UNIVERSIDAD DE LOS ANDES, FACULTAD DE INGENIERÍA
DIVISIÓN DE ESTUDIOS DE POSTGRADO
PROGRAMA DE ESTUDIOS DE POSTGRADO EN COMPUTACIÓN
MÉRIDA- VENEZUELA

DISEÑO E IMPLEMENTACION DE LA PLATAFORMA DE GESTION DE OBJETOS DE APRENDIZAJE EN LA NUBE FUENTES DE CONOCIMIENTO APLICANDO EL PARADIGMA ODA.

Autor: Ing. Jorge Omar Portilla Jaimes.
Tutores: Dr. Jose L. Aguilar C.

Trabajo de Grado presentado ante la ilustre Universidad de Los Andes como requisito parcial para optar al grado de *Magister Scientiae* en *Computación*.

Mérida, Abril 2016

Índice de contenido

1 Introducción

1.1 Motivación

En la actualidad, en la Universidad de los Andes se está desarrollando un proyecto denominado "Proyecto Madre: creación de la carrera Ingeniería en Computación e Informática", el cual está basado en la creación de una Carrera Experimental, de manera tal que permita ir construyendo un espacio universitario en el qué el "que-hacer" universitario se haga desde la reflexión sobre el "deber ser" [1]. Esa condición de experimental permite una reflexión-acción permanente sobre el modo de "hacer" universitario, de tal forma que la carrera se vaya ajustando a los resultados maduros de ese proceso. Además, por las características de la propuesta se requieren condiciones particulares (organizacional, infraestructura, recurso humano, entre otras.) que soporten las actividades que se lleven a cabo en la misma, las cuales, en las estructuras clásicas actuales de las carreras de la Universidad de Los Andes (ULA) no es posible conjugar. Algunas de las características definidas [1] son: la dinámica de la carrera es centrada en los grupos de investigación, las mallas curriculares son flexibles, el modelo pedagógico es basado en el paradigma "aprender- haciendo", es basado en un proceso de auto-formación, entre otras.

Un elemento fundamental son los grupos de investigación, sobre ellos gravitará todo el que-hacer de la carrera. Particularmente, de ellos dependerán la gestión del conocimiento del área de formación, y el desarrollo de las obras o productos tecnológicos derivados del proceso de enseñanza "aprender-haciendo".

Por otro lado, la estructura organizacional de la carrera debe responder al concepto de "nube" plasmado en [2]. El Proyecto Madre propone la conformación de nubes, que según [1]: "responden a ciertas cualidades que en las mismas se dan, llamado el *efecto nube*: sus dinámicas internas se van moviendo con el viento (con el acontecer mundial y nacional, con la industria, etc.). Dichas cualidades dinamizan las actividades, el que-hacer, etc., que en el seno de las

mismas van emergiendo, pero además, permiten que se vayan entrecruzando entre ellas, generando una poderosa sinergia educativa. Así, el concepto de nube es usado como metáfora para indicar que es un espacio difuso en el que libremente se puede navegar, se puede caminar, cuyas fronteras no están claramente definidas o indicadas, y cuyos elementos constitutivos pueden aparecer o desaparecer, así como las interrelaciones que se dan entre ellas". Específicamente, será una acumulación de cosas en un ámbito dado (mallas curriculares, fuentes de conocimiento, etc.), que representan la abundancia de ese algo necesario para la realización de los objetivos perseguidos con esas nubes. "Así, se habla más de *densidad* que de *estructura*, permitiendo que esta última surja de las dinámicas internas que se den en cada nube" [2]. En [1,2] se están considerando tres nubes: Nube de Formación, Nube de Paradigmas de Aprendizaje y Nube de Fuentes de Conocimiento.

La nube de Formación realiza toda la gestión del proceso de formación del estudiante (malla curricular, récord académico, etc.); mientras que la nube de paradigma de aprendizaje contiene todos los estilos educativos existentes, con sus características a nivel de actividades, formas de evaluación, entre otras cosas, que las definen. En lo que respecta a la Nube de Fuentes de Conocimiento, está constituida por todo el conocimiento esparcido a través del mundo, en todas sus formas, desde todas las fuentes posibles, de acuerdo a lo contemplado en [1]. según [1], "su objetivo es posibilitar el mayor acceso al conocimiento disponible a nivel mundial, pero desde una mirada crítica al mismo. Las metodologías, herramientas y técnicas que conforman esta nube deben posibilitar el acceso crítico a ese conocimiento, según las dinámicas/actividades establecidas en las otras nubes".

Por otro lado, se ha venido desarrollando una plataforma tecnológica para darle soporte a este modelo educativo [3], para lo cual ya se han elaborado trabajos para la gestión de la nube de formación [4] y de paradigmas de aprendizaje [5]. En el caso de la nube de fuentes de conocimiento, ya se tiene una primera versión de la plataforma para la búsqueda de contenidos digitales en general [6], pero es necesario contar específicamente con un sistema de recomendación de OAs [3]. Precisamente la búsqueda y recomendación especifica de OAs representa el objetivo central de este trabajo.

1.2 Planteamiento del problema

Es importante tener en cuenta que en [1] se señala que "en la nube de fuentes de conocimiento, los aspectos humanísticos y sociales de formación juegan un rol fundamental, ya que son los que permitirán una aproximación al conocimiento con el ojo crítico del papel de la ciencia y tecnología en la sociedad. En ese sentido, el proceso de adquisición de conocimiento será desde el paradigma de apropiación del conocimiento".

Basados en lo anterior, se debe considerar que la nube de fuentes de conocimiento requiere de herramientas para organizar, analizar, explotar, criticar, dicho conocimiento (es decir, de gestión del conocimiento). Tales herramientas son las que se buscan desarrollar en el presente trabajo para el caso en especifico de OAs, y ella deben posibilitar que lo establecido en la nube de aprendizaje (paradigmas, estrategias, etc.) y en la nube de formación (auto-formación, etc.), actúen en forma mancomunada, logrando un proceso de integración efectivo.

Así, el problema en especifico tratado en este trabajo consiste en desarrollar un sistema de recomendación y búsqueda de OAs, adaptado a los perfiles de los estudiante, que se acople a la plataforma del proyecto madre ya existente.

1.3 Objetivos

1.3.1 General

- Desarrollar un prototipo basado en SOA para la gestión de una nube de fuentes de conocimiento compuesta por Objetos de Aprendizaje, aplicando el paradigma ODA (Ontology Driven Architecture).

1.3.2 Objetivos específicos

- Seleccionar los estándares, normas, reglas, por los cuales se guiará la nube de fuentes de conocimiento, en particular, vinculados a los Objetos de Aprendizaje.
- Diseñar el modelo arquitectónico funcional para la gestión de conocimientos de la nube de fuentes de conocimiento.
- Desarrollar sus distintas capas ontológicas (CIM, PIM, PSM), según el paradigma ODA.
- Implementar un prototipo del sistema que considere los servicios o aplicaciones necesarias para la gestión de la nube de fuentes de conocimiento, usando el paradigma SOA.

1.4 Antecedentes

En esta sección se describen trabajos ya existentes relacionados con lo propuesto en esta tesis. Para organizar mejor la información sobre trabajos recientes, se divide en dos tipos de trabajos, que son: sistemas recomendadores de objetos de aprendizaje, y diseño de servicios web basados en arquitectura ODA.

1.4.1 Estado del arte en sistemas recomendadores de OA

En [7] se describe el desarrollo de un sistema recomendador (SR) de objetos de aprendizaje (OA), el cual 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 OAs, 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. En particular, en ese trabajo se modela el Agente Recomendador (Agente-R), como un agente BDI, el cual se encarga de realizar una recuperación flexible y presentar una lista ordenada de los mejores objetos de acuerdo con el perfil de usuario.

En [8] se presenta otro SR de OA. El SR se basa en filtrado colaborativo, el cual utiliza una adaptación del algoritmo k-vecinos, y entre otras cosas, se fundamenta en la percepción de usabilidad y utilidad que el usuario tiene acerca de los OA que descarga del repositorio. En [4] se muestra la forma como el algoritmo k-vecinos adapta el concepto de percepción con la implementación de un sistema de votación de los OA por parte de los usuarios. Al final se muestra la validación del SR, utilizando el repositorio RODAS.

En [9] se presenta un SR híbrido de OA personalizado. El SR está basado en contenidos, y tiene en cuenta el estado cognitivo del estudiante. Usa el estándar LOM (Learning Object Metadata) [10] para describir los OAs, y utiliza un perfil del estudiante en el que se guarda información de su historial de navegación y el proceso de aprendizaje llevado a cabo por el estudiante. Además, utiliza una función de similitud basada en experiencias anteriores de usuarios, para establecer la relación entre lo que el OA ofrece y lo que el estudiante requiere, y lo usa como insumo para calcular una medida de calidad del objeto. Es importante resaltar que en este trabajo se usan ontologías para modelar el dominio.

En [11] se presenta un modelo y prototipo de un Agente Recomendador Híbrido que sugiere recursos de aprendizaje y OA usando técnicas de minería de datos. El sistema realiza cosechas de información de OA y los guardas en metadatos, mediante agentes HTTP, procesamiento de lenguaje natural y minería de datos. Para que el sistema funcione, se requiere que los profesores describan sus cursos en lenguaje natural.

En [12] se propone un método de recomendación híbrido, que asiste a los usuarios en la búsqueda y selección de repositorios de OA. El método propuesto usa una combinación de diferentes técnicas de filtrado, tales como comparación de contenidos, colaborativo y demográfica. Para lograr esta meta, se usan metadatos, los perfiles de usuarios, entre otras cosas. El método de recomendación híbrido fue implementado en un sistema de búsqueda llamado DELPHOS.

En [13] se presenta un recomendador de OA semántico basado en ontologías, con miras a fortalecer los sistemas de aprendizaje personalizados. Se proponen ontologías para modelar al aprendiz, a los OA y a las reglas de emparejamiento. El trabajo contempla los estilos de aprendizaje, y propone un Framework para usar ontologías para analizar e interpretar recursos de aprendizaje en sistemas de recomendación. La recomendación consta de cuatro pasos: emparejamiento semántico entre el aprendiz y los OA, cálculo de preferencias, priorización de los OA y recomendación de OA.

En [14] se desarrolló un prototipo de sistema de recomendación de OA que ha sido utilizado para experimentar con 3 diferentes algoritmos de recomendación basados en el perfil del usuario, con el fin de determinar y comparar la precisión de sus resultados. El prototipo fue realizado con bases de datos relacionales. El componente principal del prototipo implementado consiste en una Interfaz de Programación de Aplicaciones (API por sus siglas en inglés), desarrollada en Java, que se encarga de la comunicación con el repositorio del Proyecto IGUAL. Esa API implementa, entre otras, las funciones que ejecutan los algoritmos de recomendación seleccionados.

En [15] se presenta un SR de OA aplicando competencias y filtrado colaborativo. Para comparar dos estudiantes usan el coeficiente de correlación de Pearsons, y luego recomiendan los OA que tengan buena calificación por los estudiantes similares al que realiza la solicitud. Después del filtrado de los OA por el grado de similaridad, el sistema procede a ordenarlos por competencias, de acuerdo a lo ofrecido por el objeto y lo requerido por el estudiante. El prototipo usa las matrices para evaluar la calidad del SR de *precision* y *recall*. Con *precision* se calcula la relación entre el número de ítems que el usuario consideró relevantes y el número de ítems recomendados. Con *recall* se compara el número de ítems recomendados con respecto al total de ítems sobre el que se puede recomendar.

En [16] se presenta un SR federado el cual cosecha datos de diferentes plataformas de aprendizaje online, y provee recomendaciones personalizadas. El trabajo cuenta con un cosechador (harvester), que usando servicios REST obtiene metadatos de distintos repositorios y los almacena en un repositorio central.

Además, el sistema cuenta con una maquina de recomendación (RE Engine), la cual entrega recomendaciones a usuarios a través de un Servicios Web REST.

En [17] se propone un Framework que utiliza relaciones semánticas entre OA y aprendices. Tales relaciones son el insumo fundamental para proveer recomendaciones personalizadas a los aprendices. El Framework se divide en: servicios de repositorio, Servicios de Indexado Semántico (SIS) y Servicios de usuario. Los servicios de repositorio atienden todo lo relacionado con la gestión de contenidos del repositorio de OA. Los servicios de indexado semántico se encargan de construir y manipular una ontología, indexando cada uno de los OA agregados. El proceso de indexación es realizado a través de cuatro fases: Extracción de palabras claves del OA, identificación de conceptos (usando la ontología de dominio), medida de la representatividad de conceptos (en la que se calcula similaridad de un OA con cada concepto), y por ultimo, la fase de actualización de indice. Los servicios de usuario son los encargados de procesar la consulta del usuario.

1.4.2 Estado del arte en Framework para la descripción de servicios web

En [18] se presenta un Framework ontológico estructurado llamado Web Services Framework (WSF), se trata de una plataforma basada en servicios, que utiliza lenguajes de descripción específicos para la web. Propone ontologías y modelamiento basado en ontologías, para mejorar lo propuesto en UML. Este Framework contempla los servicios web semánticos, y usa las ontologías para capturar las propiedades funcionales y no funcionales de los servicios. Busca con ello emparejar los servicios guardados en repositorios con las necesidades especificadas del cliente. El WSF cuenta con ontologías para modelar las tres capas propuestas por MDA. Plantea un proceso de transformación CIM-PIM, pero no lo implementa en forma automática.

En [19] se propone un Framework manejado por modelos para el desarrollo de aplicaciones web orientadas a servicios. Este Framework presenta el Lenguaje de Modelamiento de Presentación (PML), haciendo particular énfasis en los procesos

de transformación. El trabajo enfatiza en la fase especifica de generación de código, buscando ser práctico y aplicable las transformaciones. En el trabajo se describen los pasos iniciales para usar el Framework orientado a servicios web manejado por modelos. Utilizan modelos en Lenguaje de Modelamiento de Presentación (PML), para generar código para lenguajes de cuarta generación. Los generadores propuestos fueron implementados usando el conjunto de herramientas ofrecidas por el openArchitectureWare modelling component [20].

En [21] proponen el MWSAF (METEOR-S Web Service Annotation Framework), es un Framework para marcación semiautomática de Servicios Web con ontologías. En él se desarrollan algoritmos para emparejar y anotar archivos WSDL con las ontologías propuestas. El Framework usa ontologías de dominio para categorizar los servicios web.

1.4.3 Aspectos diferenciadores

El SR de OA propuesto en este trabajo es implementado como un servicio web semántico, lo que le posibilita ser invocado automáticamente (hacer composición, descubrimiento y ejecución automática). Este hecho le confiere la capacidad de ser re-usado por diversas plataformas en forma automática.

Además, el recomendador hace énfasis en el modelo de usuario, lo cual lo enriquece más que muchos recomendadores que se centran solo en el curso, buscando así un nivel de personalización mucho más adecuado, pues saca conclusiones basada en aspectos como por ejemplo, el historial de uso del estudiante, el estilo de aprendizaje del estudiante, entre otras cosas. En particular, el recomendador se vincula directamente con la nube de aprendizaje para analizar profundamente los paradigmas de aprendizaje apropiados para el estudiante que solicita la recomendación, priorizando los OA que cuenten con herramientas de enseñanza, formas de evaluación y actividades de aprendizaje, más apropiadas para el aprendiz para el cual se solicita la recomendación.

El hecho de que el recomendador este basado en lógica descriptiva, le permite gran poder de inferencia y formalización, lo cual garantiza que los algoritmos de

emparejamiento sean más robustos y formales, comparados por ejemplo con recomendadores basados en bases de datos, como algunos de los encontrados dentro del estado del arte.

Finalmente, nuestro recomendador de OA propone un factor de calibración que le proporciona mayor flexibilidad para balancear el nivel de hibridación del recomendador (por ejemplo, para hacer a veces prevalecer más para la recomendación el conocimiento sobre el aprendiz, o la opinión de los otros, etc.). Este hecho puede constituir una gran ventaja en el momento de ser usado por clientes externos a las nubes del Proyecto Madre, que no existe en los SR referenciados.

1.5 Metodología y alcances

En este trabajo de investigación se pretende utilizar la arquitectura dirigida por ontologías (ODA) para modelar los componentes de la arquitectura SOA del SR desarrollado, y sus relaciones, tal que los mismos permitan la gestión de la nube de fuentes de conocimiento.

El trabajo de investigación está basado principalmente en el método científico, buscando con ello dar cumplimiento a los objetivos planteados en el mismo. El diseño de las capas ontológicas se desarrolló de acuerdo a la ODA, usando técnicas de transformación para convertir los modelos independientes de la plataforma, que especifican la operación de un sistema, en modelos específicos a una plataforma que indiquen los detalles de cómo usar las capacidades del sistema.

De esta manera, el diseño general del modelo arquitectónico funcional para la gestión de la nube de fuentes de conocimiento se desarrolló usando ODA y SOA, utilizando las ventajas que ofrecen las mismas para un diseño estratificado en capas.

1.6 Organización del trabajo

El presente trabajo esta conformado por los siguientes capítulos: El capitulo uno contiene una breve introducción acerca de las generalidades de la tesis. Posteriormente se presenta el marco teórico de la tesis. El capitulo tres documenta el análisis, diseño, implementación y pruebas del Framework basado en ODA, para la descripción y composición de Servicios Web Semánticos (FODAS-WS) propuesto en este trabajo, el cual fue usado durante la fase de diseño del SR de OA de la nube de fuentes de conocimiento del Proyecto Madre presentado en esta tesis. El cuarto capitulo está dedicado a documentar la arquitectura y diseño del SR de OA, que conforma la propuesta principal de la tesis. En el capitulo cinco se plasman aspectos de implementación y pruebas de funcionamiento de la plataforma de gestión de la nube fuentes de conocimiento, basada en el SR de OA propuesto en el capítulo anterior. En el capitulo seis se plasman las conclusiones y trabajos futuros.

2 Marco Teórico.

En este capítulo se presenta los referentes teóricos necesarios para el desarrollo de la tesis. Fundamentalmente, se plasma en una primera sección aspectos relacionados con el proyecto madre en el cual está enmarcada esta tesis, una segunda sección está dedicada conceptualizar algunas Tecnologías de mejoramiento del aprendizaje (TEL), la sección tres abarca lo relacionado con sistemas recomendadores, y una cuarta sección presenta la estructura conceptual de la Arquitectura Manejada por Ontologías (ODA)

2.1 Proyecto Madre

Una idea innovadora de un proyecto educativo que mejore la calidad del proceso de aprendizaje de un estudiante, viene siendo desarrollada por el Departamento de Computación de la Universidad de Los Andes, bajo el nombre de "Proyecto Madre" [1, 2]. Las ideas planteadas se basan en una nueva carrera en el ámbito de las Tecnologías de la Información que tenga características fuera de lo convencional, como por ejemplo, una dinámica centrada en los grupos de investigación, mallas curriculares adaptables a varios perfiles de educación, y un modelo pedagógico basado en el paradigma "aprender haciendo", todo esto garantizando un proceso de auto-formación que va a ser ejecutado por el estudiante guiado por los profesores.

La gestión del conocimiento y el proceso de formación es plasmado bajo el concepto de nube. El concepto de nube se refiere a la agrupación de todos los componentes de un modelo educativo (proceso de formación, paradigmas de aprendizaje y fuentes de conocimiento). El Proyecto Madre se conforma de tres nubes, cada una gestionando lo que es propio a ella. Cada una de ellas está ajustada al modelo filosófico de las nubes, en el cual se estipula que entre nubes se puede navegar, sin que sus fronteras estén definidas claramente. Además, dicho modelo filosófico considera que los elementos de cada nube pueden aparecer y desaparecer, así como las interrelaciones entre ellas [2]. El concepto

de nube permite también un dinamismo en las actividades que realiza el estudiante en el transcurso de la carrera, haciendo que vayan emergiendo nuevas características, generando una poderosa sinergia educativa.

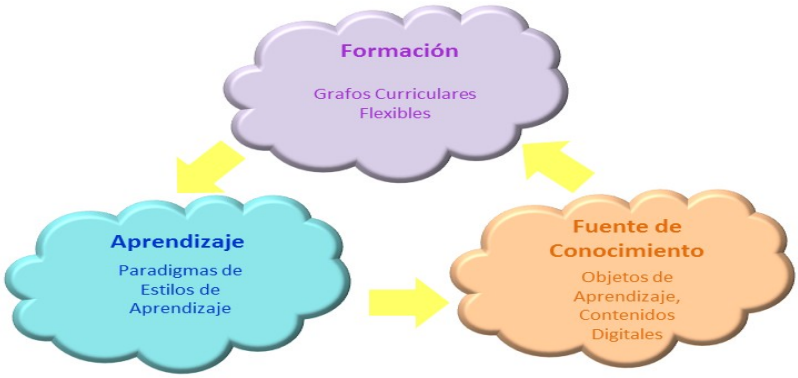

Figura 2.1 Nubes que componen el Proyecto Madre

Como se dijo antes (ver Figura 2.1), el proyecto Madre se compone de tres nubes, que son: Nube de formación, Nube de paradigmas de aprendizaje y Nube de fuentes de conocimiento. Cada una de estas tres nubes se describen a continuación [1, 2, 3]:

2.1.1 Nube de Formación

Esta nube está compuesta por todo lo relacionado con el proceso de gestión de formación del estudiante [4]. Al hablar de formación del estudiante, se hace referencia al proceso de construcción del currículo mediante el cual el estudiante se va a formar, y a la interacción que lleva a cabo el estudiante con dicho currículo. La estructura curricular está compuesta por módulos de formación que pueden constituir una materia, los cuales deben ser auto-contenidos. En la medida que el estudiante avance en su proceso de formación en el currículo, cuenta con la opción de obtener varias titulaciones. Cada una de estas titulaciones depende de la cantidad de créditos obtenidos en áreas específicas. Es importante resaltar que los créditos pueden ser de conocimiento o tecnológicos. Los créditos de conocimiento se adquieren después de completar módulos, realizar pasantías, etc. Los créditos tecnológicos se derivan de la ejecución de una obra (proyecto

científico o tecnológico, donde el estudiante aplica los conocimientos adquiridos hasta ese momento).

2.1.2 Nube de Paradigmas de Aprendizaje

Esta nube comprende lo relacionado con los paradigmas, las estrategias, las formas de evaluación y las herramientas de aprendizaje [5]. Su objetivo es aportar los mecanismos de aprendizaje necesarios para completar el proceso de auto-formación del estudiante, guiando las dinámicas de formación, y estableciendo formas de acreditar cursos, entre otras. Esta nube permite la generación de actividades que guíen la inclusión de ejes transversales vinculados al ámbito humanístico, a las artes y creatividad, formando ingenieros capaces de reconocer su entorno social.

La nube de paradigmas de aprendizaje debe posibilitar la adquisición de forma autónoma de conocimiento y competencias, aunado a esto, la nube debe proveer el soporte en tecnologías de información para la gestión de conocimiento, cursos, proyectos. El proceso de aprendizaje que se brinda es enriquecido con esta nube permanentemente, basado en las características individuales del estudiante (proceso de aprendizaje centrado en el estudiante), conectando a alumnos, docentes, tecnólogos, etc.

2.1.3 Nube de Fuentes de Conocimiento

Está constituido por todo el conocimiento esparcido a través del mundo, en todas sus formas, desde todas las fuentes posibles [1, 6]. Su objetivo es posibilitar el mayor acceso al conocimiento disponible a nivel mundial, pero desde una mirada crítica al mismo. Las metodologías, herramientas y técnicas que conforman esta nube deben posibilitar el acceso crítico a ese conocimiento, según las dinámicas/actividades establecidas en las otras nubes. En esta nube, los aspectos humanísticos y sociales de formación juegan un rol fundamental, ya que son los que permitirán una aproximación al conocimiento con el ojo crítico del papel de la ciencia y tecnología en la sociedad [1, 2]. En ese sentido, el proceso de adquisición de conocimiento se realiza desde el paradigma de apropiación del

conocimiento. Esta nube requiere de herramientas para organizar, analizar, explotar, criticar, dicho conocimiento (es decir, de gestión del conocimiento). Esas herramientas deben posibilitar lo establecido en la nube de aprendizaje (paradigmas, estrategias, etc.) y en la nube de formación (auto formación, etc.). Esto permitirá hacer énfasis en la adquisición de conocimientos, en las destrezas para usarlo y reflexionar sobre él, y la actitud como ciudadano para reconocerlo en el contexto social donde este inmerso [1, 2].

Como ya se mencionó, existen una serie de trabajos ya desarrollados en torno al proyecto madre, desde su concepción teórica y filosófica [1, 2], hasta la implementación de plataformas de gestión tecnológicas para cada nube, presentadas en [4, 5, 6, 3]. El trabajo presentado en esta tesis constituye el núcleo fundamental de la nube de fuentes de conocimiento presentada en [1, 2], pues se requiere específicamente, de un sistema recomendador de OAs, por ser un instrumento fundamental de los procesos de enseñanza-aprendizaje. Es necesario aclarar, que ya en [6] se implementó un buscador de contenidos digitales, que forma parte de la nube de conocimiento.

2.2 Tecnologías de Mejoramiento del Aprendizaje

De acuerdo con [22], las Tecnologías de Mejoramiento del Aprendizaje (TEL) tienen como objetivo diseñar, desarrollar y probar innovaciones socio técnicas que puedan soportar y mejorar las practicas de aprendizaje, tanto de estudiantes como de organizaciones. Es por tanto, un dominio de aplicación que generalmente cubre tecnologías para soportar todas las formas de enseñanza y actividades de aprendizaje. Entre las diversas Tecnologías asociadas con este campo del conocimiento se encuentran los OAs, los repositorios de OAs y los estándares para la descripción de metadatos como LOM.

2.2.1 Objetos de Aprendizaje

Según lo estipula [23], los Objetos de Aprendizaje (OA) son elementos para la instrucción, aprendizaje o enseñanza basada en computadora. No son realmente

una tecnología, son una filosofía, que se fundamenta en el paradigma orientado a objetos. Formalmente, no hay una única definición del concepto de objeto de aprendizaje, las definiciones son muy amplias. El Comité de Estandarización de Tecnología Educativa [24] dice que los OAs son "una entidad, digital o no digital, que puede ser utilizada, reutilizada y referenciada durante el aprendizaje apoyado con tecnología". Según [23], los OA son "cualquier recurso digital que puede ser reutilizado para apoyar el aprendizaje"; en [25] los definen como "una pieza digital de material de aprendizaje que direcciona a un tema claramente identificable ... que tiene el potencial de ser reutilizado en diferentes contextos". En [26] se definen los OA como una unidad de aprendizaje independiente y autónomo, que está predispuesto a su reutilización en diversos contextos instruccionales. Y por otra parte, [27] dice que "un OA es cualquier recurso que puede ser utilizado para facilitar la enseñanza y el aprendizaje, y que ha sido descrito utilizando metadatos". Todas estas definiciones son muy amplias, y en la práctica pueden resultar inoperables, ya que no hay un elemento claro que distinga a los OA de otros recursos. En particular, en este trabajo se entenderá por OA como unidades autónomas que soportan el proceso de aprendizaje llevado a cabo mediante el uso de las Tecnologías, y que se encuentran descritas mediante metadatos.

2.2.2 Repositorios de OAs

Según [28], los sistemas de Repositorios de Objetos de Aprendizaje (ROA) son la infraestructura clave para el desarrollo, almacenamiento, administración, localización y recuperación de OA. En [29] se estipula que los ROA "son un catálogo electrónico/digital que facilita las búsquedas en Internet de objetos digitales para el aprendizaje". [27], a partir de los términos "repositorio digital", "objeto de aprendizaje" y "metadato", dice que "los repositorios de objetos de aprendizaje son bases de datos con búsquedas que alojan recursos digitales y/o metadatos que pueden ser utilizados para el aprendizaje mediado". Otros autores como [27], adoptan la siguiente definición: "Un ROA es una colección de OA que tienen información (metadatos) detallada que es accesible vía Internet. Además de alojar los OA, los ROA pueden almacenar las ubicaciones de aquellos objetos almacenados en otros sitios, tanto en línea como en ubicaciones locales".

Según [30], las definiciones en su sentido general, no difieren mucho entre sí, y dejan ver claramente que estos repositorios están creados para ser utilizados en un proceso de enseñanza, lo cual lleva a que los ROA se vean como facilitadores claves para incrementar el valor de los recursos de aprendizaje dando la oportunidad de reutilizar, reorientar y hacer reingeniería para cubrir las necesidades del usuario final. En la práctica, los ROA disponibles hoy día pueden apegarse a distintos estándares , pero la tendencia es utilizar LOM o algún esquema compatible o derivado de éste.

2.2.3 Metadatos para objetos de aprendizaje

En el año 2002 se emite el estándar 1484.12.1 especificado en [10], que acredita al modelo de datos **Learning Object Metadata** (LOM) como el estándar de metadatos para OA. LOM especifica la semántica y la sintáctica de un conjunto mínimo de metadatos necesario para, completa y adecuadamente, identificar, administrar, localizar y evaluar un OA. Su propósito es facilitar a profesores, alumnos y a sistemas automáticos la tarea de buscar, compartir e intercambiar OA, permitiendo el desarrollo de catálogos que contemplan la diversidad cultural e idiomática de los contextos en los que se puedan utilizar los objetos y sus metadatos. El estándar LOM es muy extenso (76 elementos, y además es extensible), por lo que para tener una mejor organización y estructura, los metadatos se organizan en forma jerárquica. Para poder asignar valores, deben tenerse algunos conocimientos técnicos del recurso y conocimientos del campo pedagógico, por lo que se requiere de intervención humana (tal vez especializada), y difícilmente pueden llenarse los datos de forma automatizada. LOM agrupa los metadatos en nueve categorías [10], que son: general, ciclo de vida, metadatos, técnica, educativa, derechos, relación, anotación y clasificación.

2.3 Sistemas Recomendadores

El núcleo fundamental de la nube de fuentes de conocimiento diseñado e implementado en esta tesis es un sistema recomendador híbrido de OAs. Por tal

motivo en la presente sección se explican someramente los conceptos relacionados con los sistemas de recomendación.

2.3.1 Definiciones de sistemas recomendadores

En la Tabla 2.1 se sintetizan algunas definiciones importantes encontradas en la literatura asociadas con los sistemas de recomendación.

REFERENCIA	DEFINICIÓN
[31]	"Los sistemas recomendadores utilizan el conocimiento del producto (ya sea conocimiento estipulado directamente por expertos o conocimiento extraído del comportamiento de consumidores) para orientar a consumidores en la búsqueda de productos de su preferencia"
[32]	"Cualquier sistema que produce recomendaciones individualizadas como salida, o tiene el efecto de guiar al usuario en forma personalizada a los objetos interesantes o útiles en un gran espacio de posibles opciones"
[33]	"Los sistemas recomendadores utilizan las opiniones de miembros de una comunidad para ayudar a las personas de esa comunidad a identificar la información o productos más interesantes respecto de sus necesidades "
[34]	"Los sistemas de recomendación utilizan opiniones de una comunidad de usuarios, para ayudar eficazmente a las personas de esa comunidad a identificar contenidos de interés a partir de un potencial y abrumador conjunto de opciones"
[35]	"Sistemas recomendadores – Tecnología de filtrado de información personalizada, utilizada ya sea para predecir si un usuario en particular gusta de un tema en particular (problema de predicción), o para identificar un conjunto de elementos que son de interés para un determinado usuario (problema de recomendación)"
	"Un sistema de recomendación personalizada puede proporcionar servicio a los clientes basado en el comportamiento pasado de

[36]	los clientes o a través de la inferencia del comportamiento de otros usuarios con comportamientos similares. El objetivo de la personalización es ofrecer lo que ellos desean, sin cuestionarlos explícitamente, capturando el componente social y la interacción interpersonal”
[37]	“Los sistemas recomendadores sugieren temas de interés para los usuarios basados en sus preferencias explicitas e implícitas, las preferencias de otros usuarios, y los atributos de usuario y de artículos”
[38]	“Los sistemas de recomendación son agentes de información que proporcionan recomendaciones personalizadas: sugerencias para artículos que pueden ser de utilidad para un usuario... Un recomendador puede distinguirse de los sistemas de recuperación de información (information retrieval system) por la semántica de la interacción del usuario”
[39]	“.... las opiniones de otros usuarios pueden ser seleccionadas y agregadas, de tal forma que provean una predicción razonable del comportamiento de un usuario activo”

Tabla 2.1 Distintas definiciones de sistemas recomendadores

A partir de la Tabla 2.1 puede sintetizarse que un sistema recomendador es un agente de software que sugiere productos (ítems) a un usuario, acudiendo a distintos tipos de información para realizar las recomendaciones.

En un sistema recomendador, los ítems de interés y las preferencias de usuario son representadas de varias formas (por ejemplo, usando un único o múltiples atributos para describir un ítem). Particularmente en sistemas donde la recomendaciones son basadas en la opinión de otros, es crucial tener en cuenta múltiples factores que afectan la opinión de los usuarios, para así lograr mejores recomendaciones.

En [22] consideran que se tiene un conjunto C que contiene todos los usuarios del sistema, y un conjunto S que contiene todos los ítems posibles que pueden ser recomendados. Se define $U^C(s)$ como la función de utilidad dada por $U^C(s) : C \times S \rightarrow R$, que mide lo apropiado de recomendar un ítem ***s*** para un usuario ***c***. Se

supone que esta función no es conocida para el conjunto C X S, pero si se conoce para un subconjunto de este.

Por lo tanto, en el contexto de la recomendación se quiere que cada usuario $c \in C$ este habilitado para:

•Estimar (o aproximar) la función de utilidad $U^C(s)$ para un ítem s del espacio S, para el cual $U^C(s)$ no se conoce todavía; o

•Seleccionar un conjunto de N ítem que maximicen $U^C(s)$

En muchos sistemas recomendadores, la función de utilidad $U^C(s)$ es considerada como un atributo de cada ítem (por ejemplo, evaluación de desempeño). Sin embargo, la utilidad también puede considerar varios factores asociados con los atributos de cada ítem. Por tanto, el problema de recomendación se convierte en multiatributo.

2.3.2 Tipos de sistemas de recomendación

En [40] se plantean seis distintos tipos de sistemas de recomendación:

2.3.2.1 Recomendación basada en contenidos

En ella se recomienda al usuario artículos similares a los que él ha preferido en el pasado. Los sistemas de recomendación basados en contenido analizan un conjunto de artículos y/o descripciones preferidas en el pasado por un usuario, y construye un modelo o perfil de los intereses o preferencias del usuario basado en las características de estos artículos.

2.3.2.2 Recomendación colaborativa

Este tipo de recomendación recomienda al usuario artículos que personas con gustos o preferencias similares usaron en el pasado. En la recomendación colaborativa (o de filtrado colaborativo), los sistemas predicen el interés del usuario en nuevos artículos basados en las recomendaciones realizadas a otras personas con intereses similares.

2.3.2.3 Recomendación demográfica

En ella se clasifica los usuarios en clases demográficas de acuerdo con los atributos de su perfil personal, y hace recomendaciones basadas en dichas clases.

2.3.2.4 Recomendaciones basadas en utilidad

En ella se realizan recomendaciones sobre la base de un cálculo de utilidad de cada elemento para un usuario. Lo anterior requiere del planteamiento y uso de una función de utilidad.

2.3.2.5 Recomendación basada en conocimiento

En ella se sugieren artículos basados en inferencias lógicas sobre las preferencias del usuario. Es necesaria una representación del conocimiento (por ejemplo reglas) acerca de como un articulo responde a la necesidad de un usuario.

2.3.2.6 Recomendación híbrida

En este tipo de recomendación se combinan dos o más tipos de recomendación de los descritos anteriormente, con el fin de obtener un mejor rendimiento y abordar las deficiencias de cada tipo de recomendación.

2.3.3 Sistemas recomendadores en TEL

El despliegue de los sistemas recomendadores ha cobrado mayor interés a partir del instante en que el área de recuperación de información (information retrieval) se convirtió una actividad fundamental en TEL (entendido como búsqueda de recursos de aprendizaje relevantes para soportar profesores y aprendices). Lo anterior se fundamenta en el hecho de que un problema tradicional en TEL ha sido el mejorar la búsqueda de recursos de aprendizaje digitales. A consecuencia de lo anterior, el concepto de sistemas recomendadores se ha convertido en un área muy atractiva en la investigación en TEL.

Según [22], de todos los esfuerzos realizados han surgido múltiples observaciones interesantes, entre las que se destacan:

- Hay un gran número de sistemas de recomendación que han sido desplegados en configuraciones TEL.
- Las metas de recuperación de información que persiguen los sistemas recomendadores TEL son a menudo diferentes a los identificados en otros sistemas (por ejemplo, en recomendación de productos).
- Existe necesidad de identificar las particularidades de los sistemas recomendadores TEL, buscando elaborar métodos propios para sus diseños semánticos, desarrollo y evaluación.

2.3.4 Evaluación de desempeño en recomendadores

Las múltiples características de los sistemas de recomendación, hacen que se consideren múltiples dimensiones para evaluar recomendadores. Considerar solo una dimensión y métrica para evaluar la amplia variedad de sistemas de recomendación y dominios de aplicación, se aleja mucho de la realidad de obtener una matizada y completa evaluación.

En [40] y [41] se presentan una variedad de dimensiones que pueden ser usadas en la evaluación de sistemas de recomendación. Algunas de estas dimensiones describen características cualitativas mientras que otras son más cuantitativas. Tales dimensiones son explicadas en la Tabla 2.2.

Dimensión	Descripción
Exactitud (Correctness)	¿Qué tan cerca están las recomendaciones respecto de un conjunto de recomendaciones asumido como correcto?
Cobertura (Coverage)	¿En qué medida el sistema de recomendación cubre un conjunto de ítems o espacio de usuario
Diversidad (Diversity)	¿Qué tan diversos (diferentes) son los ítems recomendados en una lista?
Integridad (Trustworthiness)	¿Qué tanto confía el usuario en las recomendaciones dadas por el recomendador?
Confianza	¿Qué grado de confianza o seguridad hay en las

(confidence)	recomendaciones realizadas?
(Novedad) Novelty	¿Qué tan acertado es el recomendador recomendando ítems desconocidos por el usuario?
Sorprendibilidad (Serendipity)	¿En qué medida el sistema logra proporcionar recomendaciones sorprendentes y beneficiosas?
Utilidad (Utility)	¿Cuál es el valor que obtienen los usuarios de la recomendación realizada?
Riesgo (Risk)	¿Cuanto riesgo corre el usuario al aceptar cada recomendación?
Robustez (Robustness)	¿Qué tolerancia tiene el sistema de recomendación hacia sesgos o información falsa?
Tasa de Apren. (Learning rate)	¿Qué tan rápido puede el sistema incorporar nueva información para actualizar sus recomendaciones?
Usabilidad (Usability)	¿Qué tan fácil de usar es el sistema para los usuarios?
Escalabilidad (Scalability)	¿Qué tan escalable es el sistema respecto al número de usuarios, tamaño de datos fundamentales, y ejecución de algoritmos?
Estabilidad (Stability)	¿Qué tan consistentes son las recomendaciones respecto a cierto periodo de tiempo?
Privacidad (Privacy)	¿Existen riesgos para la privacidad de los usuarios?
Preferencias de Usuario (User preference)	¿Cómo perciben los usuarios el sistema de recomendación?

Tabla 2.2 Dimensiones de evaluación de sistemas de recomendación

En [40] se recalca que diferentes aplicaciones tienen diferentes necesidades, el diseñador del sistema debe decidir que propiedades son importantes para medir el recomendador dependiendo del tipo de aplicación.

En [41] agrupan estas dimensiones en cuatro grandes categorías. Tales categorías son: Centradas en recomendación, centrada en usuario, centrada en el sistemas y centrada en la entrega. La Figura 2.2 ilustra cómo se agrupan tales categorías. La dimensión centrada en la recomendación evalúa la calidad de las recomendaciones realizadas por el recomendador. Por otro lado, las dimensión centrada en el usuario permite medir el grado en que el sistema recomendador evaluado cumple

con las necesidades del usuario final. Las dimensión centrada en el sistema estiman la capacidad del sistema recomendador en si mismo en aspectos como estabilidad, privacidad, robustez, etc. Por ultimo, la dimensión centrada en la entrega evalúa principalmente el desempeño del sistema recomendador en lo referente al contexto en que se usa.

Centradas en Recomendación	Centradas en el Usuario
Exactitud Cobertura Diversidad Confianza	Integridad Novedad Sorprendibilidad Utilidad Riesgo
Centradas en el Sistema	**Centradas en Entrega**
Robustez Tasa de aprendizaje Escalabilidad Estabilidad Privacidad	Usabilidad Preferencias de usuario

Figura 2.2 Categorización de dimensiones de evaluación.

2.3.4.1 Relaciones entre las dimensiones

Para tener una evaluación efectiva, se deben considerar las relaciones que se dan entre las distintas dimensiones de evaluación de recomendadores. Estas relaciones describen como los cambios en una dimensión pueden afectar a las otras. La Figura 2.3 muestra las relaciones entre las dimensiones respecto del desempeño general del sistema de recomendación. Cada celda en la Figura 2.3 representa relaciones entre una dimensión respecto de las otras. Cambios en una dimensión están en concordancia con otras dimensiones. Un punto (.) en la figura representa que la mejora en una dimensión mejora la otra. Una X refleja que una dimensión tiende a tener impacto adverso sobre la otra. Las dimensiones que son independientes cuentan con un espacio en blanco en las intersecciones de la tabla ilustrada en la Figura 2.3

Metrica	Preferencias Usuario	Exactitud	Cobertura	Confianza	Integridad	Novedad	Sorprendibilidad	Diversidad	Utilidad	Riesgo	Robustez	Privacidad	Usabilidad	Estabilidad	Escalabilidad	Tasa de Aprendizaje
Preferencias Usuario	-	×	⊙		⊙				⊙				⊙	⊙		
Exactitud	×	-	⊙		⊙	×	×	×	⊙	⊙	×	×			⊙	⊙
Cobertura	⊙	⊙	-			⊙	⊙	⊙	⊙			×			⊙	
Confianza				-												
Integridad	⊙	⊙			-					⊙	⊙		⊙	⊙		
Novedad		×	⊙			-	⊙	⊙		×			⊙			
Sorprendibilidad		×	⊙			⊙	-	⊙		×			⊙			
Diversidad		×	⊙			⊙	⊙	-		×						
Utilidad	⊙	⊙	⊙						-	⊙						⊙
Riesgo		⊙			⊙	×	×	×	⊙	-	⊙	⊙				
Robustez		×			⊙					⊙	-			⊙		×
Privacidad		×	×							⊙		-				
Usabilidad	⊙				⊙	⊙	⊙						-			
Estabilidad	⊙				⊙						⊙			-		
Escalabilidad		⊙	⊙												-	
Tasa de Aprendizaje		⊙							⊙		×					-

Figura 2.3. Relaciones entre dimensiones de evaluación de sistemas recomendadores.

De acuerdo con [42], Cobertura puede afectar directamente a Exactitud, ya que cuanto más datos disponibles se tienen para generar recomendaciones más significativas son las recomendaciones. Por lo tanto, incrementa la exactitud. Por otro lado, de acuerdo a [43], a mayor exactitud genera más restricciones, y por consiguiente, decrementa a *sorprendibilidad*. Lo mismo pasa para el riesgo. Por ejemplo, recomendaciones hechas usando un ambiente de alto riesgo pueden implicar algunas restricciones. Esto decrementa sorprendibilidad, novedad y diversidad, pero incrementa exactitud, confianza y utilidad.

A continuación se presentan detalladamente, las medidas de rendimiento usadas en este trabajo.

2.3.4.2 Exactitud (precisión de la Predicción)

La exactitud/precisión de la predicción es la propiedad más discutida en la literatura de sistemas de recomendación. La mayoría de los sistemas recomendadores es una maquina de predicción. Esta máquina predice las calificaciones de un usuario sobre ítems. Las medidas de precisión de la predicción miden la precisión dada de una recomendación. En [41] se presentan tres tipos de medidas de precisión de la predicción, que se usan dependiendo del tipo de recomendación que los sistemas generan y su forma de hacerlo. Las tres formas planteadas en la literatura para medir exactitud son:

Medidas de precisión de la calificación de las predicciones.
Medidas de precisión del uso de las predicciones
Medidas de precisión del ranqueo de los ítems

2.3.4.2.1 Medidas de precisión de la calificación de las predicciones

Hay muchas aplicaciones en las que se quiere predecir la calificación que un usuario dará a un ítem. En tales casos se debe medir la precisión del sistema recomendador para predecir tales calificaciones. La medida más usada y popular para predecir las calificaciones que un usuario otorgará a un ítem es la Root Mean Squared Error (RMSE). Para calcular RMSE, se requiere conocer con anterioridad las calificaciones esperadas del usuario (obtenidos generalmente como producto de estudios previos realizados). La RMSE mide el grado similitud existente entre las calificaciones dadas por el usuario para un ítem, y las calificaciones que se esperaba que dicho usuario diera para el ítem.

2.3.4.2.2 Medidas de precisión del uso de las predicciones

En muchas aplicaciones, los sistemas recomendadores no predicen las preferencias de usuarios por ítems, sino que tratan de recomendar a los usuarios ítems que pueden utilizar (conformar un CONJUNTO DE RECOMENDACIÓN). En este caso, las medidas del uso de predicciones se utilizan para determinar el grado de utilidad que las recomendaciones de ítems realizadas representan para el usuario.

Para que los sistemas de recomendación sean de real valor estos, deben proporcionar resultados útiles que estén cerca de los intereses o intenciones de los usuarios, sin abrumarlos con resultados no deseados. Según [40], se tienen cuatro resultados posibles para los ítems recomendados y los ocultos (ítems existentes no recomendados), los cuales se sintetizan en la Tabla 2.4.

VALOR	DESCRIPCION
tp (True-positive)	Cantidad de ítems recomendados que deben realmente recomendarse (es un correcto resultado)
fp (False-positive)	Cantidad de ítems recomendados que no debieron ser recomendados
Fn (False-negative)	Cantidad de ítems no recomendados que debían haber sido recomendados al usuario
tn (True-Negative)	Cantidad de ítems no recomendados que no debieron recomendarse (es un correcto resultado)

Tabla 2.4 Posibles resultados para los ítems recomendados y no recomendados según [40]

Los resultados mostrados en la Tabla 2.4 pueden explicarse mejor usando la Tabla 2.5, la cual se obtiene a partir de la construcción de un conjunto de recomendaciones predefinido conocido como meta estándar. A partir del conjunto meta estándar se calculan los valores presentados en la Tabla 2.5, que sirven como base para el cálculo de las medidas precision y recall explicadas a continuación.

	Recomendados	No Recomendados	total
Usados Relevantes	True-positive tp=**RA**	Falso-negative fn	tp+fn=**R**
No usados Irrelevantes	False-positive fp	True-negative tn	fp+tn
total	tp+fp=**A**	fn+tn	**N**

Tabla 2.5 Clasificación de los posibles resultados de una recomendación de un item a un usuario

2.3.4.2.2.1 Precisión

Esta métrica determina la calidad de la recomendación en lo referente a la exactitud de la recomendación, de acuerdo con un conjunto de recomendaciones predefinido dado, conocido como meta estándar. Esta medida es ampliamente usada en recuperación de información (information retrieval), y representa la relación entre el número de contenidos relevantes y el total de contenidos retornados por una función de búsqueda. En el contexto de sistemas recomendadores, presicion es la relación entre el número de ítems que el usuario considera relevantes y el número de ítems recomendados. De acuerdo con [41], esta medida es calculada mediante la ecuación (2.1).

$$presicion = \frac{RA}{A} \qquad (2.1)$$

Donde RA es el número de ítem relevantes o interesantes recomendados (ítems bien recomendados por el recomendador, true-positive), y A es el número de ítems recomendados actualmente por el sistema, tomando en cuenta tanto resultados positivos como falsos positivos.

presicion mide la relación entre la cantidad de ítems bien recomendados respecto de la cantidad total de ítems que el recomendador recomendó. Lo anterior determina el grado de exactitud del recomendador. Pues, una medida de 1 en precision indica que cada uno de los ítems recomendados realmente debían recomendarse, lo cual indica un 100% de exactitud.

2.3.4.2.2.2 Recall

Según [41], recall mide el porcentaje de ítems interesantes sugeridos al usuario, con respecto al número total de ítems interesantes. Esta métrica determina el grado de completitud de la recomendación. De acuerdo con [41], esta medida se

calcula usando la ecuación (2.2) con base en los valores establecidos conforme a lo indicado en la Tabla 2.5.

$$recall = \frac{RA}{R} \qquad (2.2)$$

Donde R es el número total de ítem que se deben recomendar (cantidad de ítem que deberían recomendarse). Un valor de 1 en recall indica un 100% de completitud. Es decir, que cada uno de los ítems que deberían recomendarse fueron incluidos en la recomendación.

2.3.4.2.2.3 f-measure

Para lograr entender la calidad global del sistema recomendador, se combinan recall y presicion para generar el f-measure (ver la ecuación (2.3)). En [41] se estipula que f-measure combina recall y presicion en un único valor para propósitos de comparación. Fue propuesto por Van Rijsbergen como una medida de ponderación que mezcla recall y presicion, y arroja como resultado un valor entre 0 y 1.

$$f - measure = \frac{2 * recall * presicion}{recall + presicion} \qquad (2.3)$$

Para que f-measure obtenga un valor de 1 se requiere que recall y presicion obtengan sus máximos valores 1, lo cual indica que el recomendador siempre sugiere ítems interesantes (exactitud) y que no deja ningún ítem interesante sin recomendar (completitud).

2.3.4.2.3 Medidas de precisión del ranqueo de los ítems

Las medidas de ranqueo son usadas cuando una lista ordenada de recomendaciones se presentan al usuario de acuerdo con sus preferencias. El orden puede ser del ítem más importante o relevante al menos importante o relevante. Se usan medidas de precisión de ranqueo de ítems si se tiene disponible un ranking de referencia (benchmark o punto de referencia). La medida de precisión (exactitud) del ranking puede medirse por Normalized Distance-based Performance Measure (NDPM), la cual estipula la calidad del ranking realizado, y es calculada mediante la ecuación (2.4), de acuerdo con lo estipulado en [44]:

$$ndpm(R_u, R_s) = \frac{((2C^m) + (C^u))}{(2C)} \quad (2.4)$$

Para calcular NDPM es necesario establecer con anterioridad un ranking de referencia (R_u) para ser comparado con el ranking evaluado (R_s). La NDPM arroja un valor comprendido entre 0 y 1. Un valor de 0 indica que el ranking evaluado R_s tiene una calidad de 100%. Es decir que es 100% completo y 100% exacto. Un valor de 1 indica que el ranking evaluado R_s tiene una calidad de 0%, es decir es totalmente incompleto e inexacto.

El cálculo de la NDPM planteado en (2.4) se realiza comparando las distintas parejas existentes en los ranking R_u y R_s. Es importante recordar que un ranking es considerado en [44] como una relación matemática definida como un subconjunto del producto cartesiano OXO, así:

$$R_i = \{(O_x, O_y) / u(O_x) < u(O_y)\} \quad (2.5)$$

Donde O es el CONJUNTO DE RECOMENDACIÓN, y $u(O_i)$ es la función de utilidad explicada en secciones anteriores.

Además, en [44] se plantea una relación de indiferencia asociada con un ranking, calculada de acuerdo a la ecuación (2.6). Esta relación de indiferencia contiene como parejas a OAS cuya función de utilidad es la misma. Es decir un ítem O_i es indiferente respecto de un ítem O_j, si los dos aportan exactamente el mismo nivel de utilidad en la recomendación. Cada conjunto de ítems que tengan asociado el mismo valor de utilidad conforman una clase de equivalencia.

$$Indif(R_i) = \{(O_x, O_y) / (O_x, O_y) \notin R_i \wedge (O_y, O_x) \notin R_i\} \quad (2.6)$$

Donde cada pareja (O_x,O_y) pertenece al producto cartesiano O X O.

En [44] se considera además la definición de ranking converso, generado mediante la ecuación (2.7).

$$R_i^c = \{(O_x, O_y) / (O_y, O_x) \in R_i, donde R_i es un ranking\} \quad (2.7)$$

A partir de la comparación de estos dos ranking, se obtienen los valores de C^m, C^u y C usados en el cálculo de NDPM en (2.4). A continuación se explican cada uno de estos valores.

C^m es el aporte a la exactitud del ranking evaluado R_s, y se conforma por la cantidad de parejas que están en el ranking de referencia R_u y también están en el

ranking converso del ranking evaluado R_s^c (explicado en la ecuación (2.7)). C^m denota la cantidad de parejas que, además de estar mal rankeadas, contradicen el ranking realizado. Es decir, la cantidad de parejas que hacen al ranking R_s inexacto.

C^u es el aporte a la completitud del ranking evaluado R_s. Se conforma por la cantidad de parejas que están en el ranking de referencia R_u y no se encuentran en el ranking evaluado R_s (las parejas que no se encuentran en el ranking evaluado R_s son las que pertenecen a la relación Indif(R_s), definida mediante la ecuación (2.5). Indif(R_s) contiene aquellas parejas de objetos que tienen la misma utilidad. Es decir, que pertenecen a la misma clase de equivalencia). Por lo tanto, C^u determina la cantidad de parejas que hacen al ranking R_s incompleto.

C es la cantidad total de parejas que conforman el ranking de referencia R_u.

2.4 Arquitectura Manejada por Ontologías (ODA)

2.4.1 Ontologías en ingeniería de software

En [45,46] se propone un esquema de clasificación que permite delimitar las ideas referentes al tema. Tal clasificación busca entender cómo se aplican las ontologías en Ingeniería del Software, y cuál es el beneficio en cada caso.

La Arquitectura Manejada por Ontologías (Ontology Driven Architecture, ODA) presentada por W3C, es el punto de partida. En [45] proponen dos dimensiones para lograr una clasificación más precisa. Primero, distinguen el rol de las ontologías en el contexto de ingeniería del software entre uso en tiempo de ejecución y en tiempo de desarrollo. Segundo, miran el tipo de conocimiento que la ontología maneja. Distinguen entre el problema del dominio que el software aborda, y los aspectos de infraestructura que hace el desarrollo de software más conveniente. Basados en estas dos dimensiones en [45] plantean la matriz de la Figura 2.4. en la que se observan cuatro áreas básicas:

Desarrollo manejado por ontologías (Ontology-driven development, ODD) Subsume el uso de ontologías con el tiempo de desarrollo que describe el

problema del dominio. Por ejemplo, la arquitectura MDA propone ontologías para esta área.

Desarrollo habilitado por ontologías (Ontology-enabled development, OED) También usa ontologías en tiempo de desarrollo, pero para soportar las tareas realizadas por los desarrolladores. Por ejemplo, búsqueda de componentes o resolución de problemas de soporte.

Arquitecturas basadas en ontologías (Ontology-based architectures, OBA) Usa las ontologías como artefacto primario en tiempo de ejecución. Acá las ontologías forman parte fundamental de la lógica de la aplicación. Por ejemplo, aplicación con acceso a reglas de negocio.

Arquitecturas habilitadas por ontologías (Ontology-enabled architectures, OEA) Finalmente, las ontologías se toman para proveer infraestructura de soporte en los sistemas de software durante el tiempo de ejecución. Un ejemplo son los servicios web semánticos, donde las ontologías adicionan una capa semántica de mayor nivel que las descripciones clásicas de servicios web. En esa capa se adicionan funcionalidades para el descubrimiento automático de servicios web, emparejamiento y composición automática de servicios, etc.

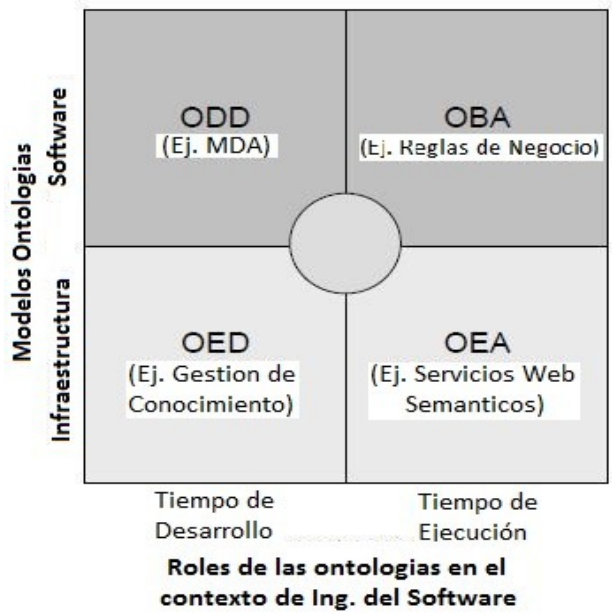

Figura 2.4. Categorías del uso de las ontologías en ingeniería del software, propuestas en [45].

2.4.2 Ingeniería manejada por modelos (Model Driven Engineering)

Según [47], el objetivo de la Ingeniería Manejada por Modelos (MDE) es manejar el desarrollo de software basado en modelado del dominio, más que en implementación y modelos computacionales. En otras palabras, MDE se focaliza en la especificación de sistemas más que en su implementación.

La Arquitectura Manejada por Modelos (Model Driven Architecture, MDA) es un campo de la MDE. De acuerdo a lo contemplado en [48], MDA especifica una metodología para el desarrollo de sistemas software separando la lógica del negocio y la lógica de la aplicación de la plataforma tecnológica subyacente. MDA especifica tres niveles en su arquitectura de acuerdo a lo ilustrado en la Figura 2.5. Tales capas o niveles son:

Representación de Modelo Independiente de la Computación (Computation Independent Model CIM) a partir de un modelo de negocios. CIM describe el contexto en el cual el sistema se usará.
Representación de Modelo Independiente de la Plataforma (Platform Independent Model PIM) a partir de CIM. Acá se describe el sistema en sí mismo, sin considerar los detalles de la plataforma donde sera implementada. Un PIM puede ser satisfecho por uno o varias plataformas arquitectónicas reales.
Representación del Modelo Especifico de la Plataforma (Representation of Platform Specific Model PSM) a partir de PIM. En este nivel se especifican las plataformas de implementación y los lenguajes. En PSM se extiende el modelo PIM, considerando los detalles para implementar el sistema en una plataforma específica.

De acuerdo con [48], MDA permite diseñar una aplicación basada en diferentes mapeos de CIM a PIM y de PIM a PSM. Los mapeos pueden ser automatizados, particularmente cuando un nivel específico del modelo especifica las reglas de mapeo en una metamodelo. En otras palabras, la metodología MDA incrementa la interoperabilidad en ambientes heterogéneos, y provee un método para

integración de sistemas, proponiendo varias capas con modelos de abstracción y procedimientos genéricos.

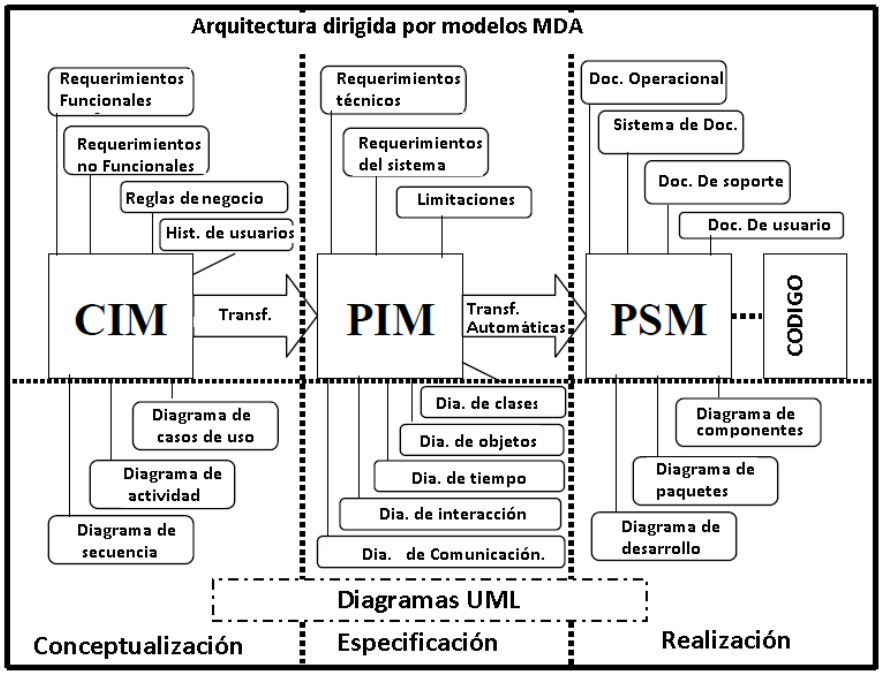

Figura 2.5 Arquitectura dirigida por modelos

2.4.3 Ontology-Driven Architecture (ODA)

El grupo de la W3C Semantic and Web Best Practices and Deployment Working Group (SWBPD) propuso una extensión natural a la metodología o arquitectura MDA que usa ontologías, buscando usar las ventajas de las tecnologías de la web semántica [48]. Esta extensión ha sido definida como Ontology Driven Architecture (ODA).

Según [46], en ODA las ontologías describen las propiedades, relaciones y comportamientos de los componentes requeridos en un proceso de desarrollo de software. Las ontologías son usadas como un modelo conceptual en el desarrollo de componentes de software, tanto en tiempo de diseño como en tiempo de ejecución. Las ontologías son divididas en módulos genéricos, para modelar tanto semántica como metadatos sintácticos.

ODA es planteada como extensión de MDA, para proveer representación de vocabularios de dominios ambiguos (como por ejemplo: requerimientos, restricciones, servicios, propiedades, etc.), chequeo de consistencia de modelos y

validación, así como para habilitar nuevas capacidades automáticas de ingeniería del software. Varios trabajos han sido desarrollados para ilustrar como ODA puede ser usado en diseño, desarrollo y administración de sistemas distribuidos. Por ejemplo, en [48] usan las ontologías para representar conocimiento sobre diversas plataformas, y muestran como esta información es usada para ejecutar configuraciones seguras de refinamiento de transformaciones entre los modelos de las plataformas.

En [49] proponen algunas ideas de cómo las tecnologías de web semántica pueden ser usadas en ODA:

Ingeniería del Software. En esta perspectiva, la idea propuesta es utilizar la Web Semántica en tareas de clasificación o de descripción de mecanismos usados por la Ingeniería del Software. La clasificación describe herramientas y técnicas para modelado semántico, durante la especificación y diseño de estados del software a lo largo de su ciclo de vida. En cuanto a mecanismo, es usada para describir, identificar, descubrir, y almacenar componentes de software, y especificar sistemas usando esos componentes, tanto en tiempo de diseño como en tiempo de ejecución.

Especificación de modelos formales. Esta perspectiva propone el uso de ontologías y tecnologías de web semántica como un medio de comunicación formal entre agentes, humanos o no, que participen en el proceso de desarrollo de software. El uso de las ontologías para la especificación de modelos formales o semi-formales, facilita la comunicación y gestión del conocimiento. Finalmente, las características de las ontologías con su riqueza semántica, in-ambigüedad y estándar de representación leíble por el computador, hacen de las ontologías un punto natural de integración entre Ingeniería del Software y Web Semántica.

Soporte en el ciclo de vida del software. Esta perspectiva propone el uso de las ontologías para describir nombres, metadatos, lenguajes, terminologías y estándares, para la especificación de un dominio durante el ciclo de vida del software.

Contenidos reusables y uso de metadatos como datos relacionales. Esta perspectiva propone el uso de metadatos y ontologías para crear esquemas de datos relacionales y modelos. El uso de metadatos y ontologías es propuesto

como un poderoso descriptor de servicios y componentes, buscando facilitar el descubrimiento de servicios basados en descripciones precisas. Las técnicas de web semántica facilitan el descubrir y usar servicios durante los diferentes estados del ciclo de vida del software.

En [48] afirman que el uso de ontologías en una combinación ODA-MDA, permite obtener importantes beneficios, por ejemplo: gestión de componentes en tiempo de ejecución; descripción de componentes mediante semántica formal basada en lógica; razonamiento y consulta de ontologías que posibiliten verificación; así como chequeo de consistencia de configuración de sistemas llevada a cabo ya sea en tiempo de diseño o en tiempo de ejecución.

2.5 Servicios web semánticos

2.5.1 Web semántica

Según [50], "La web semántica propone superar las limitaciones de la web actual mediante la introducción de descripciones explícitas del significado, y la estructura interna y la estructura global de los contenidos y servicios disponibles en la WWW." Ademas, [50] deja claro que: "Frente a la semántica implícita, el crecimiento caótico de recursos, y la ausencia de una organización clara de la web actual, la web semántica aboga por clasificar, dotar de estructura y anotar los recursos con semántica explícita, procesable por máquinas."

La web semántica usa la noción de ontología, tal como se propone en [51]: "Una ontología es una jerarquía de conceptos con atributos y relaciones, que define una terminología consensuada para definir redes semánticas de unidades de información interrelacionadas."

La Web semántica identifica un conjunto de tecnologías, herramientas y estándares, que forman los bloques constructores básicos de una infraestructura que soporta la visión de la Web asociada con un significado. La arquitectura de la Web semántica está compuesta de una serie de estándares organizados en una cierta estructura, la cual es una expresión de sus interrelaciones. Esta arquitectura

es generalmente representada por la Figura 2.6, y fue propuesta por Tim Berners-Lee bajo el nombre de “The Web semantic layer cake”.

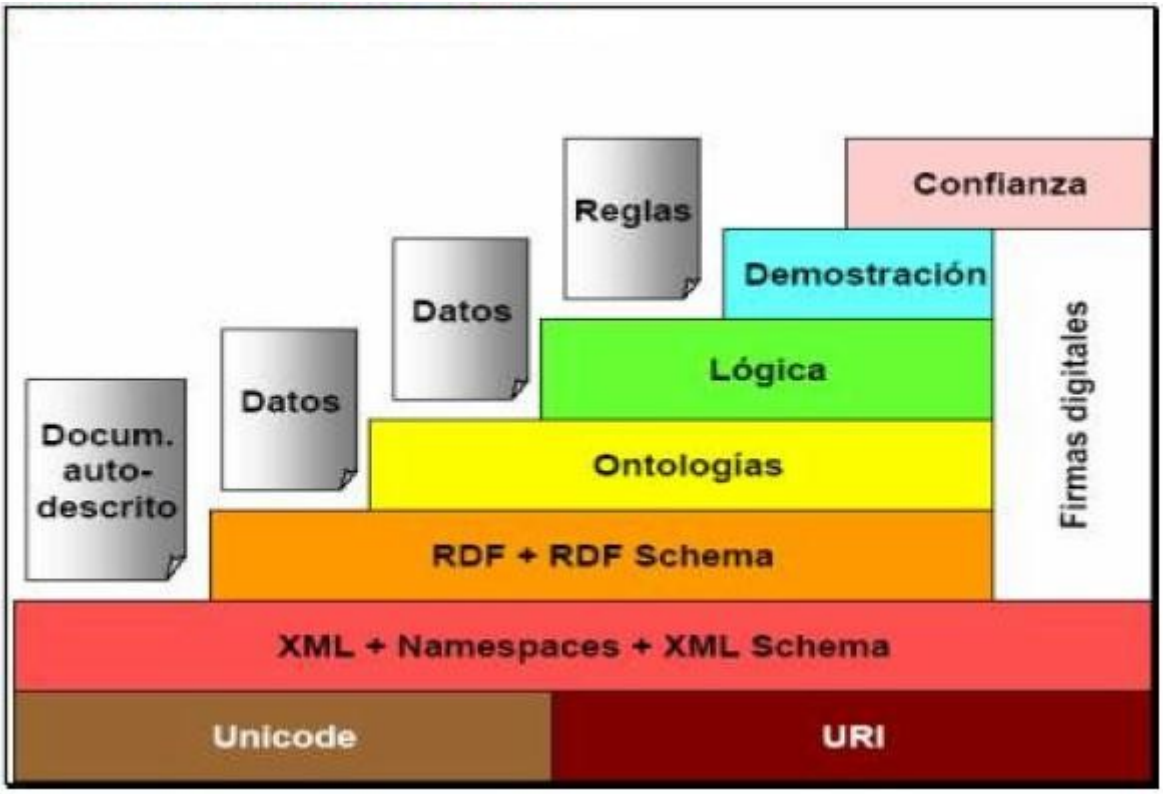

Figura 2.6 Arquitectura de la Web Semántica.

En el contexto de la presente tesis se usa el lenguaje que brinda soporte directo a la web semántica, el Lenguaje Ontológico Web[52] (OWL, por sus siglas en inglés), que se encarga de proporcionar un lenguaje basado en XML para almacenar ontologías usando lógica descriptiva, cuya sintaxis es una extensión del Framework de Descripción de Recursos [53] (RDF, por sus siglas en inglés). Además, en esta tesis se usa el sistema de razonamiento proporcionado por Fact++, descrito en [54]que implementa un demostrador de lógicas descriptivas [55].

2.5.2 Servicios Web

Según [56], un servicio web es un sistema de software diseñado para soportar interacción interoperable máquina a máquina sobre una red. El servicio web tiene una interfaz descrita en un formato procesable por la maquina (normalmente en lenguaje WSDL). Los otros sistemas interactúan con el servicio web en la forma prescrita por estos, usando mensajes SOAP, transmitidos típicamente a través de HTTP.

Una definición precisa de servicio web es la emitida por el W3C [57]: "una aplicación software identificada por una URI, cuyas interfaces y vinculaciones son capaces de ser definidas, descritas y descubiertas como artefactos XML. Un servicio web soporta la interacción con otros agentes de software mediante el intercambio de mensajes basado en XML, a través de protocolos basados en Internet".

Según [58] "Los servicios web permiten a las compañías publicar componentes y servicios en un directorio, en el que otras aplicaciones web pueden buscar e implementar dichos servicios a través de una llamada". La arquitectura de los servicios web se fundamenta en tres estándares principales, y su forma de interacción se plasma en la Figura 2.7. Los servicios web son desarrollados por un proveedor, quien realiza una descripción del servicio y su interfaz mediante un archivo escrito con el Lenguaje de Descripción de Servicios Web [59] (WSDL). Luego de generar el descriptor WSDL, el proveedor del servicio publica dicho WSDL en una Interfaz de Descripción y Descubrimiento Universal [56] (UDDI), que actúa como un directorio que recopila la información necesaria para que los potenciales clientes del servicio realicen el descubrimiento del servicio, y a partir de él la composición del servicio, mediante el intercambio de mensajes basados en el protocolo de Acceso Simple a Objetos [60] (SOAP) entre el proveedor y el cliente.

Figura 2.7 Arquitectura de los servicios Web. Tomada de [58].

2.5.3 Servicios web semánticos

De acuerdo a [61], "Aunque los servicios web permiten la comunicación entre diferentes plataformas y sistemas operativos, carecen de contenido semántico. Teniendo en cuenta que las ontologías permiten la descripción semántica de cualquier dominio del conocimiento, la incorporación de ontologías del dominio de los servicios a los servicios web, da lugar a los servicios web semánticos".

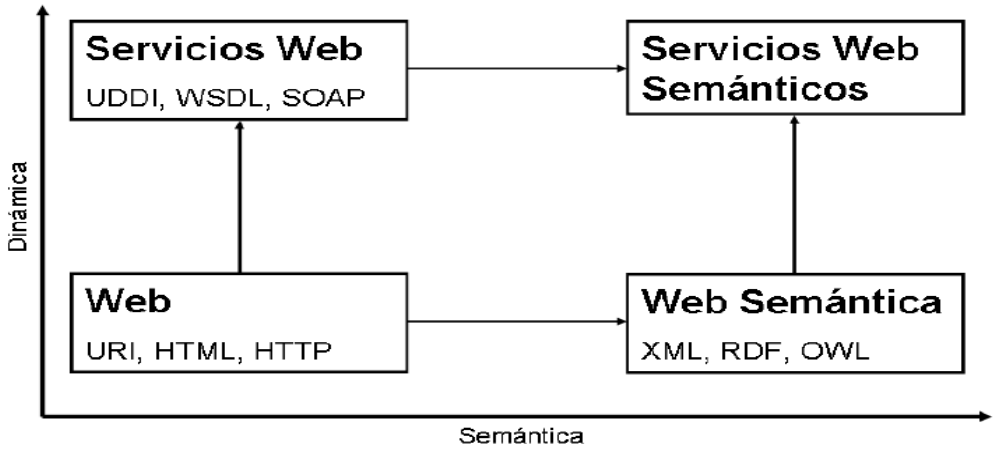

Figura 2.8 Evolución de la Web [62].

La Figura 2.8 mostrada en [62], evidencia como los servicios web semánticos se generaron como la unión entre los servicios web tradicionales y las tecnologías relacionadas con la web semántica. Dotar de semántica a los servicios web permite un aumento en su dinamismo, facilitando la composición dinámica o el propio descubrimiento de los mismos, lo cual ha sido un problema en los servicios web tradicionales, y han requerido de la intervención humana para poder ser llevados a cabo.

Con los servicios web semánticos se pretende automatizar todo lo que era semiautomático en los servicios web tradicionales, esencialmente el descubrimiento, composición, invocación e interoperación de servicios web. Esto coadyuva a lo definido en [62] " transformar la web en una infraestructura global para computación distribuida, integración de aplicaciones y automatización de procesos de negocio".

De acuerdo con [63], "Los servicios web semánticos son una línea importante de la web semántica, que propone describir no sólo información sino definir

ontologías de funcionalidad y procedimientos para describir servicios web: sus entradas y salidas, las condiciones necesarias para que se puedan ejecutar, los efectos que producen, o los pasos a seguir cuando se trata de un servicio compuesto. Estas descripciones procesables por máquinas, permitirían automatizar el descubrimiento, la composición, y la ejecución de servicios, así como la comunicación entre unos y otros.”

3 Framework Basado en ODA para la descripción y composición de Servicios Web Semánticos (FODAS-WS)

En este capitulo se presenta el análisis, diseño, implementación y pruebas de un Framework de especificación de sistemas basado en ODA, para la descripción y composición de Servicios Web Semánticos. El mismo será usado durante la fase de diseño del sistema recomendador de OAs, de la nube de conocimiento del Proyecto Madre presentado en esta tesis.

3.1 Introducción

Las ontologías ayudan a resolver uno de los problemas más importantes en la actualidad en las ciencias computacionales, como es el de interoperabilidad, que aparece claramente en áreas de investigación como: plataformas de trabajo colaborativa, automatización de procesos de negocio, inteligencia artificial, etc. Áreas que son fundamentales en la búsqueda de soluciones de computación, en los que la máquina pueda interactuar de forma autónoma e inteligente con otras maquinas, para configurar procesos complejos en forma automática. Básicamente, su capacidad de representación de conocimiento en un formato entendible por las máquinas, le permite ser un medio para compartir el mismo.

Las ontologías juegan un papel fundamental para resolver el problema de interoperabilidad, ya que son formas estructuradas para describir y formalizar vocabularios necesarios para compartir conceptualizaciones, con lo que contribuyen a resolver la heterogeneidad semántica. Sin embargo, en la actualidad aún existen desafíos para lograr el emparejamiento, la adecuación, y la integración de ontologías, para alcanzar la plena interoperabilidad.

Actualmente se cuenta con infraestructuras de computación distribuida adecuadas para la integración de datos. Los servicios web constituyen los principales representantes de tal infraestructura tecnológica, y se caracterizan por su tecnología abierta, dinámica, y con bajo nivel de acoplamiento. Estas características proveen un buen soporte para compartir recursos y realizar trabajos cooperativos en ambientes heterogéneos. Adicional a esta tecnología, se encuentra la web semántica, que representa una gran revolución en la forma de almacenar y describir conocimiento, a través del uso de ontologías.

En este capítulo se propone un Framework para la descripción de servicios web semánticos SWS basado en ODA, denominado FODAS-WS (Framework basado en ODA para la Descripción y Composición de Servicios Web Semánticos), que pretende servir como instrumento y referente para realizar la generación de código, tanto del servicio web diseñado (esqueleto de implementación del SWS), como la generación completa del cliente del SWS. De esta manera, FODAS-WS es un instrumento fundamental, tanto para el diseño de SWS, como en el descubrimiento, ejecución y composición automática de los mismos. En el ámbito de esta tesis, el Framework ODA es usado en el diseño del servicio web que encapsula el sistema recomendador de OAs de la nube de conocimiento. Este capítulo esta basado en [64].

3.2 Arquitectura del Framework basado en oda para la descripción y composición de servicios web semánticos (FODAS-WS)

El FODAS-WS contiene un conjunto estandarizado de conceptos, prácticas y criterios, para diseñar SWS, el cual sirve como referencia para enfrentar y resolver el problema de diseño, implementación (generación de esqueletos de código del SW), descubrimiento, emparejamiento y composición automática (generación del código del cliente) de SWS. Es necesario aclarar que FODAS-WS también funciona para el descubrimiento, emparejamiento y composición manual tradicional de SWS (tal como se va a usar para el diseño del SWS recomendador de OAs). Pero la verdadera potencialidad radica en que se realice de manera automática. Esto

posibilita la comunicación entre aplicaciones, sin intervención directa de un programador humano en el instante de la reutilización del servicio.

FODAS-WS representa una herramienta para la realización de diseño e implementación de servicios web que usa la arquitectura Ontology-Driven Architecture (ODA), para proporcionar un elevado nivel de definición semántica que soporte procesos de composición automática, reflejados en la generación automática del código de la aplicación cliente.

Además de ser una tecnología de generación automática de código, FODAS-WS es una herramienta que se puede usar durante el proceso de diseño de servicios web, para generar descripciones semánticas completas de la aplicación escritas en lenguaje OWL.

Actualmente, un gran porcentaje de los desarrolladores usan UML para el diseño de aplicaciones, pues su uso representa una forma efectiva de comunicación entre los equipos de desarrollo, especialmente cuando se trata de proyectos complejos. Sin embargo, a pesar de que UML constituye un excelente mecanismo de comunicación entre humanos, no lo es tanto entre maquinas. Por este motivo, FODAS-WS busca contribuir en el proceso de descubrimiento, emparejamiento y composición automática de servicios, a través de modelos ontológicos comprensibles a la maquina, almacenados en tres capas de distinto nivel de abstracción, que se llaman CAPACIM, CAPAPIM y CAPAPSM (ver figura 3.1), las cuales se ajustan completamente a lo propuesto en la arquitectura ODA. Cada uno de los modelos ontológicos que constituyen las distintas capas de FODAS-WS, se almacenan en lenguaje OWL, el cual le aporta no solo sintaxis, sino semántica y métodos formales de análisis y razonamiento sobre el conocimiento expresado.

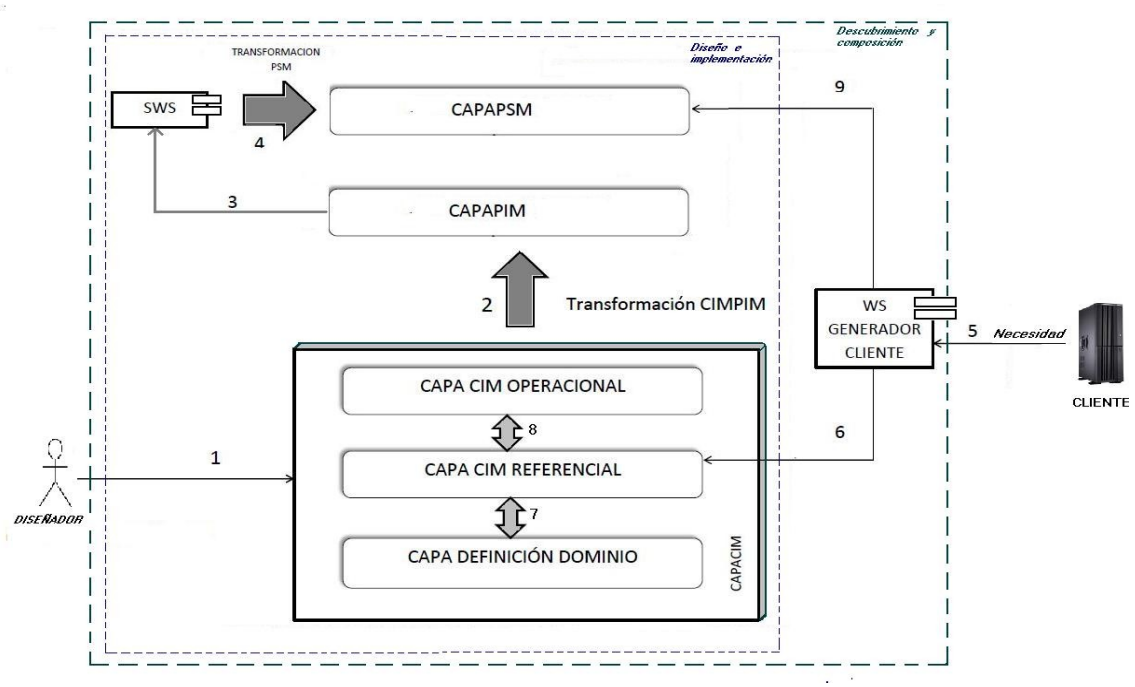

Figura 3.1. Arquitectura de FODAS-WS

FODAS-WS está implementado en java. La Figura 3.1 ilustra la arquitectura propuesta para FODAS-WS. La arquitectura de FODAS-WS posee siete componentes que interactúan entre sí, tales componentes son: CAPACIM, CAPAPIM, CAPAPSM, transformación CIMPIM, transformación PSM, SWS y WS GENERADOR CLIENTE.

CAPACIM: Representa la capa de mayor abstracción en la arquitectura ODA. Se compone de tres modelos ontológicos, en los que se modela el dominio sobre el cual el servicio web opera, y aspectos de diseño relacionados con el proceso que se implementa en el servicio, como por ejemplo: funcionalidades que provee el servicio, descripción del flujo de actividades que conforman el proceso, y descripción de la dinámica e interacciones que se generan entre los objetos participantes en el proceso. Además, está capa se encarga de describir el modelo arquitectónico del servicio implementado, y la definición semántica de las capacidades que ofrece, lo cual sirve como insumo para la realización del proceso de descubrimiento automático y de emparejamiento.

CAPAPIM: Esta capa contiene un modelo ontológico que se genera de forma automática a partir de lo descrito en la CAPACIM, mediante un proceso de transformación. En este nivel de abstracción se plasma lo necesario para generar el esqueleto de código de la implementación del servicio, en el cual se estipulan

los métodos y parámetros que el servicio requiere, así como la secuencia de llamadas de los distintos métodos. La generación automática del esqueleto de código del servicio, además de disminuir el trabajo de implementación, garantiza total compatibilidad entre lo implementado con lo diseñado, de tal forma que la composición y ejecución automática se realice con total compatibilidad.

CAPAPSM: Es la capa más especifica con que cuenta FODAS-WS. En ella se estipula lo necesario para realizar la composición y ejecución automática de los servicios. Esta capa describe semánticamente lo estipulado en un archivo WSDL, fusionado con lo contemplado en la especificación OWL-S, de reconocido uso para la descripción de servicios web semánticos. Esta capa se genera de forma automática, una vez implementado y desplegado el servicio. Con ello se busca que la información registrada en lo asociado con PSM, sea totalmente compatible con los detalles de implementación. Es importante resaltar que el hecho de generar este ultimo artefacto de diseño, después de haber sido implementado el servicio, no entra en contradicción alguna, pues esta capa es utilizada en procesos de composición de servicios que se llevan a cabo después de que el servicio este correctamente implementado y desplegado.

TRANSFORMACION CIMPIM: Es un proceso implementado como una funcionalidad de FODAS-WS, que permite realizar la generación automática de la CAPAPIM a partir de lo descrito semánticamente en la CAPACIM.

TRANSFORMACION PSM: Es un proceso implementado como una funcionalidad de FODAS-WS, que permite realizar la generación automática de la CAPAPSM a partir de lo descrito semánticamente en la CAPAPIM. Es importante resaltar que dicho proceso de transformación se realiza después de implementado y desplegado el servicio web.

SWS: Es el servicio web semántico diseñado, implementado y desplegado. Tanto su diseño como parte de la implementación del SWS, se realiza usando FODAS-WS.

WS GENERADOR CLIENTE: Es un servicio web ofrecido a cualquier aplicación. Este servicio requiere como entradas que la aplicación solicitante suministre sus necesidades expresadas en forma semántica, y de acuerdo con las políticas semánticas de FODAS-WS. WS GENERADOR CLIENTE se encarga de realizar descubrimiento automático de los distintos servicios web semánticos diseñados usando FODAS-WS, que emparejen con las necesidades solicitadas por la aplicación. Una vez descubierto todos los servicios que emparejan, el WS GENERADOR CLIENTE selecciona el servicio que mejor empareja. Posteriormente, genera código automático que implementa un cliente para el SW seleccionado, realizando además, la ejecución automática del servicio, y retornando a la aplicación solicitante los resultados, en el formato que se infirió a partir de las necesidades descritas semánticamente por la aplicación. Es importante señalar que en la actualidad, el WS GENERADOR CLIENTE no ha sido implementado, y que se propone como trabajo futuro a realizar.

En adelante se presenta la secuencia de interacciones que se genera entre los distintos componentes que intervienen en la arquitectura de FODAS-WS:

1. El diseñador describe semánticamente la CAPACIM, en la cual contempla todo lo necesario para la especificación total del servicio.
2. La transformación CIMPIM genera la CAPAPIM automáticamente, a partir de la CAPACIM.
3. Una vez generada la CAPAPIM, FODAS-WS genera el código automático que contiene el esqueleto del SWS.
4. Al momento en que se encuentre implementado y desplegado el SWS, la transformación PSM se encarga de generar de forma automática la CAPAPSM.

La secuencia de interacciones descritas hasta este instante cubren todos los aspectos de diseño necesarios para especificar un SWS (Son estos los pasos que se mostrarán en el capitulo 4, para diseñar el SWS que encapsula el sistema recomendador de OAs). A continuación se describen el resto de interacciones que se dan cuando se pretende hacer descubrimiento, composición y ejecución automática de servicios web:

1. Una aplicación cliente solicita a WS GENERADOR CLIENTE que le realice el descubrimiento, composición y ejecución automática de un servicio web que

satisfaga las necesidades descritas semánticamente, y le retorne los resultados deseados.

2. WS GENERADOR CLIENTE acude a la CAPA CIM REFERENCIAL para, a través de ella obtener todo el conocimiento necesario, para realizar el proceso de emparejamiento entre las capacidades ofrecidas por los servicios y las necesidades solicitadas por la aplicación cliente.
3. La capa CIM REFERENCIAL accesa a la CAPA DE DEFINICIÓN DE DOMINIO, para realizar todo el proceso de inferencia que se requiera como insumo, en el momento de razonar sobre el diseño del SWS almacenado en la CAPA CIM OPERACIONAL.
4. La capa CIM REFERENCIAL accesa a la CAPA CIM OPERACIONAL, para realizar el proceso de descubrimiento automático, usando como insumos las inferencias realizadas en el paso 7.
5. WS GENERADOR CLIENTE acude a la CAPA PSM, para obtener la información requerida para realizar la composición y ejecución automática, usando como insumos las inferencias realizadas en los pasos 7 y 8. Una vez ejecutado el SWS, el WS GENERADOR CLIENTE retorna a la aplicación cliente el resultado en el formato adecuado, de acuerdo a lo inferido de las necesidades solicitadas.

3.3 Diseño, implementación y pruebas de FODAS-WS

Para explicar el diseño y la implementación de FODAS-WS, vamos a usar un caso de estudio.

3.3.1 Descripción del caso de estudio

El caso de estudio consiste en el diseño e implementación de un SWS cuya capacidad es la de un sistema recomendador como los descritos en la sección 2.3. El sistema recomendador del caso de estudio está en capacidad de recomendar dos tipos de productos: OAs y documentos web. Para ello, los metadatos se encuentran almacenados en repositorios semánticos de OAs y en repositorios semánticos de archivos web, respectivamente.

La recomendación consiste en la generación de un conjunto, ya sea de objetos o de documentos, a recomendar, que se ajusten a las exigencias del usuario que lo solicitó. El recomendador posee criterios de búsqueda con lo cual genera un conjunto de productos a recomendar, a lo que se le llama CONJUNTO DE RECOMENDACIÓN. A cada uno de los miembros del CONJUNTO DE RECOMENDACIÓN, el sistema recomendador le calcula un valor de emparejamiento VE, parámetro que le sirve como criterio de priorización en el momento de recomendar (el sistema recomendador será presentado en detalle en el siguiente capítulo).

3.3.2 Diseño e implementación de la capa CIM

Como ya se mencionó, la CAPACIM es la capa de mayor abstracción dentro de FODAS-WS. Se compone de tres subcapas, cada una asociada con un modelo ontológico. Las capas se conocen como: CAPA DE DEFINICIÓN DEL DOMINIO, en la que se modela el dominio sobre el cual el servicio web opera; CAPA CIM OPERACIONAL, en la cual se definen los aspectos de diseño relacionados con el proceso que se implementa en el servicio, como por ejemplo: funcionalidades que provee el servicio, descripción del flujo de actividades que conforman el proceso, entre otras cosas; y una tercera capa, denominada CAPA CIM REFERENCIAL, que representa la capa central dentro de CIM, que se encarga de describir el modelo arquitectónico del servicio implementado, y de definir semánticamente las capacidades que ofrece, lo cual sirve como insumo para la realización del proceso de descubrimiento automático y de emparejamiento.

3.3.2.1 Capa de definición del dominio

Dentro de la CAPA DE DEFINICIÓN DEL DOMINIO de FODAS-WS, se proponen algunas definiciones:

- DOMINIO: conjunto de ELEMENTOS que, mediante ASOCIACIONES, se relacionan entre ellos a través de la ejecución de ACCIONES.
- DEFINICIÓN SEMÁNTICA DEL DOMINIO: conjunto de ASOCIACIONES, acompañado de una DEFINICIÓN SEMÁNTICA DE LAS ACCIONES, y una

DEFINICIÓN SEMÁNTICA DE LOS ELEMENTOS.

- DEFINICIÓN SEMÁNTICA DE LAS ACCIONES: establece las relaciones semánticas entre las distintas ACCIONES que existen. Se cuentan con ocho tipos de relaciones posibles entre las acciones, tal como se muestra en la Tabla 3.1.

RELACIÓN	REL. INVERSA	DESCRIPCIÓN
complementaA	complementadaPor	Una acción complementa a otra cuando al realizarla logra realizar tareas complementarias. Es decir, realiza aspectos que la otra tarea no lograba realizar
esSubAccion	esSuperAccion	Establece una jerarquía de acciones
especializaA	esEspecializadaPor	Determina cuando una acción genérica se convierte en dicha acción realizada para un escenario especifico. Por ello se asocia mediante el rol *usadoEn,* para determinar en que tipo de elemento se realiza la acción.
contrarioCon		Establece que acción realiza justo lo contrario que otra
genera		Determina cuando la realización de una acción deriva o genera otras acciones
semejanteA		Establece cuando dos acciones realizan tareas similares
profundizaA	esProfundizadoPor	Indica cuando una acción tiende a realizar lo mismo que otra pero en menor grado de profundidad.

Tabla 3.1. Relaciones existentes entre acciones.

- DEFINICIÓN SEMÁNTICA DE ELEMENTOS: determina la jerarquía entre los elementos existentes, y puntualiza los distintos elementos miembros de otro elemento, tal como se muestra en la Tabla 3.2.

RELACIÓN	REL. INVERSA	DESCRIPCIÓN
esSubElemento	esSuperElemento	Determina la jerarquía entre elementos
tieneMiembro	esMiembroDe	Define cuales elementos son miembros de cierto elemento, que los agrupa como conjunto con características similares.

Tabla 3.2. Definición semántica de elementos

- ASOCIACIÓN: describe como un ELEMENTO al ejecutar una ACCIÓN, genera otro elemento. Para describir una ASOCIACIÓN se usan tres relaciones (ver Tabla 3.3).

RELACIÓN	REL. INVERSA	DESCRIPCIÓN
tieneAntecedente	esAntecedenteDe	Esta relación determina cual es el elemento que ejecuta la acción dentro de la asociación definida.
tieneAccion	esAccionDe	Esta relación determina cual es la acción ejecutada dentro de la asociación definida
tieneConsecuente	esConsecuenteDe	Esta relación determina cual es el elemento que se genera al ejecutar la acción

Tabla 3.3. Asociaciones entre acciones

En esta capa, además de definirse ASOCIACIONES, ELEMENTOS y ACCIONES, también se especifican especializaciones de acciones, mediante las cuales se logra definir semánticamente los roles de cada elemento y las formas en que son usadas cada acción.

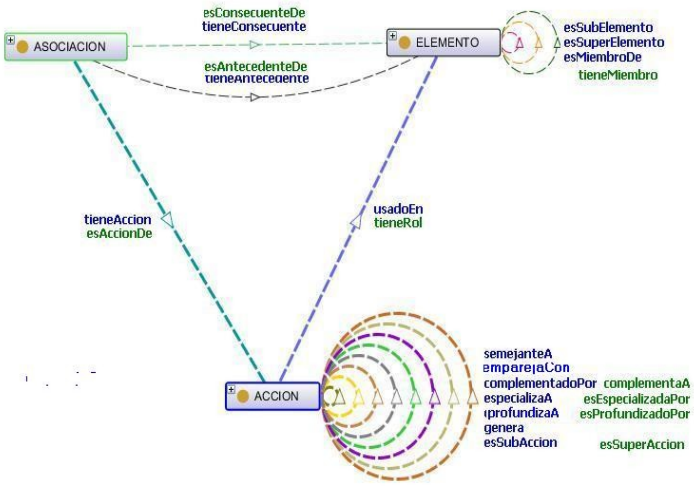

Figura 3.3 Modelo ontológico de la capa de definición del dominio.

En la Figura 3.3 se presenta el modelo ontológico usado para realizar la DEFINICIÓN SEMÁNTICA DEL DOMINIO. A continuación se muestra el modelado del dominio para el caso de estudio.

3.3.2.1.1 Descripción del dominio de los sistemas recomendadores

La Figura 3.4 muestra una representación gráfica de la definición semántica del domino de recomendación para dos tipos de PRODUCTOS, que son: OAs y documentos web. Los metadatos de cada uno de estos dos tipos de productos a recomendar, se encuentran almacenados en un repositorio semántico, conocidos como *repositorioOA* y *repositorioWeb*. Los óvalos en la Figura 3.4 representan los elementos definidos en el dominio, cada arco que une un par de óvalos representa:

- Una asociación, considerada en la forma en que se explicó antes (en el caso en que la etiqueta del arco tenga asociado un número). Cada asociación puede verse como una relación entre dos elementos que se asociación a través de una acción, identificada con la etiqueta escrita en letras mayúsculas que acompaña a cada arco usado como asociación. Por ejemplo, la asociación 1 determina que un elemento *RecOA* (Recomendador de OAs) ejecuta la acción consultar en *RepositorioOA* (Repositorio de OAs).
- Una Definición semántica de elementos (en el caso que el arco que une los óvalos no tenga asociado un número, y se encuentre acompañado de una flecha azul en su parte posterior).

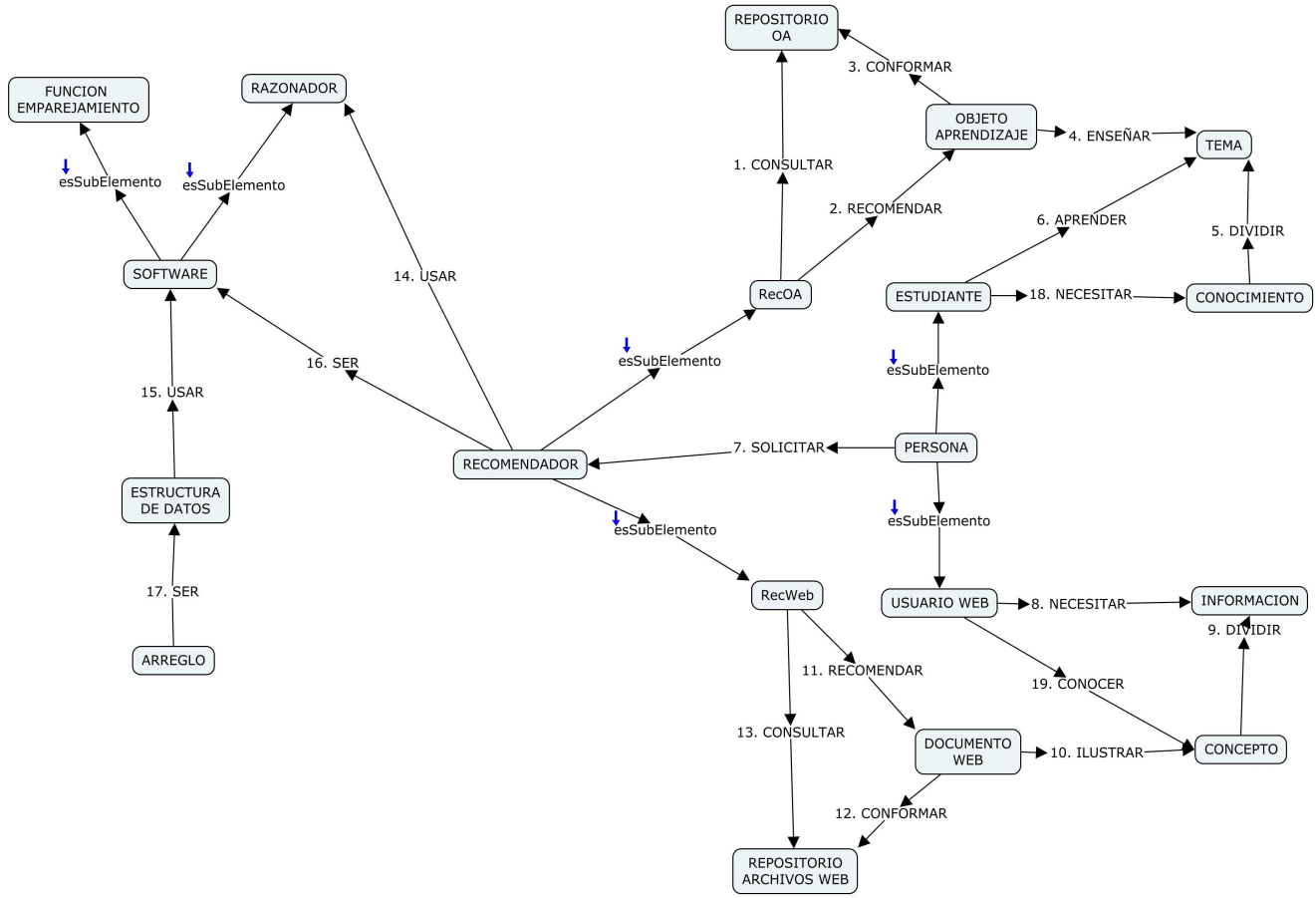

Figura 3.4. Definición semántica del dominio.

La Figura 3.5 ilustra la forma ontológica usando Protege en que se realiza la definición semántica de los elementos del dominio. Por ejemplo, se relaciona el elemento persona con el elemento estudiante mediante el object Property *esSubElemento,* con lo cual el sistema de inferencia logra entender que un estudiante es un tipo de persona con ciertas características, que lo hacen distinto al común de las personas, pero que toma todos los aspectos genéricos que cualquier persona tiene.

La Tabla 3.4 muestra un extracto de la definición semántica de las acciones que intervienen en el dominio del caso de estudio ilustrado en este capitulo. Para realizar la definición semántica de acciones se hace uso de lo explicado en la Tabla 3.1. En el dominio del caso de uso acá presentado, se utilizan las siguientes acciones: anteceder, aprender, buscar, calcular, conformar, consultar, consultarOA, consultarWeb, conocer, contener, devolver, dividir, enseñar, entregar, estudiar, ilustrar, informar, necesitar, ordenar, priorizar, recomendar, requerir, seguir, ser, sugerir, tener y usar. La definición semántica de acciones no solo comprende definir las acciones que se realizan en el dominio especificado, sino que requiere además definir las relaciones entre las acciones existentes. La Tabla 3.4 especifica

en la primera columna la relación semántica que se presenta entre cada una de las parejas de acciones especificadas en la segunda columna de la misma tabla.

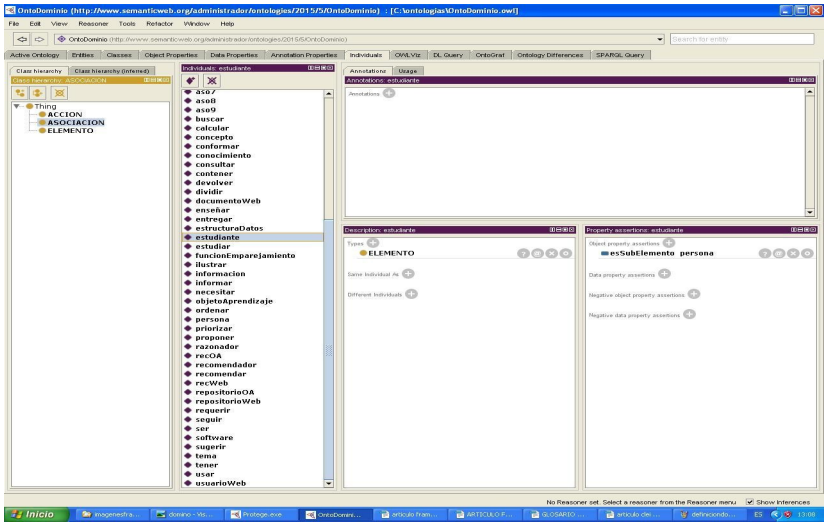

Figura 3.5. Modelo ontológico de los elementos del dominio del caso de estudio, usando Protege

Relaciones entre Acciones en el Domino	**Parejas de acciones relacionadas en el dominio**
semejanteA	aprender- conocer
	Recomendar - sugerir
	Informar - ilustrar
	Requerir - necesitar
	Ordenar - priorizar
emparejaCon	Anteceder - seguir
	Recibir - entregar
	Enseñar- aprender
	Solicitar-Recomendar
complementaA	Buscar - recomendar
	Consultar - recomendar
profundizaA	Recomendar - buscar
	Aprender - conocer
	Enseñar - ilustrar
genera	Recomendar - calcular
	Recomendar - ordenar
	Consultar - recomendar
especializaA	ConsultarOA - consultar
	ConsultarWeb - consultar

TABLA 3.4. DEFINICIÓN DE ACCIONES PARA EL DOMINIO DE RECOMENDADORES.

La Figura 3.6 ilustra el modelo ontológico de la definición de acciones, tomando como ejemplo la acción *recomendar*, que como lo estipula la Tabla 3.4, se encuentra relacionada con ocho acciones más mediante los objectProperty *genera, SemejanteA, emparejaCon y profundizaA*. Como se explicó al inició de esta sección, el hecho de que la acción *recomendar* se encuentre asociada mediante el objectProperty *genera* con las acciones *calcular* y *ordenar*, lo que le indica al sistema de razonamiento es que cuando se realice la acción r*ecomendar* ella

desencadena la ejecución de dos acciones más, que son calcular y ordenar. El hecho de que la acción recomendar se relacione con la acción *buscar* mediante *profundizaA,* le dice al sistema de razonamiento que recomendar es un tipo de acción de búsqueda, pero más especializada. De la misma manera, se puede hacer la descripción semántica del resto de relaciones entre la acción *recomendar* con sus pares. Este tipo de definiciones le proporcionan a la máquina significados, que le permiten realizar las labores de descubrimiento, emparejamiento, composición y ejecución automática. Por ejemplo; si lo que le solicita una aplicación al WS GENERADOR CLIENTE es la acción *sugerir,* el sistema de razonamiento sabe que se trata de la misma acción *recomendar,* y que es un tipo de búsqueda que da origen a las acciones *calcular* y *ordenar*.

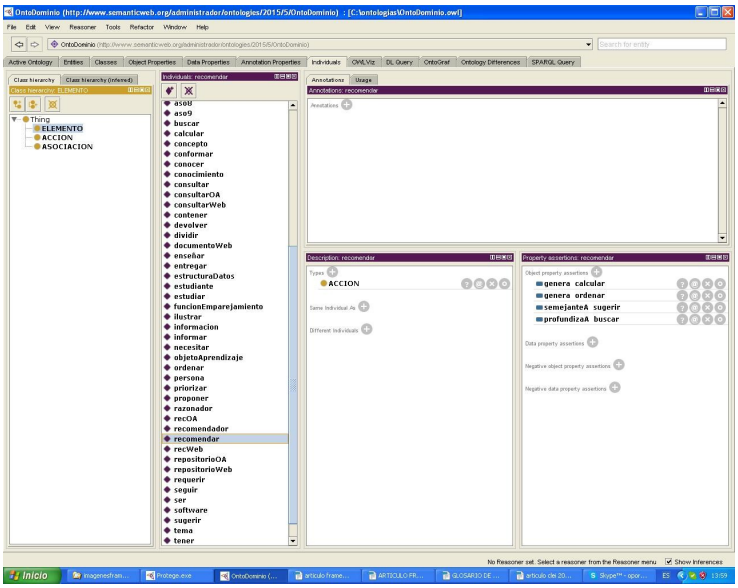

Figura 3.6. Modelo ontológico de las acciones del dominio del caso de estudio, usando Protege

Como se explicó al inicio de la sección, una *ASOCIACIÓN* se encarga de determinar como dos elementos se vinculan (asocian) mediante una acción. En la Figura 3.4 se observa que la capa de definición semántica de dominio para el caso de estudio, cuenta con 19 asociaciones . Dentro del modelo ontológico se utilizan tres objectProperty para especificar cada asociación, que son (ver tabla 3.3): *tieneConsecuente, tieneAntecedente y tieneAccion*.

En la Figura 3.7 se observa como se realiza la especificación de la *ASOCIACIÓN* aso1 (llamada así para vincularla con la ASOCIACIÓN representada con el arco cuya etiqueta tiene el número 1 en la Figura 3.4). aso1 se relaciona mediante el objectProperty *tieneConsecuente* con el elemento repositorioOA, mediante *tieneAntecedente* con el elemento recOA, y mediante el objectProperty

tieneAccion con la acción consultar. Eso indica que un recomendador de OAs recOA realiza la acción de consultar a un repositorio de OAs repositorioOA.

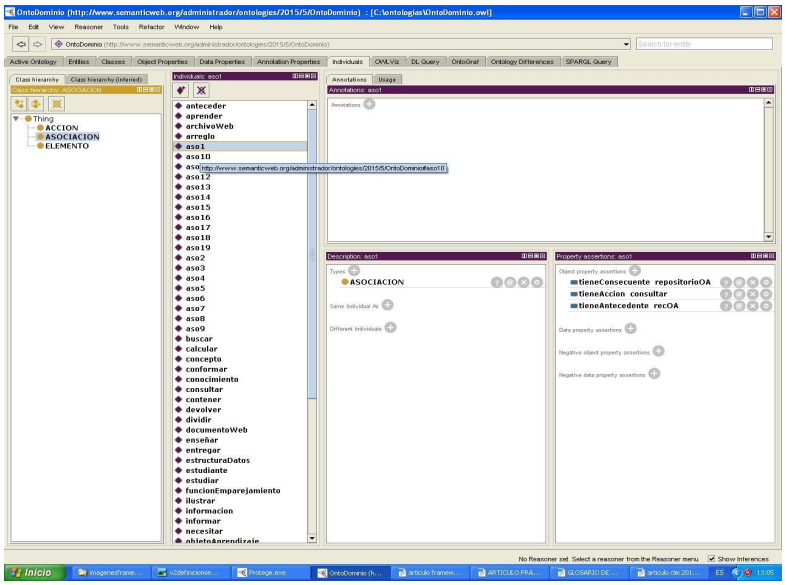

Figura 3.7. Modelo ontológico de las ASOCIACIONES del dominio para el caso de estudio, usando Protege

Es importante resaltar que la ASOCIACIÓN es fundamental, porque es en torno a ella que se da el proceso de descripción de las capacidades, los efectos, las necesidades, las precondiciones y los estados. Cada uno de ellos son definidos por un conjunto de una o más asociaciones, que se usarán en los procesos de descubrimiento y emparejamiento automático.

3.3.2.2 Diseño e implementación de la capa CIM REFERENCIAL

Es la capa en la que se contemplan los distintos referentes teóricos arquitectónicos y conceptuales a los que se ajusta el servicio web diseñado. Esta subcapa pretende aportar información relacionada con la forma en que está diseñado e implementado el servicio web, y las arquitecturas a las que se ajusta. Tal conocimiento se presenta para indicar a la máquina el modelo arquitectónico que se sigue, para que así ella use tal información en el momento de realizar emparejamiento con las necesidades arquitectónicas que tenga. Como se explicó en la sección 3, La CAPA CIM REFERENCIAL es a la que el WS GENERADOR CLIENTE acude, para a través de ella, obtener todo el conocimiento necesario para realizar el proceso de emparejamiento entre las capacidades ofrecidas por los servicios y las necesidades solicitadas por la aplicación cliente. Esta capa sirve de

enlace para que el sistema de razonamiento acuda a las capas CIM OPERACIONAL Y DE DEFINICIÓN DEL DOMINIO, para realizar las inferencias necesarias durante el proceso de descubrimiento y de emparejamiento automático.

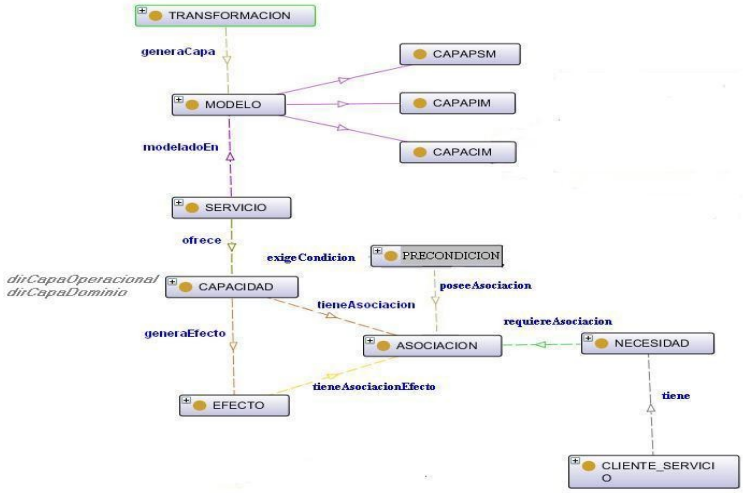

Figura 3.8. Modelo ontológico de la CAPA CIM REFERENCIAL

En la Figura 3.8 se muestra el modelo ontológico asociado a la subcapa CIMReferencial. Como se aprecia, en dicho modelo ontológico se describe que los servicios web descritos se ajustan en lo estructural a lo contemplado en la arquitectura SOA. Para ello, el modelo ontológico describe como un cliente tiene necesidades, las cuales son emparejadas mediante métodos de emparejamiento con las capacidades ofrecidas por el proveedor de un servicio. Además, se describe que tales capacidades generan efectos, los cuales consisten en asociaciones ya determinadas en la definición semántica del domino. También, el modelo ontológico describe las necesidades del cliente, descritas mediante asociaciones ya plasmadas en la definición del dominio. En esta subcapa se describe también la forma en que el servicio se diseñó, dejando explicito que se realiza mediante modelos, algunos de los cuales pueden ser obtenidos a través de transformaciones. Se especifica, además, que se cuentan con tres tipos de modelos, que son: CapaCIM, CapaPIM y CapaPSM. Es importante resaltar que el concepto CAPACIDAD cuenta con un *dataProperty dirCapaDominio,* en el que se registra el IRI de la ontología de dominio. De igual manera, se tiene el dataProperty *dirCapaOperacional,* en el que se describe lo relativo al diseño del SWS.

Con esta capa, durante el proceso de descubrimiento automático, composición automática y ejecución automática, el sistema de inferencia logra realizar un primer emparejamiento semántico considerando las precondiciones y efectos de cada capacidad, y las necesidades solicitadas por el cliente. Es la capa CIMREFERENCIAL la capa de mayor jerarquía dentro de las tres subcapas componentes de la capa CIM, pues a partir de ella se acceden las otras capas, representadas como modelos ontológicos.

En la Tabla 3.5 se ilustra la especificación de capacidades, precondiciones y efectos para el SWS del caso de prueba. También, se muestran la necesidades descritas semánticamente por la aplicación que invoca a WS GENERADOR CLIENTE. Como se aprecia en la Tabla 3.4, la acción *entregar* empareja con la acción *recibir*. Eso hace que el sistema de razonamiento infiera que la capacidad del SWS empareja con la necesidad del estudiante, pues lo que el estudiante necesita recibir (recomendacionOA) del SWS está en la capacidad de entregar. Por otro lado, las PRECONDICIONES son satisfechas, pues el *recOA* necesita como precondición saber que tema debe enseñar, que es el mismo tema que el estudiante manifiesta necesidad de aprender. Como el sistema de inferencias determina que enseñar y aprender emparejan (ver Tabla 3.4), el WS GENERADOR CLIENTE determina a través de su modulo de razonamiento que esta precondición es satisfecha. Vemos así que esta capa es como un meta-modelo que asocia a las otras ontologías para realizar tareas automáticas.

	tieneAntecedente	***tieneAccion***	***tieneConsecuente***
CAPACIDAD	recOA	entregar	recomendacionOA
PRECONDICION	recOA	recibir	recomendacionOA
	recOA	enseñar	tema
EFECTO	estudiante	recibir	recomendacionOA
NECESIDAD	estudiante	recibir	recomendacionOA
	estudiante	aprender	tema

Tabla 3.5. Definición de capacidades, precondiciones y efectos en CAPA CIM REFERENCIAL

3.3.2.3 Diseño e implementación de la capa CIM OPERACIONAL

Como ya se explicó, la arquitectura FODAS-WS cuenta con la CAPACIMOPERACIONAL, en la que se realiza una descripción funcional del servicio a implementar, la cual debe estar ajustada a lo definido en la capa de definición

semántica del dominio. En la CAPA CIMOPERACIONAL se describe no solo las funcionalidades del proceso, sino que además describe el proceso que se implementa y ejecuta en el servicio web, y la dinámica de interacción que surge en el momento de su ejecución.

El modelo, como ya se dijo anteriormente, especifica tres aspectos fundamentales del proceso implementado en el servicio web, que son:

- Funcionalidad (similar a lo descrito en los diagramas de caso de uso de UML), este aspecto es de gran utilidad en el momento de realizar descubrimiento y emparejamiento automático.
- Descripción del flujo del proceso (similar a lo descrito en los diagramas de actividad de UML), este aspecto es uno de los insumos a usar para realizar composición automática.
- Dinámica de interacción entre los objetos participantes (similar a lo descrito en los diagramas de secuencia de UML), este aspecto es el pilar central usado en el proceso de transformación entre la capa CIM y la capa PIM.

La Figura 3.9 muestra el modelo ontológico que soporta la capaCIMOperacional, el concepto central es CAPACIDAD, y este se relaciona con tres conceptos fundamentales que son CASOUSO (funcionalidad), DIAGRAMAACTIVIDADES (flujo del proceso) y DIAGRAMADESECUENCIA (dinámica de interacción).

Se requiere de la implementación de un analizador o validador entre capas, que garantice que los elementos, acciones, asociaciones, estados, precondiciones, efectos, parámetros, operaciones, actividades y demás actores que intervienen en las tres capas pertenecientes a la capa CIM, sean congruentes y estén definidos correcta y completamente. Ese es otro trabajo futuro.

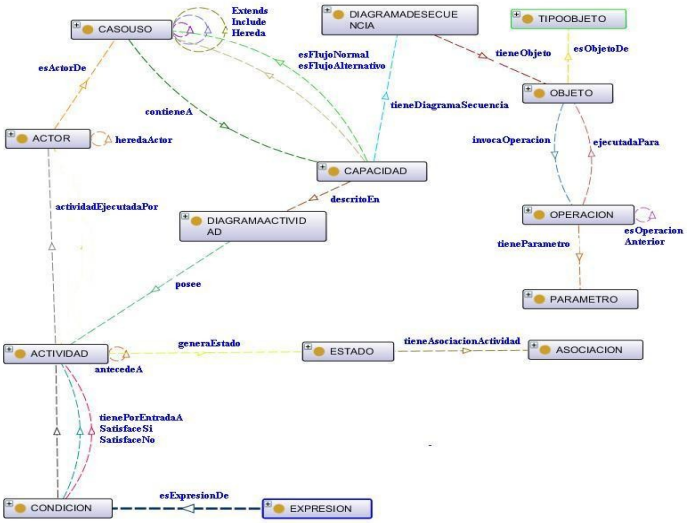

Figura 3.9. Modelo ontológico de la capa CIM OPERACIONAL

En torno al concepto CASODEUSO se describen las distintas funcionalidades, contemplando cuales se incluyen dentro de otras, cuales se extienden, y cuales se heredan, así como los distintos actores que participen en ellas. El concepto DIAGRAMAACTIVIDAD es el encargado de organizar lo referente a la descripción del flujo de las actividades que componen el proceso, por eso, mediante el objectProperty *posee* se establece que individuos del concepto ACTIVIDAD están asociadas a cada capacidad, y usando una serie de objectproperty como *satisfaceSi, tienePorEntradaA, satisfaceNo,* se especifica el orden en que las actividades se dan, y las condiciones que deben cumplir para seguir cierto flujo, así como los distintos estados por los que el sistema pasa de acuerdo a las actividades que vaya ejecutando. Mediante el concepto DIAGRAMADESECUENCIA se agrupa la descripción de los distintos objetos que interactúan a lo largo de la ejecución del proceso, especificando mediante el rol *esObjetoDe* el tipo de objeto de cada uno de ellos. De esta manera, para cada objeto participante, se estipula en esta capa que operaciones invoca, y cuales son los parámetros asociados a dichas operaciones, así como el orden entre ellas.

La capa CIMOPERACIONAL sirve de soporte para la generación automática de la capa PIM. Pero además, la capa CIMOPERACIONAL registra conocimiento fundamental para realizar descubrimiento automático a través del conocimiento referente con casos de uso, para realizar composición y ejecución automática haciendo uso del diagrama de actividades, y para describir los mecanismos de implementación usando el conocimiento asociado con el diagrama de secuencias.

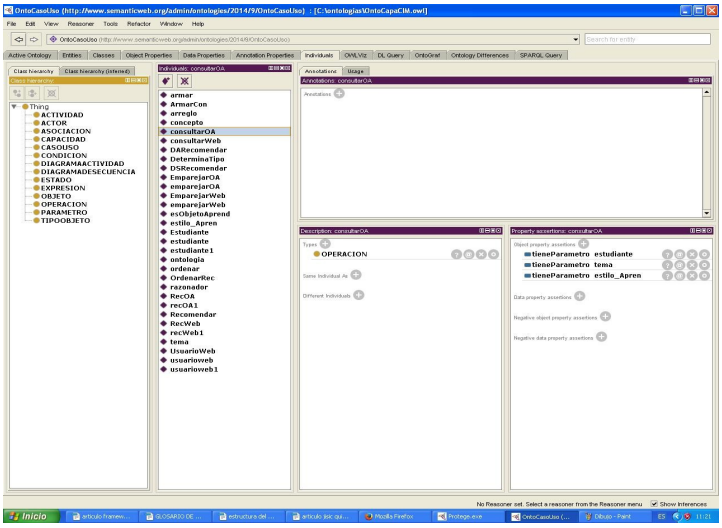

Figura 3.10. definición de la CAPA CIM OPERACIONAL

La Figura 3.10 muestra como se diseña el SWS recomendador mediante la CAPA CIM OPERACIONAL del Framework FODAS-WS. Específicamente, la Figura 3.10 muestra la forma en que se describe la operación *consultarOA,* que como se aprecia en la Tabla 3.4, es un tipo de acción definida en la capa de definición del dominio. Mediante el objectProperty *tieneParametro* se incorpora conocimiento que determina que la operación *consultarOA* tiene los parámetros *estudiante, tema* y *estilo_Apren.*

Por otro lado, cada uno de los objetos que intervienen en la realización del proceso llevado a cabo para la ejecución de una capacidad, debe ser especificado en el diseño del SWS. La Figura 3.11 muestra como se describe el objeto recOA1, indicando mediante el objectProperty *invocaOperación* que dicho objeto es a quien se le invocan las operaciones *armar, ordenar* y *emparejarOA.*

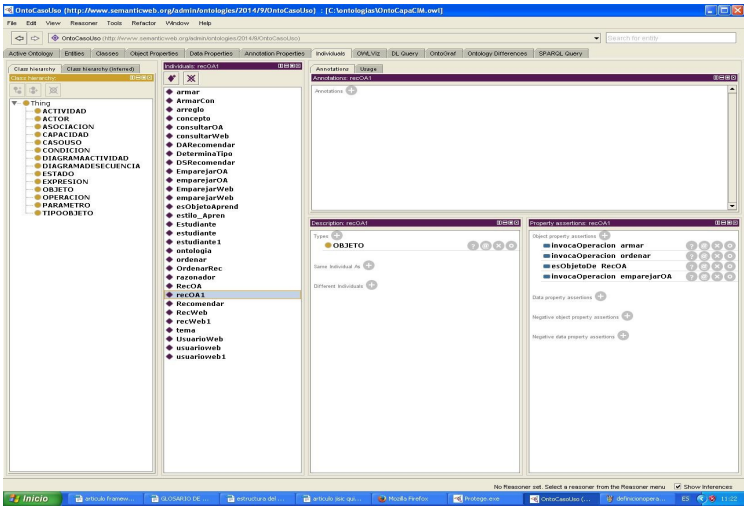

Figura 3.11. Ejemplo de especificación de un objeto que forma parte del SWS recomendador usando la capa CIM OPERACIONAL

3.3.3 Diseño e implementación de la capa PIM

Es el segundo nivel de abstracción en FODAS-WS, en él se busca expresar en forma abstracta los pilares fundamentales y la estructura a usar en la implementación del SWS, sin entrar en detalles particulares de tal implementación. Es un nivel un poco mas detallado del sistema, pero desde un punto de vista independiente de la plataforma. Contiene detalles sobre la funcionalidad del sistema y de su comportamiento, pero sin hacer referencia a las plataformas sobre las que puede ser implementado. A partir de esta capa, se genera el esqueleto de código del servicio que, además de facilitar la labor de implementación, garantiza total compatibilidad entre lo diseñado y lo implementado.

En la Figura 3.12 se aprecia el modelo ontológico asociado a la capa PIM. En tal modelo se plasma una descripción semántica muy ajustada a un diagrama de clases de UML, en ella, el concepto central es CLASE, mediante el objectProperty *tieneVariable* se especifica que variables tiene una clase, las cuales son individuos del concepto VARIABLE. De igual forma, el objectProperty *tieneMetodo* indica que métodos pertenecen a cada clase. Entre los conceptos METODO y VARIABLE se tiene un objectProperty llamado *parametroDe,* el cual estipula que parámetros tiene cada uno de los métodos. Además, el modelo ontológico cuenta con un objectProperty llamado *esSubClaseDe,* que tiene como dominio y como rango al concepto CLASE, mediante este rol se describen las distintas jerarquías de clase existentes. Como ya se mencionó, esta capa representa los conceptos abstractos que requiere un programador en el momento de implementar el servicio. Es muy importante resaltar que FODAS-WS define en este modelo ontológico el diagrama de clases del servicio a implementar.

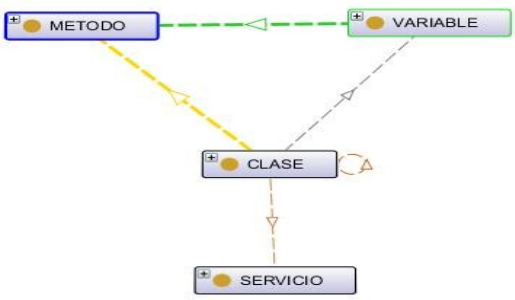

Figura 3.12. Modelo ontológico de la capa PIM.

3.3.3.1 Diseño e implementación de la transformación CIM-PIM

El fundamento principal de la arquitectura ODA es la creación de modelos ontológicos. Sin embargo, la transformación entre uno y otro de estos modelos es un tema fundamental. Lo que busca tanto la arquitectura ODA como la MDA es transformar niveles de abstracción más altos en niveles con menos abstracción.

Fundamentalmente, en FODAS-WS se consideran dos tipos de transformación. Una primera transformación CIM-PIM, que como su nombre lo indica, se encarga de tomar como insumo lo contemplado en la capa CIM para generar de forma automática el modelo PIM. La segunda transformación PIM-PSM, se realiza para generar en forma automática la capa PSM. Cada una de estas transformaciones son un proceso donde se parte de un modelo fuente para llegar a convertirlo en otro modelo (modelo destino), mediante el uso de ciertas reglas de transformación. La Figura 3.13 esquematiza la forma general, y los participantes en un proceso de transformación de modelos.

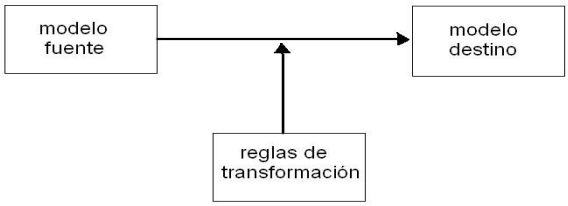

Figura 3.13. Ilustración del proceso de transformación.

Las reglas de transformación describen como se transforman los distintos elementos del modelo fuente en el modelo destino. Dentro de FODAS-WS, las reglas de transformación se constituyen en un software que se encarga de dar instrucciones al razonador para ir generando, con la ayuda del API OWLAPI, el nuevo modelo ontológico destino a partir de un modelo ontológico fuente.

En este proceso de transformación, el modelo fuente es el modelo ontológico asociado con la capa CIM operacional, y el modelo destino es el modelo ontológico asociado con la capa PIM. A lo largo de la presente sección se explicarán las reglas de transformación usadas. El insumo fundamental para realizar la transformación CIMPIM es el conocimiento descrito mediante la capa CIMOperacional.

3.3.3.1.1 Validador y analizador de la semántica definida

Antes de describir el proceso de transformación CIMPIM, es necesario indicar que el Framework FODAS-WS realiza un proceso de validación. Para ello, FODAS-WS cuenta con un analizador y validador, que se encarga de comprobar la congruencia entre la capa CIM OPERACIONAL, la capa CIMREFERENCIAL y LA CAPA DE DEFINICIÓN SEMÁNTICA DEL DOMINIO. El validador garantiza que cada uno de los aspectos especificados en la capa operacional y referencial, se encuentren descritos semánticamente en la definición semántica del dominio. El validador es incorporado en la transformación CIM-PIM, de manera que no se pueda generar la capa PIM en caso de que haya alguna inconsistencia. Para realizar tal validación, el validador corrobora un grupo de axiomas. Para ello, se define C_{DSD} como la CAPA DE DEFINICION SEMANTICA DE DOMINIO, C_{CO} como la CAPA CIM OPERACIONAL y C_{CR} como la CAPA CIM REFERENCIAL

Además, se considerando los siguientes conjuntos:

Acc= {AC_i / $AC_i \in$ Acc y AC_i es un acción definida en C_{DSD}}

E= {E_i / $E_i \in$ E y E_i es un elemento definido en C_{DSD}}

Aso= {As_i / $As_i \in$ Aso y As_i es una asociación definida en C_{DSD}}

Ope= {Op_i / $Op_i \in$ Ope y Op_i es una Operación definida en C_{CO}}

T= {T_i / $T_i \in$ T y T_i es un TipoOperación definido en C_{CO}}

Obj= {Ob_i / $Ob_i \in$ Obj y Ob_i es un Objeto definido en C_{CO}}

P= {P_i / $P_i \in$ P y P_i es un Parámetro definido en C_{CO}}

C= {C_i / $C_i \in$ C y C_i es una Capacidad definida en C_{CR}}

Activ= {$Activ_i$ / $Activ_i \in$ Activ y $Activ_i$ es una Actividad en C_{CO}}

Actor= {$Actor_i$ / $Actor_i \in$ Actor y $Actor_i$ es un Actor en C_{CO}}

Efecto= {Ef_i / $Ef_i \in$ Efecto y Ef_i es un Efecto definido en C_{CR}}

Precon= {Pre_i / $Pre_i \in$ Precon y Pre_i es un Precondicion en C_{CR}}

AsoOP= {$AsoOP_i$ / $AsoOP_i \in$ AsoOP y $AsoOP_i$ es una Asoc en C_{CO}}

AsoREF= {$AsoREF_i$ / $AsoREF_i \in$ AsoREF y $AsoREF_i$ esta en C_{CO}}

AsoDOM= {$AsoDOM_i$ / $AsoDOM_i \in$ AsoDOM y AsoDOM esta en C_{DSD}}

Tal que:

$$\forall x((C=[y|Res] \wedge y=x) \rightarrow Pertenece(x, C))$$

$$\forall x((C=[y|Res] \wedge Pertenece(x, Res)) \rightarrow Pertenece(x, C))$$

$$\forall x(tiene(x,C)\Leftrightarrow\exists y(Pertenece(y,C))\wedge Pertenece(C,x))$$

De esa manera, el validador semántico de FODAS-WS debe evaluar el cumplimiento de los siguientes axiomas:

Todo objeto definido en C_{CO} debe ser un elemento definido en C_{DSD}

$$\forall x(Pertenece(x,Obj)\rightarrow Pertenece(x,E))$$

Toda operación definida en C_{CO} debe ser una acción definida en C_{DSD}

$$\forall x(Pertenece(x,Ope)\rightarrow Pertenece(x,Acc))$$

Todo parámetro definido en C_{CO} debe ser un elemento definido en C_{DSD}

$$\forall x(Pertenece(x,P)\rightarrow Pertenece(x,E))$$

Toda capacidad definida en C_{CR} tiene un conjunto de asociaciones que pertenecen a *AsoDOM*

$$\forall x(Pertenece(x,C)\rightarrow(tiene(x,F)\wedge\forall y(Pertenece(y,F)\rightarrow Pertenece(y,AsoDOM))))$$

Toda precondición definida en C_{CR} tiene un conjunto de asociaciones que pertenecen a *AsoDOM*

$$\forall x(Pertenece(x,Precon)\rightarrow(tiene(x,F)\wedge\forall y(Pertenece(y,F)\rightarrow Pertenece(y,AsoDOM))))$$

Todo efecto definido en C_{CR} *t*iene un conjunto de asociaciones que pertenecen a *AsoDOM*

$$\forall x(Pertenece(x,Efecto)\rightarrow(tiene(x,F)\wedge\forall y(Pertenece(y,F)\rightarrow Pertenece(y,AsoDOM))))$$

Toda actividad definida en C_{CO} debe ser una acción definida en C_{DSD}

$$\forall x(Pertenece(x,Activ)\rightarrow Pertenece(x,Acc))$$

Todo actor definido en C_{CO} *d*ebe ser un elemento definido en C_{DSD}

$$\forall x(Pertenece(x,Actor)\rightarrow Pertenece(x,E))$$

Todo tipo de operación es un elemento que tiene asociados otros elementos como miembros

$$\forall x(Pertenece(x,T)\rightarrow(Pertenece(x,E)\wedge\exists y(tieneMiembro(x,y)\wedge Pertenece(y,E))))$$

Una asociación definida en C_{CO} debe ser una asociación definida en C_{DSD}

$$\forall x(Pertenece(x,AsoOP)\rightarrow Pertenece(x,AsoDOM))$$

Una asociación definida en C_{CR} debe ser una asociación definida en C_{DSD}

$$\forall x(Pertenece(x,AsoREF)\rightarrow Pertenece(x,AsoDOM))$$

3.3.3.2 Descripción de la generación automática de la capa PIM

A continuación se explica la forma en que fue implementada y diseñada la transformación CIM-PIM, aplicándose al caso de estudio.

En la capa CIM OPERACIONAL se definió que existe un objeto llamado *recOA1,* de la clase *RecOA* (un individuo de TIPOOBJETO, indicado en un diagrama de secuencias mediante el object Property *esObjetoDe*), que invoca las operaciones armar, ordenar y *emparejarOA* (a través del objectProperty *invocaOperación,* que relaciona que objetos invocan que operaciones en forma similar a como lo registra un diagrama de secuencias).

A partir de la información de que objeto invoca que operaciones, de que parámetros tiene cada operación, de que tipo de objeto es cada objeto, se puede establecer un primer bosquejo automático de la capa PIM mediante las siguientes reglas de transformación:

$$\forall x\,(tieneObjeto(DA, X) \wedge esObjetoDe(X, Y) \rightarrow GeneraClase(Y)) \quad \textbf{(RT 3.1)}$$

$$\forall x\,(tieneObjeto(DA, X) \wedge invocaOperacion(X, Y) \wedge esObjetoDe(X, Y) \rightarrow (GeneraMetodo(Y) \wedge creaTieneMetodo(Z, Y))) \quad \textbf{(RT 3.2)}$$

$$\forall x\,(tieneParametro(X, Y) \wedge ejecutadaPara(Z, X) \wedge esObjetoDe(Z, W) \rightarrow (GeneraVariable(Y) \wedge creaTieneVariable(W, Y) \wedge creaParametroDe(Y, X))) \quad \textbf{(RT 3.3)}$$

Donde:
tieneObjeto(DA,X), *esObjetoDe(X,Y)*, *invocaOperacion(X,Y)*, *tieneParametro(X,Y)* y *ejecutadaPara(Z,X),* son predicados cuya semántica es idéntica a la semántica asociada con los objectProperty cuyos nombres corresponden con el modelo ontológico de la capa CIMOPERACIONAL, plasmado en la Figura 3.9.

GeneraClase(Y), *GeneraMetodo(Y)* y *GeneraVariable(Y),* son predicados que retornan un valor de verdadero si se logró generar individuos de los conceptos CLASE, METODO y VARIABLE, respectivamente, en el modelo ontológico de la capa PIM e(ver Figura 3.12).

creaTieneMetodo(Z,Y), *creaTieneVariable(W,Y)* y *creaParametroDe(Y,X),* son predicados que retornan un valor de verdadero si se logró crear instancias de los objectProperty *tieneMetodo, tieneVariable* y *tieneParametro,* respectivamente, en el modelo ontológico de la capa PIM (ver Figura 3.12).

Con las reglas *RT 3.1*, *RT 3.2* y *RT 3.3,* se obtiene una transformación directa, de acuerdo con el conocimiento almacenado en la capa CIMOPERACIONAL. Es una primera generación (provisional) de la capa PIM, en la que se cuenta con individuos instanciados como clases, como parámetros y como métodos.

Para el caso de prueba especifico, la regla de transformación *RT 3.1 generó la clase llamada RecOA.* Por otro lado, *RT 3.2 generó* que la clase *RecOA* tiene tres métodos, llamados armar, ordenar y *emparejarOA* (todos vinculados en la capaPIM con el objectProperty *tieneMetodo*). Finalmente, *RT 3.3* generó para la clase *RecOA,* las variables razonador, arreglo, ontologia y estudiante (vinculadas en la capaPIM con el objectProperty *tieneVariable*). De igual forma, las reglas de transformación generaron la clase *RecWeb,* junto con sus tres métodos llamados *armar, ordenar* y *emparejarWeb.*

En la generación (provisional) de la capa PIM, es necesario considerar la posible existencia de jerarquías de clases. Este aspecto deriva del hecho de que varias clases pueden tener los mismos métodos o las mismas variables, en cuyo caso se hace necesario generar nuevas clases de nivel superior, de las que se extiendan cada una de ellas. Para ello, se hace necesario la incorporación de las reglas mostradas a continuación.

$$\forall x\,(tieneVariable(X,Y)\wedge tieneVariable(X,Z)\wedge MiembroClase(Y_{¿}Z)\rightarrow(creaTieneVariable(Y_{¿}Z,X)\wedge eliminaTieneVariable(X,Y)\wedge eliminaTieneVariable(X,Z)))\quad \textbf{(RT 3.4)}$$

$$\forall x\,(tieneVariable(X,Y)\wedge tieneVariable(X,Z)\wedge\neg MiembroClase(Y_{¿}Z)\rightarrow(GeneraClase(Y_{¿}Z)\wedge esSuperClaseDe(Y_{¿}Z,X)\wedge esSuperClaseDe(Y_{¿}Z,Y)\wedge creaTieneVariable(Y_{¿}Z,X)\wedge eliminaTieneVariable(X,Y)\wedge elimin$$

(RT 3.5)

$$\forall x\,(tieneMetodo(X,Y)\wedge tieneMetodo(X,Z)\wedge MiembroClase(Y_{¿}Z)\rightarrow(creaTieneMetodo(Y_{¿}Z,X)\wedge eliminaTieneMetodo(X,Y)\wedge eliminaTieneMetodo(X,Z)))\quad \textbf{(RT 3.6)}$$

$$\forall x\,(tieneMetodo(X,Y)\wedge tieneMetodo(X,Z)\wedge\neg MiembroClase(Y_{¿}Z)\rightarrow(GeneraClase(Y_{¿}Z)\wedge esSuperClaseDe(Y_{¿}Z,X)\wedge esSuperClaseDe(Y_{¿}Z,Y)\wedge creaTieneMetodo(Y_{¿}Z,X)\wedge eliminaTieneMetodo(X,Y)\wedge eliminaTie$$

(RT 3.7)

Donde:

tieneVariable(X,Y) y *tieneMetodo(X,Y)* son predicados, cuya semántica es idéntica a la semántica asociada con los objectProperty cuyos nombres corresponden con el modelo ontológico de la capa PIM (ver Figura 3.12).

MiembroClase(X) es un predicado que retorna verdadero si la clase X está incorporada en la ontología asociada con la capa PIM. *GeneraClase(Y)* es un predicado que retorna un valor de verdadero si se logró generar un individuo del concepto CLASE en el modelo ontológico de la capa PIM)ver Figura 3.12).

creaTieneMetodo(Z,Y) y creaTieneVariable(W,Y) son predicados que retornan un valor verdadero si se logró crear instancias de los objectProperty *tieneMetodo* y *tieneVariable*, respectivamente, en el modelo ontológico de la capa PIM.

eliminaTieneMetodo(Z,Y) y elimina*TieneVariable(W,Y)* son predicados que retornan un valor verdadero si se logró eliminar las instancias de los objectProperty *tieneMetodo* y *tieneVariable*, respectivamente, en el modelo ontológico de la capa PIM.

De acuerdo con la generación (provisional) de la capa PIM para el caso de estudio ilustrada en párrafos anteriores, las clases *RecOA* y *RecWeb* tienen en común los métodos *ordenar* y *armar*, y las variables *ontología*, *arreglo* y *razonador.* Este hecho implica que aplicando las reglas de transformación RT 3.4 hasta RT 3.7 sobre la capa PIM generada, se realice un proceso de refinación en el cual se verifique la existencia de jerarquías de clase, y en caso de que se cuente con ellas, la transformación proceda a crear una nueva clase de nivel superior, que contenga los métodos y variables comunes, e indique que clases heredan de ella.

La Figura 3.14 ilustra el resultado obtenido en la capa PIM, después de ejecutadas las reglas de transformación RT 3.4 hasta RT 3.7, para el caso de estudio. Como se aprecia en la Figura 3.14, las regla de transformación RT 3.4 hasta RT 3.7 detectaron que las clases *RecOA* y la *RecWeb* tienen métodos comunes (armar y ordenar) y variables comunes (ontología, arreglo y razonador). Por ello generaron una nueva clase, llamada *RecOARecWeb*, que contiene los métodos y variables comunes, y adicional a eso, indica que clases heredan de ella mediante el objectProperty *esSuperClaseDe,* con lo cual se indica que las clases *RecOA* y *RecWeb* heredan de la super clase *RecOARecWeb* (nombre obtenido de concatenar los nombres de las clases que heredan de ella).

Además de generar la clase de nivel superior, las reglas de transformación RT 3.4 hasta RT 3.7 también se encargan de eliminar de las subclases los métodos y variables que se hereden de la clase padre. La Figura 3.14 muestra, por ejemplo, como en la clase *RecOA* la regla RT 3.6 eliminó los métodos armar y ordenar de las clases *RecOA* y *RecWeb*, y como la regla RT 3.4 eliminó las variables ontología,

arreglo y razonador de las clases *RecOA* y *RecWeb*, pues ellas son heredadas de la *RecOARecWeb.*

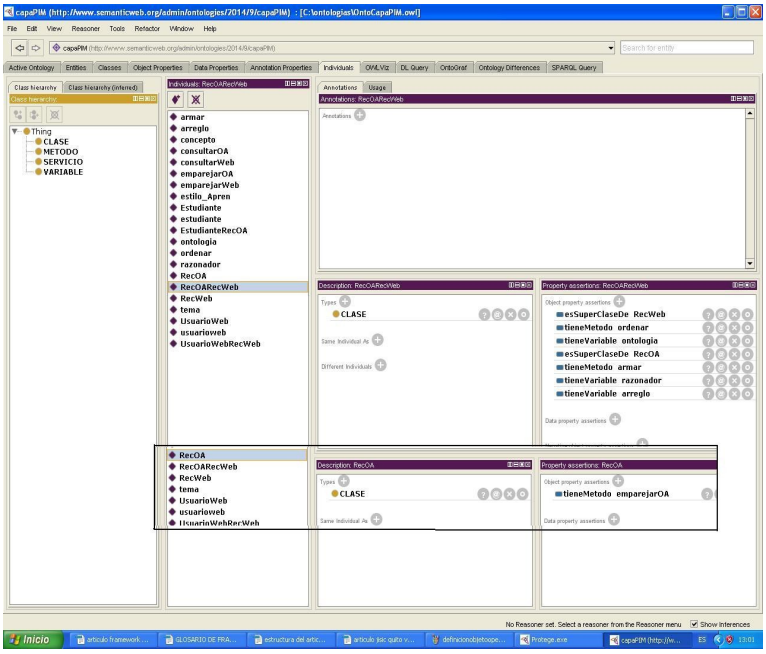

Figura 3.14. Generación automática de la capa PIM, por medio de la transformación CIMPIM.

3.3.4 Diseño e implementación de la capa PSM

La Capa PSM es la encargada de almacenar el conocimiento necesario para realizar los procesos de descubrimiento, composición y ejecución automática de los servicios web que se especifiquen usando FODAS-WS. La capa PSM está constituida por el modelo ontológico mostrado en la Figura 3.15, en el cual se almacena todo el conocimiento relacionado con los detalles de como el servicio web fue implementado en una plataforma especifica.

La instanciación de la ontología de la capa PSM, se genera de forma automática mediante la ejecución de un algoritmo denominado TRANSFORMACION PSM, incorporado como parte del Framework FODAS-WS. La Transformación PSM es un proceso idéntico al explicado mediante la Figura 3.13. El modelo fuente dentro del Framework no es directamente la capa PIM, sino que como lo ilustra la arquitectura de FODAS-WS mostrada en la Figura 3.1, el modelo fuente es el SWS ya implementado. De esta manera, la instanciación del modelo ontológico de la capa PSM se genera de forma automática a partir del archivo WSDL, que describe el servicio web, y dicho archivo se obtiene después del proceso de implementación y despliegue del servicio en un servidor web. Esto tiene la ventaja que la

descripción semántica incorporada en la capa PSM, se ajusta exacta y realmente a la forma en que el servicio fue implementado. Además, debido a que el Framework esta concebido para que a futuro realice generación automática de código del SWS, se garantiza que el conocimiento especificado en la capa PSM sea conforme lo especificado en las capas de niveles inferiores.

La capa PSM contiene aspectos análogos a los especificados en el archivo WSDL asociado al servicio Web, con la diferencia que en esta capa del FODAS-WS se pueden realizar procedimientos formales de razonamiento. En particular, este es el nivel más alto de especificación del servicio, a diferencia de los servicios web tradicionales, en los que su descripción WSDL es la única proveída al cliente, en un formato poco flexible, sin connotación semántica alguna.

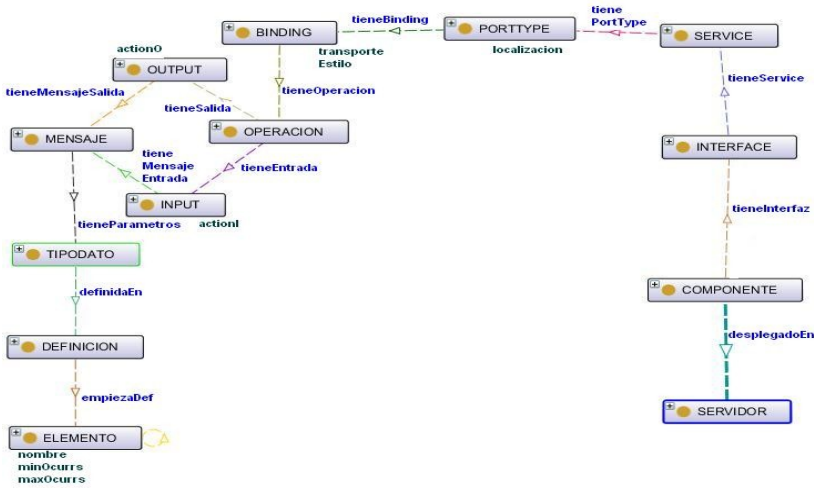

Figura 3.15. Modelo ontológico de la CAPA PSM

La Figura 3.15 muestra el modelo ontológico de la capa PSM. Se observa que los componentes (servicios web) cuentan con una o varias interfaces, representadas mediante el concepto INTERFACE, las cuales se vinculan usando el objectProperty *tieneInterfazDeRealizacion* con un individuo del concepto COMPONENTE. Las interfaces agrupan uno o mas individuos del concepto SERVICE (de forma totalmente análoga a como lo contempla WSDL y OWL-S). Un individuo de SERVICE agrupa uno o más individuos de PORTTYPE, lo que se especifica mediante el objectProperty *tienePortType. C*ada PORTTYPE agrupa una o más operaciones, que son instancias del concepto operación. Una operación esta conformada por mensajes, que pueden ser asociadas al tipo INPUT o OUTPUT. Es importante resaltar que cada PORTTYPE se asocia con un individuo de ENDPOINT mediante el objectProperty *tieneEndPoint.* Un individuo de ENDPOINT contiene la

información necesaria de como acceder al servicio, mediante los dataProperty *dirección, protocolo* y *puerto.*

Lo anterior evidencia como la capa PSM se encarga de entregar los aspectos de implementación específicos de la plataforma en que se implementó el servicio. Este modelo ontológico entrega información indispensable al momento de realizar la composición y ejecución automática. En particular, no solo dice acerca del protocolo a usar, dirección y puerto por el que se accede el servicio, sino que da la información necesaria de que mensajes SOAP se deben estructurar y en que forma. Una vez WS GENERADOR CLIENTE (encargado de hacer la composición y ejecución automática) tenga formalizado esto, debe acudir a los demás niveles de abstracción para entender el orden en que debe enviar los distintos mensajes, y lo que significa cada uno de los mensajes intercambiados con el servicio web (las distintas interacciones necesarias son ilustradas en la Figura 3.1).

3.4 Comparación con otros trabajos similares

Se realiza una comparación de FODAS-WS con otros Framework existentes, que aunque no son totalmente idénticos en funcionalidad, pertenecen a la misma área de estudio. En particular, se compara con las propuestas presentadas en [18], [19], [21]. Para ello, se definen un conjunto de criterios de comparación entre los Framework, los cuales son:

1. Se ajusta a la arquitectura ODA
2. Tiene transformaciones automáticas.
3. Propone mecanismos de descubrimiento automático de servicios web.
4. Propone mecanismos de composición automática.
5. Propone mecanismos de ejecución automática de clientes.
6. Poseé validador de semántica propio.
7. Propone generar código esqueleto para la implementación del servicio
8. Propone generar una aplicación cliente de forma automática, implementando con ello: descubrimiento, composición y ejecución automática.
9. Tiene ontologías para representar cada capa

10. Usa estándares para la representación semántica de los servicios web semánticos, tales como OWL-S [65] o WSMO [66].

Criterio	FODAS-WS	SWF	MDF-DWSOA	METEOR-S
1	SI	SI	NO	NO
2	SI	NO	NO	NO
3	SI	NO	NO	NO
4	SI	NO	NO	NO
5	SI	NO	NO	NO
6	SI	NO	NO	NO
7	SI	NO	SI	NO
8	SI	NO	NO	NO
9	SI	SI	NO	SI
10	NO	SI	SI	NO

Tabla 3.6 Comparativo de FODAS-WS respecto de otros trabajos.

FODAS-WS muestra ser el Framework más completo, pues además de permitir una descripción semántica completa del servicio, deja plasmado la conceptualización necesaria para realizar el descubrimiento automático, la composición automática y la ejecución automática de servicios web. La comparación logra estipular que FODAS-WS es el único de los trabajos que posee transformaciones automáticas entre capas, y un validador semántico que garantiza congruencia entre capas. Eso implica que minimiza lo posibilidad de inconsistencias entre el diseño del servicio web semántico, su implementación, y sus procesos de composición y ejecución.

También, FODAS-WS es el único Framework que propone mecanismos para realizar futuros trabajos en los que se logre llegar a diseñar e implementar un servicio web, al que se le denomina WS GENERADOR CLIENTE (ver Figura 3.1), que se encargue de generar aplicaciones cliente para los SWS en forma automática. Es decir, FODAS-WS a diferencia de los demás trabajos analizados, contempla la posibilidad de que a partir de la especificación semántica del SWS, se realice un proceso de descubrimiento, composición y ejecución totalmente automático.

FODAS-WS no se ajusta a los estándares existentes para realizar la descripción semántica de servicios web. Sin embargo, en ninguno de dichos estándares

realizan la descripción semántica del dominio, mediante el uso de asociaciones que representan efectos, capacidades, necesidades y precondiciones.

Por otro lado, FODAS-WS contempla la casi totalidad de los aspectos contemplados por OWL-S, ya que los dos toman como punto de partida para la descripción semántica a WSDL. Pero FODAS-WS tiene como ventaja, que minimiza cualquier posibilidad de generar inconsistencias, como ya se estipulo en este mismo apartado.

En general, FODAS-WS cuenta con las siguientes características:

- Los modelos ontológicos generados en el proceso de diseño pueden ser usados por un razonador para chequear los componentes de un sistema, y verificar sus consistencias, logrando con ello tener un diseño libre de inconsistencias que altere posteriormente su utilización.
- Los SWS diseñados con FODAS-WS permiten que se realicen inventarios de software, en los que además de conocer el software (en este caso SWS) existente, se logre realizar un chequeo de funcionalidades repetidas o inconsistentes entre ellos.
- El diseño de SWS mediante FODAS-WS representa un mecanismo de diseño comprensible para agentes de software, y el principal insumo para lograr descubrimiento, emparejamiento y composición automática de servicios web.
- El uso de FODAS-WS simplifica el proceso de diseño a través de la utilización de transformaciones automáticas entre capas, para la generación automática de las capas superiores.
- Un SWS diseñado con FODAS-WS no requiere de procesos de descripción semántica adicionales al diseño de la aplicación.
- Un SWS diseñado en FODAS-WS no solo describe los aspectos básicos contemplados en otros procedimientos de especificación semántica (como OWL-S o WSMO), sino que incluye la totalidad del diseño del servicio (funcional, estructural, etc).
- FODAS-WS sirve como instrumento para facilitar el proceso de integración y composición de aplicaciones, tanto con intervención humana como sin ella.

- FODAS-WS pretende incrementar y facilitar el proceso de reusabilidad. Tal proceso se facilita gracias a la posibilidad de realizar descubrimiento y composición automáticos de los SWS especificados mediante el Framework.
- La realización de tareas de mantenimiento de los servicios web se hace de manera mas clara y congruente, ya que el razonador puede entregar al encargado de mantenimiento información inferida que puede ser útil. Por ejemplo, análisis de funcionalidades inconsistentes entre ellas, emparejamiento entre capacidades de cada servicio y las necesidades de sus clientes, etc.

4 Especificación de la nube de fuentes de conocimiento en lo referente a gestion de OA.

En este capitulo se presenta la arquitectura, diseño y especificación semántica de la nube de fuentes del conocimiento del Proyecto Madre, presentado en [1, 2, 3, 4, 5]. La Figura 4.1 muestra, como se explicó en la sección 2.1.3, que la nube de conocimiento enmarcada en el Proyecto Madre contempla la gestión de dos tipos de recursos distintos, que son: OAs y contenidos digitales (CD). La gestión de CD es abordada en [6], y esta tesis se encarga de abordar la gestión de OAs. La plataforma encargada de gestionar OAs no tiene ningún tipo de interacción con la plataforma de gestión de CD presentada en [6], y cada una de ellas interactúa de forma independiente con las Nubes de auto-formación y de aprendizaje del Proyecto Madre.

El capitulo se compone de 3 secciones. La primera sección presenta la arquitectura de la nube fuentes de conocimiento, en la cual el componente central es un servicio web semántico que ofrece servicios de recomendación de OAs. La segunda sección se dedica a la descripción y diseño de cada uno de los componentes arquitectónicos que intervienen en la nube fuentes de conocimiento; y en la tercera sección se diseña y especifica semánticamente el servicio web semántico que encapsula las funcionalidades de recomendación de OAs, usando para ello al Framework FODAS-WS explicado en el capitulo 3 de la presente tesis. Es necesario aclarar que FODAS-WS solo está planteado para el diseño y la especificación de servicios web semánticos, motivo por el cual en la sección 3 solo se usa para describir el SWS que encapsula el recomendador de OAs.

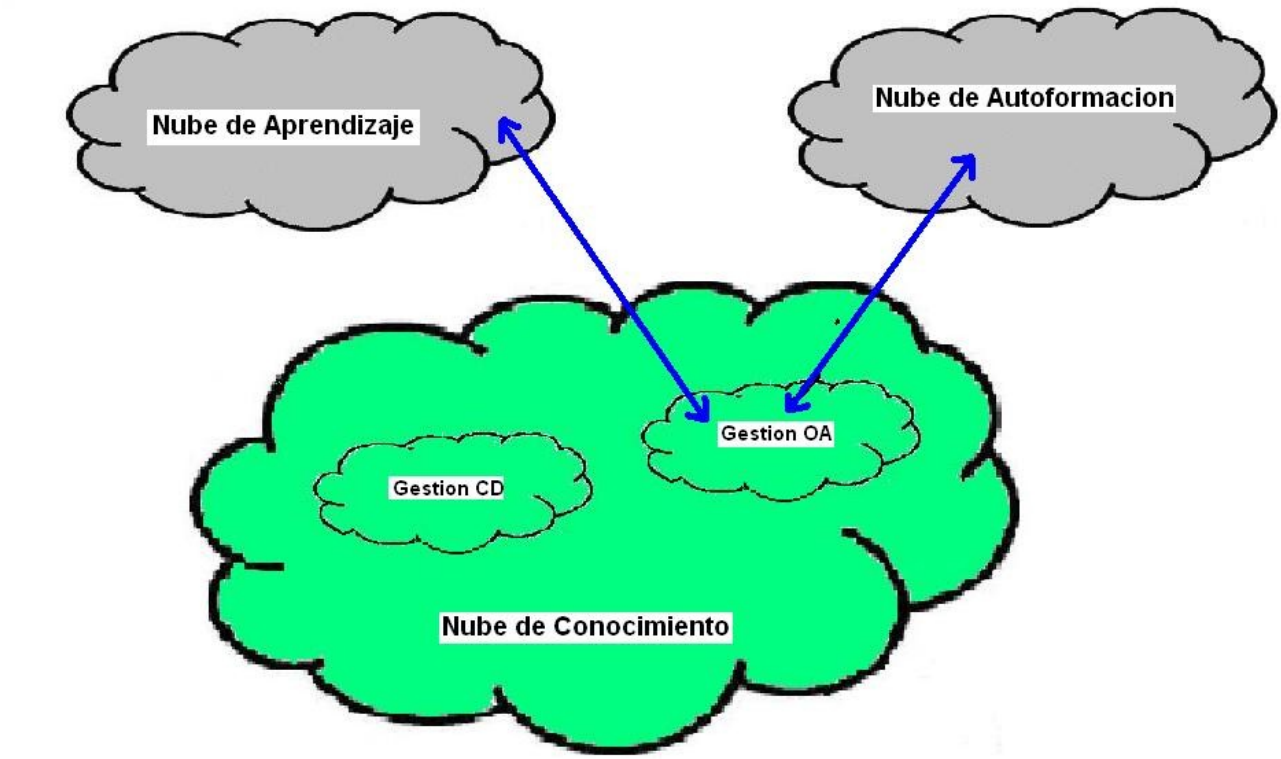

Figura 4.1 Composición interna de la nube de conocimiento

4.1 Arquitectura de la nube fuentes de conocimiento en lo referente a gestión de OAs

Esta tesis se encarga de diseñar e implementar una plataforma tecnológica basada en SOA, para gestionar los OAs de la nube de fuentes de conocimiento del proyecto madre. En la Figura 4.2 se muestra la arquitectura interna de la nube de fuentes del conocimiento en lo referente a la gestión de OAs. Como se aprecia en la Figura 4.2, el componente recomendador de OAs de la nube de fuentes del conocimiento se encuentra formada por seis componentes fundamentales:

- Un sistema recomendador de OAs
- Un repositorio semántico de OAs
- Un agente adaptativo de la usabilidad
- Un agente adaptativo de la vinculación semántica de temas
- Un agente adaptativo de la calidad y evaluación de desempeño
- Un agente adaptativo de aspectos pedagógicos

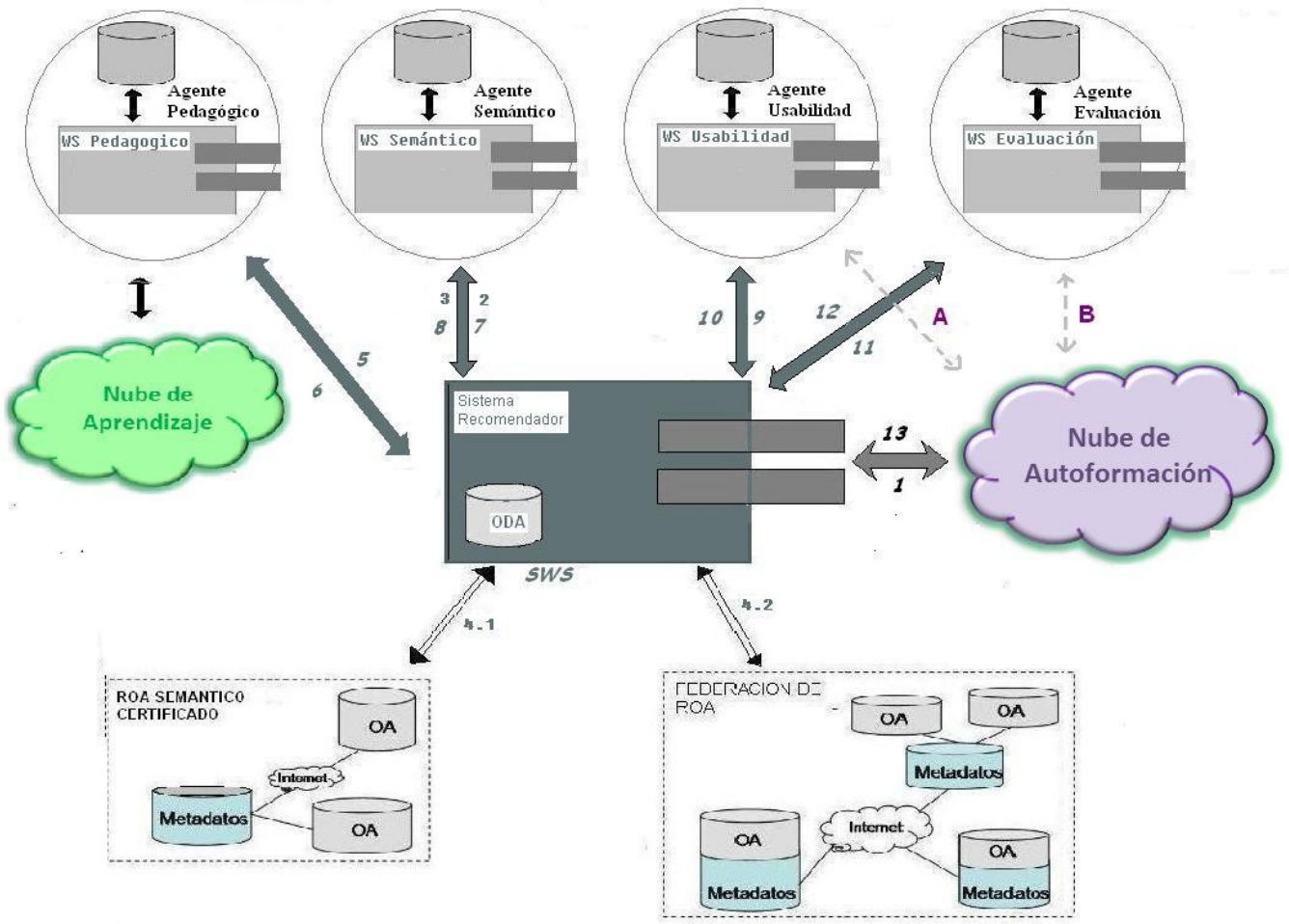

Figura 4.2. Arquitectura del componente de recomendación de OAs de la Nube de Conocimiento del proyecto Madre

Se ve en la arquitectura, como se indicó en la sección 2, que esta nube se interrelaciona con las nubes de aprendizaje y de auto-formación.

A continuación se muestra una breve descripción de cada uno de los componentes que conforman la arquitectura de la nube de fuentes de conocimiento (más adelante se presenta la especificación completa de cada una).

SISTEMA RECOMENDADOR DE OAs: Se trata de un recomendador híbrido compuesto por un componente basado en conocimiento, un componente colaborativo, y un componente basado en contenidos. El papel fundamental del sistema recomendador de OA es construir un conjunto de recomendaciones de OAs, que considera adecuados para enseñar un tema a un estudiante para el cual la nube de auto-formación solicita la recomendación. El recomendador realiza un ranking de los elementos del CONJUNTO DE RECOMENDACIÓN, de tal manera que

recomienda los objetos en forma ordenada de acuerdo con un valor de utilidad calculado por él para cada uno de los OA.

REPOSITORIO SEMÁNTICO DE OAs: Se trata de un medio de almacenamiento de conocimientos relacionados con los metadatos asociados a los OA que han sido sometidos a un proceso de certificación. El elemento fundamental del repositorio es un modelo ontológico, que se diseñó usando el estándar LOM[10]. Dicha ontología contiene los elementos considerados fundamentales, para que el recomendador pueda realizar las recomendaciones solicitadas.

FEDERACIONES DE ROA: Esta conformado por los distintos repositorios de OA distribuidos a nivel mundial que ofrecen servicios de búsqueda de OA. Estas federaciones de ROA son usadas por el sistema recomendador, solo en el caso de que no cuente con OA certificados que se ajusten a la recomendación solicitada.

AGENTE ADAPTATIVO DE LA USABILIDAD: Está compuesto de un repositorio con la información de los perfiles de usuario, en el que se almacena el conocimiento referente a las preferencias de cada usuario y al uso de OAs que cada uno realiza. Además, este Agente cuenta con un servicio web que ofrece información del perfil de un usuario, solicitado por quien lo requiera. En esta tesis, el cliente de dicho servicio web es el sistema recomendador, quien usa ese conocimiento para personalizar las recomendaciones. La información registrada en el perfil de usuario, proviene de la dinámica generada en la nube de auto-formación.

AGENTE ADAPTATIVO DE LA VINCULACIÓN SEMÁNTICA DE TEMAS: Se trata de un agente de software, que se encarga de proporcionarle al sistema recomendador los distintos temas vinculados semánticamente con el tema solicitado en la recomendación. Este agente se basa en el uso de una ontología de vinculación semántica de temas, que sirve para cualquier aplicación que la requiera.

AGENTE ADAPTATIVO DE LA CALIDAD Y EVALUACIÓN DE DESEMPEÑO: Es un agente de software que gestiona todo el conocimiento relacionado con la

calidad y la evaluación de desempeño de cada uno de los OAs que se encuentran certificados. En esta tesis, este Agente solo considera la calificación cuantitativa de calidad dada por el usuario experto, en el momento de certificar el OA. Este Agente podría ser extendido con técnicas inteligentes en trabajos futuros, que permitan re-evaluar permanentemente la calidad y el desempeño de cada uno de los OAs (aprender), de acuerdo con las dinámicas de las nubes.

AGENTE ADAPTATIVO DE ASPECTOS PEDAGÓGICOS: Proporciona un servicio web que gestiona todo el conocimiento relacionado con el aspecto pedagógico de la nube. Este Agente interactúa directamente con la nube de aprendizaje, y es el encargado de calcular el aporte de personalización pedagógica en la función de utilidad de cada objeto a recomendar, según el estilo de aprendizaje del estudiante.

Como lo muestra la Figura. 4.2, al interior de la nube de fuentes de conocimiento se llevan a cabo trece distintas interacciones:

1. La nube de auto-formación solicita al servicio web recomendador una recomendación de que OAs sugiere, para que cierto estudiante aprenda cierto tema.
2. El servicio web recomendador solicita al Agente semántico, un listado de temas vinculados semánticamente con el tema para el cual se realizó la solicitud de recomendación.
3. El Agente semántico retorna un listado que contiene la totalidad de temas vinculados semánticamente con el tema para el cual se realizó la solicitud de recomendación.
4. El sistema recomendador realiza una búsqueda de OAs, para lo cual realiza una de las siguientes dos interacciones:

   4.1 En caso de que se tengan OAs en el repositorio semántico de OAs, que enseñen temas vinculados a los temas obtenidos en la interacción 3, el recomendador obtendrá los metadatos de los OAs.

   4.2 En caso de que no se encuentre ningún OA en el repositorio semántico de OAs (es decir, un conjunto de recomendaciones certificadas vació), el recomendador realiza una cosecha (búsqueda) en los repositorios federados en el repositorio MERLOT.

5. El sistema recomendador solicita al Agente adaptativo de aspectos pedagógicos, calcular el *aporte de personalización pedagógica* asociado con el OA O_i .
6. Ese Agente retorna un valor numérico comprendido entre 0 y 1, que representa el *aporte de personalización pedagógica* del objeto O_i.
7. El sistema recomendador solicita al Agente adaptativo de la vinculación semántica, calcular el *aporte semántico cognitivo* asociado con el objeto de aprendizaje O_i .
8. Ese Agente retorna un valor numérico comprendido entre 0 y 1, que representa el *aporte semántico cognitivo* del objeto O_i para el cálculo de la utilidad generada por dicho objeto.
9. El sistema recomendador solicita al Agente adaptativo de la Usabilidad, calcular el *aporte de personalización por usabilidad* asociado con el OA O_i .
10. Ese Agente retorna un valor numérico comprendido entre 0 y 1, que representa el *aporte de personalización por usabilidad* del objeto O_i.
11. El sistema recomendador solicita al Agente adaptativo de la Calidad y Evaluación de Rendimiento, calcular el *aporte colaborativo* asociado con el OA O_i .
12. Ese Agente retorna un valor numérico comprendido entre 0 y 1, que representa el *aporte colaborativo* del objeto O_i.
13. El servicio web recomendador agrega todos los aportes de cada OA a recomendar, y retorna a la nube de auto-formación un CONJUNTO DE RECOMENDACIÓN de OAs rankeados según sus aportes. De esa manera, sugiere OAs personalizados (de acuerdo a las preferencias de usuario y a sus indicadores de uso), pero no solo asociados únicamente al tema solicitado, sino también aquellos OAs que enseñan temas relacionados semánticamente.

La interacción etiquetada con A no se encuentra asociada directamente con el proceso de recomendación que realiza la nube de fuentes de conocimiento, sino que está interacción se da cada vez que la nube de auto-formación requiere registrar conocimiento sobre el perfil de cada estudiante en el Agente adaptativo

de usabilidad. La interacción etiquetada con B tampoco se relaciona directamente con el proceso de recomendación, dicha interacción se propone que sea realizada cada vez que la nube de auto-formación requiera registrar conocimiento que modifique la evaluación de calidad de un OA. Por ejemplo, el desempeño de un estudiante al estudiar un tema usando un OA dado.

4.2 Descripción y diseño del componente de recomendación de OAs de la Nube de Fuentes del Conocimiento

A continuación se describen detalladamente cada uno de los componentes que integran el sistema de recomendación acá propuesto.

4.2.1 Repositorio semántico de OAs

El repositorio semántico de OAs tiene como componente fundamental un modelo ontológico, en el cual se almacena el conocimiento de los distintos OAs que pasan por cierto proceso de certificación, llevado a cabo por expertos que evalúan los OAs disponibles en los distintos repositorios a nivel mundial.

4.2.1.1 Modelo ontológico del repositorio semántico

En el repositorio semántico no se almacenan los OAs como tal, sino que se almacenan los metadatos que describen cada objeto, incluida la información de su ubicación. Para guardar el conocimiento de los metadatos dentro del repositorio, se utiliza una ontología. La Figura 4.3 muestra la ontología, cuyo concepto principal es el OA.

La ontología mostrada en la Figura 4.3 se ajusta al estándar LOM [10], que cuenta con amplia aceptación a nivel mundial, y se encarga de determinar qué metadatos son relevantes en el instante de describir un OA. En ese sentido, el concepto OA es caracterizado por dicho estándar. La Tabla 4.1 muestra que categorías y que elementos de cada categoría de LOM, fueron considerados en el diseño de la

ontología de metadatos del repositorio semántico de OAs (son las propiedades de la clase/concepto OA).

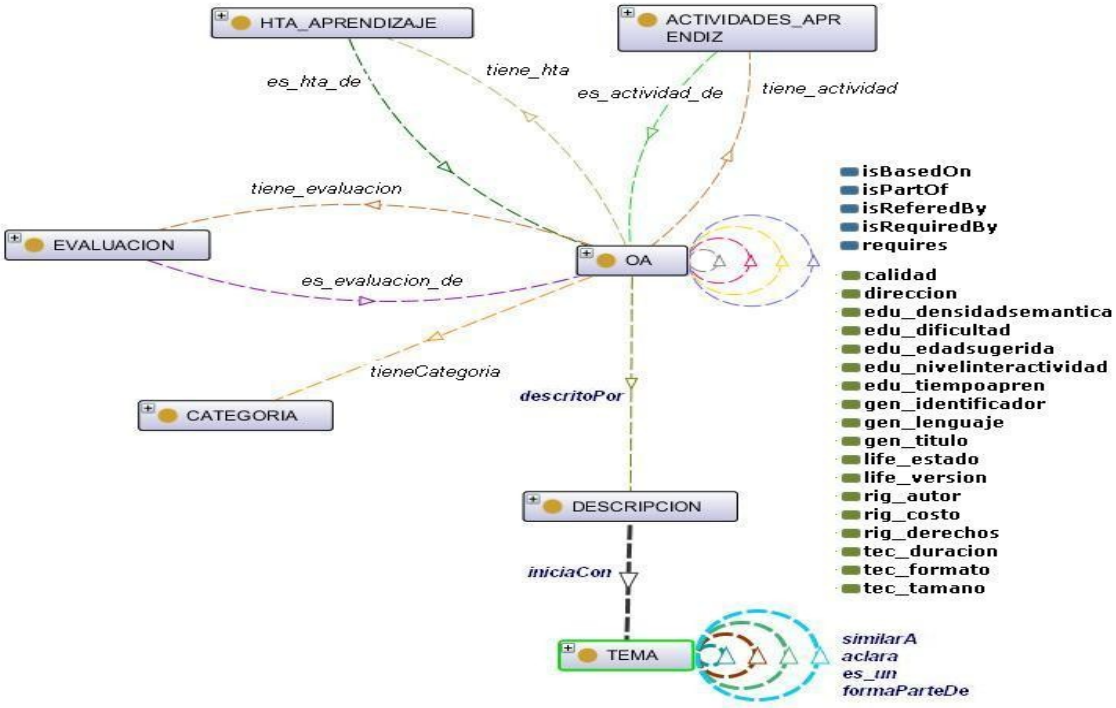

Figura 4.3 Diseño de la ontología de OAs del repositorio semántico

Los metadatos descritos en la Tabla 4.1 almacenan el conocimiento sobre cada OA, sugeridos por el estándar LOM, como propiedades de la clase OA.

CATEGORÍA	**ASPECTO**	**DESCRIPCIÓN**
General	gen_identificador	Especifica el identificador del objeto
	gen_lenguaje	Especifica el idioma en que está hecho el objeto
	gen_titulo	Determina el titulo del OA
Ciclo de vida	life_estado	Expresa el estado de completitud o condición del objeto.
	life_version	Especifica la versión del OA
Técnica	tec_duracion	Tiempo promedio en que un estudiante estudia los conceptos enseñados por el objeto
	tec_formato	Especifica el formato en que se encuentra el objeto
	tec_tamaño	Especifica el tamaño del objeto
	tec_direccion	Especifica la localización o URL del objeto
Educacional	edu_densidadsemantica	Establece que tan densos son los conceptos que enseña
	edu_dificultad	Describe lo difícil que resulta, para los destinatarios típicos, trabajar con, y utilizar este OA
	edu_edadsugerida	Edad sugerida, para los estudiantes típicos que usen el OA
	edu_nivelinteractividad	Grado en el que el aprendiz puede influir en el comportamiento del OA.
	edu_tiempoapren	Tiempo aproximado o típico, que necesitan para asimilar el OA los destinatarios objetivo típicos
Rights	rig_autor	Registra el autor o autores del objeto.
	rig_costo	Indica el costo del OA
	rif_derechos	describe los derechos de propiedad intelectual y las condiciones de uso aplicables a este OA
Relación	isBasedOn	Se usa cuando un OA enseña basándose en lo enseñado en otro.
	isPartOf	Se usa cuando el OA es parte de otro OA de mayor cobertura.

	isReferedBy	Es usado para indicar que el OA es mencionado por otros OAs.
	isRequiredBy	Se usa cuando un OA requiere de otro para cumplir sus propósitos de enseñanza
Otros	fecha	Determina la fecha de creación del objeto.

Tabla 4.1 Descripción de los elementos y categorías usadas en la ontología del estándar LOM.

Además, como se trata de un sistema recomendador híbrido, adicional al conocimiento del contenido de cada OA, se hace necesario incorporar otros metadatos que describan aspectos pedagógicos de cada OA, y su calidad. Tales metadatos se describen en la Tabla 4.2, y son plasmados como conceptos/clases en la ontología (ver figura 4.3).

CLASE	DESCRIPCION
EVALUACION	Representa cada uno de los instrumentos de evaluación que son considerados dentro de la nube de aprendizaje (ver [4], donde se detallan dichos instrumentos).
HTA_APRENDIZAJE	Representa cada una de las herramientas de enseñanza que son consideradas dentro de la nube de aprendizaje (ver [4], donde se detallan dichas herramientas).
ACTIVIDADES_APRENDIZ	Representa cada una de las actividades de aprendizaje que son consideradas dentro de la nube de aprendizaje (ver [4], donde se detallan dichas actividades).
CATEGORÍA	Estipula a que categoría o clasificación taxonómica pertenece el OA. El concepto de clasificación taxonómica es registrado conforme al aspecto 9 del estándar LOM presentado en [24]
DESCRIPCIÓN	Cada descripción es un mecanismo para determinar (describir) el tipo de temática enseñada por cada OA.
TEMA	Representa cada uno de los temas que se pueden enseñar con ese OA. Esta clase le proporciona a la nube de conocimiento capacidades cognitivas, ya que es a través de ella que se hace el enriquecimiento semántico de un OA.

Tabla 4.2 Descripción de las clases existentes en el modelo ontológico del repositorio.

La Tabla 4.3 describe cada una de las relaciones (object property) estipulados dentro de la ontología del repositorio semántico.

OBJECTPROPERTY	DESCRIPCIÓN
tiene_actividad	Determina cada actividad de aprendizaje que se asocia a un OA
tiene_hta	Determina cada herramienta de enseñanza que se asocia a un OA
tiene_evaluacion	Determina cada instrumento de evaluación que se asocia a un OA
tiene_categoria	Determina la categoría a la que pertenece el OA
descritoPor	Determina cual descripción es asociada con un OA
iniciaCon	Determina con que tema inicia una descripción.
similarA	Es usado por el agente adaptativo de la vinculación semántica de temas, para indicar que un tema es vinculado mediante COMPARACION con otro.
aclara	Es usado por el agente adaptativo de la vinculación semántica de temas, para indicar que un tema es vinculado mediante EXPLICACION con otro.
es_un	Es usado por el agente adaptativo de la vinculación semántica de temas, para indicar que un tema es vinculado mediante JERARQUIZACION con otro.
formaParteDe	Es usado por el agente adaptativo de la vinculación semántica de temas, para indicar que un tema es vinculado mediante ASOCIACION con otro.

Tabla 4.3 Descripción de cada ObjetProperty en el modelo ontológico del repositorio

4.2.1.2 Certificador de OAs

La Figura 4.4 ilustra el pseudocódigo que esquematiza el proceso de certificación de OAs, a incorporar dentro de la ontología de metadatos de OAs con que cuenta el repositorio. Para ello, el usuario experto genera un conjunto de posibles OAs a almacenar en el repositorio (llamado CAR), que se compone de todos aquellos OAs que el usuario experto encuentre almacenados en los distintos repositorios existentes a nivel mundial para cierto tema (linea 2 del pseudocódigo) . Una vez constituido el conjunto CAR, el usuario experto evaluá uno por uno los OAs (O_i) que conforman dicho conjunto (linea 4 del pseudocódigo), y con base a dicha evaluación los va certificando o no (estructura de decisión ubicada entre las lineas 5 y 7). La certificación se realiza incorporando el conocimiento de cada O_i en el repositorio semántico, con los metadatos del OA certificado por el experto (linea 6) .

```
1. INICIO
2.        Generar (CAR).
3.        PARA CADA Oi que pertenezca a CAR HACER
4.              evaluar(Oi)
5.              SI Oi es de calidad HACER
6.                    certificar(Oi)
7.              FIN SI
8.        FIN PARA
9. FIN
```

Donde:
CAR = Conjunto de **C**andidatos a **A**lmacenar en el **R**epositorio.

Figura 4.4 Pseudocódigo del proceso de certificación de OA dentro de la nube de conocimiento

El usuario experto usará la aplicación mostrada en la Figura 4.5, cada vez que requiera certificar un OA (linea 6 del pseudocódigo de la Figura 4.4). La certificación consiste en determinar los valores para cada uno de los metadatos basados en LOM, que constituyen el modelo ontológico del repositorio semántico explicado en la Figura 4.3, para el OA que se quiere certificar. Recordemos que además de los metadatos considerados por LOM, el usuario experto debe evaluar que herramientas de aprendizaje, actividades de aprendizaje y métodos de

evaluación, usa el OA, e incorporar dicho conocimiento en el repositorio. Finalmente, debe dar una calificación asociada con la calidad del OA, donde 0 indica la peor calidad y 5.0 indica el mayor nivel de calidad posible. El valor calificado por el experto para la calidad depende del nivel de profundidad, facilidad de explicación, entendibilidad, amigabilidad, didáctica, etc., que determine el experto que posee cada OA. Como trabajos futuros, se plantea que el usuario experto se reemplace por una aplicación certificadora que realice cosechado automático de OAs, y determine la totalidad de metadatos necesarios, incluida una estimación de la calidad para cada OA.

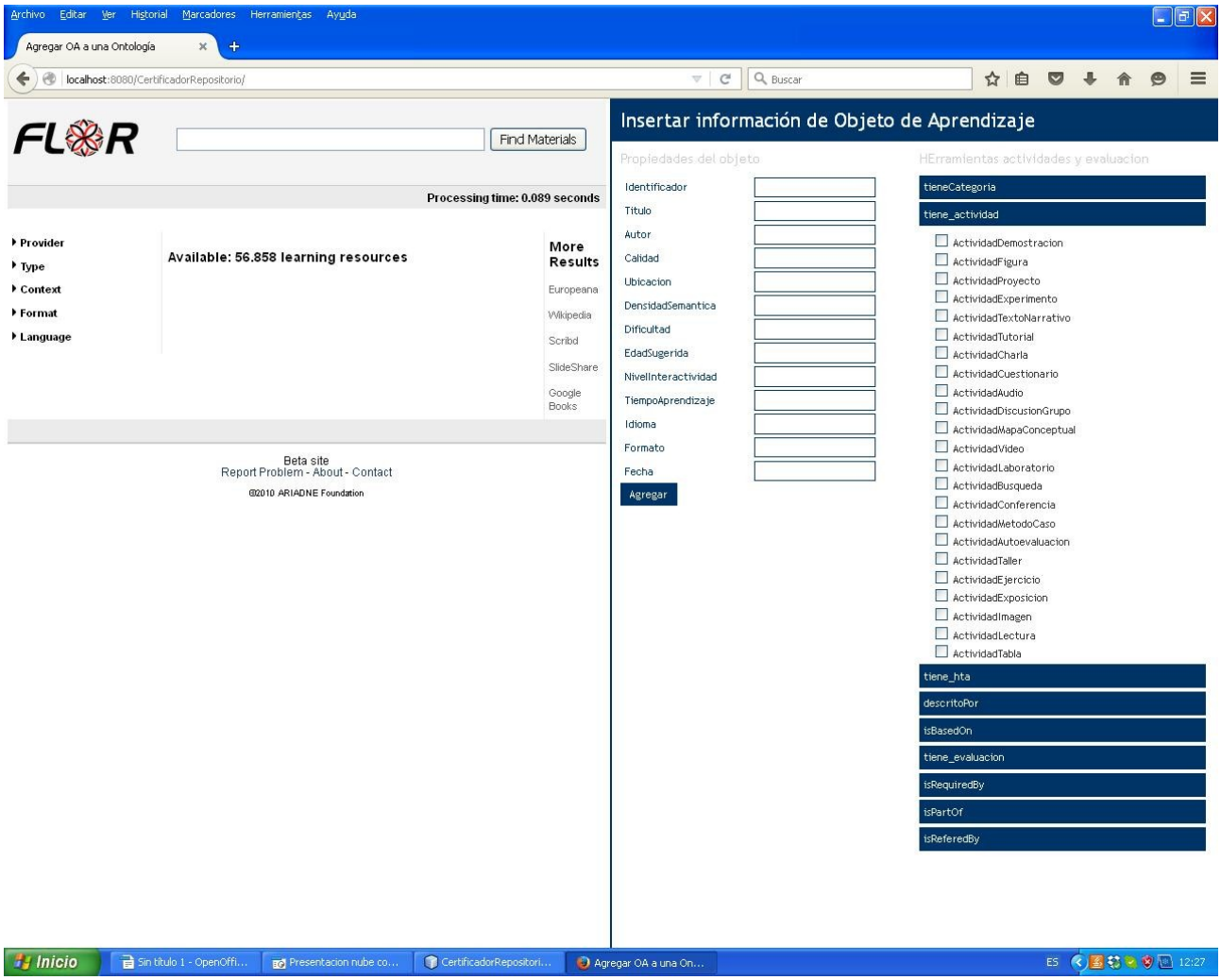

Figura 4.5 Aplicación certificadora de OAs

4.2.2 Agente adaptativo de la usabilidad

Este agente se compone de dos elementos, que son: un repositorio de perfiles de usuario, en el que se almacena el conocimiento referente a las preferencias de uso de cada usuario y al uso de OAs que realiza, y un servicio web de usabilidad que

gestiona el conocimiento de todos los perfiles de usuario almacenados, el cual se encarga de calcular el aporte de personalización por usabilidad que el sistema recomendador asigna a cada objeto en el momento de rankear cada uno de los OAs que pertenecen al CONJUNTO DE RECOMENDACIÓN.

4.2.2.1 Repositorio de perfiles de usuario

El repositorio de perfiles de usuario de este Agente consta fundamentalmente de un modelo ontológico, en el que se guarda el conocimiento de cada usuario. La Figura 4.6 muestra la ontología del repositorio de perfiles de usuario. En ese repositorio de perfiles de usuario se almacenan tres tipos de conocimiento, para determinar la usabilidad de un OA con respecto a cada estudiante, los cuales son: información personal del estudiante, las preferencias de uso de cada estudiante, y el historial de uso de cada uno de ellos.

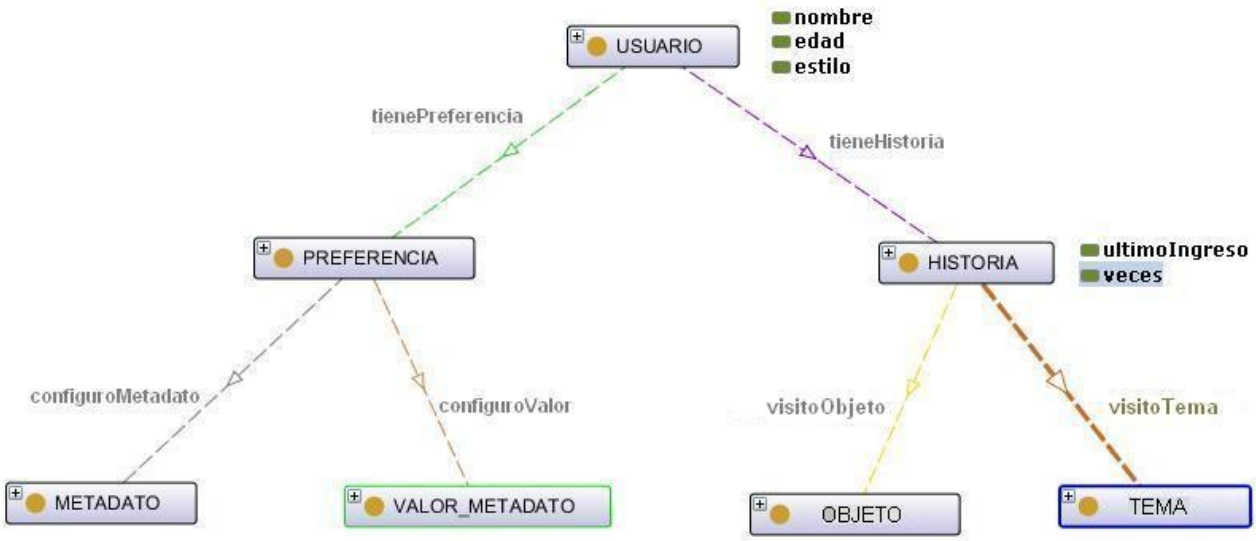

Figura 4.6 Modelo ontológico del repositorio de perfiles de usuarios.

4.2.2.1.1 Información personal

Contempla los criterios asociados con la información personal de cada estudiante, los cuales son: nombre, edad y estilo de aprendizaje. El estilo de aprendizaje del estudiante está codificado en un valor numérico compuesto por cuatro dígitos, donde cada uno de los dígitos representa una categoría dentro de lo propuesto por el modelo de estilos de aprendizaje de Felder y Silverman. Está codificación es realizada por la nube de auto-formación en el momento de realizar la inscripción del estudiante (ver [5] para más detalles de ese proceso). Para ello, la nube de

auto-formación le realiza el test de estilos de aprendizaje de Felder y Silverman al estudiante, y de acuerdo con los resultados obtenidos en el test estipula un valor numérico compuestos por cuatro dígitos $\mathbf{d_1 d_2 d_3 d_4}$, cada dígito d_i tiene un valor entre 1 y 5, que se asocia con cada una de las cuatro categorías consideradas en el modelo de Felder y Silverman [5], tal como se muestra en la Tabla 4.4.

CATEGORÍA	VALOR DÍGITO d_i	INTERPRETACIÓN
SENSORIAL-INTUITIVO	$d_1=1$	Altamente sensorial
	$d_1=2$	Medianamente sensorial
	$d_1=3$	Indiferente
	$d_1=4$	Medianamente intuitivo
	$d_1=5$	Altamente intuitivo
ACTIVO-REFLEXIVO	$d_2=1$	Altamente activo
	$d_2=2$	Medianamente activo
	$d_2=3$	Indiferente
	$d_2=4$	Medianamente reflexivo
	$d_2=5$	Altamente reflexivo
VISUAL-VERBAL	$d_3=1$	Altamente visual
	$d_3=2$	Medianamente visual
	$d_3=3$	Indiferente
	$d_3=4$	Medianamente verbal
	$d_3=5$	Altamente verbal
SECUENCIAL-GLOBAL	$d_4=1$	Altamente secuencial
	$d_4=2$	Medianamente secuencial
	$d_4=3$	Indiferente
	$d_4=4$	Medianamente global
	$d_4=5$	Altamente global

TABLA 4.4 INTERPRETACIÓN DEL VALOR NUMÉRICO USADO PARA REPRESENTAR EL ESTILO DE APRENDIZAJE DEL ESTUDIANTE.

4.2.2.1.2 Preferencias de uso (PU)

Las preferencias de uso son aquellas configuraciones iniciales determinadas por cada estudiante en el momento de su inscripción en la nube de auto-formación. Cada preferencia consta de una dupla (Metadato, Valor) configurada para un usuario en particular. Un estudiante (usuario) selecciona de entre los metadatos existentes en el modelo ontológico del repositorio semántico mostrado en la Figura 4.3, la cantidad de metadatos que considere conveniente, y para cada uno de ellos determinar su valor de preferencia, o su rango de valores de preferencia (tomados del listado de valores permitidos contemplado en el repositorio).

En la actualidad solo se cuentan implementadas dos tipos de preferencias de uso, que son: tipo de formato del objeto y máxima fecha de antigüedad del objeto. Así,

un estudiante puede configurar las dos preferencias asociando a cada preferencia un metadato y un valor para cada metadato, de la siguiente manera: (tec_formato,pdf) y (fecha,"01-01-2010"). Estas preferencias son usadas para calcular el *aporte por la personalización por preferencias de uso,* en el momento de personalizar la recomendación.

4.2.2.1.3 Historial de uso (HU)

El historial de uso se compone de las historias almacenadas en el modelo ontológico del repositorio de perfiles de usuario mostrado en la Figura 4.6. De cada historia se conocen cuatro aspectos, que son:

- OA usado
- Tema estudiado con dicho OA
- Número de veces que el estudiante ha usado el OA para estudiar el tema asociado a un modulo dado dentro de la nube de auto-formación.
- Ultima fecha en que el estudiante ha usado el OA para estudiar el tema asociado a un modulo dado dentro de la nube de auto-formación.

Estas preferencias son usadas para calcular el *aporte por la personalización por historias de uso,* en el momento de personalizar la recomendación. Cada historia de uso es almacenada en el modelo ontológico como una cuatrupla compuesta por (objeto, tema, veces, ult_fecha), donde objeto indica con que OA estudio el usuario el tema dado, y veces determina cuantas veces ese usuario ha estudiado el mismo tema con exactamente el mismo OA. Finalmente, ult_fecha registra la ultima fecha en que el estudiante estudió ese tema con ese OA. Es importante resaltar que todo el conocimiento almacenado como historias de usuario proviene de la dinámica presente en la nube de auto-formación.

Existen diferencias entre los tipos de conocimiento de usabilidad almacenados en el repositorio de perfiles de usuario. El conocimiento de preferencias de usuario es establecido por el estudiante al momento de inscribirse en la nube de auto-formación, quien lo configura de forma consciente y voluntaria. La configuración especifica las preferencias de cada estudiante asociadas a los metadatos de los OAs con los que desea estudiar. El historial de uso almacena conocimiento que no

registró el estudiante conscientemente, sino que se generó según la dinámica de uso que ha tenido el estudiante al interior de la nube de auto-formación. Este conocimiento se almacena automáticamente, en la medida que el usuario va usando un OA para estudiar cierto tema, sin que el estudiante sea consciente de ello.

4.2.2.2 Servicio web de usabilidad (WSUsabilidad)

Como se mencionó, este Agente tiene un servicio web, denominado servicio web de usabilidad. Dicho servicio es invocado de cinco formas distintas:

- Por el sistema recomendador, para obtener el conocimiento almacenado del estudiante
- Por el sistema recomendador, para calcular el aporte de personalización por usabilidad.
- Por la nube de auto-formación, para registrar información personal del usuario
- Por la nube de auto-formación, para registrar una preferencia de usuario
- Por la nube de auto-formación, para registrar una historia de uso

Para realizar la primera tarea, el servicio web de usabilidad recibe como parámetro de entrada un identificador de usuario, con el cual acude al sistema de razonamiento del modelo ontológico de la Figura 4.6 para consultar el repositorio de perfiles de usuario. Una vez consultado el repositorio de perfiles de usuario, el servicio web retorna un objeto tipo Usuario, que contiene la descripción del estudiante almacenada (ver Figura 4.7).

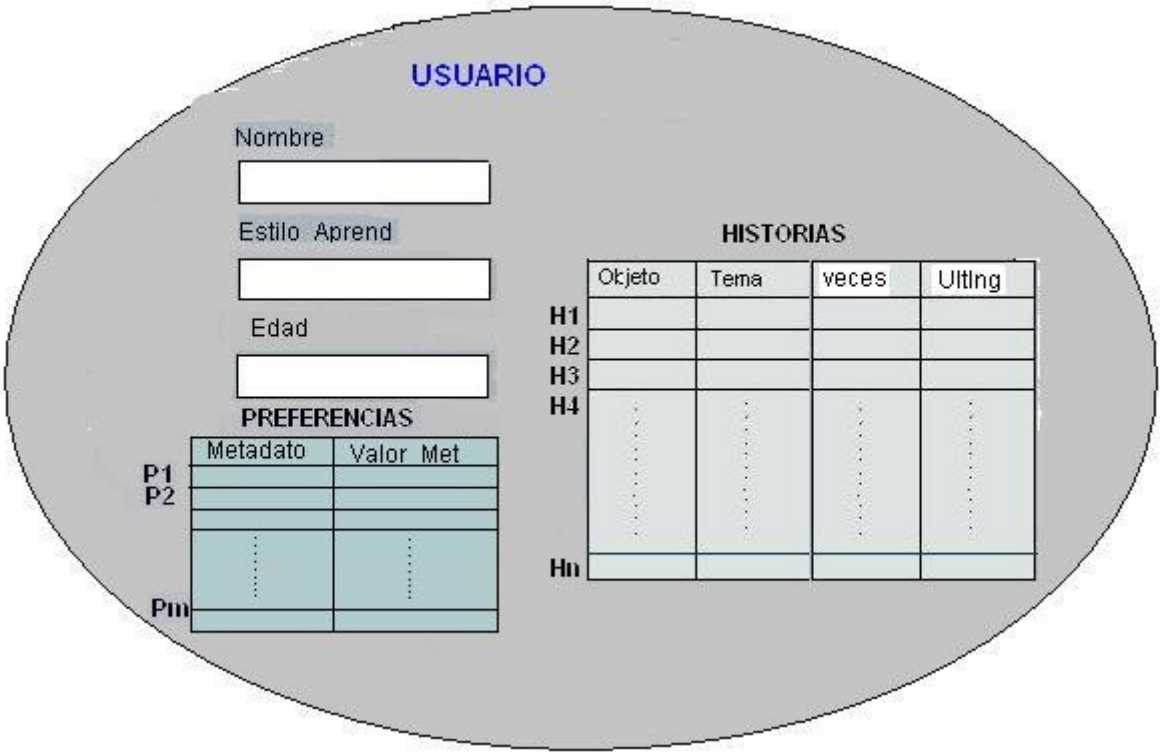

Figura 4.7 Esquema general del objeto usuario retornado por el servicio web de usabilidad, cuando se le consulta información del estudiante

La segunda tarea consiste en calcular el *aporte de personalización por usabilidad (APU)*. APU es un valor numérico entre 0 y 1, y se calcula a partir de dos los componentes de usabilidad, denominados: *aporte parcial de personalización por preferencias de uso (APPP)* y *aporte parcial de personalización por historias de uso (APPH)*. La Figura 4.7 esquematiza el conocimiento que el repositorio perfiles de usuario retorna al agente adaptativo de usabilidad para calcular el aporte de APU. Como se aprecia en la Figura 4.7, para cada usuario se obtiene la información personal del usuario (incluido su estilo de aprendizaje), el listado de historias de uso de cada OA y para cual tema se usó, el número de veces que fue usado, y la fecha de la última vez que fue usado. Además, de cada usuario solicitado se conocen sus preferencias (metadato de la preferencia y valor del mismo).

APPP consiste en un valor numérico entre 0 y 1, que indica el grado (porcentaje) de emparejamiento entre el valor de cada uno de los metadatos configurados por el usuario como su preferencia, y su valor de metadato asociado en el OA. Un 0 indica que no existe ningún grado de emparejamiento entre las preferencias del usuario con los metadatos del OA, y un 1 indica un total emparejamiento. La

ecuación (4.1) indica la formula para calcular APPP. Como se observa, *APPP* se calcula para un determinado objeto de aprendizaje O_i y un determinado usuario U_k, y arroja un valor entre 0 y 1 que indica que tanto influye la personalización por preferencias de usuario en el cálculo de la utilidad del objeto de aprendizaje O_i.

$$APPP(O_i, U_k) = \frac{\sum_{k=1}^{np} VP_{i,k}}{np} \quad (4.1)$$

Donde:

- np número de preferencias que el usuario configuró en el momento de inscribirse en la nube de auto-aprendizaje.
- $VP_{i,k}$ valor de emparejamiento entre la preferencia asociado al objeto i y la preferencia del usuario k, calculado así: 1 si empareja la preferencia j del usuario con el valor del metadato asociado en el objeto i, 0 en caso contrario.

APPH consiste en un valor numérico entre 0 y 1, que se calcula teniendo en cuenta cierta cantidad de indicadores de uso IU. Un INDICADOR DE USO (IU) es un criterio almacenado en cada una de las historias de usuario en el repositorio de perfil de usuario. Para el cálculo del APPH asociado con un OA, en particular se utilizan tres indicadores de uso, que son:

IU_1: número de veces en que el estudiante ha usado el objeto. El objetivo de este indicador de uso es determinar el rango percentil del uso que ha dado el estudiante a este objeto, respecto del número de veces que ha usado otros objetos a lo largo de su historial. Con este indicador de uso se busca establecer que tan frecuentemente se ha usado el OA, asumiendo que a mayor frecuencia de uso más recomendable es usarlo para el estudiante.

IU_2: ultima fecha en que el estudiante uso el AO para estudiar el tema. Este indicador de uso busca determinar el rango percentil de la fecha de uso respecto de las fechas de uso de cada historia almacenada en su historial. Con este indicador de uso se busca establecer el impacto del objeto respecto de que tan recientemente ha sido usado. Se asume que entre más recientemente haya usado el OA ese estudiante, más recomendable es volverlo a usar.

IU_3: Este indicador determina que tanto el estudiante ha usado el OA para estudiar el tema solicitado. Se determina calculando la proporción entre el número de veces que el estudiante ha usado el OA para estudiar el tema solicitado, respecto del total de veces que el estudiante ha usado cualquier OA para estudiar el tema. El indicador establece mayor peso a aquellos objetos que han sido usados más veces por el estudiante para estudiar dicho tema.

El RANGO PERCENTIL ASOCIADO A UN INDICADOR DE USO (RP(IU)) es una valor estadístico que indica el nivel de influencia del valor asociado al indicador de uso en el objeto al cual se le calcula AAPH. El rango percentil es una forma de comparar el valor asociado al indicador de uso de ese OA, respecto a los valores asociados para el mismo indicador de uso en el resto de los objetos almacenados en el historial de usuario del estudiante que solicitó la recomendación. El rango percentil para cada indicador de uso es un número entre 0 a 1, que indica el porcentaje de objetos cuyo indicador de uso son iguales o menores que el indicador de uso del objeto para el cual se está calculando el APPH. En otras palabras, dice cual es el nivel de influencia del indicador de uso de ese objeto, respecto del total de objetos almacenados en el historial del usuario. Por ejemplo, si un indicador de uso tiene registrado que un OA ha sido usado 10 veces por ese estudiante, es necesario determinar que tanto representa esta información respecto del historial de uso de dicho estudiante. En la medida que el rango percentil se acerque más a 1, indica que el valor de 10 veces de usar dicho OA en el historial de ese estudiante es muy significativo.

La ecuación (4.2) indica la formula para calcular APPH. Como se observa, *APPH* se calcula para un determinado objeto de aprendizaje O_i y un determinado usuario U_k, y arroja un valor entre 0 y 1 que indica que tanto influye la personalización por historias del usuario en el cálculo de la utilidad del objeto de aprendizaje O_i.

$$APUH(O_i, U_k) = \frac{(\sum_{m=1}^{N_{IU}} RP(IU_{m,k}))}{N_{UI}} \quad (4.2)$$

Donde:

- N_{IU} es el número de indicadores de uso
- $RP(IU_{m,k})$ es el rango percentil calculado para el indicador de uso m del usuario k.

Finalmente, este agente calcula *APU.* Como se dijo antes, tal aporte le indica al recomendador mediante un valor numérico entre 0 y 1, que tanto un objeto de aprendizaje O_i es indicado para un determinado usuario U_k en lo referente a usabilidad. La ecuación (4.3) muestra la forma de calcular APU.

$$APU(O_i, U_k) = APPP(O_i, U_k) * 0{,}5 + APPH(O_i, U_k) * 0{,}5 \quad (4.3)$$

donde:

$APPP(O_i, U_k)$ es calculado según la ecuación (4.1)

$APPH(O_i, U_k)$ es calculado según la ecuación (4.2)

A pesar de que este Agente pertenece a la nube de conocimiento, él interactuá permanentemente con la nube de auto-formación. Así, la nube de auto-formación permanentemente invoca métodos del servicio WSUsabilidad, para almacenar/actualizar el conocimiento que se va generando dentro del repositorio de perfil de usuario (ver en Tabla 4.5, los métodos de ese servicio que se invocan)

Metodo en WSUsabilidad	Descripción
registraInfo	Permite almacenar la información personal del estudiante (nombre, edad y estilo de aprendizaje). Este método solo lo invoca una vez la nube de auto-formación, durante el registro del estudiante en dicha nube.
registraPreferencia	Es un método invocado una vez por cada preferencia de uso que el estudiante desee registrar durante su inscripción.
registraHistoria	Es un método invocado por la nube de auto-aprendizaje, cada vez que el estudiante estudia algún tema con algún OA.

Tabla 4.5 Descripción de métodos para la incorporación de conocimiento en la nube de fuentes de conocimiento, usando WSUsabilidad

4.2.3 Agente adaptativo de la vinculación semántica de temas

La vinculación semántica de temas consiste en definir la relación de los temas de acuerdo a cuatro niveles de vinculación a considerar, que son: jerarquización, explicación, comparación y asociación. De esta manera, con la vinculación semántica se pretende potencializar la nube de conocimiento, con conocimiento que le permita no solo recomendar directamente el tema solicitado por la nube de auto-formación, sino además, recomendar temas vinculados semánticamente con

él. A continuación se explican cada uno de los cuatro niveles de vinculación semántica incorporados dentro de este Agente (ver Figuras 4.3 y 4.8).

4.2.3.1 Jerarquización

Esta relación de conceptos establece que el concepto A es un concepto incluido dentro de la jerarquía del concepto B. Con ello, la nube de conocimiento infiere que el concepto B es un concepto más genérico que el buscado, pero que puede ser recomendado, ya que este concepto permite conocer los aspectos fundamentales y más genéricos del concepto A solicitado. De esta manera, se establece una relación taxonómica entre los dos conceptos. La jerarquización se estipula mediante el objectProperty *es_un*.

4.2.3.2 Explicación

Esta relación busca expresar que un concepto C aclara o profundiza cierto aspecto del concepto A. Mediante este tipo de relación, la nube de conocimiento infiere que los conceptos relacionados mediante la relación *aclara* lo que buscan es profundizar o describir aspectos particulares del concepto A (concepto base). De esta manera, se establece una relación entre los dos conceptos. Para indicar que un concepto es explicado por otro, se recurre al objectProperty *aclara*.

4.2.3.3 Comparación

Este tipo de relación es expresada mediante el objectProperty *similarA*. Es una relación usada cuando dos conceptos tienden a ser similares en cuanto a funciones, composición o definición. En otras palabras, si el concepto A está relacionado con el concepto D mediante comparación, lo que se indica es que los conceptos A y D son muy parecidos, ya sea en cuanto a su uso, composición o definición.

4.2.3.4 Asociación

Este nivel de vinculación semántica muestra con que conceptos se puede asociar un concepto base. La asociación se realiza mediante el objectProperty *asociadoA*. Cuando un concepto A se asocia con un concepto E mediante esté relación, lo que se expresa es que dichos conceptos tienen algún grado de interrelación entre sí.

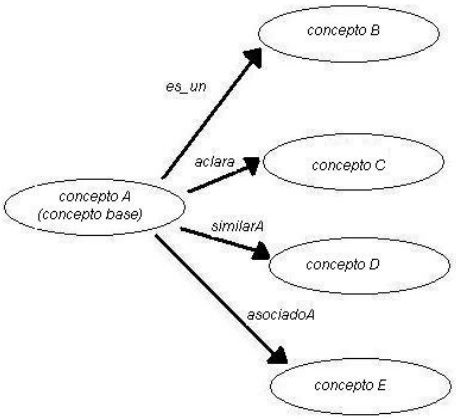

Figura 4.8 Niveles de descripción semántica usados en el modelo ontológico.

Este Agente se encarga de calcular el *aporte semántico cognitivo (ASC),* dicha funcionalidad se ofrece mediante un servicio web denominado WSSemántico. El proceso de cálculo del **ASC** consiste en determinar el NIVEL DE VINCULACION SEMANTICA (NV), el cual es un valor que indica el grado de vinculación entre un tema asociado a un OA respecto del tema solicitado, de acuerdo a los cuatro niveles de vinculación semántica antes explicados. La Tabla 4.6 ilustra la forma de calcular ASC.

NIVEL DE VINCULACION SEMANTICA (NV)	VAL	DESCRIPCION
TEMA BASE	4	El tema es el tema base
JERARQUIZACION	3	El tema está en el nivel de vinculación de jerarquización
EXPLICACION	2	El tema está en el nivel de vinculación de explicación
COMPARACION	1	El tema está en el nivel de vinculación de comparación
ASOCIACION	0	El tema está en el nivel de vinculación de asociación

Tabla 4.6 Determinación del nivel de vinculación semántica por el WSSemántico.

En general, para calcular ASC se usa la ecuación (4.4), que consiste en un valor entero comprendido entre 0 y 4. Usaremos un ejemplo para explicar como calcular

ASC para un OA cuyo tema solicitado es T_i, con base a las vinculaciones semánticas presentadas en la Figura 4.9, que se encuentran almacenadas dentro del repositorio de vinculación semántica de temas (es un grafo con la vinculación semántica entre temas).

$$ASC(O_i, T_k) = Nv(O_i, T_k) \quad (4.4)$$

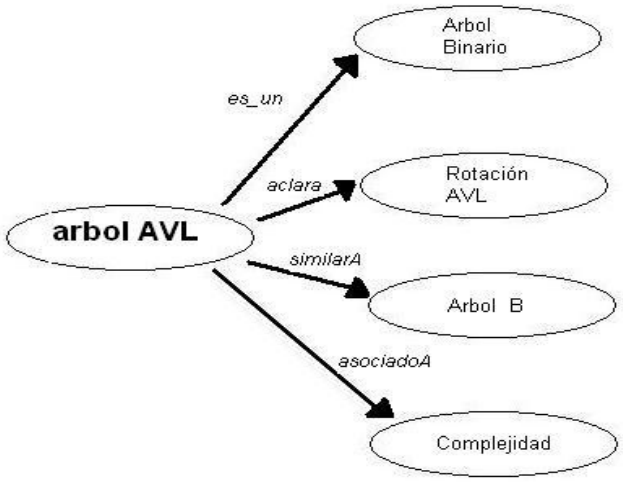

Figura 4.9 Ejemplo de vinculación semántica de temas.

De acuerdo con lo anterior, cuando se invoca el método *temasVinculados* de este Agente, se retorna la lista de temas vinculados semánticamente con él (ver Tabla 4.7).

Tema	Tipo Vinculación semántica	ASC
avl	CONCEPTO BASE	4
arbol_binario	JERARQUIZACIÓN	3
rotacionAVL	EXPLICACIÓN	2
arbolB	COMPARACIÓN	1
complejidad	ASOCIACIÓN	0

TABLA 4.7 DETERMINACIÓN DEL ASC PARA LA VINCULACIÓN SEMÁNTICA DE LA FIGURA 4.9

La ontología mostrada en la Figura 4.10 muestra el conocimiento descrito en la Figura 4.9, y por medio de ella el WSSemántico se encarga de determinar el valor del ASC de acuerdo con los criterios estipulados en la Tabla 4.6.

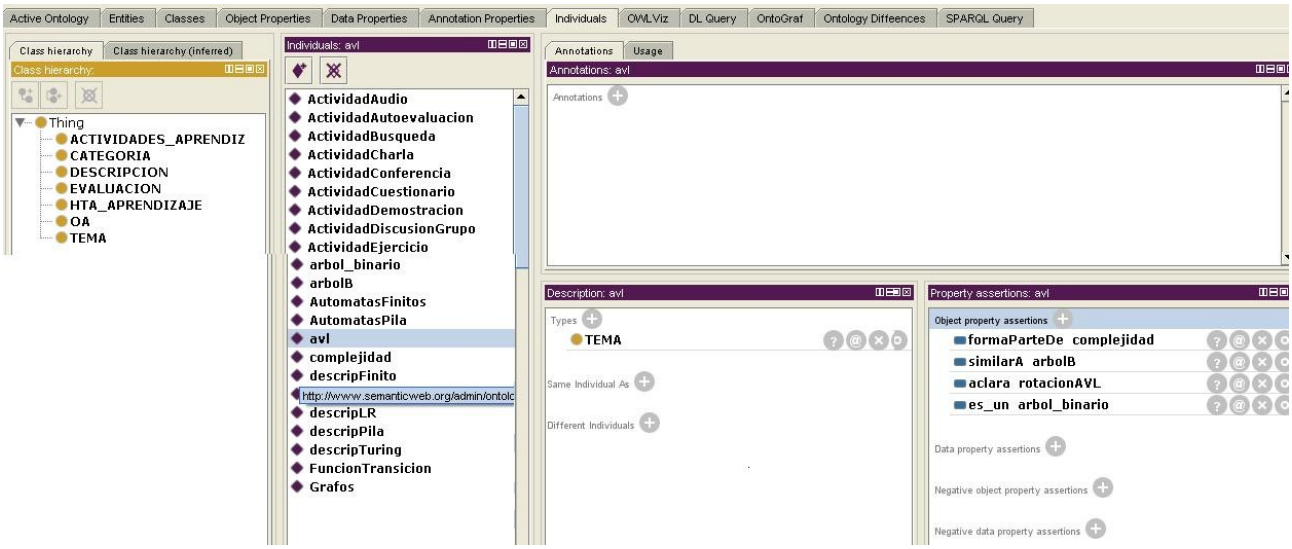

Figura 4.10 Almacenamiento de la vinculación semántica de temas del Agente Adaptativo de la vinculación semántica de temas.

4.2.4 Agente adaptativo de aspectos pedagógicos

Este Agente es el encargado de almacenar y gestionar el conocimiento relacionado con los aspectos pedagógicos. Este componente se comunica directamente con la nube de aprendizaje descrita en [5], para solicitar información de qué herramientas de aprendizaje, qué instrumentos de evaluación, y qué actividades de aprendizaje, son idóneas para el estilo de aprendizaje que corresponde al estudiante, de acuerdo por lo planteado por Felder y Silverman en [5].

El agente adaptativo de aspectos pedagógicos lleva a cabo un proceso interno mediante el cual realiza el cálculo del *aporte de personalización pedagógica (*APP). El APP es un valor numérico calculado para cada objeto de aprendizaje $\mathbf{O_i}$, dicho valor se encuentra entre 0 y 1, e indica que tanto emparejan las herramientas de aprendizaje, instrumentos de evaluación y actividades de aprendizaje de un $\mathbf{O_i}$, respecto a las herramientas de aprendizaje, instrumentos de evaluación y actividades de aprendizaje, consideradas por la nube de aprendizaje [5] como idóneas para el usuario U_k que realiza la solicitud de recomendación. La Figura 4.11 ilustra el proceso de cálculo del APP. Como se observa, el agente adaptativo debe invocar a una clase llamada *ConsultorNubeAprendizaje,* que sirve de cliente para la nube de aprendizaje, a través de la cual se determina el conjunto de actividades de aprendizaje, herramientas de aprendizaje y métodos de evaluación, idóneos para el estilo de aprendizaje del estudiante que realiza la solicitud. Una

vez el agente adaptativo conoce el conjunto de herramientas, actividades y métodos de evaluación más indicados, invoca los métodos *armaActividad*(), *armaHerramienta*() y *armaInstrumento*(), para calcular el valor normalizado de las diferencias simétricas entre el conjunto de actividades, herramientas e instrumentos de evaluación consideradas ideales para el estudiante y el conjunto de actividades, herramientas e instrumentos de evaluación que tiene registrados como metadatos el objeto de aprendizaje O_i para el cual se esta calculando el APP (ver ecuación (4.5).

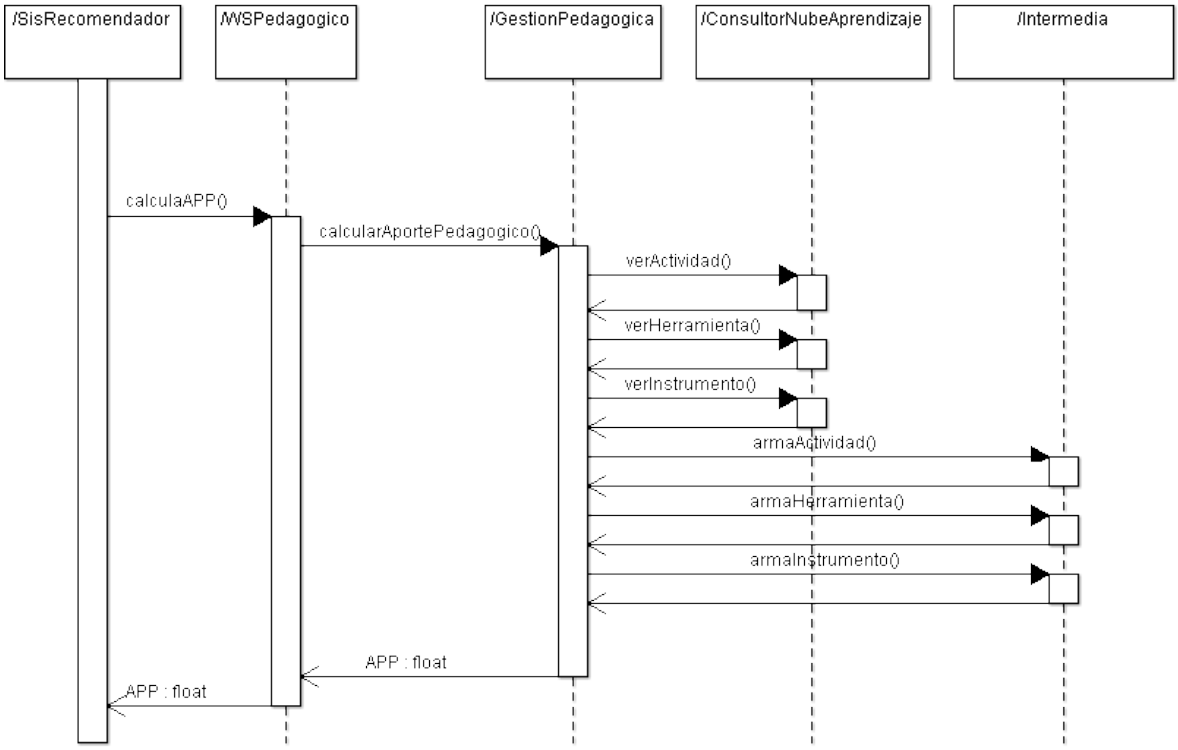

Figura 4.11 Diagrama de secuencias requerido en el Agente Adaptativo Pedagógico.

El cálculo de APP se ilustra en la ecuación (4.5).

$$APP(O_i, U_K, T_j) = \frac{(|H_p \cap H_o| + |A_p \cap A_o| + |I_p \cap I_o|)}{(|H_p| + |A_p| + |I_p|)} \quad (4.5)$$

donde:

- H_p es el conjunto de Herramientas que la nube de aprendizaje considera convenientes para el estilo de aprendizaje del estudiante U_k.
- H_o es el conjunto de Herramientas que el OA O_i tiene registrados en el repositorio semántico de OAs

- A_p es el conjunto de Actividades que la nube de aprendizaje considera convenientes para el estilo de aprendizaje del estudiante U_k.
- H_o es el conjunto de Actividades que el OA O_i tiene registrados en el repositorio semántico de OAs
- I_p es el conjunto de Instrumentos de evaluación que la nube de aprendizaje considera convenientes para el estilo de aprendizaje del estudiante U_k.
- I_o es el conjunto de Instrumentos de evaluación que el OA O_i tiene registrados en el repositorio semántico de OAs
- $|H_p \cap H_o|$ Es la cardinalidad de la intersección entre los conjuntos H_p y H_o . Es decir, es el número de herramientas de aprendizaje comunes entre las herramientas consideradas por la nube de aprendizaje ideales para el usuario U_k y lo registrado para el objeto O_i
- $|A_p \cap A_o|$ Es la cardinalidad de la intersección entre los conjuntos A_p y A_o . Es decir, es el número de actividades de aprendizaje comunes entre las actividades consideradas por la nube de aprendizaje ideales para el usuario U_k y las registradas para el objeto O_i.
- $|I_p \cap I_o|$ Es la cardinalidad de la intersección entre los conjuntos I_p y I_o . Es decir, es el número de instrumentos de evaluación comunes entre los instrumentos considerados por la nube de aprendizaje ideales para el usuario U_k y lo registrado para el objeto O_i.

Este Agente debe ser retomado en trabajos futuros, para incorporarle nuevos modelos de conocimiento y técnicas de aprendizaje, que le permitan considerar aspectos pedagógicos adicionales que aporten mayor grado de personalización y adaptabilidad.

4.2.5 Agente adaptativo de la calidad y evaluación de desempeño

Es el componente arquitectónico de la nube de conocimiento que está propuesto, para que en trabajos futuros se establezca la relación directa con la nube de auto-formación. Posee un servicio web WSEvaluación que gestiona todo el conocimiento relacionado con la calidad y la evaluación de desempeño de cada

uno de los OAs que se encuentran certificados, cuyos metadatos están almacenados en el repositorio semántico de OAs. Se proponé que a futuro se realicen trabajos al respecto, donde con lo surgido de la dinámica de la nube de auto-formación se vaya cambiando una medida de calidad del OA, dependiendo de factores como: el tipo de estudiante que estudió el objeto, los resultados de la evaluación realizada a dicho estudiante después de estudiar el objeto, la comparación de lo obtenido con el objeto evaluado respecto del desempeño con OAs similares, entre otros factores.

Este Agente calcula el *aporte colaborativo* (AC), calculado de acuerdo con la ecuación (4.6). En este momento es establecido por una calificación cuantitativa de calidad dada por el usuario experto, cuando certifica el objeto, la cual es estática, y no se actualizá con la dinámica de la nube de auto-formación. Así, la interacción etiquetada con B en la Figura 4.2 está propuesta para que en trabajos futuros el agente Adaptativo de Calidad y Evaluación de Desempeño interactué con la nube de auto-aprendizaje bajo distintos esquemas. Por ejemplo, bajo un esquema colaborativo o un proceso de aprendizaje.

$$AC(O_i, U_k) = Fcal(O_i, U_k) \quad (4.6)$$

4.2.6 Sistema recomendador de OAs

Se trata del componente arquitectónico central de la nube de conocimiento, tal como se aprecia en la Figura 4.2. Es en el sistema recomendados de OAs (SR) donde se soporta toda la dinámica de interacción de la nube de conocimiento. Es el encargado de interactuar con cada uno de los demás componentes de la nube, y con la nube de auto-formación, quien acude a solicitarle el servicio web semántico de recomendación de que OAs sirven para que cierto estudiante U_k estudie cierto tema T_k.

El sistema recomendador fue implementado como un servicio web semántico, y diseñado usando el Framework FODAS-WS presentado como parte de esta tesis. Se trata de un sistema recomendador híbrido calibrado de OAs, que recomienda a la nube de auto-formación un conjunto rankeado de OAs, al que se le denomina CONJUNTO DE RECOMENDACIÓN.

El sistema recomendador de la nube de conocimiento es híbrido, porque combina tres distintos tipos de recomendaciones:

- Recomendación basada en contenidos: Esta dada cuando se emparejan los contenidos de los OAs con lo requerido por el usuario para el cual se realiza la recomendación. El agente adaptativo por usabilidad calcula parte de este tipo de recomendación, al calcular el aporte por usabilidad en cuanto a preferencias de uso. De igual forma, el agente adaptativo pedagógico contribuye en la recomendación basada en contenidos, al calcular el aporte pedagógico.
- Recomendación colaborativa: Se encuentra en el Agente Adaptativo de la calidad y evaluación de rendimiento, quien define una medida de calidad basada en las experiencias de uso anteriores, que han generado los usuarios que han atendido a la recomendación de dicho elemento.
- Recomendación cognitiva: Se ve reflejada en el Agente Adaptativo de la vinculación semántica de temas. Este sistema no solo recomienda los temas directamente solicitados, sino temas que mediante el conocimiento almacenado en el Agente, tienen alguno de los cuatro niveles de vinculación semántica existentes.

Es importante resaltar que el sistema recomendador de OAs de la nube de conocimiento es un sistema recomendador híbrido *calibrado*. Lo anterior implica que quien solicite la recomendación, tiene la posibilidad de calibrar el peso que se le da a cada componente (agente). Para ello se utiliza un parámetro h, denominado factor de hibridación o de calibración, el cual puede contener valores comprendidos entre 0 y 1. Calibrar el recomendador con un valor de h=0 implica que la recomendación no tendrá en cuenta el factor colaborativo, y solo considerará la personalización del usuario en la recomendación. En el otro extremo, al calibrar el recomendador con un valor de h=1, implica que la recomendación realizada no tendrá en cuenta el aporte de personalización del usuario, sino únicamente el aspecto colaborativo representado por la calidad y evaluación de desempeño. El aspecto cognitivo no es objeto de calibración, debido a que el recomendador prioriza los OAs que enseñan directamente el tema

solicitado, y luego prioriza de acuerdo a los niveles de vinculación semántica mostrados en la Tabla 4.6.

El sistema recomendador se diseñó e implementó como un servicio web semántico, con el objetivo de permitir que a futuro se pueda realizar descubrimiento, composición y ejecución automática, por parte de cualquier cliente que requiera una recomendación de OAs. Cada vez que la nube de auto-formación requiere una recomendación, está realiza una solicitud al sistema recomendador, como lo muestra la Figura 4.2. El sistema recomendador retorna una colección de objetos rankeados, con base a un valor de utilidad calculado por el recomendador. La Figura 4.12 muestra el pseudocódigo del proceso de recomendación, el cual realiza las siguientes fases:

- fase de inferencia del CONJUNTO DE RECOMENDACIÓN (paso 2 del pseudocódigo)
- fase de adaptatividad de la recomendación (resto de pasos del pseudocódigo)

4.2.6.1 Fase de inferencia del CONJUNTO DE RECOMENDACIÓN

La fase de inferencia del CONJUNTO DE RECOMENDACIÓN es la primera fase, y consiste en construir el conjunto de OAs que se van a recomendar, al que se le llamará en adelante CONJUNTO DE RECOMENDACIÓN. El CONJUNTO DE RECOMENDACIÓN se compone de aquellos objetos que enseñan el tema solicitado en la recomendación, o temas que se encuentren vinculados semánticamente a él. El sistema de inferencia (proporcionado por un razonador de lógica descriptiva) es el encargado de la construcción del CONJUNTO DE RECOMENDACIÓN. Cada uno de los O_i miembros del CONJUNTO DE RECOMENDACIÓN es un OA almacenado en el repositorio semántico de OA (explicado en la sección 4.2.1) con el que el sistema de inferencia determina que el estudiante U_k (para quien la nube de auto-formación solicito la recomendación) puede estudiar cierto tema T_k (porque está vinculado semántica, según los explicados en la sección 4.2.3, con el tema solicitado). Como se muestra en el pseudocódigo de la Figura 4.12, la fase acá

explicada constituye la primera etapa realizada dentro del proceso de recomendación (realizada en la linea 2), y es el conjunto inferido en ella quien sirve como insumo para generar la adaptatividad a la recomendación solicitada.

4.2.6.2 Fase de adaptatividad de la recomendación

Es la fase durante la cual el sistema recomendador acude a los diversos agentes adaptativos para lograr una recomendación personalizada acorde con las características del estudiante que la solicita. La linea 3 del pseudocódigo mostrado en la Figura 4.12 determina que dependiendo de si el CONJUNTO DE RECOMENDACIÓN inferido en la fase anterior cuenta con objetos o no, se generan dos distintos procesos, que son:

- proceso de recomendación de OAs certificados.
- proceso de recomendación de OAs no certificados.

4.2.6.2.1 Proceso de recomendación de OAs certificados

Este proceso se lleva a cabo cuando la condición de la estructura de selección de la linea 3 del pseudocódigo mostrado en la Figura 4.12 es verdadera. Es decir, el proceso de recomendación de OAs certificados se realiza cada vez que el sistema de inferencias logró inferir/crear un CONJUNTO DE RECOMENDACIÓN usando el repositorio semántico de OAs. Lo anterior implica, que es en este proceso donde se adapta la recomendación con las características especificas del estudiante que la solicito. Esa adaptación consiste en ordenar los OAs según la relación entre las características de los usuarios y de los OAs. A mayor relación/emparejamiento, mejor clasificados quedan los OAs. El proceso cuenta con dos subprocesos, que son:

- Subproceso del cálculo de la utilidad de cada objeto O_i
- Subproceso de rankeo del CONJUNTO DE RECOMENDACIÓN, el cual se basa en el anterior cálculo.

```
1. INICIO
2.        Cr= Inferir(ROA)
3.        SI Cr NO VACIO
4.              PARA CADA Oi que pertenezca a Cr HACER
5.                    calcularUtilidad(Oi)
6.              FIN PARA
7.              Rec= Rankear(Cr)
8.        SINO
9.              Rec=consultarNoCertificado()
10.       Retornar(Rec)
11. FIN
```

Donde:

C_r = Conjunto de recomendación
ROA = Repositorio semántico de objeto de aprendizaje certificados
O_i = Objeto de aprendizaje evaluado dentro de C_r
Rec = Recomendación realizada al cliente

Figura 4.12 Proceso de recomendación llevado a cabo por el sistema recomendador.

4.2.6.2.1.1 Subproceso de cálculo de utilidad para cada objeto del CONJUNTO DE RECOMENDACIÓN

Este subproceso consiste en calcular un valor numérico comprendido entre 0 y 5, asociado a cada OA O_i que pertenezca al CONJUNTO DE RECOMENDACIÓN (construido en el subproceso anterior). Dicho valor representa el nivel de utilidad que el OA O_i tiene para el estudiante U_k, para el cual la nube de auto-formación realiza la solicitud de recomendación. El subproceso de calculo de utilidad se realiza en el ciclo repetitivo, entre las lineas 4 – 6 del pseudocódigo mostrado en la Figura 4.12.

La ecuación (4.7) muestra la definición formal de la función de utilidad, con la cual el sistema recomendador calcula el valor de la utilidad para cada OA O_i que pertenezca al CONJUNTO DE RECOMENDACIÓN. El valor calculado por la función de utilidad es de vital importancia, pues es el criterio mediante el cual el recomendador sugiere cuales objetos son más recomendables para que un

estudiante U_k estudie determinado tema, de acuerdo con la solicitud hecha por la nube de auto-formación. La función de utilidad es el mecanismo encargado de aportar adaptatividad a las recomendaciones, y constituye la columna vertebral del recomendador, pues integra los diversos componentes arquitectónicos explicados en la sección 4.1.

$$f(O_i, U_k) = ASC(O_i) + \Big(\big(\big(APU(O_i, U_k)*0{,}5 + APP(O_i, U_k)*0{,}5 \big) * (1\text{-}h) \big) + AC(O_i, U_k)*h \Big) \quad (4.7)$$

donde:

$f(O_i, U_k)$ es la utilidad calculada para el objeto O_i, según las características del usuario U_k

$ASC(O_i)$ es calculado con la ecuación (4.4)

$APU(O_i, U_k)$ es calculado con la ecuación (4.3)

$APP(O_i, U_k)$ es calculado con la ecuación (4.5)

$AC(O_i, U_k)$ es calculado con la ecuación (4.6)

h es el *parámetro de calibración (hibridación),* entregado por la nube de auto-formación en el momento de realizar la solicitud.

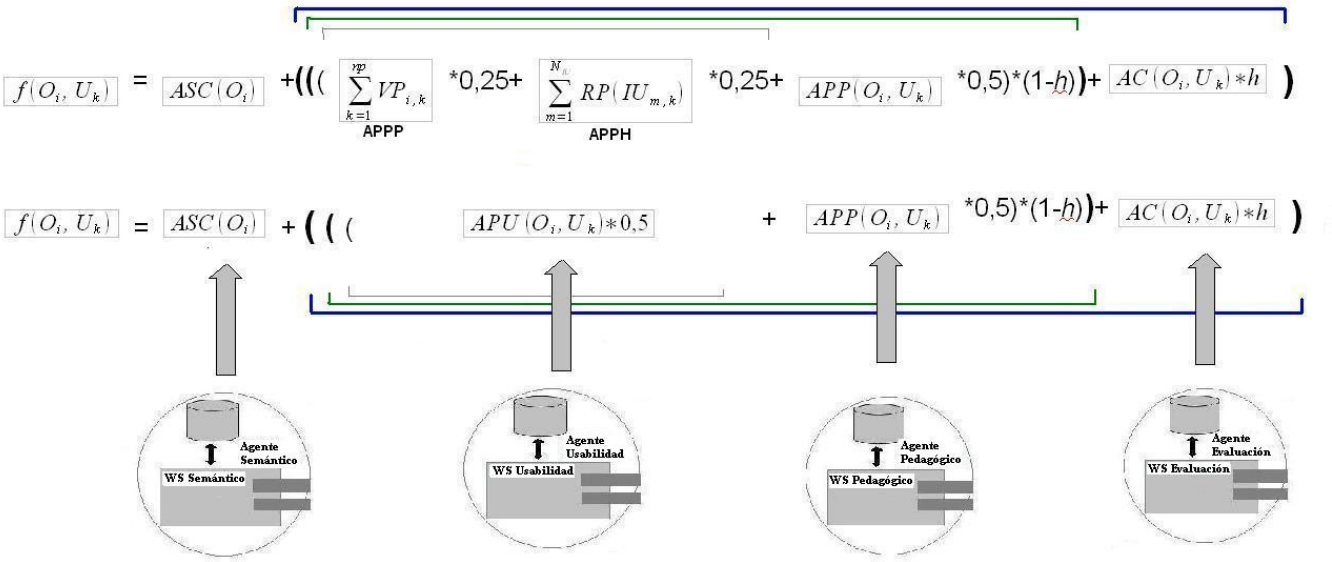

Figura 4.13 Composición de la función de utilidad del sistema recomendador.

La Figura 4.13 estipula a cual de los agentes adaptativos le corresponde cada uno de los aportes que constituyen la función de utilidad. Es importante recordar que *ASC* no se calibra.

Como ya se explicó, a medida que el valor de h crece el impacto de la personalización en la utilidad de un objeto disminuye, y el impacto del aporte colaborativo es mayor sobre la utilidad del objeto. Dentro del sistema recomendador de la nube de conocimiento, se pondera con igual peso el impacto que tiene *APU* y *APP* (cada uno tienen el 50% del total de aportes por personalización o adaptabilidad). Lo anterior no implica que dichos pesos deban permanecer estáticos en nuevas versiones del recomendador de OAs.

4.2.6.2.1.2 Subproceso de rankeo del CONJUNTO DE RECOMENDACIÓN

Este subproceso constituye la etapa final del proceso de recomendación, y fundamentalmente consta de un ordenamiento realizado a los OAs que conforman el CONJUNTO DE RECOMENDACIÓN, usando como criterio de ordenamiento la utilidad calculada en el subproceso anterior. Como *ASC* no es afectado por la calibración, permite que en el momento de rankear (ordenar) el CONJUNTO DE RECOMENDACIÓN, los OAs que enseñan temas con niveles de vinculación semántica alto al tema solicitado, permanezcan en niveles de prioridad superior a los que enseñan temas con niveles de vinculación semántica inferiores .

4.2.6.2.2 Proceso de recomendación de OAs no certificados

Este proceso es llevado a cabo en caso de que la condición de la estructura de selección de la linea 3 en el pseudocódigo mostrado en la Figura 4.12 sea falsa. Es decir, el proceso de recomendación de OAs no certificados se realiza cada vez que en el repositorio semántico de OA (explicado en la sección 2.2.1) no se cuente con OAs certificados que enseñen un tema que se encuentre vinculado semánticamente (como se explicó en la sección 4.2.3) con el tema para el cual se realizó la solicitud de recomendación. Para realizar este proceso, el sistema recomendador acude a alguna federación de OAs disponible a nivel mundial, que cuente con un API de consulta para realizar una cosecha de metadatos básicos de OAs que enseñen el tema solicitado en la recomendación, los cuales son entregados al cliente advirtiéndole que no son certificados.

4.3 Diseño y especificación semántica del servicio web semántico que encapsula el sistema recomendador de OAs usando el framework FODAS-WS

Como ya se mencionó, el sistema recomendador de OAs se implementó como un servicio web semántico basado en la arquitectura ODA. En el capítulo 3 se presentó el Framework FODAS-WS desarrollado como parte de esta tesis, para el diseño de servicios web semánticos basados en ODA. Por tanto, el sistema recomendador de OAs fue diseñado y especificado semánticamente usando FODAS-WS. En ese sentido, FODAS-WS plantea en la arquitectura mostrada en la Figura 3.1, la existencia de tres capas ontológicas: CIM, PIM y PSM. En adelante se aborda el diseño y especificación semántica de cada una de las capas usando el Framework FODAS-WS, para el sistema recomendador de OAs.

4.3.1 Diseño y especificación de la capa CIM

La capa CIM, como ya se explicó en el capitulo 3, es la capa en la que se soporta el diseño del servicio web semántico. Se compone de tres subcapas, descritas a continuación para el caso del sistema recomendador de OAs.

4.3.1.1 Diseño de la capa de definición del dominio

El modelado del dominio para el sistema recomendador de OAs se realizó mediante el Framework FODAS-WS, haciendo uso de una ontología denominada ONTODOMINIO. El dominio sobre el cual opera el servicio web semántico que encapsula el sistema recomendador de OAs, fue modelado en la sección 3.1.2.1.1. La definición semántica de dicho dominio se condensa en la ontología ilustrada en la Figura 3.4. Como bien se explicó, en la capa de definición semántica de dominio se definen las distintas posibles acciones y las relaciones que pueden darse entre ellas, tal como lo expresa la Tabla 3.4.

El Framework FODAS-WS provee el almacenamiento del conocimiento del domino como un conjunto de ASOCIACIONES, las cuales relacionan dos elementos (conceptos) del dominio a través de una acción previamente descrita en la capa

de definición de dominio. Como ya se mencionó en el capitulo 3, la capa CIMREFERENCIAL usa las asociaciones para expresar las capacidades, necesidades, precondiciones y efectos, en forma de asociaciones.

4.3.1.2 Diseño de la capa CIM REFERENCIAL

El diseño y especificación semántica de la capa CIMREFERENCIAL se realiza usando el Framework FODAS-WS mediante el uso de la ontología denominada OntoCIMReferencialRec ilustrada en la Figura 4.14. Como se explicó en el capitulo 3, la capa CIMREFERENCIAL se encarga de almacenar el conocimiento relacionado con el modelo arquitectónico del servicio web semántico que se este diseñando. En la sección 3.1.2.1.2 se realiza la descripción de dicha capa. Como el sistema recomendador de OA se ajusta a la SOA, se deben describir cada una de las capacidades que el sistema recomendador es capaz de ofrecer a sus clientes en la capa CIMREFERENCIAL, así como las precondiciones que exige dicha capacidad, y los efectos que ocasiona en el entorno.

El sistema recomendador ofrece la capacidad de *Realizar_Recomendacion,* vinculada mediante el objectProperty *ofrece*. Como lo muestra la Figura 4.14, dicha capacidad tiene asociada la asociación *aso23* (definida previamente en la capa del dominio según lo explicado en la sección 3.1.2.3.1)*,* mediante el objectProperty *tieneAsociacion*. La asociación aso23 está previamente definida en la capa de definición del modelo, y establece que el *RecOA* (Recomendador de AO) recomienda (establecido mediante la acción *recomendar*) una recomendación (descrita mediante el elemento RECOMENDACION). La capacidad *Realizar_Recomendacion* se asocia mediante el objectProperty *ExigePrecondicion* con la precondición llamada *Recomendacion_Solicitada,* la cual es definida mediante el objectProperty *poseeAsociacion* con la asociacion *aso7*. De igual forma se definen las necesidades, usando las asociaciones previamente definidas en la capa CIMREFERENCIAL. Por ejemplo, la necesidad llamada *Solicitar_Recomendacion* se asocia con la asociación llamada *aso24,* a través del objectProperty *requiereAsociacion*. La definición de esta necesidad determina que se requiere una recomendación de un conjunto de OAs rankeados.

Lo anterior es de gran utilidad cuando una aplicación cliente desea realizar descubrimiento, composición y ejecución automática del sistema recomendador. Por ejemplo, para realizar descubrimiento automático lo que se debe hacer es emparejar la necesidad del cliente *Solicitar_Recomendación* (expresada como asociación) con la capacidad *Realizar_Recomendacion* descrita semánticamente en la ontología como la asociación aso23 (ver Figura 4.14). Para los procesos de descubrimiento y composición automática son de vital importancia los dataProperty *dirCapaDominio* y *dirCapaOperacional,* pues ellos son los encargados de relacionar las capacidades definidas en la capa CIMREFERENCIAL con las descripciones realizadas en la capa de modelado del dominio y en la capa CIMOPERACIONAL, respectivamente.

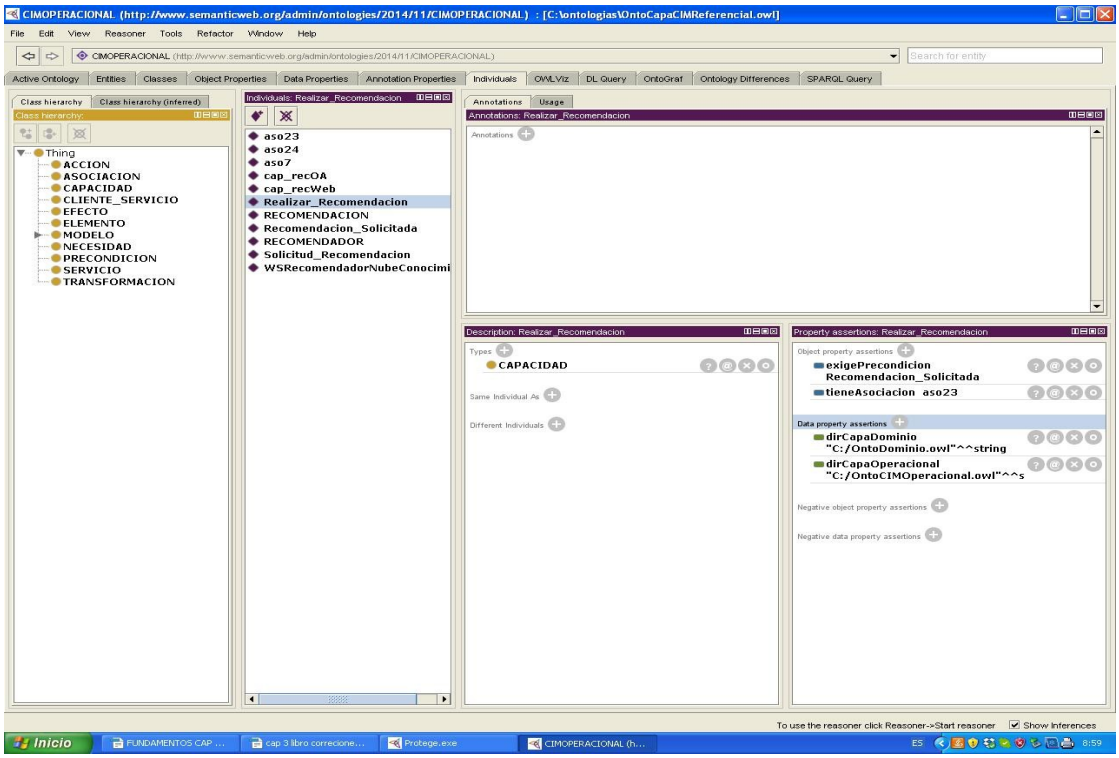

Figura 4.14 Descripción de la capa CIMREFERENCIAL

4.3.1.3 Diseño de la capa CIM OPERACIONAL

En la sección 3.1.2.1.3 se explicó la capa CIMOPERACIONAL de un servicio web semántico, diseñado mediante el Framework FODAS-WS. El aspecto fundamental dentro de la capa CIMOPERACIONAL, es el conocimiento que almacena referente a los objetos que interactúan en la implementación del proceso de recomendación, explicado en la sección 4.2.6.2.

La Figura 4.15 muestra un ejemplo de como la descripción semántica del proceso de certificación (explicado mediante la Figura 4.12) es almacenado como conocimiento al interior de la capa CIMOPERACIONAL. A manera de ejemplo, es posible ver como se describe el individuo *personalizador* instanciado a partir del concepto *OBJETO*. Es importante considerar que el objeto *personalizador* pertenece a la implementación del proceso de recomendación certificada llevado a cabo por el servicio web semántico recomendador de OA (explicado en la sección 4.2.6.2.1), y dicho objeto esta vinculado mediante el objectProperty *esObjetoDe* al TIPOOBJETO *Personalizador.* Además de lo anterior, el objectProperty *invocaOperación* vincula el objeto *personalizador* con la operación *calcularUtilidad (*que implementa el subproceso de cálculo de la función de utilidad explicado en la sección 4.2.6.2.1.1*).* Este conocimiento refleja el hecho de que la operación *calcularUtilidad* es realizada por el objeto *personalizador*. En la Figura 4.15 se muestra también la descripción de la operación *calcularUtilidad*, indicando que es un individuo de tipo OPERACION, y que tiene 12 parámetros asignados mediante el objectProperty *tieneParametro*.

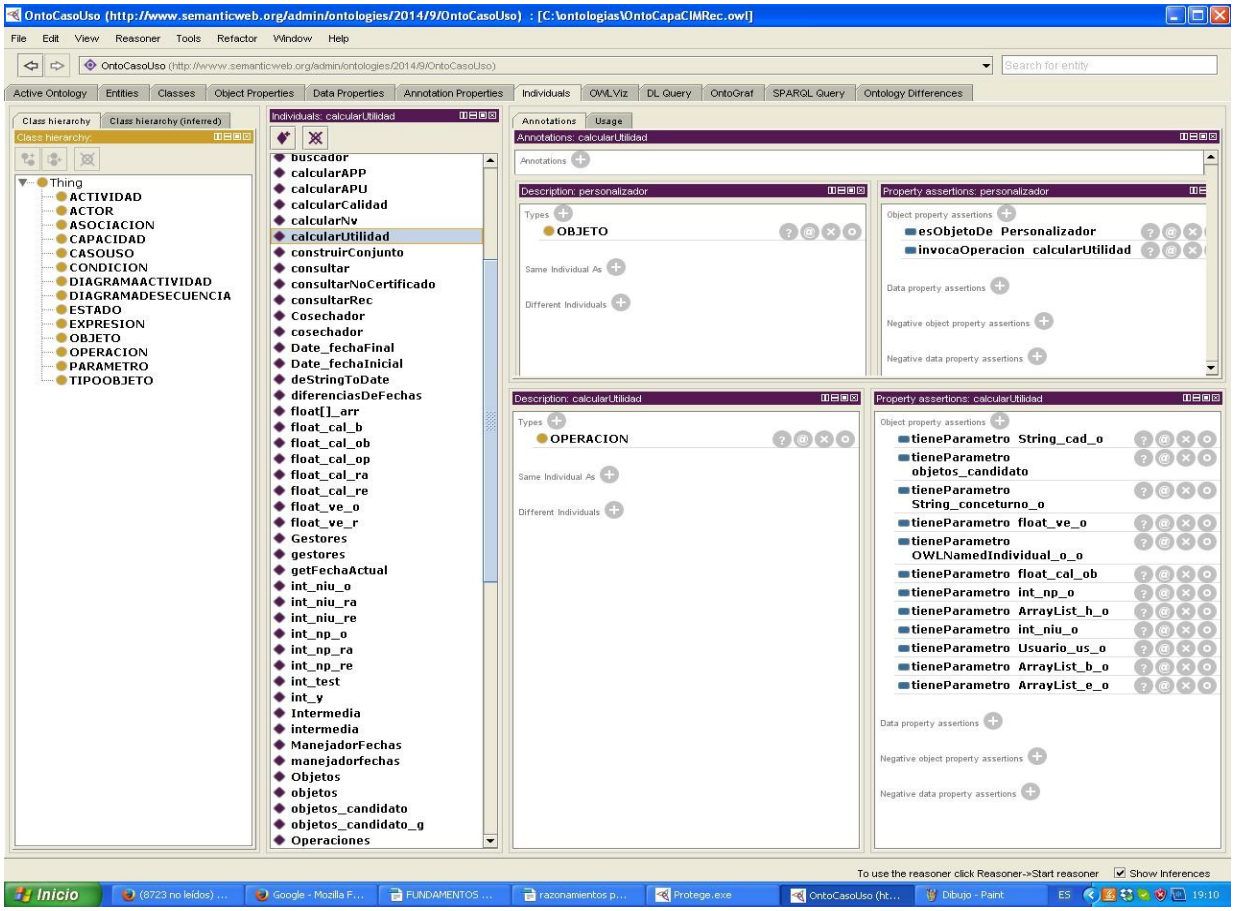

Figura 4.15 Conocimiento almacenado en la capa CIMOPERACIONAL.

4.3.2 Diseño de la capa PIM

Después de usar el Framework FODAS-WS, y ejecutar la transformación CIM-PIM (explicada en la sección 3.1.2.2 y 3.1.2.3), se genera de forma automática la capa PIM con base en el conocimiento disponible en la CAPA CIM. La Figura 4.16 ilustra el conocimiento almacenado en la capa PIM. Como se apreciá en la Figura 4.16, se definen las clases, los métodos y las variables que intervienen en la implementación del servicio web recomendador de OAs. Específicamente, la Figura 4.16 ilustra la definición de las clases *Recomendador (ver ovalo verde)* y *Personalizador (ver ovalo azul),* y a cada una de ellas se le especifica que métodos y que variables tienen, para lo cual se usan los objectProperty *tieneMetodo* y *tieneVariable,* respectivamente. Para la clase Recomendador se almaceno automáticamente que tiene el método *construirConjunto*, y que tiene cinco variables asociadas a la clase (a través del objectProperty *tieneVariable).* De igual forma, para la clase Personalizador se tiene el conocimiento almacenado que tiene el método *calcularUtilidad,* y doce variables asociadas a la clase.

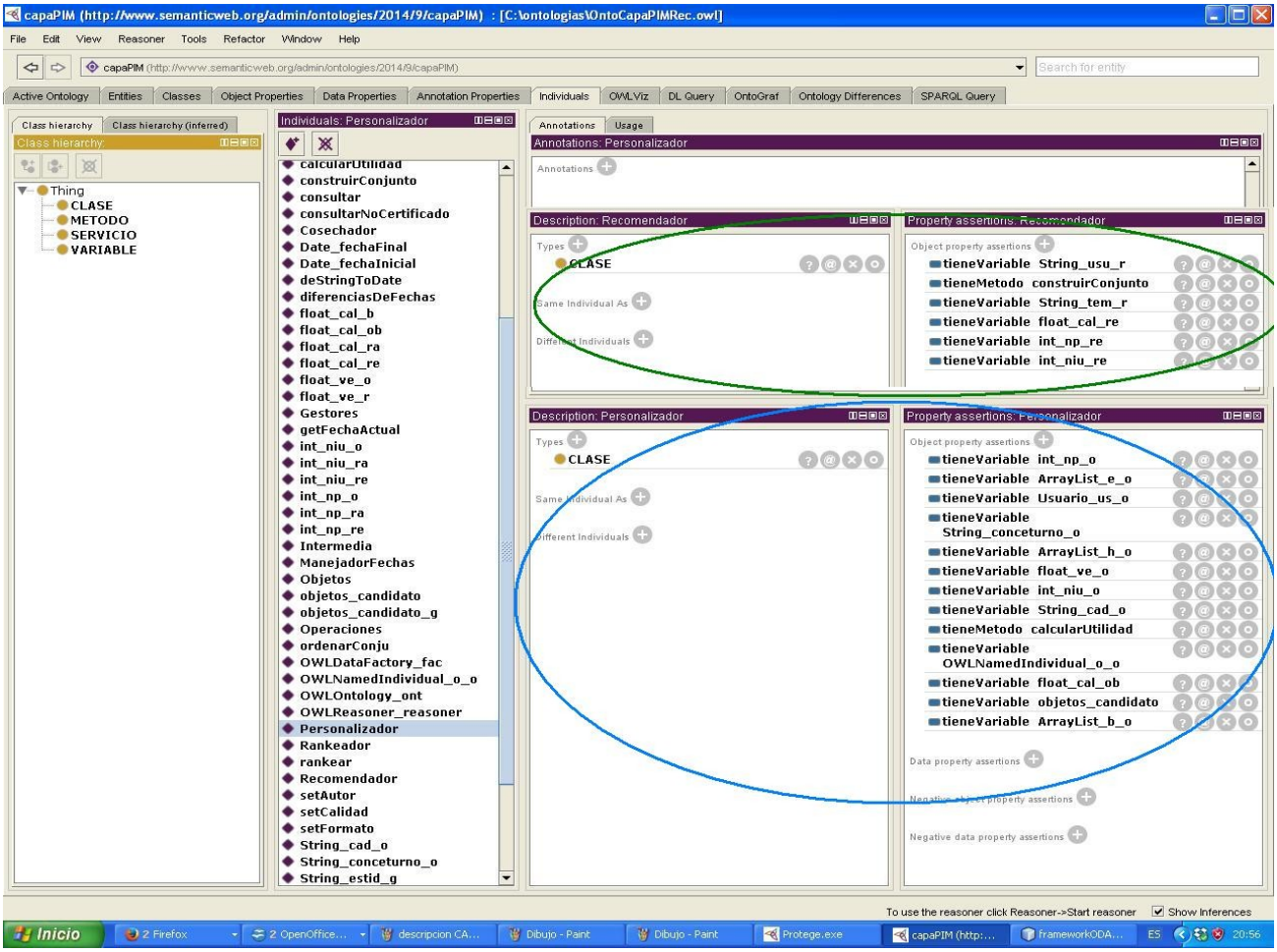

Figura 4.16 Conocimiento almacenado en la CAPAPIM

La Figura 4.17 ilustra de otra forma el conocimiento almacenado en la CAPA PIM en forma gráfica. Por ejemplo, allí se ve como el algoritmo de transformación CIMPIM del Framework FODAS-WS generó automáticamente la clase Personalizador, que contiene variables como us:p (de tipo Usuario) y conceturno_p (de tipo String), y los métodos *emparejaPedagogico*() y *emparejaUsabilidad*(). (ver Figuras 4.16 y 4.17).

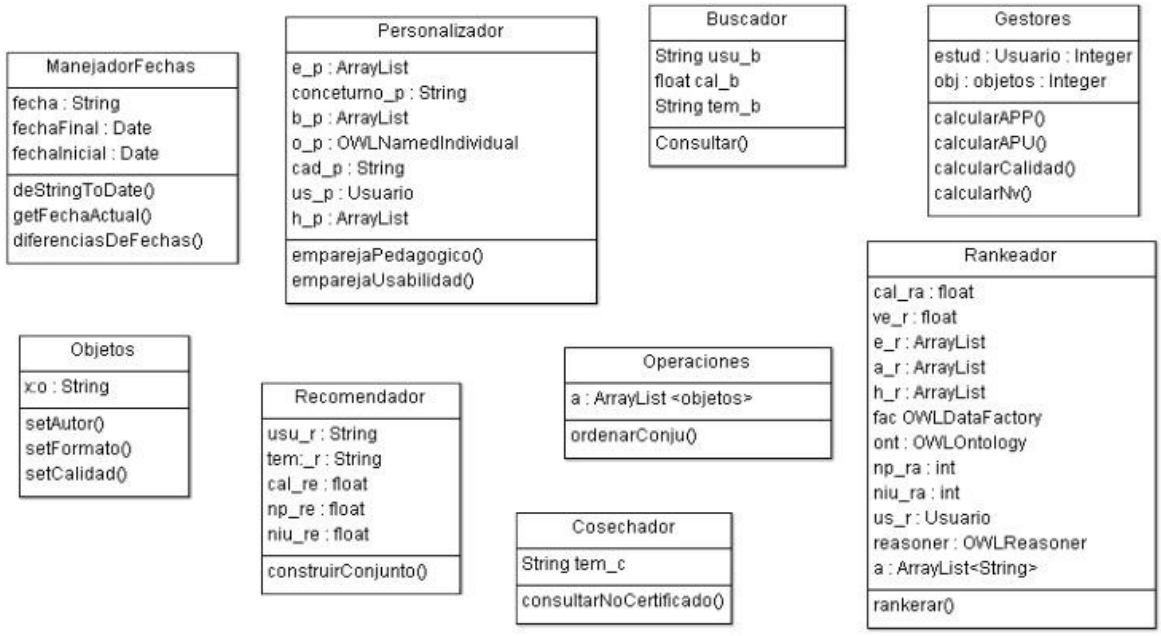

Figura 4.17 Ilustración gráfica del conocimiento almacenado en la capa PIM.

4.3.3 Diseño capa PSM

El conocimiento almacenado en la CAPAPSM es generado automáticamente por la transformación PIM-PSM (explicada en la sección 3.1.3). Como esta capa ofrece detalles de implementación del servicio en una plataforma especifica, esta etapa de diseño se genera una vez el servicio ha sido implementado. Tal proceso de diseño es realizado después de la implementación, buscando que no exista incongruencia alguna entre lo implementado y lo descrito como conocimiento en esta capa. Ese conocimiento es almacenado en una ontología escrita en OWL, denominada *OntoCapaPSM*, y contiene conocimiento extraído principalmente de la información que aporta el archivo WSDL que describe al servicio web recomendador de OA, una vez es implementado y desplegado en el servidor de aplicaciones. La Figura 4.18 muestra el modelo ontológico de la capa PSM presentado en la figura 3.15, en la herramienta Protegé.

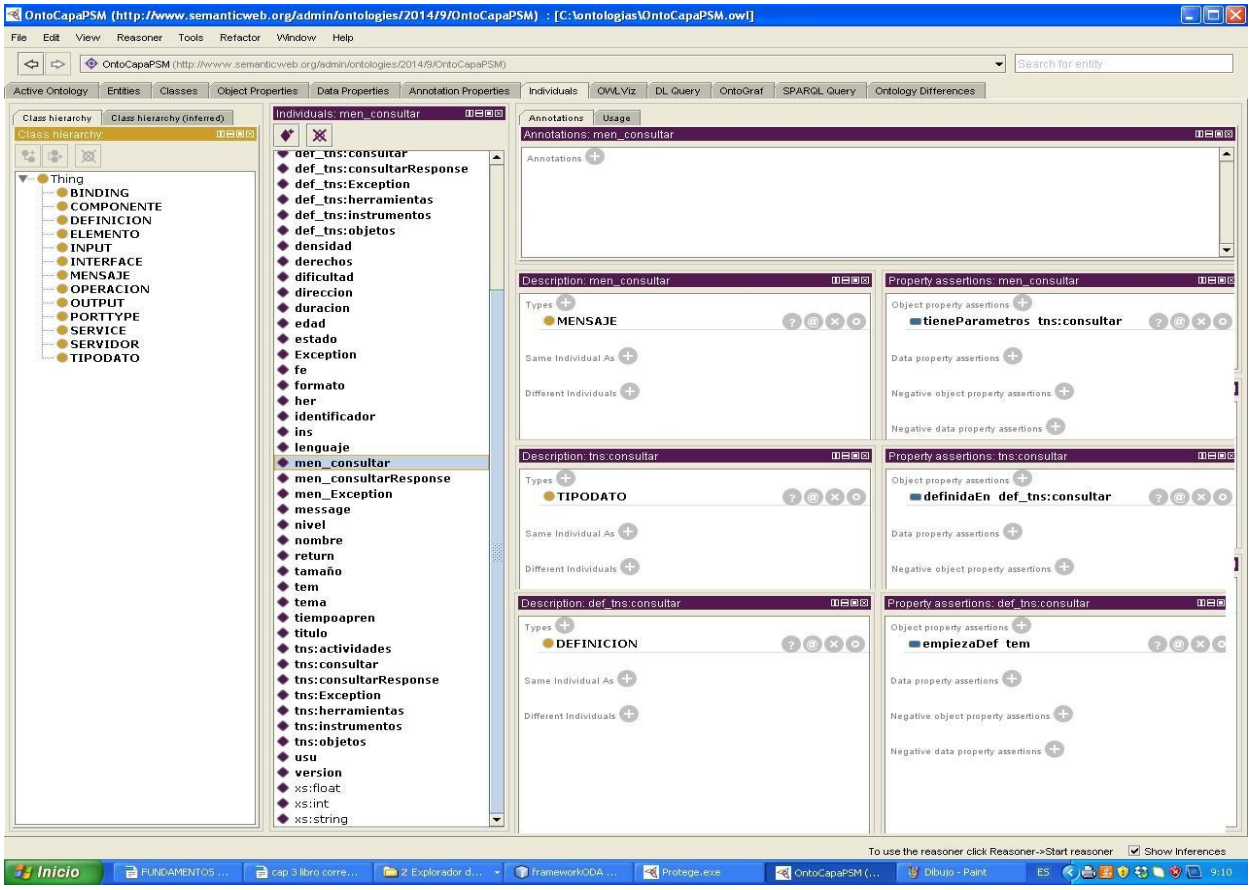

Figura 4.18 Conocimiento almacenado en la CAPAPSM

5 IMPLEMENTACIÓN Y PRUEBAS DE LA PLATAFORMA DE GESTIÓN DE LA NUBE DE FUENTES DE CONOCIMIENTO

El presente capitulo contempla la implementación y pruebas realizadas a los distintos componentes arquitectónicos que conforman la nube de fuentes de conocimiento. La sección uno describe brevemente la forma en que se implementó la plataforma tecnológica que soporta la nube de fuentes de conocimiento. Una segunda sección se dedica a la realización de pruebas a la nube de conocimiento implementada. La tercera y ultima sección del capitulo se dedica a realizar una evaluación del desempeño del sistema de recomendación, en particular, de la calidad de sus recomendaciones.

5.1 Implementación de la plataforma de gestión de OAs de la nube fuentes de conocimiento

Fundamentalmente, la plataforma de gestión consta de un servicio web semántico que contiene como funcionalidad principal un sistema recomendador de OAs, encapsulado e implementado en un componente denominado *RecomendadorNubeConocimientoODA*. Además, la plataforma de gestión de la nube de conocimiento cuenta con la implementación de los cuatro agentes adaptativos, explicados en el capitulo 4, cada uno como un servicio web tradicional. La Figura 5.1 muestra un diagrama de componentes de la plataforma de gestión de la nube de conocimiento. La Nube de Auto-formación actúa como cliente de la Nube de Conocimiento a través de la interfaz *Buscador,* especificada a través del Framework FODAS-WS. Los componentes de color verde oscuro de la Figura 5.1 representan componentes externos a la nube de conocimiento (implementados como servicios web tradicionales) que interactúan directamente con la nube de conocimiento.

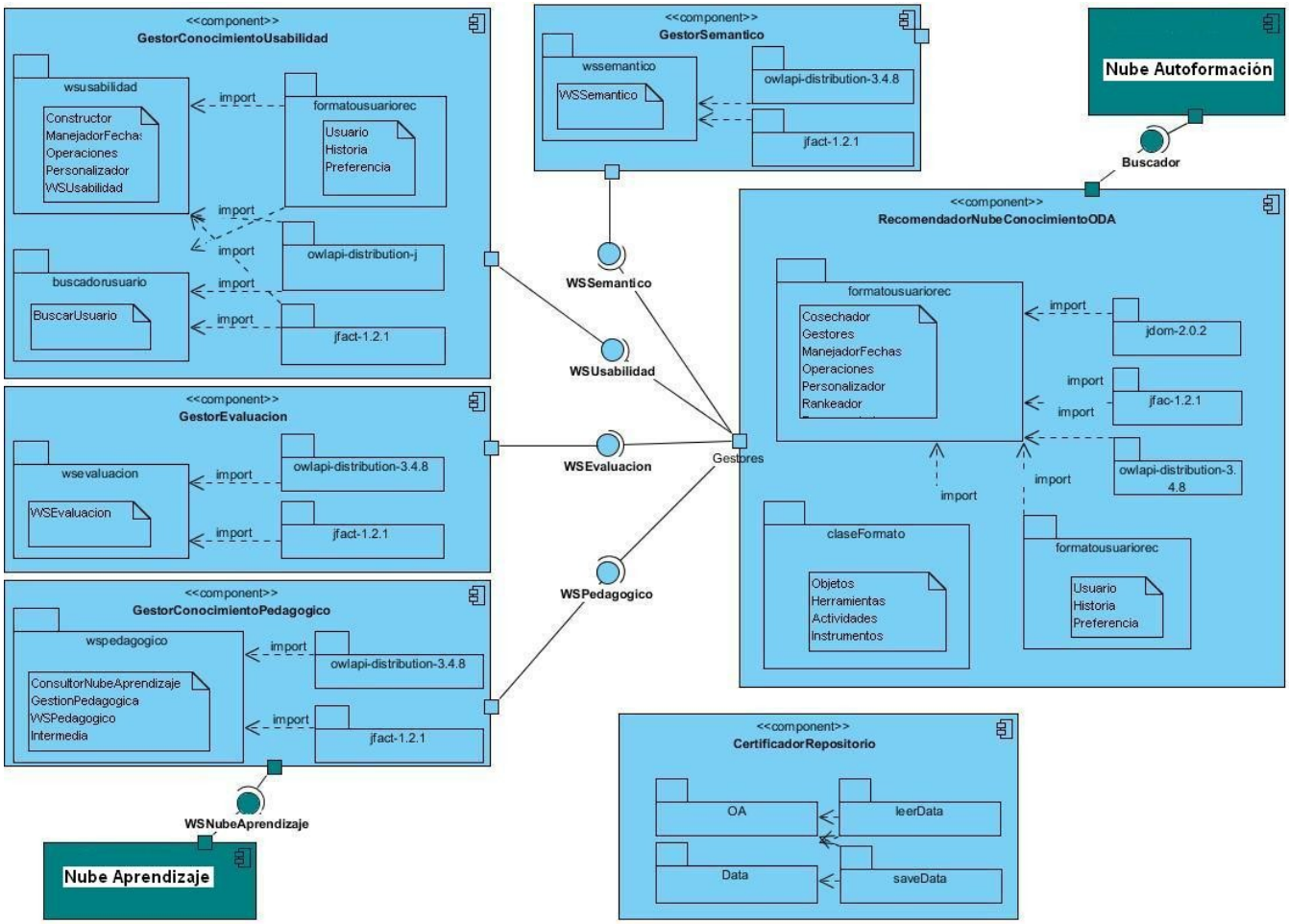

Figura 5.1 Diagrama componentes de la plataforma de gestión de OAs de la nube de conocimiento.

En adelante se describe brevemente cada uno de los componentes que conforman la plataforma de gestión de OAS de la nube fuentes de conocimiento mostrada en la Figura 5.1.

RecomendadorNubeConocimientoODA: Encapsula el servicio web semántico que implementa el sistema recomendador de OA, explicado en la sección 4.2.6.
GestorConocimientoUsabilidad: Encapsula el servicio web que implementa el agente adaptativo de usabilidad, explicado en la sección 4.2.2.
GestorEvaluación: Encapsula el servicio web que implementa el agente adaptativo de la calidad y evaluación de desempeño, explicado en la sección 4.2.5.
GestorConocimientoPedagogico: Encapsula el servicio web que implementa el agente adaptativo de aspectos pedagógicos, explicado en la sección 4.2.4.
GestorSemantico: Encapsula el servicio web que implementa el agente adaptativo de vinculación semántica de temas, explicado en la sección 4.2.3.

La plataforma de gestión de la nube de conocimiento cuenta con una aplicación web certificadora (encapsulada en el componente denominado *CertificadorRepositorio*), que está adjunta al repositorio semántico de OAs, que se ajusta a lo explicado en la sección 4.2.1.2.

Para la implementación de la plataforma de gestión se usaron los siguientes paquetes externos:

- owlapi-distribution-3.4.8: implementa el API OWLAPI encargado de proporcionar la interfaz de comunicación entre las aplicaciones java y el sistema de razonamiento en lógica descriptiva.
- Jfact-1.2.1: implementa los servicios de razonamiento en lógica descriptiva, necesarios para usar las distintas ontologías.
- Jdom-2.0.2: implementa el API que permite leer y escribir documentos XML .

Además de los paquetes externos, dentro de la plataforma de gestión de la nube de conocimiento se implementan otros paquetes adicionales, específicos de la plataforma:

- claseformato: implementa las clases para realizar una representación orientada a objetos de los distintos metadatos que contiene un OA, extraídos del modelo ontológico del repositorio semántico explicado en la sección 4.2.1.1.
- formatousuariorec: implementa las clases para realizar una representación orientada a objetos de la información de cada perfil de usuario, explicado en la sección 4.2.2.2.
- leerdata: implementa las clases para leer el conocimiento en una ontología basada en lógica descriptiva, usando el API OWLAPI.
- savedata: implementa las clases para escribir el conocimiento en una ontología basada en lógica descriptiva, usando el API OWLAPI.

La Figura 5.2 muestra el diagrama de despliegue de los distintos nodos que conforman la plataforma de gestión de la nube fuentes de conocimiento. Como se aprecia en la Figura 5.2, los distintos componentes que encapsulan servicios web (nodos coloreados con color azul claro) consisten en servidores Glassfish 4, y su

acceso se realiza mediante el protocolo HTTP. Dichos nodos representan la plataforma de despliegue de cada uno de los componentes explicados a partir de la Figura 5.1, cuyos nombres corresponden exactamente con el nombre del componente asociado a cada nodo.

Como la arquitectura es completamente distribuida, cada componente puede estar desplegado en un servidor de aplicaciones distinto. Los componentes gestionados por servidores de archivos (nodos coloreados con color gris) representan los tres repositorios de conocimiento de la nube: repositorio semántico de OAs, repositorio de perfiles de usuario, y repositorio de descripciones semánticas de servicios web. Además, existen dos nodos (coloreados con coloreados con color verde oscuro) que se incluyen en el diagrama de despliegue, pero que no forman parte directa de la implementación de la plataforma de gestión de la nube. Dichos nodos representan la nube de aprendizaje y la nube de auto-formación.

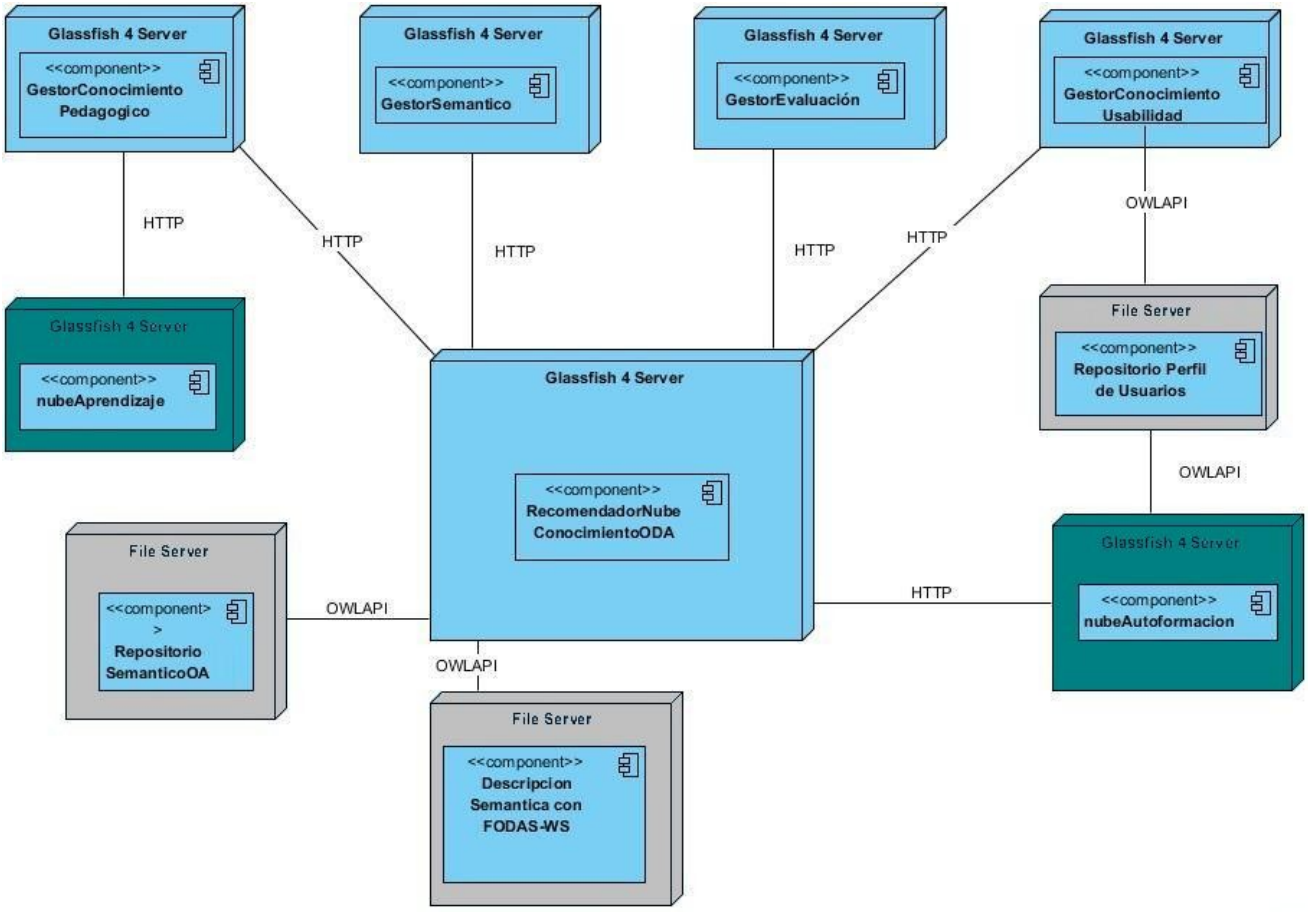

Figura 5.2 Diagrama de despliegue de la plataforma de gestión de OAs de la nube de conocimiento.

5.2 Pruebas a la Nube de Conocimiento

Esta sección se dedica a describir el protocolo experimental aplicado a la nube de fuentes de conocimiento, buscando con ello garantizar el correcto funcionamiento de la arquitectura presentada en esta tesis.

5.2.1 Protocolo experimental aplicado a la nube fuentes de conocimiento

5.2.1.1 Objetivo general del protocolo experimental

Comprobar el correcto funcionamiento de la nube de fuentes de conocimiento.

5.2.1.2 Pruebas que conforman el protocolo experimental

El protocolo experimental de la nube de fuentes de conocimiento se compone de distintas pruebas, que buscan comprobar el correcto funcionamiento de la nube. El protocolo experimental considera dos tipos distintos de pruebas: Pruebas de funcionamiento interno de la nube, y pruebas de integración de la nube dentro del Proyecto Madre. La Tabla 5.1 ilustra las distintas pruebas consideradas dentro de cada tipo.

5.2.1.2.1 Pruebas de funcionamiento interno

Componente de la Nube	Objetivo de la Prueba
WSPedagógico	Funcionamiento del Agente Adaptativo de aspectos Pedagógicos
WSSemántico	Funcionamiento del Agente Adaptativo de la Vinculación Semántica de Temas
WSUsabilidad	Funcionamiento del Agente Adaptativo de la Usabilidad respecto de *aporte (APPP)*
WSUsabilidad	Funcionamiento del Agente Adaptativo de la Usabilidad en lo referente al *aporte (APPH)*
WSEvaluación	Funcionamiento del Agente Adaptativo de la Calidad y Evaluación de Desempeño
WSRecomendador	Funcionamiento general del Sistema Recomendador de OAs
ROA Certificados	Funcionamiento de las herramientas de gestión del repositorio
Federacion de ROAs no certificados	Prueba de la búsqueda (cosechado) de OAs no certificados

5.2.1.2.2 Pruebas de integración con las otras nubes

Nube	Objetivo de la Prueba
Nube de aprendizaje	integración con la nube de aprendizaje
Nube de Auto-formación	integración con la nube de auto-formación (desarrollo de cliente)

5.2.2 Especificación de las pruebas de funcionamiento interno

Las pruebas de funcionamiento interno se centran en garantizar que los resultados generados al interior de la nube sean coherentes y correctos.

5.2.2.1 Prueba del funcionamiento del Agente Adaptativo de Aspectos Pedagógicos.

5.2.2.1.1 Objetivo

Evaluar el funcionamiento y precisión del Agente Adaptativo de aspectos Pedagógico, en lo referente al cálculo de $APP(O_i, U_k)$ usando la ecuación (4.5)

5.2.2.1.2 Componentes

- Servicio web que implementa el Agente
- Nube de Aprendizaje
- Sistema recomendador de AOs
- Servidor de aplicaciones Glassfish

5.2.2.1.3 Procedimiento

1. Desplegar el servicio web WSPedagogico que implementa el Agente en el servidor Glassfish
2. Desplegar y ejecutar el sistema recomendador semántico que actuá como cliente.

3. Consultar el sistema recomendador semántico para un usuario (estudiante) dado, llamado cli1, con el tema AutomatasPila, un factor de calibración de 0.0; y calcular $APP(O_i, U_k)$.
4. Consultar el sistema recomendador semántico para un usuario (estudiante) dado, llamado cli2, con el tema AutomatasPila, un factor de calibración de 0.0; y calcular $APP(O_i, U_k)$.

5.2.2.1.4 Resultados esperados

Se esperan los siguientes resultados:

1. La consola de administración del servidor Glassfish 4.0 deberá mostrar que el servicio web WSPedagogico está desplegado correctamente.
2. El navegador debe visualizar la pagina principal del cliente que emula la nube de auto-formación, quien invoca el servicio del sistema recomendador.
3. En el navegador se solicita la recomendación para el estudiante cli1 con el tema AutomatasPila y el factor de calibración 0.0, y se invoca el cálculo de $APP(O_i, U_k)$ para cada uno de los objetos recomendados.
4. En el navegador se solicita la recomendación para el estudiante cli2 con el tema AutomatasPila y el factor de calibración 0.0, y se invoca el cálculo de $APP(O_i, U_k)$ para cada uno de los objetos recomendados .

5.2.2.1.5 Resultados obtenidos

1. Se desplegó correctamente el servicio web WSPedadogico.
2. Se visualizó correctamente la pagina principal del cliente que emula la nube de auto-formación.
3. En la Figura 5.3 se aprecia la recomendación cargada en la pagina principal del cliente que emula la nube de auto-formación, para el estudiante cli1 con el tema AutomatasPila y un factor de calibración de 0.0. El valor $APP(O_i, U_k)$ para cada uno de los objetos recomendados se muestra en la Tabla 5.2.
4. En la Figura 5.4 se aprecia la recomendación cargada en la pagina principal del cliente que emula la nube de auto-formación para el estudiante cli2 con el tema AutomatasPila y un factor de calibración de 0.0. El valor $APP(O_i, U_k)$ para cada uno de los objetos recomendados se muestra en la Tabla 5.2.

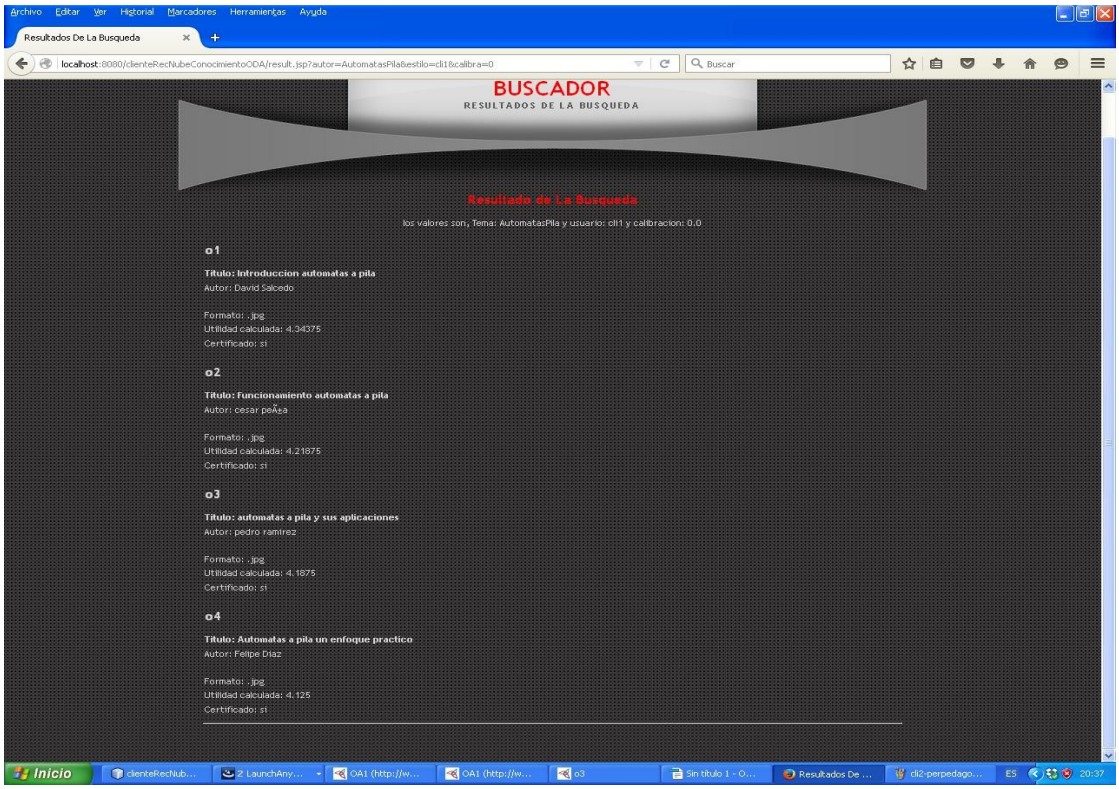

Figura 5.3 Recomendación cargada en el cliente para el cli1 con el tema AutomatasPila.

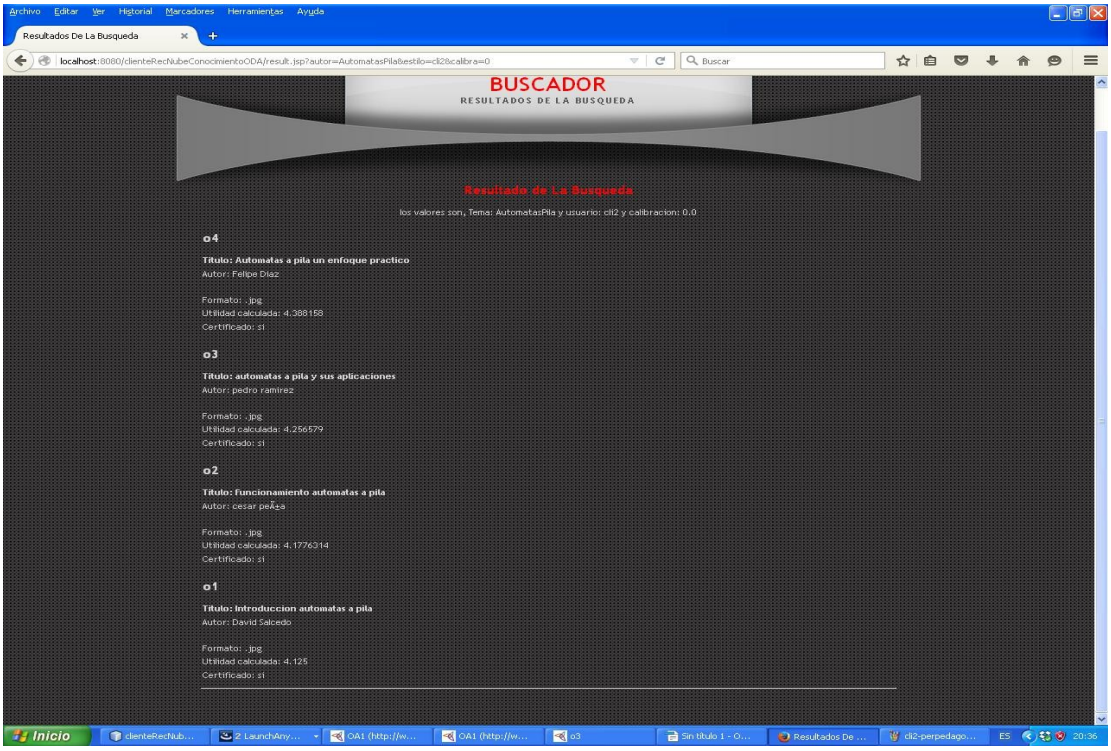

Figura 5.4 Recomendación cargada en el cliente para el cli2 con el tema AutomatasPila.

PRUEBA DEL FUNCIONAMIENTO DEL GESTOR PEDAGOGICO				
HIBRIDACION	*0*			
cli1	**o1**	**o2**	**o3**	**o4**
utilidad	4.34375	4.21875	4,1875	4,125
APP	0.4375	0.1875	0,125	0
cli2	**o4**	**o3**	**o2**	**o1**
utilidad	4.388158	4.256579	4.1776314	4,125
APP	0.5263158	0.2631579	0.10526316	0

Tabla 5.1 Resultados obtenidos para el aporte de personalización pedagógica

5.2.2.1.6 Conclusiones

La prueba evidencia la influencia del aporte de $APP(O_i, U_k)$. En particular, el estudiante cli1 tiene un estilo de aprendizaje (5555), bastante distinto al estilo de aprendizaje del estudiante cli2 (2121). Este agente ordena los OAs a recomendar según el emparejamiento entre las herramientas de aprendizaje, actividades de aprendizaje y métodos de evaluación que tenga el OA, respecto de las herramientas de aprendizaje, actividades de aprendizaje y métodos de evaluación que la nube de aprendizaje considere como ideales para enseñar a un estudiante con dicho estilo de aprendizaje. De tal forma que el 0,4375 calculado como APP para el objeto o1 del cliente cli1 indica que ese objeto de aprendizaje posee el 43.75% de las herramientas de aprendizaje, actividades de aprendizaje y métodos de evaluación que se consideran ideales para enseñar al estudiante cli1, que es para quien se solicita la recomendación. Vemos así que el orden en la prioridad de la recomendación cambia dramáticamente dependiendo del estilo de aprendizaje del estudiante a quien se le solicita la recomendación, para un mismo tema, factor de calibración y grupos de objetos a recomendar.

5.2.2.2 Prueba del funcionamiento del Agente Adaptativo de la Vinculación Semántica de temas.

5.2.2.2.1 Objetivo

Evaluar el funcionamiento y precisión del Agente, en lo referente al cálculo del aporte semántico cognitivo, calculado según la ecuación (4.4)

5.2.2.2.2 Componentes

- Servicio web WSSemantico que implementa el Agente
- Modelo ontológico de vinculación semántica de temas
- Sistema recomendador de OAs
- Servidor de aplicaciones Glassfish

5.2.2.2.3 Procedimiento

1. Desplegar el servicio web que implementa el Agente en el servidor Glassfish
2. Desplegar y ejecutar el sistema recomendador semántico que actuá como cliente.
3. Consultar el Agente Adaptativo de la vinculación semántica para el tema avl, para que genere un listado de los temas vinculados semánticamente a dicho tema.
4. Consultar el Agente Adaptativo de la vinculación semántica para calcular el aporte semántico para cada uno de los OAs recomendados para el tema avl, respecto al estudiante cli1, con algún nivel de vinculación semántica con el tema solicitado por el estudiante.

5.2.2.2.4 Resultados esperados

Se esperan los siguientes resultados:

1. La consola de administración del servidor Glassfish 4.0 debe mostrar que el servicio web WSSemantico está desplegado correctamente.
2. El navegador debe visualizar la pagina principal del cliente que emula la nube de auto-formación, quien invoca el servicio del sistema recomendador.
3. El navegador debe visualizar el listado de temas vinculados semánticamente con el tema consultado.
4. El navegador debe mostrar los resultados de la recomendación realizada, donde se evidencie el aporte semántico cognitivo de cada OA.

5.2.2.2.5 Resultados obtenidos

1. Se desplegó correctamente el servicio web WSSemantico que implementa el Agente.
2. Se ejecuto correctamente la aplicación web cliente que emula la nube de auto-formación, a través de la cual se invoca el sistema recomendador de OAs.
3. Se visualizó la lista de temas vinculados semánticamente con el tema avl (ver Figura 5.5).
4. Se establece el aporte semántico en la recomendación (ver Figura 5.6).

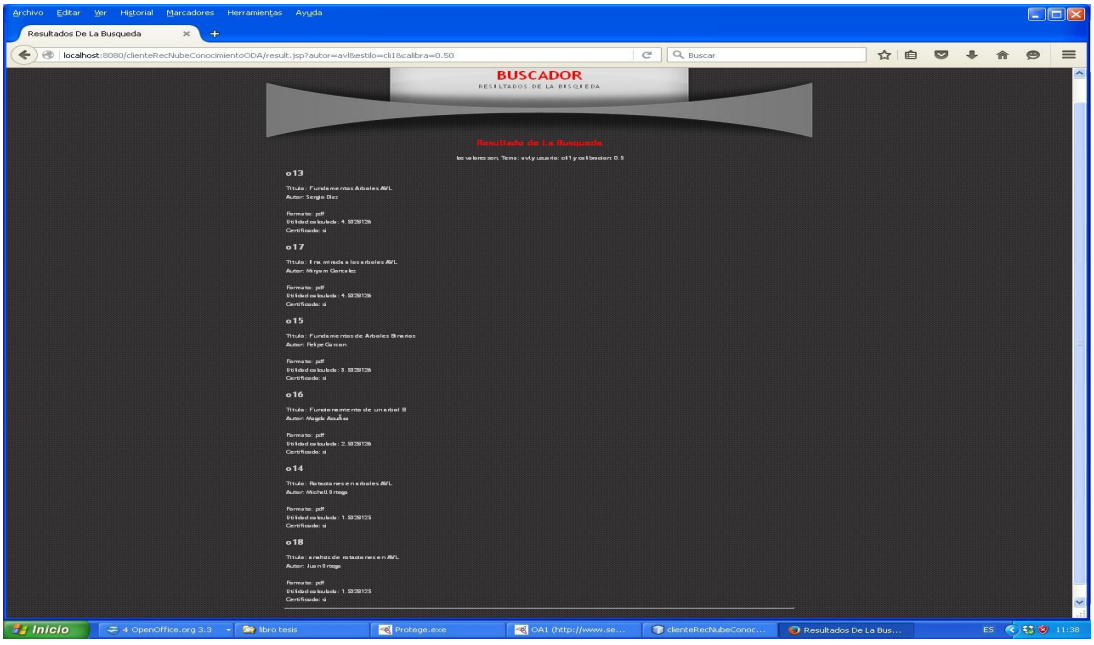

Figura 5.5 Recomendación sugerida por el sistema recomendador para el cliente cli1 con el tema avl

OA	utilidad	ASC	Tema	Nivel vinculación
o13	4,53	4	avl	TEMA BASE
o17	4,47	4	avl	TEMA BASE
o15	3,53	3	Arbol Binario	JERARQUIZACION
o16	2,53	2	Arbol B	COMPARACION
o14	1,53	1	RotacionAVL	EXPLICACION
o18	1,47	1	RotacionAVL	EXPLICACION

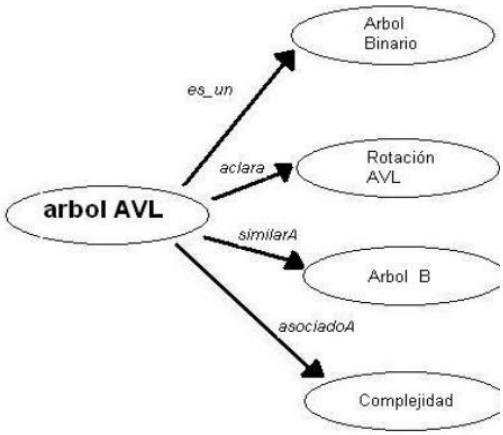

Figura 5.6 Síntesis de los resultados obtenidos en la prueba de funcionamiento del Agente Adaptivo de la Vinculación Semántica de Temas.

5.2.2.2.6 Conclusiones

El aporte semántico cognitivo que calcula el Agente Adaptativo de la vinculación semántica de temas, permite priorizar los temas de acuerdo a los niveles de vinculación semántica planteados en el modelo de conocimiento del sistema recomendador. La Figura 5.6 evidencia como el aporte semántico cognitivo $ASC(O_i)$, calculado con la ecuación (4.4), cambia de acuerdo con el nivel de vinculación semántica que tenga el tema asociado con el OA con respecto del tema de consulta (avl). El valor 4 de ASC para los objetos o13 y o17 indican la vinculación directa entre los temas asociados a esos OAs con el tema de la consulta. Ese valor se decrementa en el resto de los OAs, en función de la relación semántica con el tema buscado, teniendo los menores valores la vinculación semántica por explicación (casos o14 y o18). Eso, por supuesto incide en el valor final de utilidad de cada OA.

5.2.2.3 Prueba del funcionamiento del Agente Adaptativo de la Usabilidad en lo referente al aporte parcial de personalización por preferencias de uso (APPP).

5.2.2.3.1 Objetivo

Evaluar el funcionamiento y precisión del Agente Adaptativo de la usabilidad en lo referente al cálculo de APPP, usando la ecuación (4.1).

5.2.2.3.2 Componentes

- Servicio web WSUsabilidad que implementa el Agente
- repositorio de perfiles de usuario
- Sistema recomendador de OAs
- Servidor de aplicaciones Glassfish

5.2.2.3.3 Procedimiento

1. Desplegar el servicio web WSUsabilidad que implementa el Agente en el servidor Glassfish
2. Desplegar y ejecutar el sistema recomendador semántico que actuá como cliente.
3. Consultar el sistema recomendador semántico para un usuario (estudiante) dado, llamado cli1, con el tema MaquinasTuring, y un factor de calibración de 0.0 usuario. Calcular únicamente APPP) usando la ecuación (4.1).
4. Consultar el sistema recomendador semántico, para un usuario (estudiante) dado, llamado cli2, con el tema MaquinasTuring y un factor de calibración de 0.0. Calcular únicamente APPP) usando la ecuación (4.1).

5.2.2.3.4 Resultados esperados

1. La consola de administración del servidor Glassfish 4.0 deberá mostrar que el servicio web WSUsabilidad está desplegado correctamente.
2. El navegador debe visualizar la pagina principal del cliente que emula la nube de auto-formación, quien invoca el servicio del sistema recomendador.
3. En el navegador se cargará la recomendación realizada para el estudiante cli1, con el tema MaquinaTuring y factor de calibración 0.0, y se podrá extraer el valor de APPP para cada uno de los objetos recomendados.
4. En el navegador se cargará la recomendación realizada para el estudiante cli2, con el tema MaquinaTuring y factor de calibración 0.0 y se podrá extraer el valor de APPP para cada uno de los objetos recomendados.

5.2.2.3.5 Resultados obtenidos

1. Se desplegó correctamente el servicio web WSUsabilidad.
2. Se visualizó correctamente la pagina principal del cliente que emula la nube de auto-formación.
3. En la Figura 5.7 se aprecia la recomendación para el estudiante cli1, con el tema MaquinaTuring, y un factor de calibración de 0.0. Los valores

calculados de APPP de cada uno de los objetos recomendados, se muestran en la Tabla 5.4.

4. En la Figura 5.8 se aprecia la recomendación para el estudiante cli2, con el tema MaquinaTuring, y un factor de calibración de 0.0. Los valores calculados de APPP de cada uno de los objetos recomendados, se muestran en la Tabla 5.4.

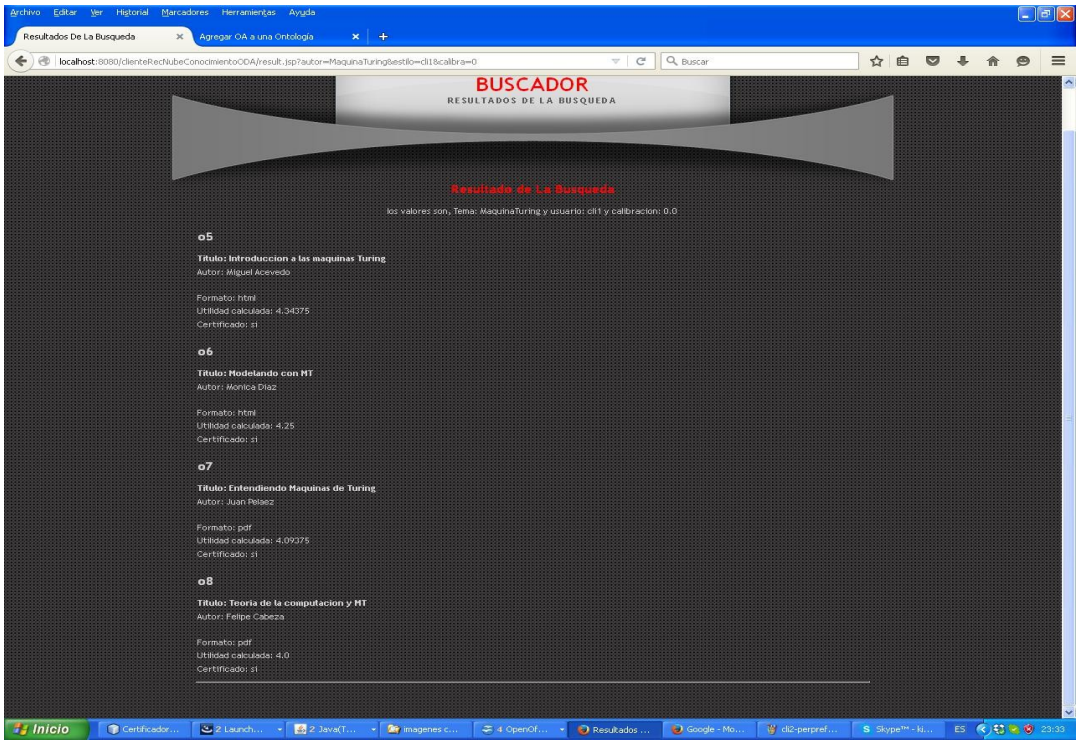

Figura 5.7 Recomendación para el cliente cli1 con el tema MaquinaTuring.

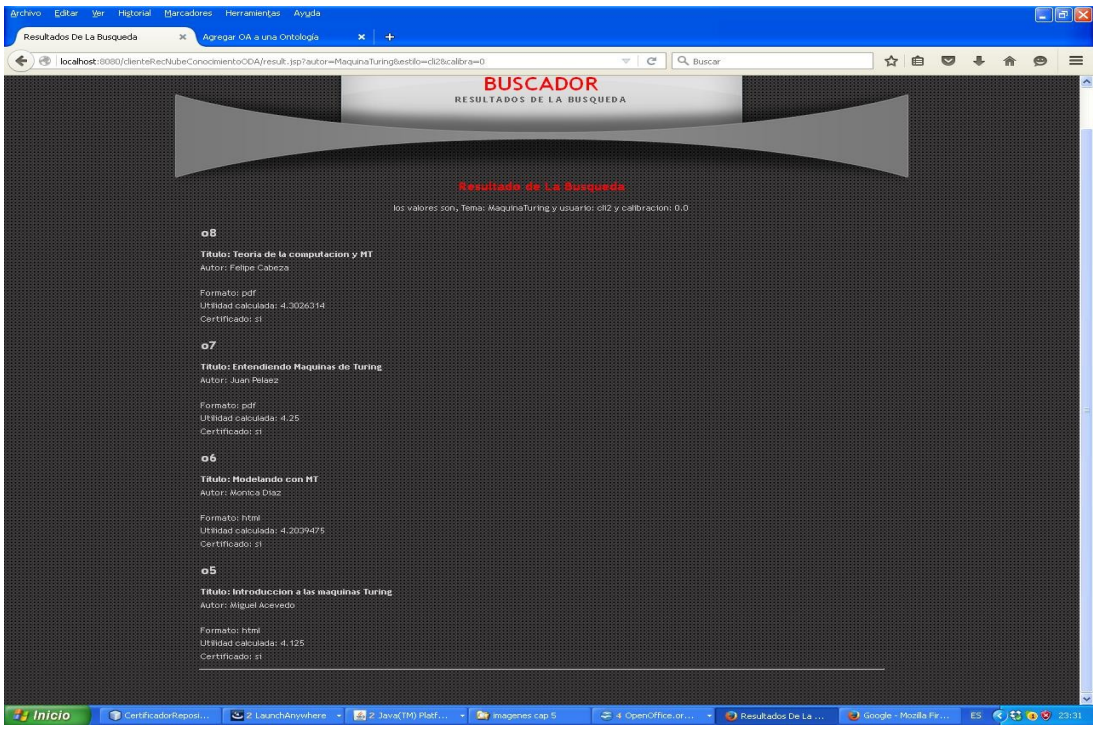

Figura 5.8 Recomendación para el cliente cli2 con el tema MaquinaTuring.

PRUEBA DE PERSONALIZACION DE PREFERENCIAS DE USO				
HIBRIDACION	*0*			
cli1	**o5**	**o6**	**o7**	**o8**
utilidad	4.34375	4,25	4.09375	4
APPP	1	1	0	0
cli2	**o8**	**o7**	**o6**	**o5**
utilidad	4.3026314	4,25	4.2039475	4,125
APPP	1	1	0,5	0,5

Tabla 5.4 Resultados obtenidos para el aporte parcial por personalizacion por preferencias de uso

5.2.2.3.6 Conclusiones

La prueba permite comprobar el correcto funcionamiento del Agente Adaptativo de la Usabilidad en lo referente a APPP. La Tabla 5.4 muestra las recomendaciones entregadas para los estudiantes cli1 y cli2, los cuales cuentan con preferencias de uso distintas. El valor de APPP influye en el orden de prioridad de los OAs para cada estudiante. Por ejemplo, para el caso del OA o8, se encuentra que para el estudiante cli1 su APPP es 0 (lo cual indica que cumple con el 0% de las preferencias de ese usuario), y para el estudiante cli2 su APPP es 1 (lo cual indica que el objeto o8 cumple con el 100% de las preferencias de uso del estudiante cli2). Es bueno resaltar que el hecho de que un OA cuente con un APPP de 0 en cierta recomendación, no implica que el objeto no forme parte del CONJUNTO DE RECOMENDACIÓN, solo que la prioridad del objeto dentro de la recomendación es más bajo.

5.2.2.4 Prueba del funcionamiento del Agente Adaptativo de la Usabilidad en lo referente al aporte parcial de personalización por historias de uso (APPH).

5.2.2.4.1 Objetivo

Evaluar el funcionamiento y precisión del Agente Adaptativo de la usabilidad en lo referente al cálculo de APPH, usando la ecuación (4.2).

5.2.2.4.2 Componentes

- Servicio web WSUsabilidad que implementa el Agente
- repositorio de perfiles de usuario
- Sistema recomendador de OAs
- Servidor de aplicaciones Glassfish

5.2.2.4.3 Procedimiento

1. Desplegar el servicio web WSUsabilidad que implementa el Agente en el servidor Glassfish
2. Desplegar y ejecutar el sistema recomendador semántico que actuá como cliente.
3. Consultar el sistema recomendador semántico para un usuario (estudiante) dado, llamado cli1, con el tema AutomatasFinitos ,y un factor de calibración de 0.0. Calcular únicamente APPH .
4. Consultar el sistema recomendador semántico para un usuario (estudiante) dado, llamado cli2, con el tema AutomatasFinitos, y un factor de calibración de 0.0. Calcular únicamente APPH.

5.2.2.4.4 Resultados esperados

1. La consola de administración del servidor Glassfish 4.0 deberá mostrar que el servicio web WSUsabilidad está desplegado correctamente.
2. El navegador debe visualizar la pagina principal del cliente que emula la nube de auto-formación, quien invoca el servicio del sistema recomendador.
3. En el navegador se cargará la recomendación realizada para el estudiante cli1, con el tema AutomatasFinitos y el factor de calibración 0.0. Además, se podrá extraer el valor de APPH para cada uno de los objetos recomendados.
4. En el navegador se cargará la recomendación realizada para el estudiante cli2, con el tema AutomatasFinitos y el factor de calibración 0.0. Además, se podrá extraer el valor de APPP para cada uno de los objetos recomendados.

5.2.2.4.5 Resultados obtenidos

1. Se desplegó correctamente el servicio web WSUsabilidad.
2. Se visualizó correctamente la pagina principal del cliente que emula la nube de auto-formación.
3. En la Figura 5.9 se aprecia la recomendación cargada en la pagina principal del cliente que emula la nube de auto-aprendizaje, para el estudiante cli1, con el tema AutomatasFinitos, y un factor de calibración de 0.0. Los valores calculados de APPH de cada uno de los objetos recomendados, se muestran en la Tabla 5.5.
4. En la Figura 5.10 se aprecia la recomendación cargada en la pagina principal del cliente que emula la nube de auto-formación, para el estudiante cli2, con el tema AutomatasFinitos, y un factor de calibración de 0.0. Los valores calculados de APPH de cada uno de los objetos recomendados, se muestran en la Tabla 5.5.

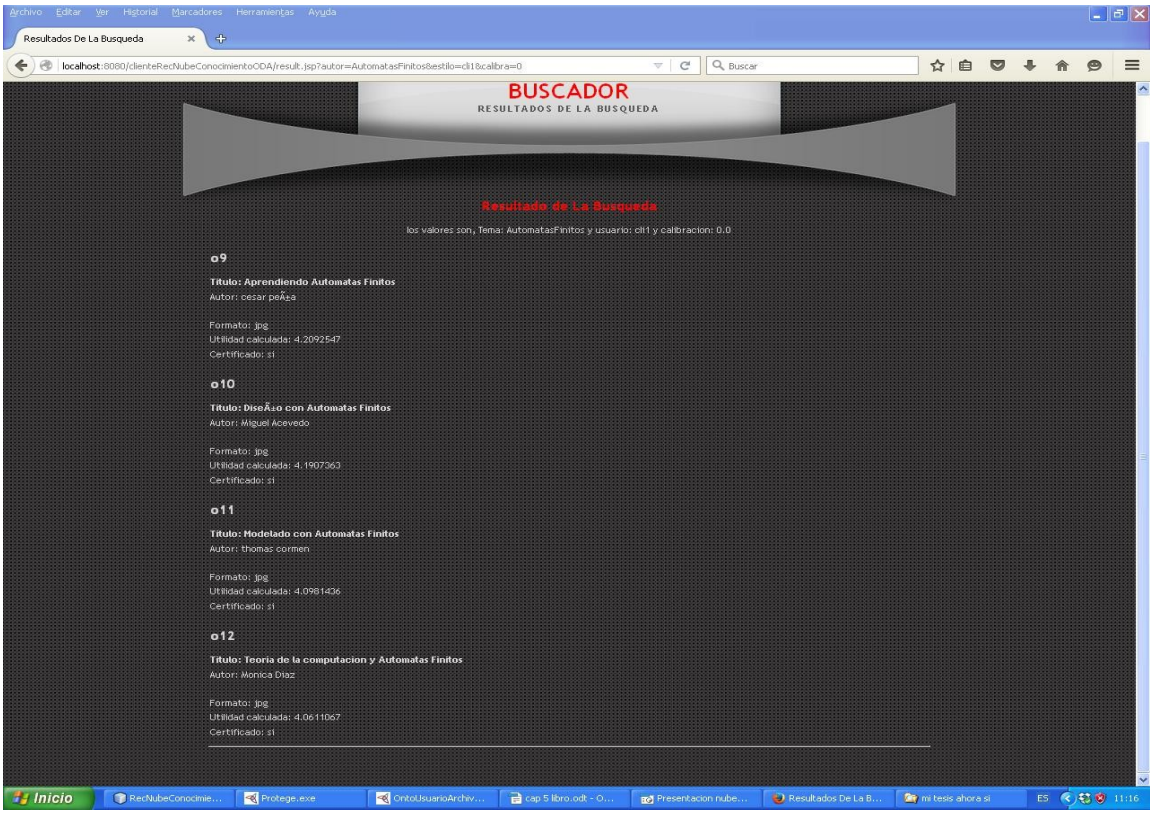

Figura 5.9 Recomendación para cli1, con el tema AutomatasFinitos.

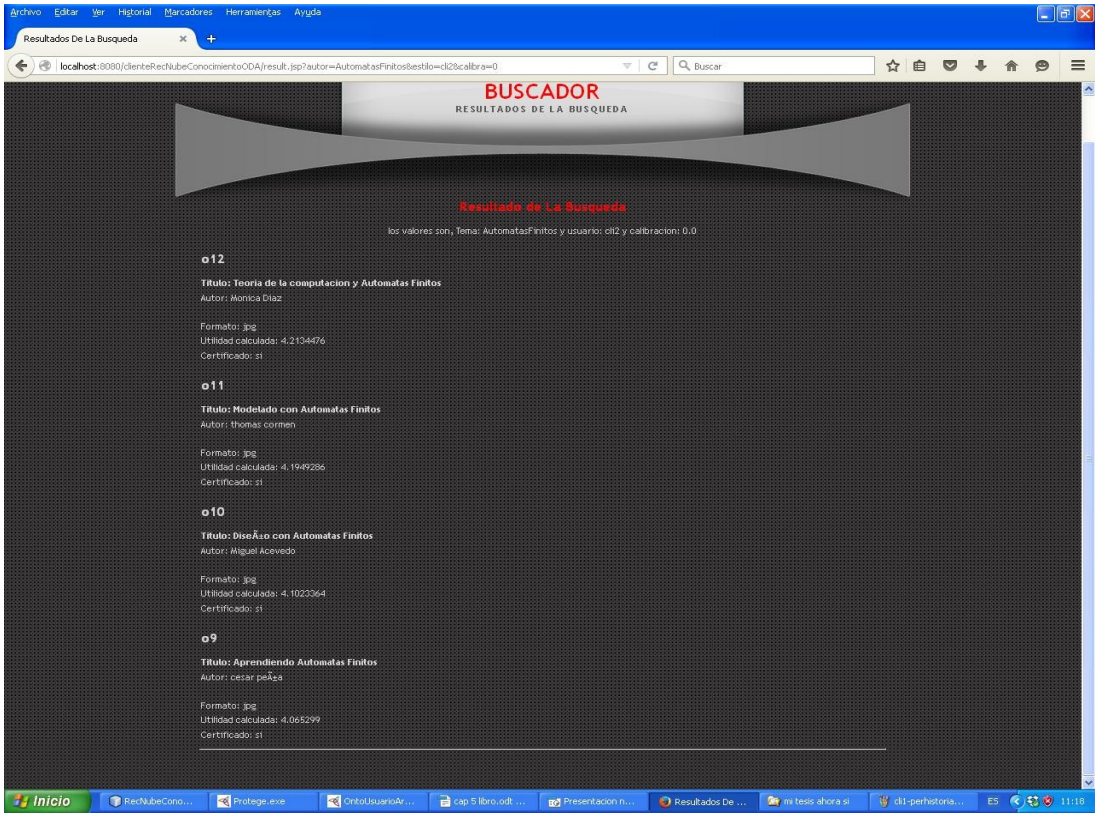

Figura 5.10 Recomendación para cli2, con el tema AutomatasFinitos.

PRUEBA DE PERSONALIZACION DE HISTORIAL DE USO				
HIBRIDACION	*0*			
cli1	**o9**	**o10**	**o11**	**o12**
utilidad	4.2092547	4.1907363	4.0981436	4.0611067
APPH	0.2092547	0.19073619	0.0981436	0.06110656
cli2	**o12**	**o11**	**o10**	**o9**
utilidad	4.2134476	4.1949286	4.1023364	4.065299
APPH	0.21344735	0.19492882	0.10233625	0.065299205

Tabla 5.5 Resultados obtenidos para el aporte parcial de personalización por historias de Uso

5.2.2.4.6 Conclusiones

La prueba permite comprobar el correcto funcionamiento del Agente Adaptativo de la Usabilidad, en lo referente a APPH. La Tabla 5.5 muestra las recomendaciones entregadas para los estudiantes cli1 y cli2, los cuales cuentan con historias de uso distintas. APPH influye en el orden de prioridad de los OAs para cada estudiante. Un mayor valor de APPH para un objeto dado implica que ha sido usado más veces respeto al total de usos de OAs que tiene ese estudiante, y/o que lo ha usado recientemente dicho estudiante, y/o que ese OA ha sido usado para estudiar ese tema con mayor frecuencia por el estudiante. Por ejemplo, en la Tabla 5.5 el objeto o9 tiene un valor de APPH de 0,2092 para el estudiante cli1, y el objeto o11 un valor de APPH de 0,0611. Eso indica que el estudiante cli1 tiene mayor afinidad con el objeto o9, ya sea porque lo ha usado

más recientemente o más frecuentemente. Con esta prueba se corrobora como el agente adaptativo de usabilidad influye directamente en la adaptatividad de las recomendaciones entregadas.

5.2.2.5 Prueba del funcionamiento del Agente Adaptativo de la Calidad y Evaluación de Desempeño.

5.2.2.5.1 Objetivo

Evaluar el funcionamiento y precisión del Agente de Adaptabilidad de Calidad y Evaluación de desempeño en lo referente al cálculo de $AC(O_i, U_k)$, usando la ecuación (4.6)

5.2.2.5.2 Componentes

- Servicio web WSEvaluacion que implementa el Agente.
- Sistema recomendador de OA
- Servidor de aplicaciones Glassfish

5.2.2.5.3 Procedimiento

1. Desplegar el servicio web WSEvaluación que implementa el Agente en el servidor Glassfish
2. Desplegar y ejecutar el sistema recomendador semántico que actuá como cliente.
3. Consultar el sistema recomendador semántico para un usuario (estudiante) dado, llamado cli1, con el tema AutomatasPila y un factor de calibración de 0.99. Calcular únicamente e.

5.2.2.5.4 Resultados esperados

1. La consola de administración del servidor Glassfish 4.0 debe mostrar que el servicio web WSEvaluación está desplegado correctamente.

2. El navegador debe visualizar la pagina principal del cliente que emula la nube de auto-formación, quien invoca el servicio del sistema recomendador.
3. En el navegador se cargará la recomendación realizada para el estudiante cli1, con el tema AutomatasPila y el factor de calibración 0.99. Además, se podrá extraer el valor de $AC(O_i, U_k)$ de cada uno de los objetos recomendados.

5.2.2.5.5 Resultados obtenidos

1. Se desplegó correctamente el servicio web WSEvaluación.
2. Se visualizó correctamente la pagina principal del cliente que emula la nube de auto-formación.
3. En la Figura 5.11 se aprecia la recomendación cargada en la pagina principal del cliente que emula la nube de auto-formación, para el estudiante cli1, con el tema AutomatasPila, y un factor de calibración de 0.99. Los valores de $AC(O_i, U_k)$ de cada uno de los objetos recomendados, se muestran en la Tabla 5.6.

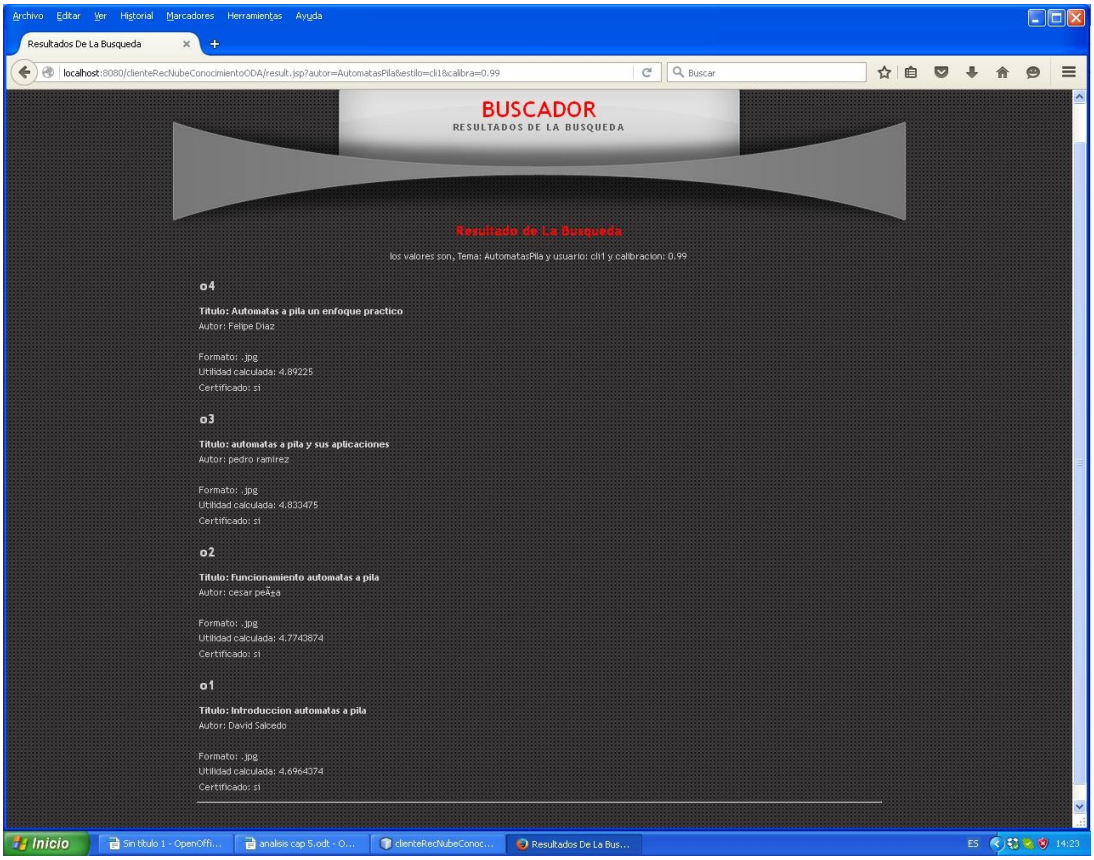

Figura 5.11 Recomendación para el cli1, con el tema AutomatasPila.

Objeto aprendizaje	Valor utilidad	Aporte Colaborativo
o4	4,89	0,89
o3	4,83	0,83
o2	4,77	0,77
o1	4,7	0,7

Tabla 5.6 Resultados obtenidos para el aporte colaborativo

5.2.2.5.6 Conclusiones

La prueba evidencia la influencia del aporte de $AC(O_i, U_k)$ en el sistema recomendador. Es importante resaltar que en esta tesis, el aporte Colaborativo es obtenido de una forma estática, a partir de una noción de calidad estipulada por el experto que introduce los metadatos del OA en el repositorio. Lo anterior implica, que a mayor calificación de calidad del objeto (un valor entre 0 y 5), dada a juicio del experto, mayor es el valor del aporte colaborativo.

5.2.2.6 Prueba del funcionamiento general del Sistema Recomendador de OAs

5.2.2.6.1 Objetivo

Evaluar el funcionamiento y precisión del sistema Recomendador de OAs, en lo referente a la construcción del CONJUNTO DE RECOMENDACIÓN, y al ordenamiento de dicho CONJUNTO DE RECOMENDACIÓN.

5.2.2.6.2 Componentes

- Servicio web que implementa el Sistema Recomendador de OAs
- Agente Adaptativo de aspectos Pedagógico
- Agente Adaptativo de la Usabilidad
- Agente Adaptativo de la Vinculación Semántica de Temas
- Agente Adaptativo de la Calidad y Evaluación de Desempeño
- Cliente que emula la nube de auto-formación
- Servidor de aplicaciones Glassfish

5.2.2.6.3 Procedimiento

1. Desplegar el servicio web WSRecomendador que implementa el Sistema Recomendador de OAs en el servidor Glassfish
2. Desplegar y ejecutar el cliente que emula la nube de auto-formación.
3. Invocar el Sistema Recomendador de OA para el estudiante cli1, con el tema AutomatasPila, con cuatro distintos valores para el factor de calibración (0.25, 0.50, 0.75, y 0.99), los cuales representan distintos niveles de calibración de la hibridación del recomendador.

5.2.2.6.4 Resultados esperados

1. La consola de administración del servidor Glassfish 4.0 deberá mostrar que el servicio web WSRecomendador está desplegado correctamente.
2. El navegador debe visualizar la pagina principal del cliente que emula la nube de auto-formación, quien invoca el servicio del Sistema Recomendador de OAs.
3. En el navegador se cargarán cada una de las cuatro distintas recomendaciones solicitadas para el estudiante cli1, con el tema AutomatasPila, y factores de calibración (0.25, 0.50, 0.75, y 0.99), respectivamente. De cada una de estas cuatro recomendaciones, se puede verificar el valor de la función de utilidad $f(O_i, U_k)$, calculada según ecuación (7), para cada uno de los OAs de cada CONJUNTO DE RECOMENDACIÓN.

5.2.2.6.5 Resultados obtenidos

1. Se desplegó correctamente el servicio web WSRecomendador.
2. Se visualizó correctamente la pagina principal del cliente que emula la nube de auto-formación.
3. En la Tabla 5.7 se visualizan los resultados obtenidos para las cuatro distintas recomendaciones solicitadas, en las cuales se varia únicamente el factor de calibración. Además, la Tabla 5.7 presenta cada una de las cuatro

recomendaciones, rankeadas de acuerdo al valor de la utilidad calculada para cada objeto.

PRUEBA DE HIBRIDACION DEL RECOMEDADOR				
F. CALIBRACION				
0,25	**o1**	**o2**	**o3**	**o4**
utilidad	4,4328	4,359	4,3506	4,3187
0,5	**o1**	**o3**	**o4**	**o2**
utilidad	4,5218	4,5137	4,5125	4,4993
0,75	**o4**	**o3**	**o2**	**o1**
utilidad	4,7062	4,6768	4,6396	4,6109
0,99	**o4**	**o3**	**o2**	**o1**
utilidad	4,8922	4,8334	4,7743	4,6964

Tabla 5.7 Resultados arrojados por el Sistema Recomendador de OAs

5.2.2.6.6 Conclusiones

Esta prueba condensa el funcionamiento general del Sistema Recomendador de OA, y evidencia la correcta ejecución de cada uno de las etapas que lo componen. Es importante evidenciar la influencia notoria del factor de calibración de hibridación del recomendador, respecto de las utilidades calculadas para cada uno de los objetos. Para ello, basta con observar el orden de rankeo de la recomendación dada para un factor de calibración de 0,25 y el rankeo dado para una recomendación con un factor de calibración de 0,99. Se puede tomar como ejemplo la utilidad calculada para el objeto o1 en las distintas recomendaciones. La Tabla 5.7 muestra como el objeto o1 tiene una utilidad de 4,4328 para el factor de calibración de 0.25, y una utilidad de 4,6964 para el factor de calibración de 0.99. La diferencia entre las utilidades calculadas para el mismo objeto o1 se debe a que en el segundo cálculo de utilidad de o1 influye únicamente el aporte colaborativo. Se nota, además, como a pesar de que el OA o1 incrementa su valor de utilidad para la recomendación con factor de calibración de 0.99, pierde su primer puesto en el ranking que poseía en las recomendaciones anteriores, obteniendo el ultimo puesto en el ranking entregado con un factor de calibración de 0.99.

5.2.2.7 Prueba del funcionamiento de la herramienta de gestión del Repositorio Semántico de OAs

5.2.2.7.1 Objetivo

Evaluar el funcionamiento y precisión del Certificador de OAs, que sirve como herramienta de administración del repositorio semántico de OAs.

5.2.2.7.2 Componentes

- Certificador de OAs
- Servidor de aplicaciones Glassfish

5.2.2.7.3 Procedimiento

1. Desplegar la aplicación web Certificador que gestiona OAs del repositorio semántico de OAs en el servidor Glassfish
2. Ejecutar la aplicación web Certificadora de OAs.
3. El usuario experto debe realizar búsquedas en el panel izquierdo de la aplicación web certificadora, para encontrar OAs candidatos a analizar y certificar.
4. Usar la aplicación web Certificador para que un usuario experto almacene metadatos de OAs certificados.

5.2.2.7.4 Resultados esperados

1. La consola de administración del servidor Glassfish 4.0 deberá mostrar que la aplicación web Certificador está desplegada correctamente.
2. El navegador debe visualizar la pagina principal de la aplicación web Certificador.
3. La aplicación certificadora en su panel izquierdo permitirá realizar búsquedas de OAs, para evaluar el interés de los mismos y determinar cuales certificar.

4. La aplicación certificadora, en su panel derecho, permitirá al usuario experto certificar los OAs que considere convenientes. La aplicación certificadora, además, permitirá registrar el conocimiento necesario de cada uno de los metadatos LOM del OA, así como de sus herramientas de aprendizaje, actividades de aprendizaje, y métodos de evaluación.

5.2.2.7.5 Resultados obtenidos

1. Se compiló y desplegó correctamente la aplicación web Certificador.
2. Se visualizó correctamente la pagina principal de la aplicación web certificadora de OAs del repositorio.
3. El usuario experto consultó distintos tipos de repositorios de OAs disponibles a nivel mundial, y se visualizó los metadatos disponibles para cada OA en los distintos repositorios.
4. La Figura 5.12 muestra la forma en que se almacenó el conocimiento de los metadatos LOM del OA o1 dentro del modelo ontológico del repositorio semántico de OAs. Para realizar tal incorporación de conocimiento en el repositorio, se acudió a usar el panel derecho del certificador mostrado en la Figura 4.5.

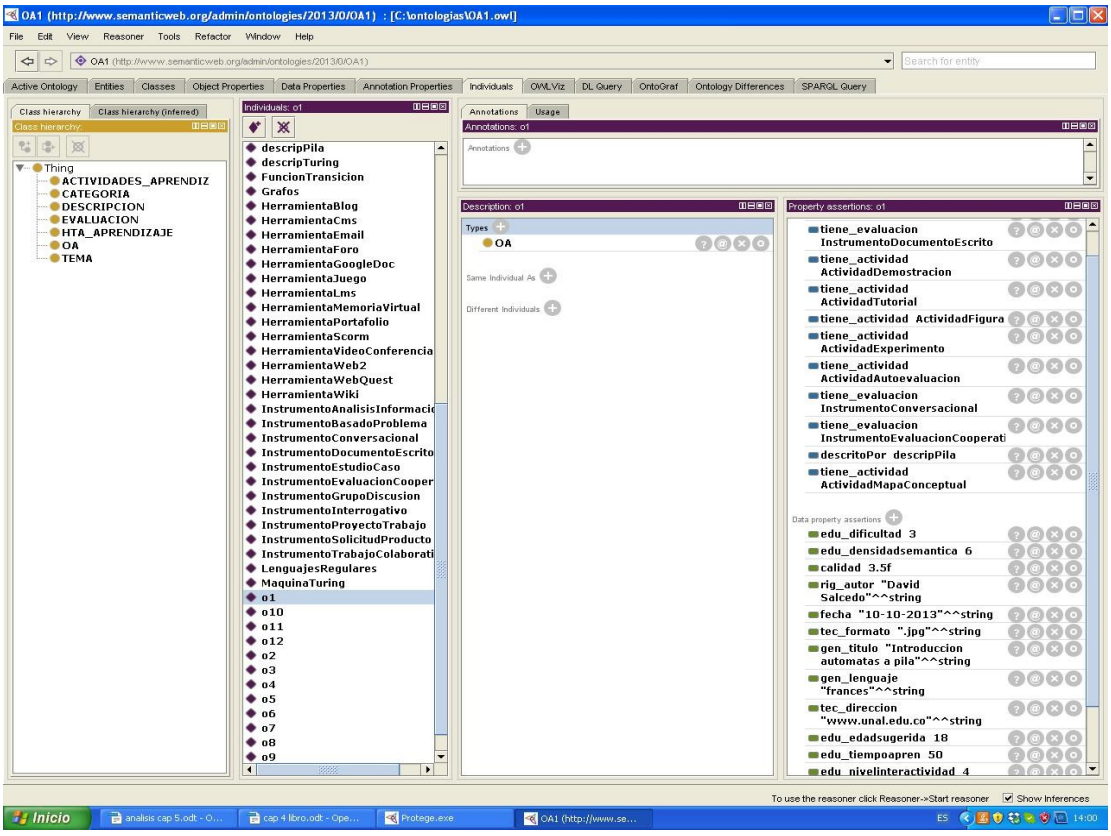

Figura 5.12 OAs certificados almacenados por el Certificador de OA.

5.2.2.7.6 Conclusiones

Esta prueba evidencia el correcto funcionamiento de la herramienta de gestión de conocimiento del repositorio semántico de OAs. Mediante esta prueba se plasmó como el usuario experto puede incorporar nuevos OAs certificados. Este repositorio registra conocimiento de las distintas actividades de aprendizaje, herramientas de evaluación y métodos de evaluación, que posee cada OA, a juicio del experto(ver Figura 5.12).

5.2.2.8 Prueba de la búsqueda (cosechado) de OAs no certificados

5.2.2.8.1 Objetivo

Determinar el correcto funcionamiento del sistema recomendador de OAs, en el momento de realizar búsquedas (cosechado) de OAs no certificados.

5.2.2.8.2 Componentes

- Recomendador de OAs
- Cliente que emula la nube de auto-formación
- Servidor de aplicaciones Glassfish

5.2.2.8.3 Procedimiento

1. Desplegar el servicio web WSRecomendador que implementa el Sistema Recomendador de OAs en el servidor Glassfish
2. Ejecutar la aplicación web cliente que emula la nube de auto-formación.
3. Consultar el sistema recomendador de OAs para un usuario (estudiante) dado, llamado cli1, con el tema graph, y un factor de calibración de 0,5. Es importante anotar que al interior del repositorio certificado de OAs no se encuentra conocimiento de ningún objeto que enseñe dicho tema, ni tema alguno vinculado semánticamente a él.

4. Buscar OAs en las distintas federaciones de OAs disponibles a nivel mundial, tal como se muestra en la Figura 5.13, la búsqueda se realiza invocando el servicio web Rest ofrecido por MERLOT (llamada resaltada dentro de un ovalo de color azul)

5.2.2.8.4 Resultados esperados

1. La consola de administración del servidor Glassfish 4.0 deberá mostrar que el servicio web WSRecomendador está desplegada correctamente.
2. El navegador visualizará la pagina principal de la aplicación cliente que emula la fuente de auto-formación.
3. El sistema recomendador emitirá una recomendación, cuyo conjunto de recomendaciones será compuesto por OAs no certificados, que se encuentran registrados en algún repositorio de OAs disponible a nivel mundial.
4. Recibir el resultado de la búsqueda en formato xml desde MERLOT, como representante de las distintas federaciones de OA disponibles a nivel mundial. La Figura 5.13 muestra como a partir de la linea 51, se realiza el procesamiento de la información recibida mediante el API jdom.

5.2.2.8.5 Resultados obtenidos

1. Se compiló y desplegó correctamente el servicio web WSRecomendador.
2. Se visualizó correctamente la pagina principal de la aplicación web cliente que emula la nube de auto-formación.
3. La Figura 5.14 muestra la recomendación obtenida por el cliente, que fue emitida por el sistema recomendador. La recomendación obtenida está compuesta de OAs no certificados, que se encuentran almacenados en el repositorio de OA MERLOT

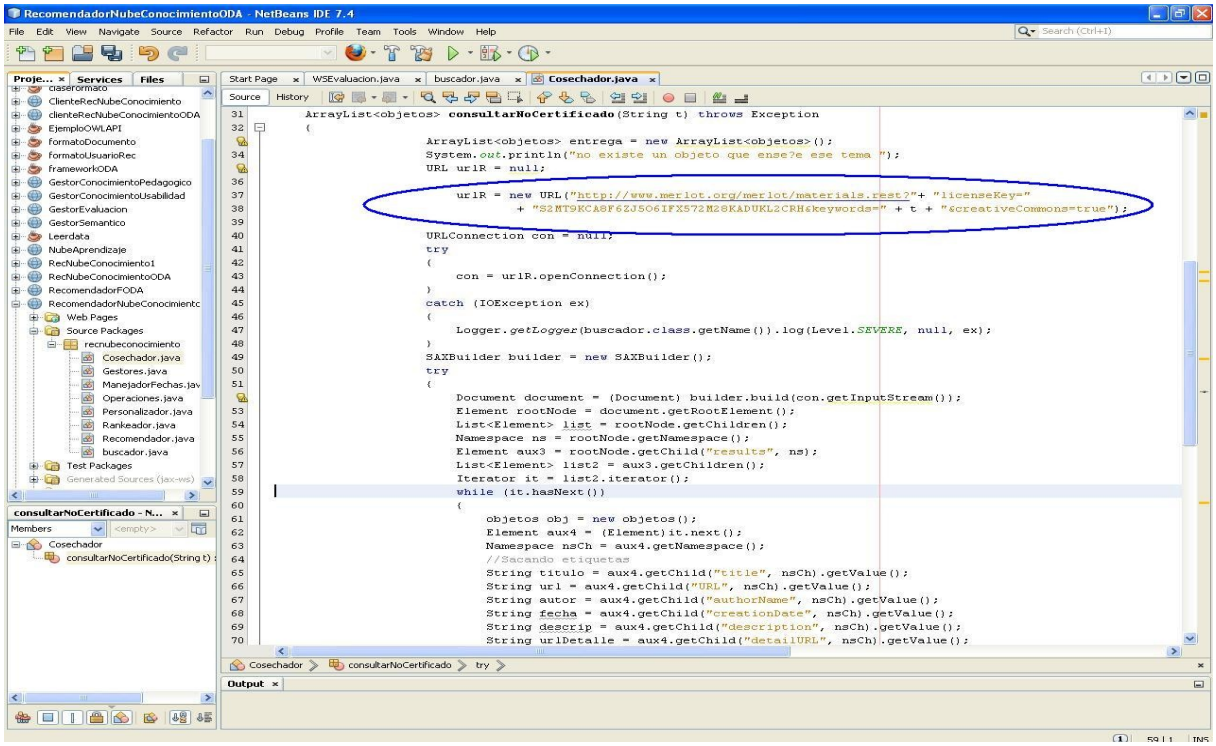

Figura 5.13 Proceso de invocación a MERLOT

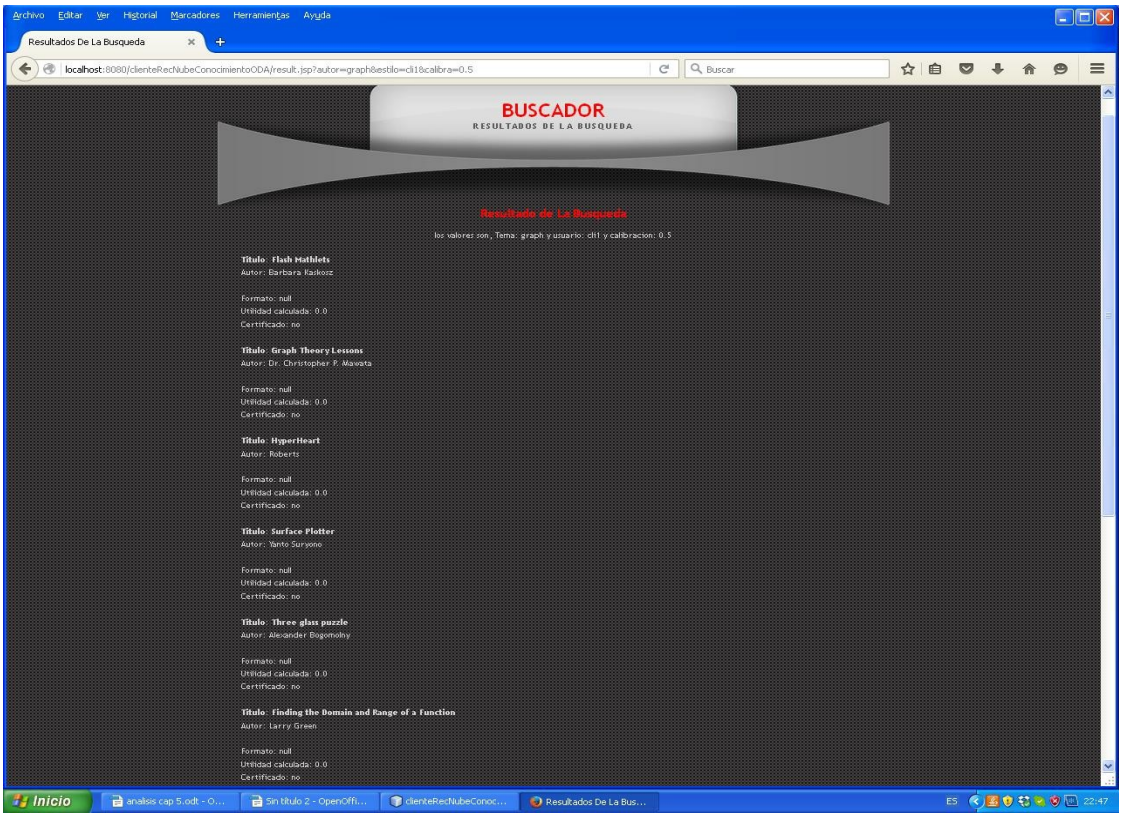

Figura 5.13 Recomendación de OAs no certificados.

5.2.2.8.6 Conclusiones

Esta prueba evidencia el correcto funcionamiento del sistema recomendador en lo referente a la realización de cosechado de OAs no certificados, disponibles en repositorios de trayectoria mundial. Como se aprecia en la Figura 5.13, el sistema recomendador emite la recomendación de ocho distintos OAs, disponibles para el tema graph en el repositorio de OA MERLOT. Debido a que se trata de objetos no certificados, no se cuenta con un rankeo para esta recomendación, pues los metadatos que proveen los repositorios de aprendizaje de donde provienen, no son totalmente compatibles con los metadatos requeridos para calcular la función de utilidad. La prueba evidencia que es posible el cosechado de OAs disponibles no solo en MERLOT, sino en cualquier repositorio que presente un API de consulta. Sin embargo, debe plantearse como trabajo futuro la implementación de un cosechador semántico de OAs que posibilite la adaptación e incorporación de los OAs cosechados, al repositorio semántico de OAs con que cuenta la nube de conocimiento.

5.2.3 PRUEBAS DE INTEGRACIÓN DE LA NUBE DE CONOCIMIENTO CON LAS DEMAS NUBES DEL PROYECTO MADRE

Como se mencionó al inicio del capitulo, es necesario probar la integración de la nube de fuentes de conocimiento, con las nubes de aprendizaje y auto-formación del proyecto madre. Para ello, se planteo dentro del protocolo experimental la realización de dos pruebas de integración, que plasmen la forma en que la nube de conocimiento se acopla con las nubes de aprendizaje y de auto-formación, respectivamente.

5.2.3.1 Prueba de integración de la nube de fuentes de conocimiento con la nube de aprendizaje

5.2.3.1.1 Objetivo

Determinar el nivel de integración de la nube de fuentes de conocimiento con la nube de aprendizajes

5.2.3.1.2 Componentes

- Nube de aprendizaje
- Agente Adaptativo de aspectos Pedagógicos
- Servidor de aplicaciones Glassfish

5.2.3.1.3 Procedimiento

1. Desplegar el servicio web WSNubeAprendizaje que implementa la nube de aprendizaje en el servidor Glassfish
2. Desplegar y Ejecutar el servicio web WSPedagogico que implementa el Agente Adaptativo de aspectos Pedagógicos.
3. Invocar el Agente Adaptativo de aspectos Pedagógicos, para que a través de él se consulte la nube de aprendizaje invocando el servicio web WSNubeAprendizaje, y entregándole como parámetro de entrada el estilo de aprendizaje 2121 de un estudiante codificado de acuerdo a la Tabla 4.4.
4. Invocar el Agente Adaptativo de aspectos Pedagógico. para que a través de él se consulte la nube de aprendizaje invocando el servicio web WSNubeAprendizaje, y entregándole como parámetro de entrada el estilo de aprendizaje 5555 de un estudiante codificado de acuerdo a la Tabla 4.4.

5.2.3.1.4 Resultados esperados

1. La consola de administración del servidor Glassfish 4.0 deberá mostrar que el servicio web WSNubeAprendizaje está desplegado correctamente.

2. La consola de administración del servidor Glassfish 4.0 deberá mostrar que el servicio web WSPedagogico está desplegado correctamente.
3. La nube de aprendizaje retorna un listado de herramientas de aprendizaje, actividades de aprendizaje y métodos de evaluación, sugeridos para que un estudiante con estilo de aprendizaje codificado con 2121, aprenda de la forma más optima posible.
4. La nube de aprendizaje retorna un listado de herramientas de aprendizaje, actividades de aprendizaje y métodos de evaluación, sugeridos para que un estudiante con estilo de aprendizaje codificado con 5555, aprenda de la forma más optima posible.

5.2.3.1.5 Resultados obtenidos

1. Se compiló y desplegó correctamente el servicio web WSNubeAprendizaje.
2. Se desplegó y ejecutó correctamente el servicio web WSPedagogico.
3. Se obtuvo un listado de herramientas de aprendizaje, actividades de aprendizaje y métodos de evaluación, sugeridos para que un estudiante con estilo de aprendizaje codificado con 2121, aprenda de la forma más optima posible, tal como lo muestran las Tablas 5.8, 5.9 y 5.10
4. Se obtuvo un listado de herramientas de aprendizaje, actividades de aprendizaje y métodos de evaluación, sugeridos para que un estudiante con estilo de aprendizaje codificado con 5555, aprenda de la forma más optima posible, tal como lo muestran las Tablas 5.8, 5.9 y 5.10

HERRAMIENTAS DE APRENDIZAJE DADAS POR LA NUBE DE APRENDIZAJE		
ESTILO DE APRENDIZAJE 2121	ESTILO DE APRENDIZAJE 5555	COMPARTIDAS
HerramientaPortafolio HerramientaCms HerramientaLms HerramientaScorm		HerramientaWeb2 HerramientaVideoConferencia HerramientaMemoriaVirtual HerramientaWiki HerramientaGoogleDoc HerramientaEmail HerramientaBlog HerramientaWebQuest HerramientaForo

Tabla 5.8 Herramientas de aprendizaje sugeridas por la nube de aprendizaje para los estilos de aprendizaje codificados como 2121 y 5555, de acuerdo a lo explicado en la Tabla 4.4

ACTIVIDADES DE APRENDIZAJE DADAS POR LA NUBE DE APRENDIZAJE		
ESTILO DE APRENDIZAJE 2121	ESTILO DE APRENDIZAJE 5555	COMPARTIDAS
ActividadTabla ActividadLaboratorio ActividadTaller ActividadExposicion ActividadCuestionario ActividadMetodoCaso ActividadBusqueda ActividadProyecto	ActividadTutorial ActividadDemostracion ActividadExperimento ActividadImagen ActividadMapaConceptual ActividadFigura ActividadAutoevaluacion	ActividadDiscusionGrupo ActividadAudio ActividadCharla ActividadEjercicio ActividadVideo ActividadLectura ActividadTextoNarrativo ActividadConferencia ActividadPortafolio

Tabla 5.9 Actividades de aprendizaje sugeridas por la nube de aprendizaje para los estilos de aprendizaje codificados como 2121 y 5555, de acuerdo a lo explicado en la Tabla 4.4

METODOS DE EVALUACION DADOS POR LA NUBE DE APRENDIZAJE		
ESTILO DE APRENDIZAJE 2121	ESTILO DE APRENDIZAJE 5555	COMPARTIDAS
InstrumentoBasadoProblema InstrumentoSolicitudProducto InstrumentoGrupoDiscusion InstrumentoEstudioCaso	InstrumentoConversacional InstrumentoEvaluacionCooperativa InstrumentoDocumentoEscrito	InstrumentoInterrogativo InstrumentoProyectoTrabajo InstrumentoAnalisisInformacion InstrumentoTrabajoColaborativo

Tabla 5.10 Métodos de evaluación sugeridos por la nube de aprendizaje para los estilos de aprendizaje codificados como 2121 y 5555, de acuerdo a lo explicado en la Tabla 4.4

5.2.3.1.6 Conclusiones

Esta prueba evidencia la integración de la nube de fuentes de conocimiento con la nube de aprendizaje. La Tabla 5.8 muestra las herramienta de aprendizaje sugeridas para dos distintos estilos de aprendizaje, codificados de acuerdo con la Tabla 4.4. Es importante resaltar como para cada estilo de aprendizaje, la nube de aprendizaje sugiere distintas herramientas. Debe notarse que por ejemplo, para el estilo de aprendizaje 5555 todas las herramientas sugeridas por las nubes de aprendizaje son también sugeridas para el estilo de aprendizaje 2121, lo cual evidencia que las herramientas de aprendizaje, métodos de evaluación y actividades de aprendizaje, no son exclusivos para cada estilo de aprendizaje, sino que se solapan de acuerdo a las características del paradigma de aprendizaje de cada individuo. Las herramientas de aprendizaje, actividades de aprendizaje y métodos de evaluación, sugeridos para un estilo de aprendizaje, son aquellas que las teorías de aprendizaje consideran ideales para que un individuo con ese estilo de aprendizaje aprenda de forma optima.

5.2.3.2 Prueba de integración de la nube de fuentes de conocimiento con la nube de auto-formación

5.2.3.2.1 Objetivo

Determinar el nivel de integración de la nube de fuentes de conocimiento con la nube de auto-formación.

5.2.3.2.2 Componentes

- Cliente que emula la nube de auto-formación
- Sistema Recomendador de OAs
- Servidor de aplicaciones Glassfish

5.2.3.2.3 Procedimiento

1. Desplegar el servicio web WSRecomendador que implementa el Sistema Recomendador de OAs en el servidor Glassfish
2. Desplegar y Ejecutar la aplicación web que emula la nube de auto-formación. La nube de auto-formación fue emulada, ya que la nube real que se tiene implementada en la actualidad carece de algunos componentes arquitectónicos actuales contemplados dentro del proyecto madre. En especifico: no posee el repositorio semántico de perfil de usuarios, o no solicita el factor de calibración del recomendador.
3. Ejecutar a través del cliente que emula la nube de auto-formación, una solicitud de recomendación idéntica a la realizada en la prueba interna de funcionamiento general del sistema recomendador realizada en este capítulo.

5.2.3.2.4 Resultados esperados

1. La consola de administración del servidor Glassfish 4.0 deberá mostrar que el servicio web WSRecomendador está desplegado correctamente.

2. La consola de administración del servidor Glassfish 4.0 deberá mostrar que la aplicación web cliente que emula la nube de auto-formación está ejecutándose correctamente.
3. El cliente que emula la nube de auto-formación deberá visualizar las distintas recomendaciones solicitadas en la prueba de funcionamiento general del sistema recomendador. Esto evidencia la invocación del sistema de recomendación al interior de la nube de auto-formación, para realizar las distintas labores de gestión académica que requiera.

5.2.3.2.5 Resultados obtenidos

1. Se compiló y desplegó correctamente el servicio web WSRecomendador.
2. Se desplegó y ejecutó correctamente la aplicación web cliente que emula la nube de auto-formación.
3. El cliente que emula la nube de auto-formación permitió visualizar las distintas recomendaciones solicitadas en la prueba de funcionamiento general del sistema recomendador, sintetizadas en la Tabla 5.7. Estas pruebas arrojan como resultado las disponibilidad de las recomendaciones, debidamente rankeadas al interior de la aplicación con que se emula la nube de auto-formación.

5.2.3.2.6 Conclusiones

Esta prueba no pone en evidencia la integración de la nube de fuentes de conocimiento con la nube de auto-formación. Ahora bien, para esta prueba se emula la nube de auto-formación con una aplicación web cliente, debido a que se requiere una nueva versión de la nube de auto-formación, donde se consideren aspectos como la gestión del repositorio de perfiles de usuario, la generación dinámica de las medidas de calidad de los OAs, la inclusión de un factor de calibración en el recomendador, entre otros aspectos. Los resultados plasmados en la aplicación cliente que emula la nube de auto-formación son válidos, como si las recomendaciones fueran entregadas directamente a la implementación física de esta nube, pues se está garantizando la entrega de una recomendación compuesta de OAs rankeados (ordenados según la adecuación de los mismos al

usuario a quien se le hace la solicitud), tal como lo requiere la nube de auto-formación.

5.3 Evaluación de desempeño de la nube de fuentes de conocimiento

El componente fundamental al interior de la nube de fuentes de conocimiento es el Sistema Recomendador de OAs. Como se mencionó en el capitulo 2, existen múltiples dimensiones para evaluar sistemas recomendadores. Tales dimensiones son divididas, según [41], en cuatro categorías. En el marco de la presente tesis se brinda especial atención a la categoría de las dimensiones de evaluación de desempeño centradas en la recomendación, debido a que el insumo fundamental aportado por la nube de fuentes de conocimiento son las recomendaciones rankeadas de OAs.

Dentro de las dimensiones consideradas para medir desempeño, este trabajo hace especial énfasis en *Exactitud*, debido a que es la que esta más directamente asociada con indicadores de calidad y desempeño del sistema recomendador.

5.3.1 Evaluación de desempeño del sistema recomendador de OAs en cuanto a exactitud

La exactitud (correctness) mide la precisión de las recomendaciones que un sistema recomendador dá. Como se explicó en el capitulo 2, en [41] se presentan tres formas distintas de medir exactitud, las cuales deben ser usadas dependiendo del tipo de recomendación que el sistema genera, y su forma de hacerlo. En el contexto de la presente tesis se utilizan dos de tales medidas:

- *Medidas de precisión basada en predicciones (clasificación):* El sistema recomendador de OAs construye un CONJUNTO DE RECOMENDACIÓN (explicado en la sección 4.2.6.2.1). Dicho proceso consiste en clasificar los OAs del repositorio semántico en dos grupos, los que se deben recomendar (que conforman el CONJUNTO DE RECOMENDACIÓN) y los que no deben recomendarse. Las medidas de precisión basadas en predicciones explicadas en la sección 2.3.4.2.2, son usadas para medir la exactitud y

completitud del CONJUNTO DE RECOMENDACIÓN construido por el recomendador.

- *Medidas de precisión basadas en el rankeo de ítems:* El sistema recomendador de OAs realiza un proceso de rankeo de los objetos que recomienda. Con ello se busca que la recomendación entregada se de en un orden de prioridad, basada en la función de utilidad calculada para cada OA. Con esta medida (explicada en la sección 2.3.4.2.3) se evaluá la calidad del proceso de recomendación de AOs certificados (ver sección 4.2.6).

5.3.1.1 Evaluación de desempeño del sistema recomendador respecto de la exactitud y completitud del CONJUNTO DE RECOMENDACIÓN.

Evaluar el desempeño del sistema recomendador en lo referente al subproceso de construcción del CONJUNTO DE RECOMENDACIÓN (explicado en la sección 4.2.6.2.1), implica construir previamente un conjunto optimo al que se le denomina *conjunto meta estándar* (standard gold). El *conjunto meta estándar* es el CONJUNTO DE RECOMENDACIÓN optimo, construido previamente para cada recomendación. Es decir, cada vez que se solicite recomendación para un tema distinto, se requiere contar con un CONJUNTO DE RECOMENDACIÓN meta estándar (previamente calculado), que contendrá cada uno de los OAs registrados en el repositorio semántico de OAs que enseñan el tema solicitado, o temas vinculados semánticamente a él, y que teóricamente deben ser recomendados en recomendaciones donde se solicite dicho tema (independientemente del estudiante y factor de calibración). El estudiante que solicita y el factor de calibración, únicamente influirán en el orden de rankeo del CONJUNTO DE RECOMENDACIÓN. Para medir el desempeño del recomendador en la construcción del CONJUNTO DE RECOMENDACIÓN, se muestra un ejemplo que sirva de ilustración. Por ejemplo, para el tema AutomatasPila se construyó en forma analítica el *conjunto meta estándar* mostrado en la Tabla 5.11, de acuerdo con el conocimiento almacenado en el repositorio semántico de OAs certificados.

TEMA	AutomatasPila			
OAs	o1	o2	o3	o4

Tabla 5.11 Conjunto *meta estándar* para el tema AutomatasPila.

Para medir el desempeño en la construcción del CONJUNTO DE RECOMENDACIÓN se consideraron los resultados obtenidos en la prueba de funcionamiento general del recomendador (ver sección 5.2.2.6). Tal prueba consta de la realización de cuatro recomendaciones distintas para el mismo tema (AutomatasPila) y estudiante (cli1), pero con distintos valores en el factor de calibración. Los resultados arrojados en dicha prueba se aprecian en la Tabla 5.7, y muestran cada una de las recomendaciones realizadas.

Como se explica en la sección 2.3.4.2.2, para realizar las medidas de precisión basadas en predicción es necesario realizar una clasificación de los OAs que conforman el CONJUNTO DE RECOMENDACIÓN en cuatro categorías: verdaderos positivos (tp), falsos positivos (fp), falsos negativos (fn) y verdaderos negativos (tn). Tal clasificación se realiza comparando los OAs que contiene el CONJUNTO DE RECOMENDACIÓN, respecto a los OAs que conforman el conjunto *meta estándar*. Como resultado, se obtiene una tabla similar a la Tabla 2.5. Para esta prueba, usando el conjunto meta estándar mostrado en la Tabla 5.11, se obtuvo la clasificación mostrada en la Tabla 5.12. Los componentes de la Tabla 5.12 fueron calculados de la siguiente manera:

CR= {CR_i / $CR_i \in$ CONJUNTO DE RECOMENDACIÓN}

CME= {CME_i / $CME_i \in$ conjunto meta estándar}

tp= $|CR \cap CME|$

fp = $|CR - CME|$

fn = $|CME - CR|$

tn = $|CME|$ *- (tp+fp+fn)*

Por ejemplo, tp en la Tabla 5.12 es 4 debido a que son 4 los objetos considerados como verdaderos positivos, es decir se incluyeron cuatro objetos en el CONJUNTO DE RECOMENDACIÓN que se encuentran también en el conjunto meta estándar. Los valores de la Tabla 5.12 sirven como base para el cálculo de las medidas de desempeño recall y precisión (ver sección 2.3.4.2.2), que permiten estimar la completitud y la exactitud del CONJUNTO DE RECOMENDACIÓN. Se tiene una única clasificación para las cuatro distintas pruebas mostradas en la Tabla 5.7, ya

que siempre se obtiene el mismo CONJUNTO DE RECOMENDACIÓN, pues siempre se realiza para el mismo tema, que es el único parámetro que influye en la construcción del CONJUNTO DE RECOMENDACIÓN.

	OA Recomendados	OA no Recomendados	total
OA Usados - Relevantes	tp = 4	fn =0	tp+fn = 4
OA no usados - Irrelevantes	fp = 0	tn = 0	fp + tn = 0
total	tp+fp =4	fn+tn=0	4

Tabla 5.12 Clasificación de los OAs recomendados en la recomendación dada en la Tabla 5.7

Pasemos ahora a definir las dos medidas de desempeño, precisión y recall.

5.3.1.1.1 cálculo de precisión

Precisión calcula el porcentaje de OAs bien recomendados, respecto del total de OAs que se recomendaron para esa solicitud de recomendación (ver sección 2.3.4.2.2.1). De esa manera se determina, que tan exacta es la recomendación respecto de lo que se recomendó. La recomendación será mucho más exacta, en la medida que todos los objetos recomendados se encuentren dentro del conjunto *meta estándar (*no recomendó objetos que no debe recomendar)*.*

Usando la ecuación 2.1, y teniendo en cuenta lo plasmado en la Tabla 5.12, se calcula la precisión:

$$presicion(R_u)=\frac{RA}{A}=\frac{(t_p)}{(t_p+f_p)}=\frac{4}{(4+0)}=1$$

El valor de 1 para el indicador precisión indica que el CONJUNTO DE RECOMENDACIÓN construido por el recomendador tiene una exactitud de 100%. Es decir, el CONJUNTO DE RECOMENDACIÓN no incluye ningún OA que no se debe recomendar en esa recomendación. En nuestro recomendador el valor de precisión siempre va a ser 1 para cualquier recomendación, debido a que fp siempre va a ser 0. Tal afirmación se realiza basados en [68] donde se muestra que el algoritmo de tableaux para las lógicas SHOIQ es completo y correcto. El hecho de que como lo afirma [54] Fact++ use el algoritmo de tableaux para las lógicas SHOIQ, permite demostrar que los procesos de inferencia realizados por Fact++ son correctos y completos (el proceso de razonamiento infiere toda la información requerida, según lo establecido en el modelo ontológico). En particular, el hecho que sea correcto implica que fp=0. En el caso de nuestro sistema recomendador, fp = 0 garantiza que el CONJUNTO DE RECOMENDACIÓN sea totalmente exacto.

Eso lo garantiza porque antes de hacer la evaluación de la función de utilidad, el proceso de inferencia solamente incluye en el CONJUNTO DE RECOMENDACIÓN OAs que estén vinculados semánticamente con el tema solicitado en la recomendación, es decir, nunca incluye un falso positivo en el CONJUNTO DE RECOMENDACIÓN.

5.3.1.1.2 cálculo de recall

Recall calcula el porcentaje de OAs bien recomendados respecto del total de AOs que se deben recomendar para esa solicitud de recomendación. Ella determina que tan completa es la recomendación respecto de lo que se recomendó, siendo mucho más completa (no dejo de recomendar objetos que debía recomendar) en la medida que todos los objetos que se deben recomendar se encuentren dentro del CONJUNTO DE RECOMENDACIÓN construido para dicha solicitud de recomendación.

Usando la ecuación 2.2, y teniendo en cuenta lo plasmado en la Tabla 5.12, se calcula recall :

$$recall(R_u)=\frac{RA}{R}=\frac{(t_p)}{(t_p+f_n)}=\frac{4}{(4+0)}=1$$

El valor de 1 para el indicador recall determina que el CONJUNTO DE RECOMENDACIÓN construido por el recomendador tiene una completitud de 100%. Es decir, el CONJUNTO DE RECOMENDACIÓN no deja de incluir ningún OA que debería recomendar en esa recomendación. El valor de recall va a ser siempre 1 para cualquier recomendación de nuestro sistema, debido a que fn siempre va a ser 0. Igualmente, basado en [68], Fact++ infiere todos los OAs requeridos, según lo establecido en el modelo ontológico, para un usuario dado. El hecho de que sea completo implica que fn=0. De esta manera, en el caso de nuestro sistema recomendador, fn=0 garantiza que el CONJUNTO DE RECOMENDACIÓN sea totalmente completo. El sistema recomendador lo garantiza porque antes de hacer la evaluación de la función de utilidad, el proceso de inferencia incluye en el CONJUNTO DE RECOMENDACIÓN todos los OAs vinculados semánticamente con el tema solicitado en la recomendación, es decir, nunca deja de incluir un OA que esté vinculado semánticamente con el tema solicitado en la recomendación, en el CONJUNTO DE RECOMENDACIÓN (no genera un falso negativo).

5.3.1.1.3 cálculo de f-measure

Finalmente se calcula f-measure [40], como una medida capaz de combinar la información aportada por recall y precisión. f-measure logra determinar si el CONJUNTO DE RECOMENDACIÓN es totalmente completo y totalmente exacto. Usando la ecuación 2.3, se calcula f_measure.

$$f-measure(R_i)=\frac{(2*precision(R_i)*recall(R_i))}{(precision(R_i)+recall(R_i))}=\frac{(2*1*1)}{(1+1)}=1 \qquad (5.3)$$

5.3.1.1.4 Análisis de resultados

Obviamente, al ser recall y precisión iguales a 1 f-measure también lo será, es decir, el CONJUNTO DE RECOMENDACIÓN es totalmente completo y exacto. Lo anterior implica que el sistema de recomendación no agrega OAs al CONJUNTO DE RECOMENDACIÓN que no debe agregar (precisión = 1), y que el CONJUNTO DE RECOMENDACIÓN construido por el sistema recomendador es totalmente completo, es decir, no falta ningún OA que se deba recomendar dentro del CONJUNTO DE RECOMENDACIÓN (*recall* = 1). Eso lo garantiza el proceso de inferencia de los OAs a recomendar, vinculados semánticamente al tema solicitado (ver pseudo-código de la figura 4.12)

5.3.1.2 Evaluación de desempeño del sistema recomendador respecto del rankeo de OAs que realiza

Como se explicó en la sección 2.3.4.2.3, la medida de desempeño basada en distancia normalizada (NDPM) es muy usada para determinar la calidad del rankeo realizado por un sistema recomendador.

Para el caso de esta tesis, se evaluá la calidad de los ranking realizados en la prueba de funcionamiento general del sistema recomendador de OAs mostrada en la Tabla 5.7. Para ello se calcula el valor de la NDPM de cada una de las solicitudes de recomendación realizadas para cada distinto factor de calibración mostrado en la Tabla 5.7.

5.3.1.2.1 Determinación del ranking de referencia

Para determinar el ranking de referencia (explicado en la sección sección 2.3.4.2.3), se tuvieron en cuenta los siguientes aspectos asociados con los metadatos de cada uno de los OAs almacenados en el repositorio semántico de OAs de la nube de fuentes de conocimiento.

- Cantidad de herramientas de aprendizaje que emparejan entre las estipuladas para el OA y las sugeridas por la nube de aprendizaje para el estilo de aprendizaje del estudiante que solicita la recomendación.
- Cantidad de actividades de aprendizaje que emparejan entre las estipuladas para el OA y las sugeridas por la nube de aprendizaje para el estilo de aprendizaje del estudiante que solicita la recomendación.
- Cantidad de instrumentos de evaluación que emparejan entre las estipuladas para el OA y las sugeridas por la nube de aprendizaje para el estilo de aprendizaje del estudiante que solicita la recomendación.
- Calidad estipulada en el repositorio para el OA.
- Nivel de vinculación semántica entre el tema asociado con el OA y el tema solicitado en la recomendación.
- Número de preferencias de usuario almacenados en el agente de adaptabilidad de usabilidad que emparejan con las características del OA.
- Valor de los indicadores de uso asociados con el OA dentro del historial de uso almacenado en el agente de adaptabilidad de usabilidad.

Cada uno de los ranking de referencia es un ranking que se ajusta al definido mediante la ecuación (2.5). Las Tablas 5.13, 5.14, 5.15 y 5.16 estipulan cada uno de los ranking de referencia construidos usando como base los metadatos almacenados para cada OA, para los factores de calibración de 0.25, 0.50, 0.75 y 0.99, respectivamente.

Ranking de Referencia	(o1,o2)	(o1,o3)	(o1,o4)	(o2,o3)	(o2,o4)	(o3,o4)
Ranking obtenido	(o1,o2)	(o1,o3)	(o1,o4)	(o2,o3)	(o2,o4)	(o3,o4)
Valores calculados	**C^m**	**0**	**C^u**	**0**	**C**	**6**

Tabla 5.13 Cálculos realizados para la NDPM para la recomendación mostrada en la Tabla 5.7 con factor de calibración de 0,25.

La medida de NDPM, para el factor de calibración 0,25, usando la ecuación 2.4, es:

$$ndpm(R_{0,25}) = \frac{((2*0)+(0))}{(6)} = 0$$

Tal valor de NDPM evidencia que el ranking obtenido es idéntico al esperado (ver tabla 5.14).

Ranking de Referencia	(o1,o3)	(o1,o4)	(o1,o2)	(o3,o4)	(o3,o2)	(o4,o2)
Ranking obtenido	(o1,o3)	(o1,o4)	(o1,o2)	(o3,o4)	(o3,o2)	(o4,o2)
Valores calculados	**C^m**	**0**	**C^u**	**0**	**C**	**6**

Tabla 5.14 Cálculos realizados para la NDPM para la recomendación mostrada en la Tabla 5.7 con factor de calibración de 0,50.

La medida de NDPM para el factor de calibración 0,50 es:

$$ndpm(R_{0,50}) = \frac{((2*0)+(0))}{(6)} = 0$$

De nuevo, el valor de 0 para la NDPM evidencia que el ranking obtenido es idéntico al esperado (ver tabla 5.15).

Ranking de Referencia	(o4,o3)	(o4,o2)	(o4,o1)	(o3,o2)	(o3,o1)	(o2,o1)
Ranking obtenido	(o4,o3)	(o4,o2)	(o4,o1)	(o3,o2)	(o3,o1)	(o2,o1)
Valores calculados	**C^m**	**0**	**C^u**	**0**	**C**	**6**

Tabla 5.15 Cálculos realizados para la NDPM para la recomendación mostrada en la Tabla 5.7 con factor de calibración de 0,75

La medida de NDPM para el factor de calibración 0,75 es:

$$ndpm(R_{0,75}) = \frac{((2*0)+(0))}{(6)} = 0$$

De nuevo, el valor de 0 para la NDPM evidencia que el ranking obtenido es idéntico al esperado (ver tabla 5.16).

Ranking de Referencia	(o4,o3)	(o4,o2)	(o4,o1)	(o3,o2)	(o3,o1)	(o2,o1)
Ranking obtenido	(o4,o3)	(o4,o2)	(o4,o1)	(o3,o2)	(o3,o1)	(o2,o1)
Valores calculados	**C^m**	**0**	**C^u**	**0**	**C**	**6**

Tabla 5.16 Cálculos realizados para la NDPM para la recomendación mostrada en la Tabla 5.7 con factor de calibración de 0,99.

Finalmente, la medida de NDPM para el factor de calibración 0,99 es:

$$ndpm(R_{0,99}) = \frac{((2*0)+(0))}{(6)} = 0$$

Igualmente, el valor de 0 para la NDPM evidencia que el ranking obtenido es idéntico al esperado.

La razón de lo anterior, es debido a que al determinar la calidad del ranking realizado, lo que se está calculando indirectamente es la calidad de la implementación de la función de utilidad. Es decir, NDPM determina con que nivel de confiabilidad la función de utilidad asigna un valor de utilidad a cada objeto, conforme con la expresión matemática expresada mediante la ecuación (4.7). Los resultados anteriores permiten generalizar la conclusión de que NDPM valdrá 0 para cualquier recomendación, debido a que los cálculos manuales realizados para calcular el ranking de referencia en cada caso, permitió evidenciar la correcta implementación del calculo de la función de utilidad en el sistema recomendador. Además, permite concluir que dicha función de utilidad describe correctamente los diferentes criterios usados para evaluar los OAs, según las preferencias del usuario a quien se le hace la recomendación.

5.3.2 Comparación del sistema recomendador con otros trabajos

Para comparar el sistema recomendador presentado en esta tesis con otros trabajos, se pueden asumir dos tipos de comparaciones: una comparación cuantitativa (en la que se compare la completitud, exactitud y calidad del rankeo de la recomendación), y/o una comparación cualitativa, en la que se comparen cualidades especificas de los distintos trabajos.

En lo referente a una comparación cuantitativa, solo uno de los trabajos presentados en el estado de arte realiza una evaluación de desempeño [15]. Por tal motivo, únicamente con él se pueden realizar comparaciones cuantitativas de desempeño. De acuerdo con la Tabla 2 de [15], el recomendador de OAs que ellos presentan tiene un *recall* de 0,4642, valor más bajo que el obtenido por el recomendador presentado en está tesis. Nuestro sistema recomendador es totalmente completo en el momento de construir los conjuntos de recomendación, mientras que el recomendador presentado en [15] solo considera un 46,42% de los objetos a recomendar. En cuanto al valor de precisión, tanto el sistema recomendador de OA presentado en esta tesis como el presentado en [15] tienen un valor de 1, lo cual garantiza una exactitud del 100% en los OAs recomendados.

A continuación se realiza una comparación cualitativa del sistema recomendador desarrollado en esta tesis con otros trabajos. La comparación a realizar se dividió en dos aspectos:

- Comparación cualitativa de los distintos aspectos asociados con la adaptatividad del proceso de recomendación.
- Comparación cualitativa de los aspectos asociados con el diseño, la implementación y funcionalidad del sistema recomendador.

5.3.2.1 Comparación cualitativa de los aspectos de adaptatividad

La Tabla 5.6 muestra una comparación cualitativa realizada desde el punto de vista de la adaptatividad, entre varios sistemas recomendadores de OAs. En particular, está basada en los siguientes criterios:

1. Realiza recomendaciones rankeadas de acuerdo con una función de utilidad que evalúa la adaptabilidad de cada objeto recomendado.
2. Usa un repositorio semántico de OAs, cuyos metadatos permiten construir conjuntos de recomendación adaptables.
3. Cuenta con un mecanismo de calibración, que permite mezclar los componentes de adaptabilidad por personalización y colaborativo.
4. Utiliza algoritmos de recomendación estándares para el cálculo de la adaptabilidad
5. Personaliza y adapta la recomendación con base en los estilos y paradigmas de aprendizaje.
6. Tiene un componente de usabilidad que establece el aporte adaptativo (de personalización) con base a los perfiles de usuario.
7. Plantea el uso de agentes adaptativos, que analizan cada criterio de adaptabilidad de cada objeto a recomendar.
8. Propone un agente adaptativo de la evaluación del rendimiento y calidad de los OAs, basado en las experiencias de recomendaciones dadas en el pasado a otros usuarios.

C	Nuestro Enfoque	[7]	[11]	[12]	[15]	[17]
1	SI	SI	NO	SI	NO	NO
2	SI	NO	NO	SI	NO	SI
3	SI	NO	NO	NO	NO	NO
4	NO	NO	SI	SI	SI	NO
5	SI	NO	NO	NO	SI	NO
6	SI	SI	NO	SI	SI	NO
7	SI	NO	NO	NO	NO	NO
8	SI	NO	NO	SI	NO	NO

TABLA 5.6. COMPARACIÓN RESPECTO A LS ADAPTATIVIDAD, ENTRE DIFERENTES SISTEMAS RECOMENDADORES DE OAS

Desde el aspecto adaptativo, el sistema recomendador adaptativo de OAs de la nube de fuentes del conocimiento es el más completo, pues cumple con la mayor cantidad de criterios de comparación. Como desventaja, tiene el hecho de no utilizar algoritmos estándares de recomendación usados en otros trabajos (criterio 4). Esos algoritmos aportan medidas estadísticas estándares para el cálculo de similaridad. Sin embargo, a pesar de no usar tales medidas, el trabajo acá presentado logra determinar una similaridad ontológica, al incluir los distintos niveles de vinculación semántica como criterio de selección de OAs (explicado en la sección 4.2.3).

Por otro lado, la arquitectura propuesta en este trabajo es la única que contempla el factor de calibración como mecanismo de hibridación (criterio 3). Junto con [12, 15] esta arquitectura usa un agente adaptativo de la evaluación del rendimiento y calidad de los OAs basado en las experiencias de recomendaciones dadas en el pasado a otros usuarios (criterio 8). En el caso de la arquitectura propuesta en este trabajo, eso significa que ella no debe sufrir ninguna modificación en el momento en que la nube de auto-formación implemente un mecanismo de realimentación que permita crear un modelo dinámico e inteligente para gestionar la evaluación de rendimiento y la calidad de cada uno de los OAs que se incorporen en el repositorio.

Adicionalmente, a pesar de que [15] contempla aspectos pedagógicos en el momento de realizar la recomendación (criterio 5), el sistema recomendador adaptativo acá presentado es quien da mayor cobertura en dicho aspecto, pues no solo tiene en cuenta el estilo de aprendizaje del estudiante que solicita la recomendación, sino que además contempla los aspectos didácticos relacionados con el paradigma de aprendizaje asociado a los contenidos de los OAs (como son, las herramientas de aprendizaje incorporadas en cada OA, los instrumentos de

evaluación de cada OA, y las actividades de aprendizaje desarrolladas por cada OA).

A pesar de que tanto en [12] como en [7] se proponen mecanismos de rankeo basados en el calculo de una función de utilidad (criterio 1), no tienen en cuenta aspectos de adaptatividad pedagógica como los contemplados por el agente de adaptatividad pedagógica de nuestro sistema recomendador. Además, nuestro recomendador es el único que cuenta con un repositorio de OAs certificados (criterio 2). [12] contempla le existencia de un repositorio de OAs, pero no cuenta con el proceso de certificación de objetos.

Nuestro recomendador es el único que incorpora agentes adaptativos independientes, que contribuyen al cálculo de la función de utilidad (y con ello, a la capacidad adaptativa del recomendador). En [7] se usan agentes, pero no se trata de agentes adaptativos, sino de agentes BDI que sirven como mecanismos de implementación de ciertas etapas del proceso.

Al igual que nuestro recomendador, los trabajos presentados en [7,12,15] cuentan con mecanismos que tienen en cuenta aspectos de usabilidad (criterio 6). Sin embargo, nuestro recomendador es el único que fusiona lo referente a preferencias de uso con el historial de uso de cada estudiante. También, la arquitectura propuesta es la única respecto a la literatura revisada que mezcla tres ámbitos de recomendación de OAs: basada en contenidos, en filtrado colaborativo, y en conocimiento.

Después de comparar cada uno de los criterios, se puede establecer que ninguno de los trabajos involucrados en esta comparación tienen una total similaridad con nuestro recomendador. El recomendador más cercano es el presentado en [12], el cual es similar en un 62.5% de los criterios de comparación establecidos (5 de 8).

5.3.2.2 Comparación cualitativa basada en aspectos asociados con el diseño, la implementación y funcionalidad

La Tabla 5.7 muestra los resultados obtenidos al realizar una comparación de nuestro sistema recomendador de OAs, con respecto a otros sistemas recomendadores, teniendo en cuenta criterios asociados con el diseño, la

implementación y funcionalidad de los mismos. A continuación se enumeran los criterios de comparación usados:

1. Realizan una evaluación de desempeño en lo referente a la exactitud y completitud de la recomendación.
2. Realizan una evaluación de desempeño en lo referente a la calidad del ranking de la recomendación.
3. Cuenta con una herramienta para gestionar y certificar OAs.
4. Busca OAs externos al repositorio.
5. Está implementado como un servicio web semántico.
6. Usa modelos de conocimiento basado en ontologías para gestionar el conocimiento necesario.
7. Su diseño está basado en la arquitectura ODA.
8. Utiliza el estándar LOM para describir los metadatos de cada OA.

C	**Nuestro Enfoque**	**[7]**	**[11]**	**[12]**	**[15]**	**[17]**
1	SI	NO	NO	NO	SI	NO
2	SI	NO	NO	NO	NO	NO
3	SI	NO	NO	NO	NO	NO
4	SI	SI	SI	NO	NO	NO
5	SI	NO	NO	NO	NO	NO
6	SI	NO	NO	NO	NO	SI
7	SI	NO	NO	NO	NO	NO
8	SI	SI	SI	SI	NO	SI

Tabla 5.7. Comparación cualitativa respecto de diseño, implementación y funcionalidad.

El sistema recomendador de OAs de la nube de fuentes de conocimiento es el único que está implementado como un servicio web semántico (criterio 5), diseñado basado en la arquitectura ODA (criterio 7). Lo anterior implica que, como se contempla en [62,63], queda abierta la posibilidad para que aplicaciones clientes de la nube de fuentes de conocimiento puedan realizar descubrimiento, composición y ejecución automática del sistema recomendador.

También, el sistema recomendador es el único que cuenta con una herramienta para que un usuario experto certifique OAs, para incorporar nuevos objetos dentro de un repositorio certificado (criterio 3). Así, las recomendaciones realizadas por el sistema recomendador son constituidas generalmente por OAs, cuya evaluación y determinación de calidad son garantizadas por el juicio de un usuario experto en

el área, pero no solo se limita a ellas, porque puede buscar externamente (criterio 4). Así, además de tales recomendaciones certificadas, el sistema recomendador de OAs es el único que puede acudir a realizar una búsqueda de OAs disponibles en repositorios mundiales, únicamente en caso de no contar con OAs certificados para recomendar.

El hecho de que se use ontologías (basadas en lógica descriptiva, criterio 6), garantiza que nuestro trabajo entregue un conjunto de recomendaciones exacto y completo. Además, a nuestro sistema recomendador es al único que se le evalúa la calidad del ranking, usando la medida NDPM (criterio 2). Sin embargo, junto con [15], son los únicos que miden la exactitud y completitud de la recomendación (criterio 1).

La selección de LOM como estándar para la representación de los metadatos del recomendador fue muy acertada, pues la casi totalidad de los trabajos usan dicho estándar (Criterio 8), lo cual muestra su gran nivel de aceptación y de utilización.

6 Conclusiones y Trabajos Futuros

6.1 Conclusiones

Este trabajo, conjuntamente con el presentado en [6], forman parte de la nube de fuentes de conocimiento. En este trabajo se desarrolla el componente que se encarga de gestionar todo lo referente a OAs, específicamente, un sistema recomendador híbrido calibrado que realiza dos tipos de recomendaciones: una recomendación de OAs certificados (que garantiza la calidad de los objetos recomendados), o una recomendación no certificada (que recomienda OAs cosechados de repositorios disponibles a nivel mundial). Este trabajo se diferencias con el presentado en [6] en el tipo de recursos que tratan, en [6] se propone un recomendador de contenidos digitales y acá se propone un sistema recomendador semántico de OAs.

Para cumplir con el objetivo de seleccionar los estándares, normas, o reglas, a usar en el sistema recomendador, se escogió el estándar LOM para definir el conocimiento asociado con los metadatos de OAs. Según la literatura revisada, LOM es el estándar con más nivel de aceptabilidad y uso. Por otro lado, se seleccionó el lenguaje OWL, basado en lógica descriptiva, como formalismo de representación de conocimiento, debido a su alto poder de expresividad y a sus características de inferencia.

Con respecto al objetivo de realizar un modelo arquitectónico para la gestión de OAs en la nube de fuentes de conocimiento, este trabajo propone una arquitectura para la gestión de OAs en la nube fuentes de conocimiento en el capítulo 4, cuya implementación se presenta en el capítulo 5. Dicha arquitectura es completamente modular, lo que permite y facilita que cada uno de sus componentes arquitectónicos puedan ser actualizados, modificados y reemplazados, sin generar alteraciones en los demás componentes (basado en la idea de agentes adaptativos).

También, en este trabajo se planteo el objetivo de realizar el diseño del modelo arquitectónico basado en el paradigma ODA. En particular, en la sección 4.3 se presenta la especificación semántica del sistema recomendador según el paradigma ODA. Además, se diseñó e implementó como parte de este trabajo un Framework para la descripción semántica de servicios web semánticos, denominado FODAS-WS(ver capitulo 3). Dicho Framework cuenta con las tres capas clásicas del paradigma ODA, que son: CIM, PIM y PSM, y cada una de ellas cuenta con un grupo de componentes que permiten la especificación, diseño y desarrollo de servicios web semánticos, como de sus clientes.

Finalmente, se alcanzo el objetivo de diseñar e implementar un prototipo que realice la gestión de OAs en la nube de fuentes de conocimiento. En particular, el diseño del servicio web semántico se realizó usando FODAS-WS. Implementar el sistema recomendador como servicio web semántico, permite que a futuro se pueda realizar descubrimiento, composición y ejecución automática, por parte de cualquier cliente que requiera una recomendación de OAs.

De esta manera, en este trabajo se logro el Objetivo general de desarrollar un prototipo basado en SOA, para la gestión de OAs en la nube de fuentes de conocimiento, aplicando el paradigma ODA. En particular, el sistema recomendador propuesto tiene las siguientes cualidades diferenciadoras, con respecto a otros trabajos:

- Su función de utilidad, que constituye el instrumento matemático que pondera el impacto de los distintos componentes vinculados a la arquitectura. Dicha función es el mecanismo integrador e hibridizador de la arquitectura. Además, posibilita reconfigurar el recomendador, lo cual lo hace flexible y escalable. La función de utilidad entrega las recomendaciones priorizadas, personalizadas y adaptadas a cada usuario.
- Su carácter híbrido calibrado, lo cual permite balancear las recomendaciones entregadas, teniendo en cuenta las experiencias de recomendaciones anteriores, el emparejamiento de los metadatos de los OAs con respecto a los intereses o cualidades del estudiante, y la relación semántica entre el tema solicitado y cada OA encontrado.

- Su repositorio semántico basado en LOM, que además incluye la caracterización de aspectos pedagógicos, como los instrumentos de evaluación, y las herramientas y actividades de aprendizaje.
- Su mecanismo de personalización híbrido, que mezcla las preferencias de uso *(APPP)* y las historias de uso *(APPH).* Esto permite que el recomendador pueda determinar con que objetos se siente más cómodo el estudiante, teniendo en cuenta tanto las manifestaciones directas realizadas por el estudiante (al registrar las preferencias de uso), como las realizadas de forma indirecta (almacenadas en las historias de uso).
- Su agente adaptativo de vinculación semántica de temas, el cual puede ser usado en distintos contextos (en otras aplicaciones), pues lo que busca es establecer relaciones semánticas entre conceptos.
- Su conjunto de recomendación, el cual es exacto y completo (no deja sin recomendar ningún OA que deba recomendar, ni incluye en la recomendación ninguno que no deba recomendar). Además, la priorización (ranking) que se hace de los OAs contenidos en el conjunto de recomendación es la correcta. Los valores obtenidos por el sistema recomendador en las métricas recall, precisión, f-measure y NDPM, constatan lo anterior.
- Su diseño y especificación como servicio web, usando el Framework FODAS-WS, lo que implica que no solo se cuenta con un diseño que puede ser entendido por humanos, sino que puede ser entendido por aplicaciones de software. Esto último posibilita la realización de descubrimiento, composición y ejecución automática de servicios, pero además, disminuye las inconsistencias entre el diseño del servicio web semántico, su implementación, y sus procesos de composición y ejecución. Para ello, FODAS-WS usa modelos ontológicos comprensibles a la máquina, almacenados en tres capas de distinto nivel de abstracción, que se llaman CAPACIM, CAPAPIM y CAPAPSM (las cuales se ajustan completamente a lo propuesto en la arquitectura ODA): realiza transformaciones automáticas entre capas; y usa un validador semántico que garantiza que el conocimiento almacenado en las distintas capas sea consistente y completo.

En general, el sistema recomendador propuesto se considera un sistema recomendador adaptativo por varios motivos. Por un lado, el hecho de que el sistema recomendador considere estilos de aprendizaje, herramientas de aprendizaje, instrumentos de evaluación y actividades de aprendizaje idóneas para cada estudiante, es evidencia de la capacidad adaptativa pedagógica del sistema recomendador. Además, el recomendador tiene capacidad adaptativa por usabilidad, que garantiza que a cada usuario se le estén generando recomendaciones personalizadas respecto a sus preferencias de uso. Por otro lado, el hecho de que el sistema recomendador involucre un componente colaborativo, permite que las recomendaciones entregadas contemplen experiencias anteriores de recomendación. Finalmente, la vinculación semántica de temas posibilita que las recomendaciones contemplen un componente cognitivo que genere recomendaciones de no solo OAs directamente relacionados con el tema, sino que a través de un proceso de razonamiento, además recomienda objetos con múltiples relaciones semánticas con el tema solicitado.

Otros aspectos a destacar del sistema recomendador son:

- El hecho de que se realice recomendaciones de OAs no certificados, garantiza que la capacidad de respuesta del sistema recomendador sea alta, debido a que la probabilidad de encontrar OAs para cierto tema en las federaciones de repositorios de OAs es alta. Sin embargo, las recomendaciones no certificadas corren el riesgo de ser de menor calidad, ya que la búsqueda se realiza únicamente considerando palabras claves (vinculación semántica).
- El hecho de que el agente adaptativo pedagógico sea quien se comunica directamente con la nube de aprendizaje, permite que el sistema recomendador sea independiente del proceso de caracterización pedagógica (realizada por la nube de aprendizaje). Además, cualquier modificación en la forma de calculo del aporte pedagógico proporcionado por este agente, no requiere modificar el sistema recomendador.
- El proceso de certificación es manual, y requiere que el usuario que incorpore el conocimiento de los metadatos de cada OA sea experto. Dicha experticia será quien garantice la calidad del conocimiento que se incorpore para los objetos que se registren en el repositorio como certificados. Por

otro lado, el valor de calidad de cada OA es determinado por el experto según sus criterios, algo que deberá ser mejorado en trabajos futuros.

- La determinación del conjunto *meta estándar* (para calcular recall y presicion) y del *ranking de referencia* (para calcular NDPM), deben ser realizados con anterioridad a la ejecución de las pruebas. Próximos trabajos deberán analizar como formalizar el proceso de obtención de los mismos, de tal manera de hacer menos subjetivos los procesos de calculo de cada métrica usada en la tesis.

Con respecto al Framework FODAS-WS, es importante clarificar que la subcapa CIM REFERENCIAL registra las capacidades, necesidades, precondiciones y efectos del servicio (según la arquitectura SOA). La subcapa CIM OPERACIONAL describe cada una de las capacidades definidas en la capa CIM REFERENCIAL, y sirve de base para los procesos de descubrimiento (su conocimiento sobre las funcionalidades de cada capacidad del servicio) y de composición automática (su conocimiento sobre el proceso llevado a cabo por cada capacidad del servicio).

Por otro lado, durante la realización de este trabajo se generaron otros trabajos relacionados con el mismo, en particular:

- En [67] se diseñó e implementó un sistema recomendador de patrimonios web basado en nuestro sistemas recomendador, que ofrece dos ventajas fundamentales sobre las herramientas disponibles para almacenar el patrimonio de la web. La primera, su nivel de expresividad semántica, pues permite no solo buscar y recomendar documentos web asociados con un tema base, sino que además recomienda documentos asociados con otros niveles de descripción semántica. Una segunda ventaja es su nivel de personalización, ya que normalmente tales herramientas solo ofrecen búsquedas convencionales, sin considerar a que tipo de usuario va dirigido.
- En [69] se diseñó e implementó un sistema recomedador híbrido calibrado de ontologías biomédicas basado en nuestra propuesta, cuyos clientes pueden ser aplicaciones o humanos interesados en ontologías que describan terminología biomédica, o temas específicos como: genética, manejo de virus, enfermedades y patologías especificas, entre otros. En el sistema recomendador, cada uno de los usuarios cuenta con su perfil definido, de

acuerdo tanto a sus preferencias de investigación, como a sus niveles de conocimiento y formas de trabajo. Eso implica que cuando un agente investigador usa el sistema recomendador, obtiene como respuesta una recomendación personalizada y priorizada, no solo de acuerdo al tema biomédico del que necesita conocer, sino también acorde con su perfil de usuario.

2. Trabajos Futuros

Algunos trabajos futuros que se derivan desde el presente trabajo son:

- Implementar un modulo de generación automática de código al interior de FODAS-WS, que construya el esqueleto del código de los servicios web semánticos que se especifiquen en el Framework. Con dicho trabajo se garantiza que las implementaciones de los servicios web diseñados sean totalmente acorde con las especificaciones escritas mediante lógica descriptiva. Para su realización, se debe considerar el conocimiento almacenado en las distintas capas CIM, PIM y PSM.
- Diseñar e implementar un servicio web ws-generator en FODAS-WS, que reciba peticiones de aplicaciones web, y sirva de intermediario para realizar descubrimiento, composición y ejecución automática de servicios web semánticos especificados con FODAS-WS, entregando los resultados deseados a la aplicación web cliente en el formato que lo requiera. Dicho servicio web es de gran importancia, ya que cualquier aplicación web cliente podría hacerle solicitudes al servicio de intermediación, para usar en forma automática servicios web semánticos especificados con FODAS-WS. Al igual que antes, para realizar tal trabajo se requiere considerar el conocimiento almacenado en las distintas capas CIM, PIM y PSM.
- Implementar un validador entre capas para el Framework FODAS-WS, que garantice que los elementos, acciones, asociaciones, estados, precondiciones, efectos, parámetros, operaciones, actividades y demás aspectos que intervienen en las tres capas, sean congruentes y estén definidos correcta y completamente.

- Diseñar e implementar un cosechador de OAs no certificados, que busque en las diferentes federaciones de OAs disponible a nivel mundial, los evalué y los certifique, basado en el modelo de metadatos de OAs establecido en este trabajo
- Diseñar e implementar un agente adaptativo de evaluación de la calidad de los OAs, de acuerdo con las dinámicas de las nubes. En particular, en función de la dinámica de la nube de auto-formación se debería ir cambiando la medida de calidad del OA (aprendizaje), dependiendo de factores como: el tipo de estudiante que estudió el objeto, los resultados de la evaluación realizada a dicho estudiante después de estudiar el objeto, la comparación de lo obtenido con el objeto evaluado respecto del desempeño con OAs similares, entre otros factores.
- Diseñar e implementar un agente de vinculación semántica de temas que contemple una asociación entre los niveles de vinculación semántica y las competencias de aprendizaje requeridas en cada tema.
- Diseñar e implementar un agente de usabilidad, que contemple nuevos indicadores de uso, y que posibilite agregar más conocimiento al perfil de cada usuario, en aspectos como: indicadores psicológicos, intereses culturales de cada estudiante, niveles de conocimiento del estudiante respecto a cada tema, actividades a las que el estudiante dedica su tiempo, objetivos de aprendizaje del estudiante en el mediano y largo plazo, etc. Los tipo de indicadores de uso, así como su generación e instanciación, deben ser enriquecidos, extraídos y reconfigurados en tiempo de ejecución, usando técnicas de minería, tratamiento estadístico, entre otros.
- Diseñar e implementar un agente adaptativo pedagógico, que no solo se comunique con la nube de aprendizaje para determinar herramientas, instrumentos y actividades pedagógicas, sino que usando el conocimiento aportado por el agente adaptativo de usabilidad y el agente adaptativo de evaluación de desempeño, logre determinar nuevas estrategias pedagógicas personalizadas para cada estudiante.
- Incorporar un modulo de aprendizaje y optimización de recomendaciones, que permita automatizar la calibración del nivel de hibridación óptimo, para un usuario o solicitud dada.

- Incorporar en el recomendador una funcionalidad, para armar la secuencia ideal de OAs recomendados para un curso o módulo dado.

Referencias

[1] J. Aguilar, K. Moreno, D. Hernandez, J. Altamiranda, M. Viloria, "Propuesta de un modelo educativo utilizando el paradigma de la nube", *ReVeCom*, Vol. 1, No. 2, pp. 1-11, 2014.

[2] J. Aguilar, J. Altamiranda. "Propuesta filosófica de un nuevo modelo educativo para carreras en el área de ciencias de la computación", *Revista Educare*, Vol. 19, No. 1, pp. 70-94, 2015.

[3] J. Aguilar, J. Fuentes, K. Moreno, R. Dos Santos, O Portilla, J. Altamiranda, D. Hernández, "Computational platform for the educational model based on the cloud paradigm", *Proc. IEEE Frontiers in Education Conference*, pp. 2399-2407, 2015.

[4] K. Moreno, J. Aguilar, J. Altamiranda, "Plataforma web para la gestión de un proceso de auto-Formación basado en ontologías", *Proceedings of the Conferencia Nacional de Computación, Informática y Sistemas (CoNCISa 2014)*, pp. 69-80, 2014 .

[5] J. Fuentes, J. Aguilar, "Management system of learning paradigms using ODA". *Proc. XL Conferencia Latinoamericana en Informática (CLEI 2014),* Montevideo, Uruguay, 2014.

[6] R. Dos Santos, J. Aguilar, "Sistema buscador de contenidos digitales de la nube de conocimiento del Proyecto Madre", *ReVeCom*, Vol. 2, No. 1, pp. 36-47, 2015.

[7] A.Casali, V. Gerling, C. Deco C. Bender, "Sistema inteligente para la recomendación de objetos de aprendizaje". *Revista Generación Digital,* Vol. 9, No. 1, pp. 88-95, 2011.

[8] M. Caro, J. Hernández, J Jiménez. "Diseño de un sistema de recomendación en repositorios de objetos de aprendizaje basado en la percepción del usuario: caso rodas", *Ciencia e Ingeniería Neogranadina*, Vol. 21, No. 1, pp, 51-72, 2011.

[9] A. Ruiz-Iniesta, G. Jiménez-Díaz y M. Gómez-Albarrán. "Personalización en recomendadores basados en contenido y su aplicación a repositorios de Objetos de Aprendizaje". *IEEE-RITA*, Vol. 5, No. 1, pp. 31-38, 2010.

[10] *IEEE. 1484.12.1 Standard for Learning Object Metadata. ANSI/IEEE,* 2002.

[11] J. Solís, M. Chacón-Rivas, C. Garita. "Agente híbrido recomendador de Objetos de Aprendizaje". *Proc. Latin-American Community on Learning Objects LACLO*, 2014.

[12] A. Zapata, V. Menendez, M. Prieto, C.Romero. "A hybrid recommender method for Learning Objects design and evaluation of digital content for education" *International Journal of Computer Applications (IJCA),* Vol. 1, pp. 1-7, 2011.

[13] N. Pukkhem. "Ontology-based semantic approach for Learning Object recommendation". *ACEEE Int. Journal on Information Technology* , Vol. 3, No. 4, pp. 12-21, 2013.

[14] G.Carrillo, X. Ochoa. "Recomendación de objetos de aprendizaje basado en el perfil del usuario y la información de atención contextualizada". *Proc. Latin-American Community on Learning Objects LACLO*, 2013.

[15] S. Cazella, E. Reategui, P. Behar. "Recommendation of Learning Objects applying collaborative filtering and competencies", Proc. *Intl Conf. Key Competencies in the Knowledge Society, 324*, pp.35-43, 2010.

[16] L.Zhou1, S.Helou, L. Moccozet, L.Opprecht, O. Benkacem, C.Salzmann, D. Gillet. "A federated recommender system for online learning environments", *Lecture Notes in Computer Science*, , Vol. 7558, pp 89-98, 2012.

[17] S. Fraihat, Q. Shambour. "A framework of semantic recommender system for e-Learning". *Journal of Software* Vol. 10, No 3, pp 317-330, 2015.

[18] Pahl, C. "Layered ontological modelling for web service-oriented Model-Driven Architecture". *Lecture Notes in Computer Science*, Vol 3748, pp 88 – 102, 2005.

[19] A. Achilleos, G. Kapitsaki, G. Papadopoulos, "A Model-Driven framework for developing web service oriented applications". *Lecture Notes in Computer Science*, Vol 7059, pp 181-195, 2011.

[20] O*penArchitectureWare, http://www.eclipse.org/gmt/oaw*

[21] A. Patil, S. Oundhakar, A. Sheth, K. Verma, "METEOR-S web service annotation framework". *Proc.13th Intl Conf on the World Wide Web*, pp 553-562, 2004.

[22] N. Manouselis, H.Drachsler, K. Verbert, O. Santos (Eds), *Recommender Systems for Technology Enhanced Learning* Springer-Verlag New York 2014

[23] D. Wiley, "Connecting learning objects to instructional design theory: a definition, a metaphor, and a taxonomy". En *The Instructional Use of Learning Objects, (D. Wiley Ed.)* 2000. (http://reusability.org/read/chapters/wiley.doc)

[24] IEEE. *Learning Object Metadata Working Group*. 2001.

[25] R. Mason, M. Weller, C. Pegler, "Learning in the connected economy". *The Open University course team, IET, Open University*, 2003.

[26] E. Morales, F. Garcia "Quality content management for e-Learning: general sigues for a decisión support system". *Proc. 7th Intl Conf. on Enterprise Information Systems*. 2005.

[27] C. Lopez. "Los Repositorios de Objetos de Aprendizaje como soporte a un entorno e-learning." Tesis Doctoral, Universidad de Salamanca, España. 2005.

[28] *Advanced Distributed Learning Initiative. "Emerging and Enabling Technologies for the design of Learning Object Repositories Report"*. 2002.

[29] CANARIE. "White Paper for a Learning Object Repository". Canada, 2001.

[30] D. Porter, J . Curry, B. Muirhead, N. Galan, "A Report on Learning Object Repositories". CANARIE Inc. 2002.

[31] J. Schafer, J. Konstan, J. Riedl, "E-commerce recommendation applications". *Data Mining and Knowledge Discovery*, Vol 5, pp 115-153, 2001.

[32] R. Burke, "Hybrid recommender systems: survey and experiments". *User Modeling and User-Adapted Interaction*, Vol 12, pp 331-370, 2002.

[33] J. Konstan, "Introduction to recommender systems: algorithms and evaluation". *ACM Transactions on Information Systems*, Vol 22, pp 1-4, 2004.

[34] J. Herlocker, J. Konstan, L. Terveen, J. Riedl, "Evaluating collaborative filtering recommender systems". *ACM Transactions on Information Systems*, Vol 22, No 1, pp 5-53, 2004.

[35] M. Deshpande, G. Karypis, "Item-based top-n recommendation algorithms". *ACM Transactions on Information Systems*, Vol 22, No1, pp 143-177, 2004.

[36] L. Hung, "A personalized recommendation system based on product taxonomy for one-to-one marketing online". *Expert Systems with Applications*, Vol 29, pp 383-392, 2005.

[37] J. Schafer, J. Konstan, J. Riedl, "E-Commerce recommendation applications". *Data Mining and Knowledge Discovery*, Vol 5, pp 115-153, 2001.

[38] R. Burke, "Hybrid web recommender systems". *Lecture Notes in Computer Science*, Vol. 4321, pp 377-408, 2007.

[39] M. Ekstrand, J. Riedl, J. Konstan, "Collaborative filtering recommender systems". *Foundations and Trends in Human-Computer Interaction*, Vol 4, No. 2, pp 81-173, 2010.

[40] F. Ricci, L. Rokach, B. Shapira, P. Kantor, *Recommender Systems Handbook*. Springer-Verlag, New York, NY, USA, 2010.

[41] M. Robillard, W. Maalej, R. Walker, T. Zimmermann, *Recommendation Systems in Software Engineering*. Springer, 2014.

[42] N. Good, J. Schafer, J. Konstan, A. Borchers, B. Sarwar, J. Herlocker, J. Riedl, "Combining collaborative filtering with personal agents for better recommendations". *Proc. 16th National Conf on Artificial intelligence* , pp 439 - 446, 1999.

[43] M. Ge, C. Delgado-Battenfeld, D. Jannach, "Beyond accuracy: evaluating recommender systems by coverage and serendipity". *Proc 4th ACM conf. on Recommender systems,* pp. 257–260, 2010.

[44] Y. Yao. "Measuring Retrieval Effectiveness Based on User Preference of Documents·. *Journal of the American Society for Information Science*. Vol 46, No. 2, pp 143 – 155, 1995.

[45] H. Happel, S. Seedorf, "Applications of ontologies in software engineering". *Proc. International Workshop On Semantic Web Enabled Software Engineerin* , pp 1-14, 2006.

[46] A. Anandaraj, G. Dheepak, K. Raja, "Study of ontology in software modelling process and life cycle". *International Journal of Research and Reviews in Software Engineering*, Vol. 1, No. 1, pp 13-17, 2011.

[47] J. Montalvo, "A multimedia ontology-driven architecture for autonomic quality of service management in home networks". *Tesis Doctoral, INSA de Toulouse*, 2012.

[48] P. Tetlow, J. Pan, D. Oberle, E. Wallace, M. Uschold, E. Kendall, "Ontology driven architectures and potential uses of the semantic web in systems and software engineering". *W3C Working Draft*, 2006.

[49] *OMG, The Model-Driven Architecture - Guide Version 1.0.1. OMG Specification*, 2003.

[50] T. Berners-Lee, J. Hendler, O. Lassila, "The Semantic Web". *Scientific American*, May 2001.

[51] R. Gruber. "A Translation approach to portable ontology specifications". *Knowledge Acquisition*, Vol 5, No. 2, pp. 199-220, 1993.

[52] O*WL Web Ontology Language Reference. W3C Recommendation,* 2004. https://www.w3.org/TR/owl-ref/

[53] *Resource Description Framework (RDF): Concepts and Abstract Syntax W3C Working Draft,* 2002 (https://www.w3.org/TR/rdf-concepts/).

[54] D. Tsarkov , I. Horrocks. "FaCT++ description logic reasoner: system description". *Proc. Int. Joint Conf. on Automated Reasoning, pp. 292-297 ,* 2006.

[55] F. Baader, D. Calvanese, D. Guinness, P. Nardi, *The Description Logic Handbook Theory, implementation, and applications*. Cambridge University press, 2010.

[56] *Web Services Architecture W3C Working Group, 2004 (*https://www.w3.org/TR/ws-arch/.

[57] G.Alonso, F. Casati, H. Kuno, V. Machiraju, *Web Services. Concepts, Architectures and Applications* , Springer, 2004.

[58] F. Sánchez, "Sistema basado en tecnologías del conocimiento para entornos de servicios web semánticos". *Revista Iberoamericana de Inteligencia Artificial*, Vol 13, No 43, pp 32-35, 2009.

[59] *Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, W3C Working Draft*, 2003 (https://www.w3.org/TR/wsdl20/).

[60] *SOAP Version 1.2 Part 1 Messaging Framework, W3C Recommendation*, 2003 (https://www.w3.org/TR/soap12-part1/).

[61] P. Alvez, P. Foti y M. Scalone. *Proyecto Batuta - Generador de aplicaciones orquestadoras*. Uruguay, 2006 (https://peruti.files.wordpress.com/2007/09/toolorquesta.pdf).

[62] M. Fuentes, J. Muñoz, F. Álvarez, J. Vanderdonkt, M. Orey, "MIRROS: Intermediary model to recovery learning objects". *Computación y Sistemas*, Vol 13, No 4, pp 129-141, 2010.

[63] P. Castells, "La web semántica". Escuela Politécnica Superior. Madrid, 2003 (http://arantxa.ii.uam.es/~castells/publications/castells-uclm03.pdf).

[64] O. Portilla, J. Aguilar, "Framework basado en ODA para la descripción y composición de Servicios Web Semánticos (FODAS-WS)", *Latin American Journal of Computing*, Vol 2, No 2, pp. 15-24, 2015

[65] *OWL-S: Semantic Markup for Web Services*. 2004.

[66] *Web Service Modeling Ontology (WSMO)*. 2005

[67] O. Portilla, J. Aguilar, C. León, "Semantic recommender system for the recovery of the preserved web heritage," *Proc. Latin American Computing Conference (CLEI 2015)*, pp. 1-11, 2015.

[68] C. Duc, M. Lamolle, O. Cur, "An EXPSPACE tableau-based algorithm for SHOIQ". *Proc. Intl Workshop on Description Logics* , pp. 136-146, 2012.

[69] O. Portilla, J. Aguilar, J. Altamiranda "*Sistema recomendador hibrido calibrado de ontologias biomedicas*". Universidad de los Andes – Merida, Centro de Estudios en Microelectrónica y Sistemas Distribuidos CEMISID, 2014.