Sistema inteligente para la recomendación de objetos de aprendizaje

Ana Casali, Valeria Gerling, Claudia Deco y Cristina Bender

Fecha de recepción: 31 de Mayo de 2010
Fecha de aprobación: 23 de Septiembre de 2010

LACLO 2010 Best Papers

RESUMEN

En este trabajo se describe el desarrollo de un sistema recomendador de objetos de aprendizaje. Este sistema ayuda a un usuario a encontrar los recursos educativos que le sean más apropiados de acuerdo a sus necesidades y preferencias. La búsqueda se realiza en diferentes repositorios de objetos de aprendizaje, donde cada objeto tiene metadatos descriptivos. Se utilizan estos metadatos para recuperar aquellos objetos que satisfagan no sólo el tema de la consulta, sino también el perfil de usuario, teniendo en cuenta sus características y preferencias.

El sistema tiene una arquitectura multiagente que incluye varios tipos de agentes con diferentes funcionalidades. En particular, en este trabajo se modela el Agente Recomendador (Agente-R), como un agente BDI graduado. Este agente se encarga de realizar una recuperación flexible y presentar una lista ordenada con los mejores recursos de acuerdo con el perfil de usuario. Se ha implementando un prototipo y se presenta un caso de uso.

Palabras Clave—: Sistemas recomendadores, sistemas multiagente, objetos de aprendizaje, metadatos, perfil de usuario.

ABSTRACT

This paper describes the development of a recommender system of learning objects. This system helps a user to find educational resources that are most appropriate to his/her needs and preferences. The search is performed in different repositories of learning objects, where each object has descriptive metadata. This metadata is used to retrieve objects that satisfy not only the subject of the query, but also the user profile, taking into account his/her characteristics and preferences.

A multi-agent architecture that includes several types of agents with different functionalities is used. In particular, this paper models the Recommender Agent (Agent-R) as a graded BDI agent. This agent is responsible for making a flexible retrieval and it gives as result an ordered list with the best resources according to the user profile. A prototype was implemented and an use case is presented.

Keywords—: Recommender systems, multiagent systems, learning objects, metadata, user profile.

I. INTRODUCCIÓN

En el dominio de la educación existe gran cantidad y diversidad de material que puede contribuir al proceso enseñanza-aprendizaje. Un objeto de aprendizaje (OA) es "cualquier recurso digital que puede ser utilizado repetidamente para facilitar el aprendizaje" [1]. Los OAs pueden ser utilizados por un alumno que desea aprender un tema, o pueden ser utilizados por un docente que desea preparar material didáctico para su clase. Estos usuarios pueden obtener OAs por medio de la realización de consultas en repositorios accesibles vía web. Estos repositorios de OAs contienen abundante material útil e interesante sobre cada tema. Todo usuario que realiza una misma búsqueda temática, obtiene como resultado la misma lista de OAs.

Los usuarios en general revisan sólo los primeros resultados. En muchos casos estos primeros resultados no son los adecuados si la búsqueda se realiza sólo considerando las palabras claves de la temática de interés. Esto se debe a que los usuarios poseen distintas características personales y preferencias, que deberían también ser consideradas en el momento de la búsqueda. Para ayudar a resolver este tipo de

---

Ana Casali: acasali@fceia.unr.edu.ar. Doctora en Informática de la Universidad de Girona, España. Directora del Departamento de Ciencias de Computación y de la carrera Licenciatura en Ciencias de la Computación de la Facultad de Ciencias Exactas, Ingeniería y Agrimensura, Universidad Nacional de Rosario, Argentina. Investigadora del Centro Internacional Franco Argentino de Ciencias de la Información y de Sistemas CIFASIS, Rosario, Argentina.

Valeria Gerling: valeria.gerling@gmail.com. Licenciada en Ciencias de la Computación de la Universidad Nacional de Rosario, Argentina. Integrante de proyectos de investigación de la Facultad de Ciencias Exactas, Ingeniería y Agrimensura, Universidad Nacional de Rosario, Argentina.

Claudia Deco: deco.clau@gmail.com. Doctora en Ingeniería, Universidad Nacional de Rosario. Magister en Informática de la Universidad de la República, Uruguay. Investigadora del Departamento de Investigación Institucional y Profesora Pro-Titular en la Facultad de Química e Ingeniería Rosario, Universidad Católica Argentina.

Cristina Bender: cbender@uca.edu.ar. Magister en Informática de la Universidad de la República, Uruguay. Investigadora del Departamento de Investigación Institucional y Profesora Pro-Titular en la Facultad de Química e Ingeniería Rosario, Universidad Católica Argentina.

problema surgen los sistemas recomendadores, que son capaces de seleccionar, de forma automática y personalizada, el material que mejor se adecúe a las preferencias o necesidades de un usuario. El enfoque es lograr la personalización de los resultados de una búsqueda, utilizando las características y preferencias del usuario, modeladas en perfiles personales, y metadatos con las descripciones semánticas de cada objeto. La base para el razonamiento del sistema recomendador es considerar los metadatos de cada recurso educativo y la importancia relativa de cada una de las preferencias del usuario al momento de considerar la elección de un material.

En este trabajo se presenta la arquitectura e implementación de un prototipo de un sistema recomendador cuyo objetivo es devolver una lista ordenada de los objetos de aprendizaje más adecuados de acuerdo a un perfil de usuario. La búsqueda se realiza en repositorios, o en federaciones de repositorios, accesibles vía web y con metadatos descriptivos de estos objetos que incluyen características educacionales. Para la experimentación del prototipo se utiliza el repositorio Ariadne[12], por ser uno de los que contiene gran cantidad de material en diferentes idiomas y tiene información referida a los metadatos educacionales.

El resto del trabajo se organiza de la forma siguiente: en primer lugar se describen los sistemas recomendadores y sistemas multiagentes. Luego se presenta la arquitectura del sistema multiagente propuesto en el cual se modela como un agente BDI graduado a uno de los agentes principales del sistema. A continuación se presentan detalles de la implementación y se analiza un caso de uso para evaluar el funcionamiento de la arquitectura propuesta. Finalmente se presentan las conclusiones.

II. SISTEMAS RECOMENDADORES

Un sistema recomendador le brinda a un usuario aquellos resultados de una búsqueda de información cercanos a sus necesidades [2]. Entre las aplicaciones potenciales de los sistemas recomendadores, el dominio de la educación parece ser un buen candidato ya que las ofertas de recursos educativos están en constante crecimiento.

Una de las posibilidades para el diseño de sistemas recomendadores es utilizar arquitecturas de agentes. Los agentes son entidades de software que poseen la suficiente autonomía e inteligencia como para poder encargarse de tareas específicas con poca o ninguna supervisión humana. El objetivo de estos agentes es explorar y filtrar las mejores opciones a partir de un perfil de usuario (preferencias, características, etc.) considerando un importante número de posibilidades diferentes. Esto involucra la construcción de un modelo o perfil de usuario el cual puede ser obtenido de forma implícita o explícita. Una taxonomía detallada de sistemas recomendadores puede verse en [3] y las principales técnicas para su desarrollo pueden agruparse en: sistemas de filtrado colaborativo, filtrado basado en contenidos, filtrado basado en conocimiento y sistemas híbridos.

En los últimos años, se ha incrementado el diseño e implementación de sistemas multiagentes para abordar el desarrollo de sistemas distribuidos complejos y, en particular, se han utilizado para el desarrollo de sistemas recomendadores. Esta tecnología de agentes es importante a la hora de modelar diferentes características que se espera de estos sistemas como por ejemplo: generar y considerar el perfil del usuario, inferir y agregar información proveniente de fuentes heterogéneas y distribuidas, obtener sistemas escalables, abiertos y seguros, y realizar la tarea requiriendo la menor intervención de las personas.

Una de las arquitecturas de agentes más notorias es el modelo de agente BDI (*Belief-Desire-Intention*) propuesto por Rao y Georgeff [4]. Este modelo está basado en la representación explícita de las creencias (*B*) del agente que representan el estado del entorno, sus deseos (*D*) que representan sus motivaciones, y las intenciones (*I*) del agente que modelizan sus metas u objetivos. Esta arquitectura ha evolucionado en el tiempo y ha sido utilizada en importantes aplicaciones de sistemas multiagentes. Con el propósito de hacer que la arquitectura BDI sea más flexible, Casali et al. [5] han propuesto un modelo general para diseñar agentes BDI graduados (*g-BDI*). Este modelo permite especificar arquitecturas capaces de tratar con la incertidumbre del entorno y con actitudes mentales graduadas con el fin de desarrollar agentes que puedan tener una mejor performance en entornos dinámicos e inciertos. En el modelo g-BDI los grados en las creencias van a representar en qué medida el agente cree que una fórmula es cierta. Los grados, positivos o negativos, en los deseos permiten al agente establecer respectivamente, diferentes niveles de preferencia o de rechazo. Las graduaciones en las intenciones también estarán dando una medida de preferencia, pero en este caso, modelarán la relación costo-beneficio que le significa al agente alcanzar esa meta. A partir de la representación de estas tres actitudes y según cómo interactúen unas con otras, se pueden modelar distintos tipos de agentes que tendrán diferentes comportamientos.

Según Zaiane [6] los sistemas recomendadores han surgido con el comercio electrónico y propone usarlos en el dominio de la educación. Siguiendo esta propuesta, Romero et al. [7] plantea el uso de técnicas de minería de datos para recomendar la navegación entre links, y Soonthornphisaj et al. [8] propone un sistema que integra el material recomendado antes de dárselo al usuario. Zhu et al. [9] propone una arquitectura de un sistema de recomendación personalizado en educación, la cual adopta una tecnología multiagente y consiste de seis agentes de software, los cuales coordinan el trabajo de forma jerárquica entre ellos para ofrecer una gama de funciones primarias.

En este trabajo, se propone un modelo basado en el filtrado de contenidos. Se presenta una arquitectura multiagente para la

[12] http://www.ariadne-eu.org/

recuperación de recursos educativos a partir de una federación de repositorios, con el fin de asistir a un estudiante o a un docente a elegir objetos educativos acordes a sus características personales y preferencias. Esta plataforma multiagente incluye varios tipos de agentes en concordancia con las distintas funcionalidades del sistema, y se ha modelado a uno de los principales componentes del sistema (el Agente-R), como un agente g-BDI, el cual se encarga de la recuperación de los mejores objetos de aprendizaje acordes al perfil del usuario. También, se presenta la implementación de un prototipo preliminar de este sistema y se muestra que el sistema propuesto es viable y puede ser de ayuda, a través del análisis de un caso de estudio.

III. ARQUITECTURA DEL SISTEMA RECOMENDADOR

Para diseñar este sistema se ha elegido una arquitectura multiagente para poder manejar información heterogénea y distribuida (como la que se encuentra en los distintos repositorios), y tener un alto grado de modularidad y autonomía.

Además estas arquitecturas son altamente escalables, abiertas y pueden realizar una tarea requiriendo la menor intervención de las personas. La arquitectura propuesta consta de los siguientes agentes: el Agente Interfaz (Agente-I), que se encarga de capturar los datos ingresados por el usuario; el Agente Refinador Semántico (Agente-RS) cuyo objetivo es producir la estrategia de búsqueda asociada al interés del usuario; el Agente Perfil de Usuario (Agente-PU) tiene la tarea de construir el perfil de usuario; los Agentes Buscadores (Agente-Bi) se encargan de encontrar los OAs que satisfacen la temática y las restricciones de interés del usuario en los distintos repositorios; el Agente Mediador (Agente-M) integra lo encontrado por cada Agente-Bi y soluciona conflictos, y por último el Agente Recomendador (Agente-R) cuyo objetivo es seleccionar los mejores objetos de acuerdo al perfil del usuario. La arquitectura se muestra en la Fig. 1 y a continuación se detalla la funcionalidad de los agentes que la componen.

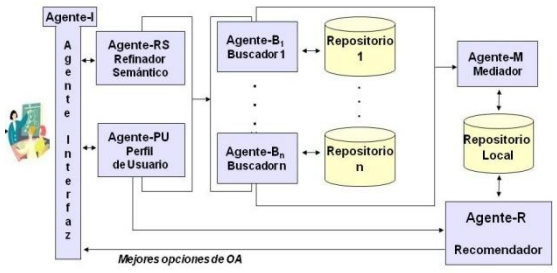

Fig. 1. Arquitectura del sistema multiagente

El agente *Interfaz* (Agente-I) interactúa con el usuario a través de una interfaz gráfica. Captura los datos ingresados por el usuario, y le despliega el resultado de su búsqueda. Este agente le provee al Agente-PU las preferencias y restricciones del usuario y al Agente-RS la temática sobre la búsqueda. Luego recibe del Agente-R la recomendación (ranking de OAs) que dará al usuario.

El agente *Refinador Semántico* (Agente-RS) produce la estrategia de búsqueda temática. Cuando el usuario hace una consulta, a través del agente Interfaz, ingresa un conjunto de términos que describen el tema de su interés. El Agente-RS desambigua estos términos y los expande semánticamente incorporando sinónimos y conceptos relacionados. La salida de este agente es una estrategia de búsqueda que es enviada a cada agente buscador.

El agente *Perfil de Usuario* (Agente-PU) recibe del Agente Interfaz los datos del usuario así como sus preferencias y restricciones, con el objetivo de construir su perfil. Este agente provee a los agentes buscadores (Agente-B_i) algunas restricciones (que llamaremos primarias) que puedan colaborar en el filtrado de los OAs; por ejemplo, si el material tiene costo o no lo tiene. Además, este agente le provee al agente recomendador las preferencias y restricciones restantes del usuario, para que los objetos sean adecuadamente ordenados.
Cada agente *Buscador* (Agente-B_i) busca en un repositorio los OAs que satisfagan la estrategia de búsqueda del usuario y, eventualmente, alguna de las restricciones primarias. Para poder realizar la búsqueda primero adapta la estrategia de búsqueda a la sintaxis del repositorio al que accede, luego se comunica con el repositorio a través de un protocolo de comunicación para realizar la consulta correspondiente y de esta manera obtener una respuesta. Cuando recibe esta respuesta, transforma el resultado de la búsqueda a un formato común (por ejemplo, al lenguaje XML), y se lo provee al agente mediador el cual lo deposita en un repositorio local. En esta arquitectura se tienen tantos agentes buscadores como repositorios haya, dado que la manera de consultar varía de acuerdo al protocolo necesario para comunicarse con el repositorio correspondiente.

El agente *Mediador* (Agente-M) integra la información encontrada por cada uno de los agentes buscadores, y soluciona conflictos de manera que los datos sean consistentes. Por ejemplo, si este agente recibe de dos agentes buscadores información que describen un mismo objeto, complementa los metadatos que lo describen en el repositorio para obtener una descripción más completa del objeto. Por último, este agente deposita la información que generó en el repositorio local, para que el Agente-R pueda trabajar con estos datos.

El agente *Recomendador* (Agente-R) es el que finalmente recomienda al usuario los OAs que satisfacen adecuadamente el perfil del mismo. La salida es una lista ordenada de objetos, donde el primero es el que más se adecúa al deseo del usuario. Este agente tiene en cuenta las preferencias y restricciones que fueron provistas por el Agente-PU y utiliza un conjunto de reglas y funciones para determinar cuáles de los OAs son los que debe recomendar. En la próxima sección se describe el Agente-R, que se ha diseñado utilizando el modelo BDI graduado.

A. Diseño del Agente Recomendador (Agente-R)

El Agente Recomendador se ha especificado tomando como

base al modelo *g-BDI* [5], el cual se fundamenta en un sistema multi-contextos compuesto por contextos y reglas que representan sus interrelaciones. En el modelo de agente *g-BDI* se tienen contextos para representar sus creencias (contexto *Belief*), sus deseos (contexto *Desire*) y sus intenciones (contexto *Intention*), y se elige un modelo adecuado para razonar sobre los grados en cada uno de estos contextos. Los detalles de esta arquitectura pueden verse en [10]. A continuación se presentan las características de estos contextos en el Agente-R.

Contexto Belief (BC): Este contexto modela la información del Agente-R sobre el entorno educativo, esto es, modela las características que presentan los OAs. Estas características están descriptas a través de metadatos definidos de acuerdo al estándar LOM y escritos en XML. Del conjunto de metadatos de LOM, se ha seleccionado un subconjunto de aquellos que describen las características de los OAs que consideramos más relevantes para el recomendador, por ejemplo, el idioma, el grado de interacción del usuario con el objeto, el contexto académico, etc.

Con estos datos del objeto, el agente aplicará las reglas para inferir su grado de creencia b_{ij} en que un objeto O_i satisfaga las distintas preferencias p_j del usuario, representado por la fórmula $B(O_i; p_j; b_{ij})$. Se han establecido grupos de reglas para los distintos tipos de preferencias utilizando distintas distancias definidas especialmente para cada tipo de preferencia. El detalle de los conceptos utilizados y sus definiciones puede verse en [10]. Algunas de las reglas utilizadas, según cada metadato, son las siguientes:

- **Metadato *Interacción***
  (R1) Si InteractivityLevel(O_i)= 'low' Entonces B(O_i,interacción=baja, 1)
  Esta regla representa: "si el grado de interactividad (representado por el metadato InteractivityLevel) del O_i es bajo (*low*), entonces la creencia del agente respecto a que ese objeto va a satisfacer la preferencia del usuario *interacción=baja* es máxima (su valor es 1)".
  (R2) Si InteractivityLevel(O_i)= 'high' Entonces B(O_i,interacción=baja, 0.2)

- **Metadato *Rol***
  (R3) Si IntendedEndUserRole(O_i)=[learner,. . .] Entonces B(O_i,rol=alumno, 1)
  (R4) Si IntendedEndUserRole(O_i)=[a,b,c,teacher,. . .] and a,b,c ∈ {learner, author, manager}
  Entonces B(O_i,rol=docente, 0.4)

- **Metadatos *Contexto académico* y *Nivel de conocimiento***
  (R5) Si Context(O_i)= 'higher education' and Dificulty(O_i)= 'dificult'
  Entonces
  B(O_i,(contexto=terciario/universitario) and (nivel=inicial), 0.6)
  (R6) Si Context(O_i)= 'school' and Difficulty(O_i)= 'difficult'
  Entonces B(O_i,(contexto=terciario/universitario) and (nivel=inicial), 0.8)

- **Metadato *Estilo de aprendizaje***
  (R7) Si LearningResourceType(O_i)=[exercise,lecture] Entonces B(O_i,estilo=práctico, 0.6)
  (R8) Si LearningResourceType(O_i)=[exercise,narrative text,slide] Entonces B(O_i,estilo=mixto, 0.5)

En el caso del metadato *Idioma*, si el usuario elige uno o más idiomas y al menos uno de ellos pertenece al idioma del OA, entonces *B* tendrá valor 1 y el valor de prioridad utilizado para calcular el grado de intención será igual al máximo de prioridades de cada uno de los idiomas que fue elegido.

Contexto Desire (DC): El deseo global del Agente-R es encontrar el OA que mejor satisfaga al usuario teniendo en cuenta la temática, las restricciones y las preferencias. En este contexto se representan las preferencias (por ejemplo, el idioma o el contexto académico) y las restricciones (por ejemplo, su duración máxima) que el usuario tiene respecto a los OAs. Tanto las preferencias como las restricciones pueden ser graduadas (con valores en [0; 1]) expresando lo que el usuario desea o rechaza en diferentes grados de un recurso educativo.

La Tabla 1 muestra un ejemplo del perfil de un usuario, donde se observa que tiene como preferencia que el idioma del OA sea español con prioridad 1, inglés con prioridad 0.8 y francés con un menor grado de preferencia (0.6). Además, el usuario desea, con grado 0.8, que la interacción con el OA sea baja. En cuanto a restricciones, indica con prioridad 0.8 que no desea objetos cuya duración supere los 60 minutos. Formalmente esto se representa: D^+(idioma, español, 1), D^+(idioma, inglés, 0.8), D^+(idioma, francés, 0.6), D^+(interacción, baja, 0.8), D^-(duración máxima, 60, 0.8).

TABLA 1. Extracto del perfil de un usuario.

Preferencia	Prioridad	Descripción
Lengua Materna="español"	1	Idiomas que el usuario desea que presenten los documentos.
Idioma="español"	.8	
Idioma="inglés"	.6	
Idioma="francés"		
Interacción="baja"	.8	Grado de interacción que el usuario desea tener con el objeto.
Restricción		**Descripción**
Duración Máxima="60"	.8	Tiempo máximo que el usuario espera que le tome trabajar con el objeto.

Contexto Intention (IC): Para esta aplicación, las intenciones serán los objetivos educativos que se intentarán alcanzar a través del mejor objeto (u objetos) seleccionado. En este contexto, las intenciones dependerán de las restricciones del usuario respecto a los OAs, sus preferencias (que se espera que se traduzcan en beneficio para el usuario al aprender a través de un determinado OA) y también de la satisfacción esperada de las preferencias a través de un recurso educativo que cuenta con ciertas características representadas en sus metadatos. Por ejemplo, en qué medida un OA cuyo tipo de recurso está

catalogado en su metadato como [ejercicio, lectura] satisface a la preferencia de que el recurso sea "práctico". Estas variables son combinadas a través de una regla adecuada que determina el grado de intención de cada OA para satisfacer el conjunto de preferencias del usuario. Este grado de intención se utiliza para ordenar los OAs en la recomendación.

Luego, el agente Recomendador calcula las intenciones de alcanzar sus objetivos. Esto es satisfacer un conjunto P de n preferencias (p_j) de aprendizaje, a través de un recurso O_i considerando distintos factores que provienen de distintos contextos. Uno de los factores es la prioridad dada (d_j) por el usuario a cada preferencia (p_j, d_j), por ejemplo (*interacción=baja*, 0.8). Otro factor es el grado de satisfacción de cada preferencia p_j por las características de un recurso $O_i(b_{ij})$, representado por la fórmula $B(O_i, p_j, b_{ij})$, por ejemplo $B(O_i$,*estilo=práctico*, 0.6) representa que la satisfacción esperada del objeto O_i respecto a la preferencia del usuario de que el recurso sea de estilo práctico es de 0.6. También se podrían incluir factores tales como el costo o la confianza en la fuente proveedora del recurso (Universidad, Institución, Autor, etc.). Se pueden utilizar distintas funciones para computar este grado de intención asociado a cada OA, modelando distintos comportamientos del Agente-R. Por ejemplo, este cálculo puede plantearse como el promedio de las satisfacciones esperadas de las distintas preferencias:

$$I(O_i, P) = \frac{\sum_{j=1}^{n} d_j \times b_{ij}}{n}$$

IV. IMPLEMENTACIÓN DEL PROTOTIPO DEL SISTEMA RECOMENDADOR

El prototipo se implementó utilizando el lenguaje SWI-Prolog[13] y el entorno de desarrollo de aplicaciones web de código abierto Ruby on Rails[14] escrito en el lenguaje de programación Ruby. El lenguaje SWI-Prolog se utiliza para implementar el Agente Recomendador. SWI-Prolog es un lenguaje de programación que soporta la deducción lógica y es una implementación de código abierto del lenguaje Prolog. Cuenta con un conjunto rico de características tales como soporte multihilo, portabilidad a muchas plataformas y abundante documentación.

El lenguaje Ruby se utilizó para desarrollar el Agente Interfaz dado que es un lenguaje dinámico y de código abierto enfocado en la simplicidad y productividad. También se utilizó para desarrollar el Agente Buscador ya que soporta consumir servicios web con SOAP, lo cual es necesario para comunicarse con los repositorios donde el protocolo es SQI.

Mediante una interfaz gráfica el usuario puede buscar aquellos objetos que satisfagan su perfil. La interfaz también mantiene su historial de búsquedas pasadas, almacenando la temática elegida y sus preferencias (idiomas, rol, interacción, etc.). Esto se puede apreciar en la Fig. 2 donde se muestra la ventana que permite al usuario ingresar sus preferencias de búsqueda y en la Fig. 3 donde se muestra el despliegue del historial.Toda persona que desee utilizar el sistema debe crear un usuario de búsqueda, el cual será su identificador dentro del sistema recomendador. Para poder administrar la base de datos, la cual contiene los registros de las búsquedas y de los usuarios, se necesita un usuario con perfil de administrador. Este perfil, a través de la interfaz, puede ver el historial de búsqueda de todos los usuarios, y también sus datos personales. Además, puede observar qué objetos han sido recomendados por el sistema, pudiendo analizar el valor de sus metadatos. Con el propósito de poder evaluar el desempeño del recomendador, el administrador puede acceder a distintas estadísticas obtenidas a partir de las búsquedas realizadas.

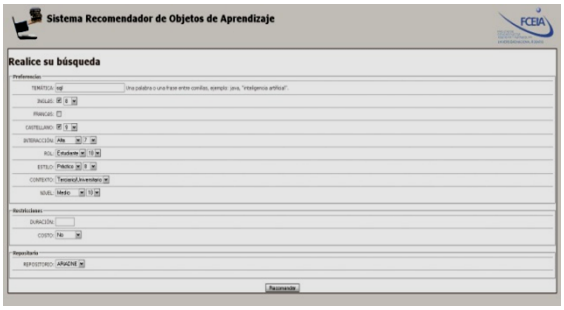

FIG. 2. Formulario de búsqueda de la Interfaz.

El Agente Interfaz guarda en un archivo los datos referentes a la búsqueda del usuario, es decir, la lengua materna, la temática buscada, y por cada una de las preferencias y restricciones, escribe los valores que eligió el usuario junto con las prioridades deseadas. Por medio de este archivo el Agente Interfaz le comunica estos datos al Agente Recomendador, y le provee al Agente Buscador la temática de interés para que éste busque los objetos en uno o más repositorios.

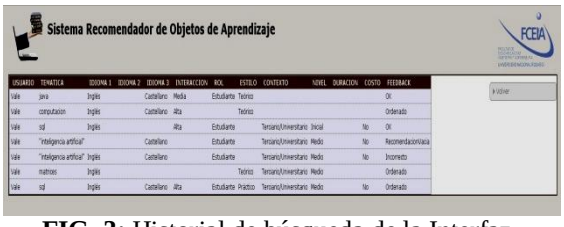

FIG. 3: Historial de búsqueda de la Interfaz.

El Agente Recomendador le provee al Agente Interfaz la lista de objetos que satisfacen el perfil del usuario. Para esto, genera un archivo con el resultado que usa el Agente Interfaz para mostrar en pantalla los 10 mejores. Para cada objeto encontrado, se muestra su identificación (ID), y en caso de que las posea, las siguientes características: título, descripción y dónde localizarlo. La Fig. 4 ilustra el resultado de una recomendación junto con el detalle de uno de los objetos recomendado por el sistema.

[13] http://www.swi-prolog.org/
[14] http://rubyonrails.org/

FIG. 4: Resultado de la recomendación y detalle del décimo objeto.

Una vez que el sistema le recomienda al usuario una lista ordenada de objetos, éste puede expresar su opinión a través de la interfaz. En la Fig. 5 se observa una ampliación del ángulo superior derecho de la Fig. 4, donde se muestran tres botones con las opciones que puede elegir el usuario:

- *Resultado Correcto:* Cuando la lista de objetos recomendada satisface al usuario.
- *Ordenar de otra manera:* Cuando el usuario está conforme con los objetos pero el orden no coincide con su preferencia. El usuario puede indicar cuáles son los cinco primeros mejores y en caso de que desee puede escribir alguna sugerencia a través de la interfaz, como se resalta en la Fig. 6.
- *Resultado Incorrecto:* Cuando el usuario no está satisfecho con la lista de objetos recomendada. En este caso el usuario puede escribir su opinión a través de la interfaz.

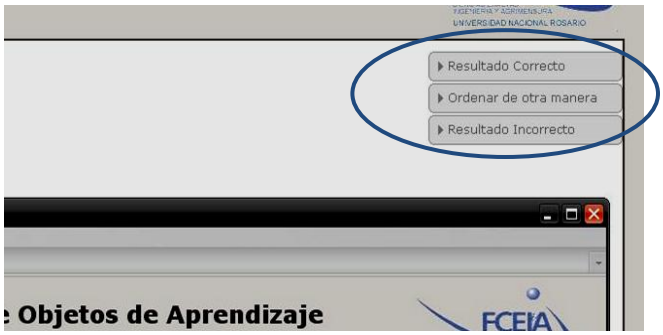

FIG. 5: Opciones para la evaluación de la lista de objetos recomendados.

La lista de objetos se prepara con los resultados provistos por los Agentes Buscadores. Cada uno de éstos envía una consulta a un repositorio (o federación de repositorios) y espera una repuesta. En este primer prototipo los repositorios accedidos son Ariadne y FLOR[15].

El Agente Buscador se comunica con éstos a través del protocolo SQI y el lenguaje de consulta VSQL. Como repuesta recibe un archivo XML que contiene los metadatos, según el estándar LOM, de cada uno de los objetos que satisfacen la temática. Cada Agente Buscador crea un archivo XML por cada uno de los objetos, y los almacena en el repositorio local. Además, genera un archivo con la cantidad de objetos y el nombre de cada archivo XML de cada objeto. Este último archivo es utilizado por el Agente Recomendador para generar la lista ordenada que el Agente Interfaz presenta al usuario.

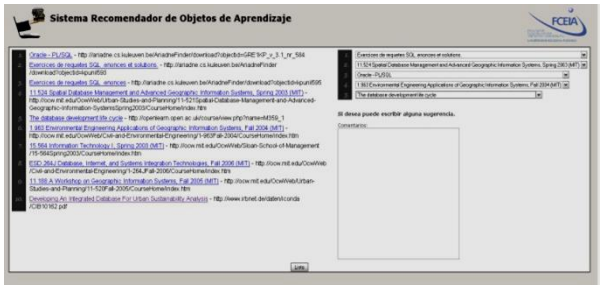

FIG. 6: Interfaz. Posibilidad de cambio en el orden.

V.CASO DE USO

Para probar el recomendador se utiliza el repositorio Ariadne, dado que tiene cargado un alto porcentaje de los metadatos descriptivos de los objetos y además, cuenta con información de los metadatos de la categoría Educacional (Educational) del estándar LOM. La existencia de estos metadatos le permite al recomendador, basándose en las preferencias del usuario, construir la lista ordenada de los OAs que satisfacen al perfil del usuario.

Las pruebas del prototipo fueron realizadas por alumnos y docentes de la carrera Licenciatura en Ciencias de la Computación de la Universidad Nacional de Rosario. Por este motivo, se decidió realizar una consulta sobre el tema *lenguaje Java*, dado que es conocido para estos usuarios y además el repositorio Ariadne cuenta con material de este tema de acceso posible y en idiomas conocidos por la mayoría de estos usuarios.

La búsqueda a través de la interfaz del tema *Java*, filtró 27 de los 263 objetos, dejando sólo aquellos resultados cuyo idioma es inglés, francés o español y que además están disponibles y son de acceso libre. En la Tabla 2 se presentan, a modo de ejemplo, los metadatos más relevantes de algunos de los objetos recuperados.

Para probar el recomendador se contó con la ayuda de usuarios con diferentes perfiles y se consideró el feedback otorgado por cada uno para establecer si el Agente Recomendador satisface, en algún grado, estos perfiles. En la Tabla 3 se muestran los distintos perfiles de usuario de acuerdo a la información extraída por el Agente Interfaz. El Recomendador entrega en cada caso una lista de objetos ordenados de acuerdo a las preferencias del usuario. Por ejemplo, para el Usuario2 el sistema le recomienda la siguiente lista ordenada de OAs:

{Id12, Id11, Id13, Id7, Id6, Id8, Id4, Id15, Id26, Id2}

En la Tabla 2 se muestran las características correspondientes a

[15] http://ariadne.cti.espol.edu.ec/lacloFinder/

estos diez OAs, ordenados por identificación del objeto. A partir de esta salida del sistema, los usuarios ingresan su respuesta, según la pantalla mostrada en la Fig. 5. El Usuario2 está conforme con los objetos pero el orden no coincide con su preferencia y sugiere *Ordenar de otra manera*, indicando que a su entender los primeros tres objetos deberían ser: {Id7, Id6, Id8}.

Tabla 2. Metadatos relevantes de algunos de los objetos recuperados.

Tabla 2. Metadatos relevantes de algunos de los objetos recuperados.

OA	Language	Learning Resource Type	Interactivity level	Intended End-User Role	Context	Difficulty	Typical Learning Time (min)
Id2	fr	narrative text	very low	learner	higher education	difficult	10
Id4	fr	narrative text	unknown	learner	higher education	unknown	15
Id6	fr	slide	unknown	learner	higher education	unknown	90
Id7	fr	slide	unknown	learner	higher education	unknown	135
Id8	fr	slide	unknown	learner	higher education	unknown	90
Id11	en	slide	medium	learner	higher education	medium	60
Id12	en	slide	high	learner	higher education	medium	90
Id13	en	narrative text	unknown	learner	higher education	unknown	3
Id15	fr	narrative text	unknown	learner	higher education	unknown	120
Id26	fr	narrative text	medium	learner	higher education	medium	200

TABLA 3. Perfiles de los usuarios.

Perfil	Usuario1	Usuario2	Usuario3	Usuario4	Usuario5	Usuario6
Lengua materna	español	Español	español	español	español	español
Temática	Java	Java	Java	Java	Java	Java
Preferencia por Idioma Español	10	10	10	10	10	10
Preferencia por Idioma Inglés	9	10	10	10	10	10
Preferencia por Idioma Francés	0	9	9	9	9	9
(Rol,Preferencia)	(alumno,10)	(alumno,10)	(alumno,10)	(docente,8)	(docente,8)	(docente,7)
(Interacción,Preferencia)	(media,5)	(ignorar,-)	(ignorar,-)	(ignorar,-)	(alta,7)	(ignorar,-)
(Estilo,Preferencia)	(teórico,9)	(teórico,9)	(práctico,9)	(teórico,10)	(teórico,10)	(práctico,10)
(Contexto,Nivel,Preferencia)	(U,i,9)	(U,i,10)	(U,m,10)	(U,i,9)	(U,a,9)	(U,i,7)
(Duración máxima,Preferencia)	(0,-)	(0,-)	(0,-)	(0,-)	(0,-)	(90,9)
Costo	no	no	no	no	no	no

Para evaluar la cercanía entre el ranking recomendado por el Agente-R y el ranking propio del usuario, se utiliza la distancia de Manhattan, la cual es adecuada para capturar distancias entre posiciones.

El cálculo de esta distancia se realiza entre las posiciones de los tres primeros objetos ordenados por el usuario con las posiciones correspondientes en la lista recomendada, teniendo en cuenta el grado de intención de cada uno de los objetos en la lista. En la Tabla 4 se presentan las distancias calculadas sobre los cinco casos satisfactorios. Se excluye al Usuario1, dado que evaluó la respuesta del recomendador como *Resultado Incorrecto*, y por lo tanto no hay orden sugerido.

TABLA 4. Distancias calculadas sobre los casos satisfactorios.

Usuario	Distancia	Peor Caso
Usuario2	6.0	66.0
Usuario3	14.0	55.0
Usuario4	9.0	66.0
Usuario5	25.0	67.0
Usuario6	14.0	72.0
Promedio	13.6	65.2

El desempeño del Agente Recomendador depende fundamentalmente de la carga de metadatos de los objetos en el repositorio y de la calidad de esta información. Teniendo en cuenta esto y que el promedio de distancias ha resultado 13.6, con un promedio 65.2 en los peores casos, se puede afirmar que se observa un buen desempeño de este agente y, que en la mayoría de los casos, la lista de objetos recomendada no es tan distante a la elegida por el usuario. Sin embargo, los resultados podrían no satisfacer al usuario dado que, aunque un documento cumpla con las preferencias establecidas en la interfaz, esta especificación puede no transmitir lo que el usuario deseaba, ya que existen factores subjetivos difíciles de captar y modelizar.

En las pruebas realizadas, se han observado algunas dificultades vinculadas a la información de los metadatos. En algunos casos se encontró una mala asignación de valores en algunos metadatos de alguno de los objetos. Por ejemplo, se encontró un documento cuyo metadato Idioma (Language) es *inglés*, pero sólo tiene el título en inglés y el cuerpo del documento está en francés. Además, hay metadatos que si bien pueden estar catalogados correctamente desde el punto de vista del tipo de archivo, no llegan a describir adecuadamente el contenido del documento.

Por ejemplo, se encontró un documento donde el metadato Tipo de Recurso (Learning Resource Type) contenía el valor *texto narrativo*, pero era en realidad un programa en Java para resolver el problema del vendedor ambulante y que podría clasificarse como *ejemplo*. Esta clasificación del recurso como *texto narrativo* hace que al aplicar las reglas se tenga el máximo grado de creencia de que el objeto satisface el estilo teórico, lo cual no es cierto.

VI. CONCLUSIONES

En este trabajo se ha presentado la arquitectura y la implementación de un sistema recomendador de objetos de aprendizaje. Esta arquitectura se basa en un sistema multiagente lo cual permite trabajar de una forma flexible y escalable, con la información heterogénea y distribuida proveniente de los distintos repositorios de objetos de aprendizaje. Se diseñó uno de los agentes de esta arquitectura,

el Agente Recomendador, como un agente BDI graduado. Se ha implementado un prototipo del sistema recomendador utilizando los lenguajes Ruby on Rails para el Agente Interfaz y los Agentes Buscadores, y SWI-Prolog para el Agente Recomendador.

A partir de esta implementación se desarrolló un caso de estudio que arrojó resultados promisorios respecto a los rankings de OAs recomendados por el sistema. En esta primera etapa, los usuarios han evaluado si los objetos recomendados le son útiles y si el orden es el correcto, de acuerdo a su visión. A partir del caso de estudio se observa que la propuesta del Sistema Recomendador, del Agente-R y de las reglas definidas para computar el ranking de objetos, es promisoria.

Una evaluación más profunda de la recomendación brindada es dificultosa, ya que a través de la selección de los recursos más adecuados se busca alcanzar mejores resultados del proceso de enseñanza-aprendizaje, lo cual es complejo de evaluar. Un inconveniente que se presentó para realizar la prueba del prototipo es la falta de información en muchos de los metadatos educacionales de los OAs en los repositorios evaluados. Otra dificultad presentada es el acceso limitado a la descripción de los objetos o a ellos mismos.

Actualmente se trabaja en la integración de este sistema recomendador como parte de un asistente que ayude al docente a ensamblar objetos de aprendizaje de acuerdo a un diseño instruccional, considerando las características y preferencias de los alumnos a los cuales es dirigido.

AGRADECIMIENTOS

Este trabajo es financiado parcialmente por el Proyecto JARDIN LACCIR-RFP2008 (Latin American and Caribbean Collaborative ICT Research and Microsoft), por el Proyecto 219308 de la Secretaría de Estado de Ciencia, Tecnología e Innovación (Provincia de Santa Fe) y por el Proyecto de Investigación Institucional Mejora de la enseñanza de las ciencias en la carrera de Ingeniería Ambiental utilizando un sistema de recomendación de búsqueda personalizada de recursos educativos (Facultad de Química e Ingeniería, Universidad Católica Argentina).

REFERENCIAS

[1] D. Wiley. “Connecting Learning Objects to Instructional Design Theory: A definition, a metaphor, and a taxonomy”, in *Instructional Use of Learning Objects*, edited by D. A.Wiley, Ed. Association for Instructional Technology, 2002.

[2] O. Niinivaara. “Agent-Based Recommender Systems”, Technical Report, Dept. of CS, University of Helsinki, 2004.

[3] M. Montaner, B. López, J.L. de la Rosa. “A Taxonomy of Recommender Agents on the Internet”, in *Artificial Intelligence Review*, edited by Kluwer Academic Publishers, 19: 4, 285-330, 2003.

[4] A. Rao and M. Georgeff. “BDI Agents from Theory to Practice”, in *Technical Note 56, AAII*, 1995.

[5] A. Casali, L. Godo, C. Sierra. “Graded BDI Models For Agent Architectures”, in *CLIMA V*, edited by J. Leite and P. Torroni *LNAI* 3487, Springer-Verlag, Berling Heidelberg, 126-143, 2005.

[6] O. R. Zaiane. “Building a recommender agent for e-learning systems”, in Proceedings of *International Conference on Computers in Education*, pp: 55-59, 2002.

[7] C. Romero, S. Ventura, J. Delgado, P. de Bra. “Personalized Links Recommendation Based On Data Mining in Adaptive Educational Hypermedia Systems”, in *Second European Conference on Technology Enhanced Learning (EC-TEL 2007)*. Crete, Greece, 2007.

[8] N. Soonthornphisaj, E. Rojsattarat, S. Yimngam. “Smart E-Learning Using Recommender System”, in *LNCS*. Springer Berlin, Heidelberg, Volume 4114. Computational Intelligence pp 518-523. 2006.

[9] F. Zhu, H. Ip, J. Cao. “PeRES: A Personalized Recommendation Education System Based on Multi-agents & SCORM”, in *Advances in Web Based Learning (ICWL 2007)*, LNCS 4823/2008, pp 31-42, 2008.

[10] V. Gerling. “Un Sistema Inteligente para Asistir la Búsqueda Personalizada de Objetos de Aprendizaje”, Tesis de grado, Licenciatura en Ciencias de la Computación, Universidad Nacional de Rosario, 2009.