendizaje deductivo basado en conceptos. Creemos pues necesario avanzar en la creación de aplicaciones que permitan al alumno construir expresiones para mostrar sus conocimientos sin dejar de lado la supervisión conceptual de su actividad. La arquitectura descrita pone de manifiesto dos cuestiones a nuestro juicio fundamentales:

- La complejidad intrínseca de un sistema distribuido de las características de *RamSys* con capacidad de interpretar secuencias de acciones no predeterminadas.

- La necesidad de tender hacia el uso de estándares que posibiliten la comunicación entre procesos y, por ende, faciliten la reusabilidad de las componentes de software empleadas [Mizoguchi et al. 97].

En nuestro caso estamos haciendo un uso intensivo de componentes desarrolladas en centros o por instituciones de reconocido nivel como son SMI de Stanford, RIACA, en la Universidad Técnica de Eindhoven, o el consorcio MONET, que agrupa a numerosos investigadores de universidades y empresas tanto europeas como americanas. Esta circunstancia nos ha facilitado el proceso de diseño y desarrollo del prototipo. Además nos garantiza la fiabilidad de los componentes utilizados pues han sido contrastados previamente.

Actualmente hemos constatado que es necesario ampliar las clases y relaciones descritas por la ontología con otras nuevas que se adapten a los requisitos del contexto en que estamos trabajando. Además, hemos comprobado que sería necesario hacer una carga parcial de las ontologías que describen objetos matemáticos, dado que actualmente las cargamos en su totalidad, lo cual carga en exceso al razonador y ralentiza el proceso de comprobación de la integridad de la base. Por tanto sería necesario establecer algún mecanismo de filtrado. Sin embargo esto supone una dificultad no sólo por el hecho del filtrado y la carga selectiva de partes de la ontología, sino porque el alumno, en su actividad, puede tratar de hacer uso de conceptos o relaciones que no se haya previsto usar para la resolución de un determinado ejercicio. Por tanto habría que ir, tal vez, hacia un modelo de carga dinámica de las definiciones de clases en las ontologías.

Finalmente queremos comentar también algo al respecto de los aspectos puramente didácticos de la herramienta. Actualmente, salvo la verificación por parte del sistema de la consecución de objetivos por parte del alumno, no nos hemos planteado la introducción de ningún tipo de ayuda didáctica. La actividad del alumno con el sistema es enteramente libre y no guiada. No se le puede ofrecer ningún tipo de ayuda o indicación en el caso de que este la necesite si no sabe cómo actuar. Sin embargo pensamos que, por el tipo de información de los problemas y del dominio que hemos representado, es posible extraer información, aunque sea muy básica, de, por ejemplo, el tipo de objetos que debe utilizar para efectuar la construcción de una demostración. Este sería, tal vez, un primer nivel de ayuda. Por el momento no hemos contemplado la posibilidad de otras características didácticas de la herramienta. Tanto para estas cuestiones como para otras que puedan surgir está abierta la posibilidad a cualquier tipo de colaboración.

## Agradecimientos

Este trabajo fue subvencionado por la Comisión Interministerial de Ciencia y Tecnología (CICYT), proyecto número TIC2002-01948; así como por el programa de Becas de Colaboración del Ministerio de Educación y Ciencia ECI/2298/2004.

## Referencias

[Baader 03] Baader, F, The description logic handbook. Theory, implementation and applications, 2003.

[Breda et al. 03] Breda, A, Neto, MT, Costa N, "The role of non-euclidean geometry in the opening out of deductive reasoning", International Conference on Multimedia and Information & Communication Technologies in Education, 2003, pp. 1127-1131.

[Cabri] Cabri-geometry, accesible en la dirección http://www.cabri.net/cabri2/accueil-e.php en enero de 2006.

[Cinderella] Cinderella, accesible en la dirección http://www.cinderella.de/tiki-index.php en enero de 2006.

[Díez 02] Díez, F, "Diseño y resolución interactiva de ejercicios que involucren Cálculo Simbólico", Tesis doctoral, 2002.

[Díez et al. 04] Díez, F, Moriyón, R, "Design and interactive resolution of exercises of Mathematics that involve symbolic computations", Computing in Science & Engineering, vol. 6, no. 1., 2004, pp. 81-84.

[Dirat et al. 99] Dirat, L, Buffa, M, Fedou, JM, Sander, P, "JOME: OpenMath on the Web",

Fourth International Conference on Technology in Mathematics Teaching, 1999.

[Geometer] Geometer's Sketchpad, accesible en la dirección http://www.keypress.com/sketchpad/ en enero de 2006.

[Gómez-Pérez et al. 04] Gómez-Pérez, A, Fernández-López, M, Corcho, O, Ontological Engineering. London, 2004.

[Jena] JENA: a Semantic Web Framework for Java, accesible en la dirección http://jena.sourceforge.net/ en enero de 2006.

[Haarslev et al. 01] Haarslev, V, Möller, R, "RACER System Description", International Joint Conference on Automated Reasoning, 2001.

[Mizoguchi et al. 97] Mizoguchi, R, Ikeda, M, Sinitsa, K, "Roles of Shared Ontology in AI-ED Research", Artificial Intelligence in Education, 1997. pp. 538.

[MONET] The MONET Consortium, accesible en la dirección http://monet.nag.co.uk en enero de 2006.

[Paulos 00] J. A. Paulos. El hombre anumérico. El analfabetismo matemático y sus consecuencias. Tusquets Ed. Barcelona, 2000.

[Protégé] Protégé: an ontology editor and knowledge-base framework, accessible en http://protege.stanford.edu/index.html en enero de 2006.

[Quinney 02] Quinney, D. "calculus machina: an intelligent tutor providing computer based support for teaching undergraduate calculus", 2nd International Conference on the Teaching of Mathematics, 2002.

[Riem 04] Riem, MN, "The OpenMath Guide. A practical guide on using OpenMath.", 2004, http://www.riaca.win.tue.nl/products/openmath/guide/omguide.pdf

[Wolfram 99] Wolfram, S, The Mathematica Book, 4th ed., 1999.

[Xambó et al. 02] Xambó, S, Eixarch, R, Marquès, D, "WIRIS: An internet platform for the teaching and learning of mathematics in large educational communities". Contributions to Science, vol. 2, no. 2, 2002, pp. 269-276.