Framework Basado en ODA para la Descripción y Composición de Servicios Web Semánticos (FODAS-WS)
ODA Based Framework for Description and Composition of Semantic Web Services (FODAS-WS)

Jose Aguilar y Omar Portilla

***Resumen*— Este artículo presenta el Framework FODAS-WS, basada en la arquitectura ODA (Ontology Driven Architecture). FODAS-WS es un marco de referencia que sirve para diseñar y describir Servicios Web Semánticos (SWS) automáticamente. Con FODAS-WS se busca facilitar las tareas de descubrimiento, composición y ejecución automática de SWS. Para describir un servicio se utilizan tres capas de abstracción: CAPACIM, CAPAPIM y CAPAPSM. La CAPACIM abarca una descripción semántica del dominio y de las funcionalidades del servicio, y es descrita manualmente por el diseñador. Las capas CAPAPIM y CAPAPSM son generadas por procesos de transformación automáticos, basados en las capas que le siguen en mayor nivel de abstracción. CAPAPIM usa como insumo la salida de CAPACIM, y CAPAPSM la salida de CAPAPIM. CAPAPIM tiene un modelo ontológico para generar el esqueleto del código del servicio, en el cual se estipulan los métodos y parámetros que el servicio requiere. CAPAPSM especifica la composición y ejecución automática de servicios.**

***Palabras clave*— Servicios Web Semánticos, ODA Diseño de SWS.**

***Abstract*—This article presents the FODAS-WS Framework, based on the ODA architecture (Ontology Driven Architecture). FODAS-WS is a framework used to design and describe Semantic Web Services (SWS) automatically. FODAS-WS facilitates the tasks discovery, the composition and automatic execution of SWS. To describe a service are used three layers of abstraction: CAPACIM, CAPAPIM and CAPAPSM. The CAPACIM covers a semantic description of the domain and the functionality of the service, and is described by the designer manually. The CAPAPIM and CAPAPSM layers are generated by automated transformation processes based on the layers that follow a greater level of abstraction. CAPAPIM uses as input the CAPACIM output, and CAPAPSM the CAPAPIM output. CAPAPIM has an ontological model to generate the skeleton of the service code, in which are defined the methods and parameters set required by the service. CAPAPSM automatically specifies the composition and execution of services.**

***Index Terms*—Semantic Web Services, ODA, Design SWS.**

I. Introducción

Según [1], la globalización, cooperación y colaboración presentadas en la actualidad han cambiado substancialmente el mundo del software. Actualmente, un usuario humano de forma individual no puede producir conocimiento y ser competente. Pero, en colaboración con otros y con organizaciones, puede constituir una rica fuente de conocimiento y competencias. El conocimiento puede ser usado, almacenado, descubierto e intercambiado por cualquier usuario (humano, dispositivo industrial inteligente, robot, agente software, etc.) en ambientes heterogéneos y amplios.

En la actualidad se cuenta con infraestructuras de computación distribuida adecuadas para la integración de datos. Los servicios web se constituyen como 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. Las ontologías juegan un papel fundamental

Dr. Jose Aguilar ha sido parcialmente financiado por el Proyecto Prometeo del Ministerio de Educación Superior, Ciencia, Tecnología e Innovación de la República de Ecuador).

J. A. está con la Universidad de Los Andes, Facultad de Ingeniería, Escuela de Ingeniería de Sistemas, CEMISID, Mérida, Estado Mérida, Venezuela, Tlf. +58-274-2402880, Fax: +58-274-2402872, E-mail: aguilar@ula.ve. Además, J. A. actualmente es Investigador Prometeo en la Universidad Técnica Particular de Loja, Loja, Ecuador,

Omar Portilla es profesor del programa de Ingeniería de Sistemas de la Universidad de Pamplona, Colombia. Está culminando el Postgrado de Computación, Facultad de Ingeniería, Universidad de Los Andes, Mérida, Estado Mérida.

en la interoperabilidad, ya que son formas estructuradas para describir y formalizar los vocabularios necesarios para compartir conceptualizaciones, con lo que contribuyen a resolver la heterogeneidad semántica, proveyendo una conceptualización compartida en cierto dominio de interés.

Este artículo propone el uso de ontologías organizadas mediante la arquitectura ODA, para el diseño de SWS, y para la realización de descubrimiento, composición y ejecución automática de ellos. En las siguientes secciones se propone un Framework para la descripción de SWS basado en ODA, denominado FODAS-WS, que pretende servir como instrumento para realizar la generación de código, tanto del servicio web diseñado (esqueleto de implementación del SWS), como del cliente del SWS. Esto permite el descubrimiento, la ejecución y la composición automática de SWS.

II. MARCO TEÓRICO

A. Servicios Web Semánticos

De acuerdo a lo planteado en [2]: "Un servicio web es un componente de software modular bien definido que expone una interfaz sobre la red. El uso de los servicios web se hace a través del intercambio de mensajes XML a través del protocolo HTTP".

El objetivo del servicio web es soportar una infraestructura abierta para aplicaciones web. También, deben tener una descripción específica que determine su definición, para qué está hecho el servicio web, y cuáles servicios se pueden invocar.

Los SWS son considerados por [3], como una extensión a los servicios web, que proporcionan ventajas como el descubrimiento automático de servicios, la invocación automática del servicio, y la composición e interoperabilidad del servicio web.

B. Generación automática de código

Según [5], un generador de código automático es una herramienta que genera, a partir de determinados patrones, el código fuente de una aplicación. El uso de estas herramientas reduce el tiempo que se necesita para el desarrollo del software, como también asegura que el grado de errores de programación permanezcan acotados, reduciendo por tanto los tiempos de depuración y puesta a punto. Estos sistemas constan de procedimientos que generan el código fuente, en función de lo expresado en el diseño de la aplicación o modelo de datos.

Dentro del ámbito de la generación automática de código las propuestas más populares son las relativas a la implementación de soluciones a partir de diagramas UML [8]. En este enfoque, se intenta traducir los esquemas de relaciones entre clases que representan a una base de datos, a clases e interfaces implementadas en un lenguaje en particular, tal como lo plantean [6] y [9].

C. Arquitectura MDA y ODA

En [10] y [11] se propone una clasificación para entender cómo se aplican las ontologías en Ingeniería del Software, y cuál es el beneficio en cada caso. Primero, distinguen el rol de las ontologías en el contexto de ingeniería del software entre su uso en tiempo de ejecución y en tiempo de desarrollo. Segundo, miran el tipo de conocimiento que la ontología maneja. Además, 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 [10] plantean la matriz de la Figura 1, en las que se observan cuatro áreas básicas:

- Ontology-driven development (ODD): Inserta el uso de ontologías en el tiempo de desarrollo, para describir el problema del dominio. Por ejemplo, la arquitectura MDA propone ontologías para esta área.
- 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.
- 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.
- 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 SWS, donde las ontologías adicionan una capa semántica de mayor nivel que las descripciones clásicas de servicios web. En esa capa, se adiciona funcionalidades para el descubrimiento automático de servicios web, y emparejamiento y composición automática de servicios.

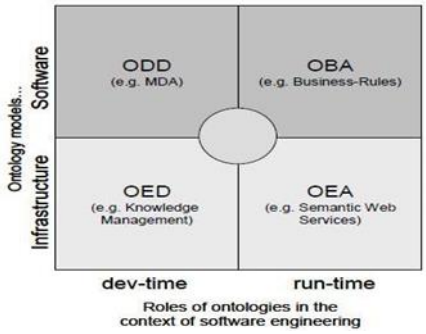

Figura 1. Uso de categorías para ontologías en ingeniería del software propuestas en [10].

Aunque las cuatro áreas parecen distintas en primera instancia, ellas se entrecruzan en algunas aplicaciones.

II.C.1) Ingeniería Manejada por Modelos (Model Driven Engineering)

Según [12], el objetivo de la Ingeniería Manejada por Modelos (MDE) es manejar el desarrollo de software basado en el modelado del dominio. En otras palabras, MDE se focaliza en la especificación de sistemas, más que en su implementación. MDA es un campo de la MDE, que especifica una metodología para el desarrollo de sistemas de software separando la lógica del negocio de la lógica de la plataforma tecnológica subyacente [13]. MDA especifica tres niveles en su arquitectura:

- Modelo Independiente de Computación (Computation Independent Model CIM): describe el contexto en el cual el sistema se usará.
- Modelo Independiente de Plataforma (Platform Independent Model PIM): Describe el sistema en si mismo, sin considerar detalles del uso de plataformas. Un PIM puede ser satisfecho por uno o varias plataformas arquitectónicas reales.
- Modelo Especifico de Plataforma (Representation of Platform Specific Model PSM): En este nivel, se toma en cuenta el ambiente de implementación y los lenguajes.

II.C.2) Ontology-Driven Architecture (ODA)

El grupo de la W3C Semantic and Web Best Practices and Deployment Working Group (SWBPD) propuso a ODA como una extensión a MDA, que usa ontologías para explotar las ventajas de las tecnologías de web semántica.

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. Así, las ontologías son divididas en módulos genéricos, para modelar tanto semántica como metadatos sintácticos.

ODA provee representación de vocabularios de dominios ambiguos (como por ejemplo: requerimientos, restricciones, servicios, propiedades, etc.), chequeo de consistencia de los modelos, así como habilita nuevas capacidades automáticas en ingeniería del software. Varios trabajos han sido desarrollados para ilustrar como ODA puede ser usado en diseño, desarrollo y administración de sistemas distribuidos.

De acuerdo con [13], ODA permite integrar las ventajas de las tecnologías de web semántica en la metodología MDA, en las siguientes áreas:

- Sistemas e Ingeniería del Software, para utilizar la Web Semántica como una herramienta para modelado semántico, durante la especificación y diseño del software a lo largo de su ciclo de vida. También puede ser usada para describir, identificar, descubrir, y almacenar artefactos, y diseñar equipos.
- Especificación de modelos formales, para 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.
- Soporte del ciclo de vida del software. Esta perspectiva propone el uso de ontologías, como proveedor de lenguaje, terminología y estándares, para la especificación del dominio durante el ciclo de vida del software.
- Definición de Contenidos reusables y metadatos, como un poderoso descriptor de servicios y componentes, buscando facilitar el descubrimiento de servicios basados en descripciones precisas.

D.ESTADO DEL ARTE

En [14] 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. El WSF cuenta con ontologías para modelar las tres capas propuestas por MDA. Plantea un proceso de transformación de CIM a PIM, pero no lo implementa en forma automática.

En [15] 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.

En [16] 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.

En [17] se sigue y se adapta la metodología MDD, para su aplicación con el desarrollo de un tipo de software: Servicios web semánticos, basándose en la especificación OWL-S.

En [18] sintetizan los enfoques de desarrollo dirigido por modelos, aplicados al desarrollo de servicios web semánticos bajo dos categorías: los basados en principios y metodologías de Ingeniería de Software, y los basados en la aplicación formal de UML [19] y [20].

III. 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 diseñar, implementar (generación de esqueletos de código del SW), descubrir automáticamente, emparejar y componer automáticamente (generación del código del cliente), SWS. FODAS-WS representa una herramienta para la realización de diseño e implementación de servicios web que usa ODA. Proporciona un elevado nivel de definición semántica, que puede soportar procesos de composición automática reflejados en la generación automática del código de la aplicación cliente.

FODAS-WS busca contribuir en el proceso de descubrimiento, emparejamiento y composición automática de servicios, mediante modelos ontológicos comprensibles a la máquina, almacenados en tres capas de distinto nivel de abstracción, que se llaman CAPACIM, CAPAPIM y CAPAPSM. Estas capas se ajustan completamente lo propuesto en la arquitectura ODA. Cada uno de los modelos ontológicos que constituyen las distintas capas de FODAS-WS, se describen en lenguaje OWL, el cual le aporta no solo sintaxis, sino semántica y métodos formales de análisis, inferencia y razonamiento sobre el conocimiento expresado.

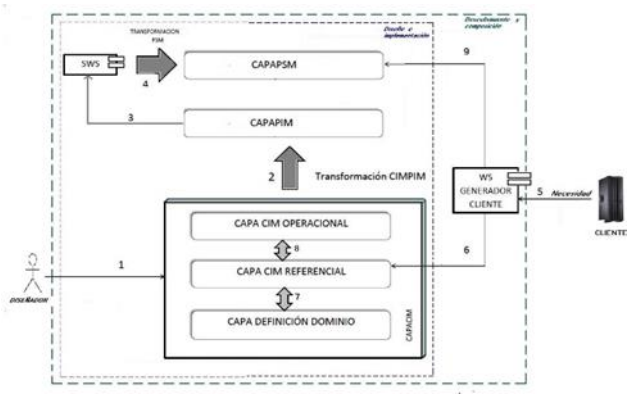

Figura 2. Arquitectura de FODAS-WS

FODAS-WS es implementado en java, como una estructura conceptual y tecnológica, que sirve de base para el diseño, implementación, y descripción de SWS, así como para la generación automática de código del cliente del servicio. La Figura 2 ilustra la arquitectura propuesta para FODAS-WS y sus distintos componentes e interrelaciones.

La arquitectura de FODAS-WS posee siete componentes que interactúan entre sí, los cuales son:

- *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 interacción 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. Esta capa posibilita la generación automática del esqueleto de código del servicio, además de disminuir el trabajo de implementación, garantizando 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 específica con que cuenta FODAS-WS. En ella se estipula lo necesario para realizar la composición y ejecución automática de servicios. Esta capa describe semánticamente lo estipulado en un archivo WSDL, fusionado con lo contemplado en la especificación OWL-S . 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.
- *TRANSFORMACIÓN CIMPIM:* permite realizar la generación automática de la CAPAPIM, a partir de lo descrito semánticamente en la CAPACIM.
- *TRANSFORMACIÓN PSM:* 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 SWS diseñado, implementado y desplegado.
- *WS GENERADOR CLIENTE:* Es un servicio web propuesto para ser implementado en trabajos futuros. Requiere como entradas las necesidades del cliente, expresadas en forma semántica, de acuerdo con las políticas semánticas de FODAS-WS. WS GENERADOR CLIENTE se encarga de realizar descubrimiento automático de los distintos SWS diseñados usando FODAS-WS, que emparejen con las necesidades solicitadas por la aplicación. Una vez descubierto todos los servicios candidatos a ser seleccionados, el WS GENERADOR CLIENTE, mediante la aplicación de mecanismos de emparejamiento, selecciona el servicio que mejor empareja. Posteriormente, genera código automático que implementa al cliente para el SW seleccionado, realizando la ejecución automática del servicio, retornando a la aplicación solicitante los resultados arrojados en el formato que se infirió a partir de las necesidades descritas semánticamente que proporcionó.

A continuación se presenta la secuencia de interacciones que se generan entre los distintos componentes que intervienen en la arquitectura de FODAS-WS, para los aspectos de diseño e implementación necesarios:

1. El diseñador describe semánticamente la CAPACIM, en la cual contempla todo lo necesario para la especificación total del servicio.
2. 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.

A continuación se describen la secuencia 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 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 la CAPA DE DEFINICION 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 la CAPA CIM OPERACIONAL, para realizar el proceso de descubrimiento automático, usando como insumos las inferencias realizadas en el paso 3.

5. WS GENERADOR CLIENTE acude a la CAPA PSM, para obtener la información requerida para realizar composición y ejecución automática, usando como insumos las inferencias realizadas en los pasos 3 y 4. Una vez ejecutado el SWS, el WS GENERADOR CLIENTE retorna a la aplicación cliente, en el formato adecuado de acuerdo a lo inferido de las necesidades solicitadas, los resultados.

IV. DISEÑO, IMPLEMENTACIÓN Y PRUEBAS DE FODAS-WS

El caso de estudio presentado, consiste en el diseño e implementación de un servicio web semántico que básicamente es un sistema recomendador. Según [21] y [22], los sistemas RECOMENDADORES son SOFTWARE que ayudan a emparejar a USUARIOS con PRODUCTOS. Se consideran como agentes software que contienen los intereses individuales y particulares de un consumidor, y hacen recomendaciones de acuerdo a ello. Ellos proveen sugerencias de calidad a un consumidor, en el instante en que requiera buscar y seleccionar algún tipo de producto online. De esta manera, se obtiene un tipo de recomendación personalizada de acuerdo con las características y necesidades de los usuarios. El sistema recomendador del caso de estudio está en capacidad de recomendar dos tipos de productos: Objetos de aprendizaje y documentos web. Los metadatos de esos productos se encuentran almacenados en repositorios semánticos de objetos de aprendizaje y en repositorios semánticos de archivos web respectivamente.

A. DISEÑO E IMPLEMENTACIÓN DE LA CAPA CIM

Como ya se mencionó, la CAPACIM 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; y CAPA CIM REFERENCIAL, que 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.

IV.A.1) CAPA DE DEFINICIÓN DEL DOMINIO

En la CAPA DE DEFINICIÓN DEL DOMINIO, se proponen algunas definiciones:

- DOMINIO: Se conforma de un conjunto de ELEMENTOS que, mediante ASOCIACIONES, se relacionan entre ellos a través de la ejecución de ACCIONES.
- DEFINICIÓN SEMÁNTICA DEL DOMINIO: Es un conjunto de ASOCIACIONES, acompañado de una DEFINICIÓN SEMÁNTICA DE ACCIONES, y una DEFINICIÓN SEMÁNTICA DE ELEMENTOS.
- DEFINICIÓN SEMÁNTICA DE ACCIONES: Consiste en el establecimiento de relaciones semánticas entre las distintas ACCIONES que existen. Se cuentan con 8 ocho tipos de relaciones posibles entre las acciones, tal como se muestra en la Tabla 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 realiza para un escenario especifico.
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 1. Especificación de Acciones.

- DEFINICIÓN SEMÁNTICA DE ELEMENTOS: Se encarga de determinar la jerarquía entre los elementos existentes, y de puntualizar los distintos elementos miembros de otro elemento (ver Tabla 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 2. Definición de elementos.

- ASOCIACIÓN: La asociación es la encargada describir como un ELEMENTO, al ejecutar una ACCIÓN, genera otro elemento. Para describir una ASOCIACIÓN se usan seis relaciones, tal como se explica en la Tabla 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	Determina cual es la acción ejecutada dentro de la asociación definida
tieneConsecuente	esConsecuenteDe	Determina el elemento que se genera al ejecutar la acción

Tabla 3. Descripción de Asociaciones.

Además de definir 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 en cada acción.

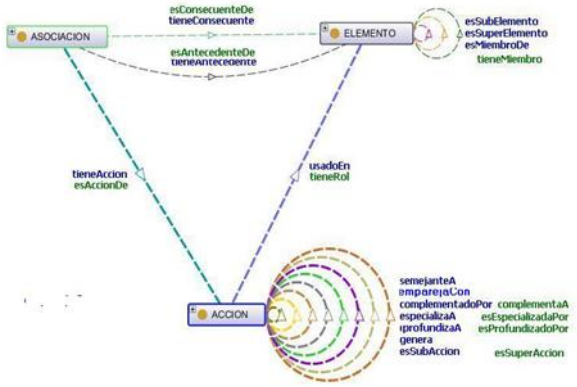

Figura 3. Modelo ontológico de la capa de definición del dominio.

En la Figura 3 se presenta el modelo ontológico usado para realizar la DEFINICIÓN SEMÁNTICA DEL DOMINIO. Como ya se explicó, la definición semántica del dominio consta de una definición semántica de acciones, y una definición semántica de elementos y un conjunto de asociaciones. El modelado de dominio para el caso de uso es mostrado en la Figura 4.

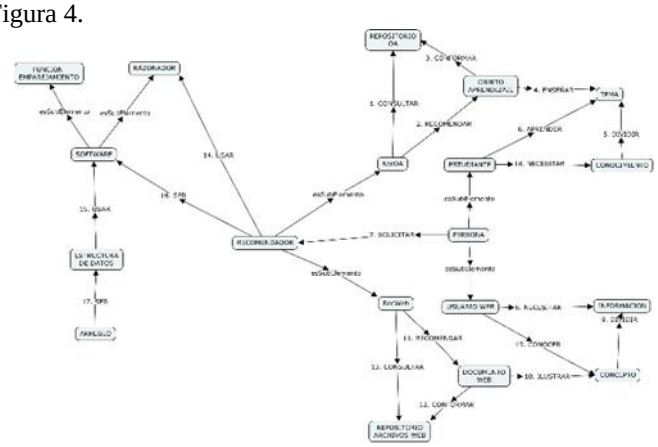

Figura 4. Ilustración de la definición semántica del dominio.

La Figura 4 muestra la representación gráfica de la definición semántica del domino de recomendación. 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 4, representan los elementos definidos en el dominio, cada arco que une un par de óvalos representan una asociación. Cada asociación puede verse como una relación entre dos elementos, a través de una acción identificada con la etiqueta que acompaña a cada arco. Por ejemplo, la asociación1 determina que un elemento RecOA (Recomendador de Objetos de aprendizaje) ejecuta la acción consultar respecto del elemento RepositorioOA (Repositorio de Objetos de Aprendizaje).

La definición semántica de acciones en el caso de uso acá presentado, utiliza las acciones definidas en la Tabla 4, las cuales se registran en la ontología.

Relación	Parejas de acciones relacionadas
semejanteA	aprender- conocer
	Recomendar - sugerir
	Informar - ilustrar
	Requerir - necesitar
	Ordenar - priorizar
emparejaCon	Anteceder - seguir
	Enseñar- aprender
complementaA	Buscar - recomendar
	Consultar - recomendar
profundizaA	Recomendar - buscar
	Aprender - conocer
	Enseñar - ilustrar
genera	Recomendar - calcular
	Recomendar - ordenar
especializaA	ConsultarOA - consultar
	ConsultarWeb - consultar

Tabla 4. Definición de acciones para el dominio de recomendadores.

Por otro lado, ASOCIACIÓN se encarga de determinar cómo dos elementos se vinculan (asocian) mediante una acción. En la Figura 4 se observa la capa de definición semántica de dominio en la que se cuentan con 19 asociaciones, representadas como arcos que unen a los elementos que constituyen óvalos.

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

IV.A.1.a) 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 el emparejamiento con las necesidades arquitectónicas que tenga. 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 automático y de emparejamiento.

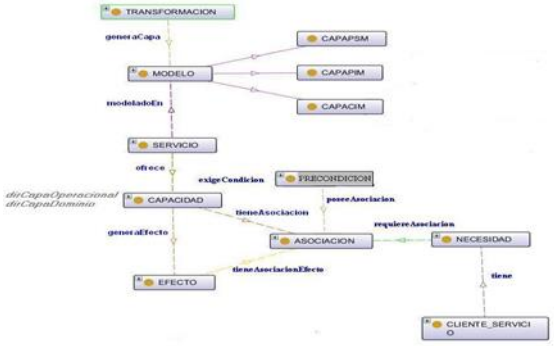

Figura 5. Modelo ontológico de la CAPA CIM REFERENCIAL

En la Figura 5 se muestra el modelo ontológico asociado a la subcapa CIMReferencial, 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. Se describe además que tales capacidades generan efectos, los cuales consisten en asociaciones ya determinadas en la definición semántica del domino.

Lo que busca la capa es que durante el proceso de descubrimiento automático, composición automática y ejecución automática, el sistema de inferencia logre 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 representados en modelos ontológicos.

En la Tabla 5 se ilustra la especificación de capacidades, precondiciones y efectos definidos para el SWS del caso de prueba. También, se muestra la necesidades descritas semánticamente por la aplicación que invoca a WS GENERADOR CLIENTE. Como se aprecia en la Tabla 2, la acción entregar empareja con la acción recibir. Lo cual 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), el SWS está en capacidad de entregar. Como se aprecia, las PRECONDICIONES son satisfechas, pues el recOA necesita como precondicion saber que tema debe enseñar, que es el mismo tema que el estudiante necesita aprender. Como el sistema de inferencias determina que enseñar y aprender emparejan (ver Tabla 5), el WS GENERADOR CLIENTE determina, a través de su módulo de razonamiento, que esta precondición es satisfecha.

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

IV.A.1.b) DISEÑO E IMPLEMENTACIÓN DE LA CAPA CIM OPERACIONAL

En la CAPA CIMOPERACIONAL se especifica el modelado del negocio y sus requerimientos. Es la capa encargada describir, no solo las funcionalidades del proces, y quien lo usa, 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 ejecución.

La Figura 6 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).

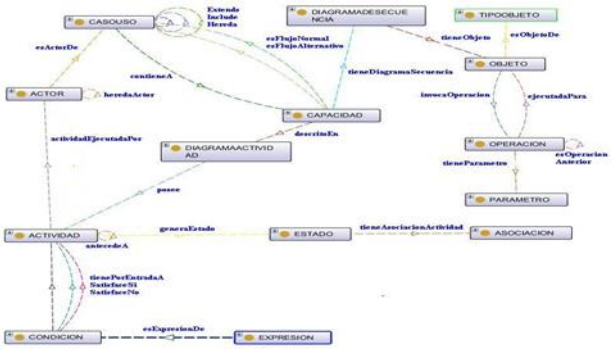

Figura 6. Modelo ontológico de la capa CIM OPERACIONAL

En torno al concepto CASODEUSO, se describen las distintas funcionalidades. El concepto DIAGRAMAACTIVIDAD es el encargado de organizar lo referente a la descripción del flujo de las actividades que componen el proceso, Mediante el concepto DIAGRAMADESECUENCIA se agrupa la descripción de los distintos objetos que interactúan a lo largo de la ejecución del proceso.

IV.A.2) DISEÑO E IMPLEMENTACIÓN DE LA CAPA PIM

En la Figura 7 se aprecia el modelo ontológico asociado a la capa PIM. La estructura fundamental del modelo ontológico de la capa PIM, está basada en lo propuesto en los diagramas de clases. Las reglas de transformación lo que buscan es que el razonador infiera el conocimiento necesario para generar el ABOX.

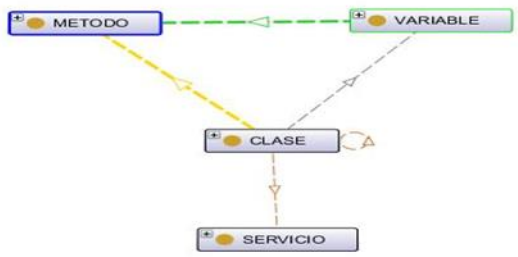

Figura 7. Modelo ontológico de la capa PIM.

El Framework FODAS-WS realiza un proceso de validación. Para ello, FODAS-WS cuenta con un analizador y validador, que se encargan 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 debe estar incorporado en la transformación CIMPIM, de manera que no se pueda generar la capa PIM en caso de que haya alguna inconsistencia.

IV.A.3) DESCRIPCIÓN DE LA GENERACIÓN AUTOMÁTICA DE LA CAPA PIM

La transformación CIMPIM de FODAS-WS realiza una primera generación de la capa PIM, en la que se cuenta con individuos instanciados como clases, como parámetros y como métodos. En el caso específico del individuo llamado RecOA, se determina que las reglas de transformación generaron que la RecOA tiene tres métodos llamados armar, ordenar y emparejarOA (todos vinculados en la capaPIM con el objectProperty tieneMetodo). También, es importante recalcar que en la transformación se genera para la clase RecOA, las variables llamadas razonador, arreglo, ontologia y estudiante (vinculadas en la capaPIM con el objectProperty tieneVariable). De igual forma, para el individuo llamado RecWeb, la transformación generó tres métodos, llamados armar, ordenar y emparejarWeb.

A partir de esta transformación básica, es necesario considerar la posible existencia de jerarquías de clases. La Figura 14 ilustra el resultado obtenido en la capa PIM, después de ejecutada esta regla de transformación en el caso de estudio. Como se aprecia en la Figura 14, la regla de transformación detectó que la RecOA y la RecWeb tienen métodos comunes (armar y ordenar) y variables comunes (ontología, arreglo y razonador), por ello genero una nueva clase, llamada RecOARecWeb, que contiene los métodos y variables comunes. Adicional a eso, indica que clases heredan de ella mediante el objectProperty esSuperClaseDe, con el 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 también se encargan de eliminar de las subclases los métodos y variables, que se hereden de la clase padre. La figura 8 indica, por ejemplo, como en la clase RecOA la regla de transformación retiro, tanto los métodos llamados armar y ordenar, como las variables llamadas ontología, arreglo y razonador, pues ellas son heredadas de la RecOARecWeb.

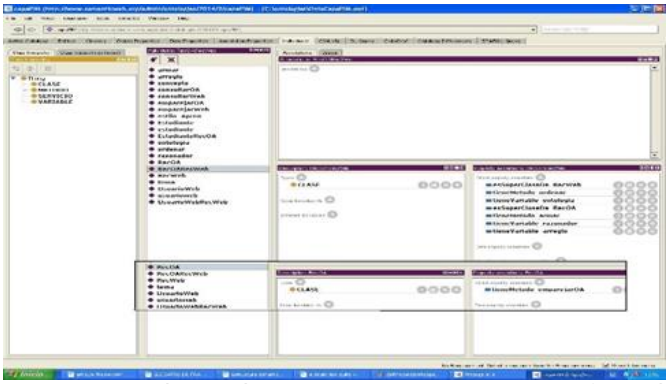

Figura 8. Generación automática de la capa PIM por medio de la transformación CIMPIM.

B. DISEÑO E IMPLEMENTACIÓN DE LA CAPA PSM

Esta capa especifica aspectos análogos a los especificados en el archivo WSDL asociado al Web Service, con la diferencia que en esta capa del FODAS-WS, el razonador puede realizar procedimientos formales de razonamiento, y además, contemplar que este solo 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, y en un formato sin connotación semántica alguna.

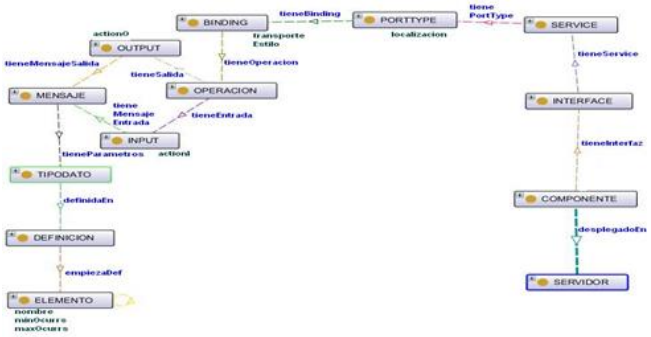

Figura 9. Modelo ontológico de la CAPA PSM

La Figura 9 muestra la capa PSM. Se observa que los componentes (servicios web) cuentan con una o varias interfaces, representadas mediante el concepto INTERFACE. Una vez WS GENERADOR CLIENTE tenga formalizado ésto, 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.

V. ANÁLISIS DE RESULTADOS

La Tabla 6 muestra los resultados obtenidos al realizar una comparación de FODAS-WS con otros Framework existentes, que aunque no son totalmente idénticos en funcionalidad, pertenecen a la misma área de estudio [39], [40] y [41]. Los

resultados obtenidos con FODAS-WS se evalúan mediante una comparación de sus capacidades propuestas, respecto de las capacidades postuladas por esos trabajos similares. A continuación se enumeran los criterios de comparación entre los Framework usados:

1. Se ajusta a la arquitectura ODA
2. Tiene transformaciones automáticas.
3. Propone mecanismos de descubrimiento, composición y ejecución automáticos.
4. Poseé validador de semántica propio.
5. Propone ofrecer funcionalidades completas de composición mediante un servicio WS
6. Propone generar código esqueleto para la implementación del servicio
7. Propone generar código del cliente automático para implementar composición y ejecución automática.
8. Genera PSM a partir del servicio ya implementado, para garantizar total coherencia.
9. Tiene ontologías para representar cada capa
10. Usa estándares para representar las capas ontológicas

Criterio	FODAS-WS	SWF [14]	MDF-DWSOA [15]	METEOR-S [16]
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	SI	NO
7	SI	NO	NO	NO
8	SI	SI	NO	SI
9	SI	SI	NO	SI
10	NO	SI	SI	NO

Tabla 6. Comparativo entre FODAS-WS y otros trabajos

Es importante en este análisis considerar, que 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, composición y ejecución automática de servicios web.

VI. CONCLUSIONES Y TRABAJOS FUTUROS

El uso de FODAS-WS propuesto cuenta con las siguientes ventajas:

- Usando los modelos ontológicos generados en el proceso de diseño, un razonador puede chequear los componentes de un sistema y verificar su consistencia.
- Con el uso de diseños ODA se pueden realizar inventarios de software, en los que además de conocer el software existente, se logre realizar un chequeo de funcionalidades repetidas o inconsistentes.
- 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, y la generación automáticas de las capas superiores.
- Un SWS diseñado con FODAS-WS no requiere de procesos de descripción semántica adicionales al diseño de la aplicación. Esto minimiza la posibilidad de que la aplicación no se ajuste a lo descrito en tales descripciones semánticas, y reduce los tiempos de diseño y especificación, ya que a medida que se está diseñando la aplicación también se está describiendo semánticamente, eliminando con ello un proceso adicional que puede dar pie a errores humanos en la descripción del servicio.
- Con un servicio web semántico diseñado en FODAS-WS, no solo se describe los aspectos básicos contemplados en otros procedimientos de especificación semántica (como OWL-S o WSMO), sino que se permite conocer la totalidad del diseño del servicio, no solo en lo funcional sino también en lo estructural.
- La realización de tareas mantenimiento de los servicios web se hace de manera más clara y congruente, ya que el razonador puede entregar al encargado de mantenimiento información inferida que puede ser útil.
- FODAS-WS resulta ser el framework que propone una funcionalidad más completa para el área de descripción de SWS. Además, propone mecanismos de descubrimiento, composición y ejecución automática de SW, que posibilitan la automatización de procesos de negocio.

A partir del presente trabajo, se propone realizar los siguientes trabajos futuros:

- Diseñar e implementar un módulo que se encargue de generar el código del servicio web diseñado en forma automática.
- Diseñar e implementar un servicio web que ofrezca los servicios de descubrimiento, composición y ejecución automática de SWS diseñados con FODAS-WS, cuyos clientes especifiquen semánticamente sus necesidades.
- Diseñar e implementar un prototipo que use el framework FODAS-WS, para la automatización de procesos de negocio.

REFERENCIAS

[1] P. Pan, C. Wang, G. Horng, and S. Cheng. The development of an Ontology-Based Adaptive Personalized Recommender System. in Electronics and Information Engineering (ICEIE), 2010 International Conference On. 2010.

[2] OBJECT MANAGEMENT GROUP, OMG (2003). MDA Guide Version 1.0.1 [online]. Massachusetts (USA): Object Management Group. http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf

[3] TIMM, Jhon T. E. & GANNOD, Gerald C. A model-driven approach for specifying semantic Web services. In: 3rd. IEEE International Conference

on Web Services, 2005, ICWS 2005, IEEE. Proceedings of the ICWS 2005, Vol. 1, p. 313-320. ISBN: 0-7695-2409-5.
[4] HYUK YANG, Jin & JEONG CHUNG. Automatic Generation of Service Ontology from UML Diagrams for Semantic Web Services. In: First Asian conference on The Semantic Web, ASWC 2006, Beijing (China). p. 523-529. ISBN: 3-540-38329-8 978-3-540-38329-1
[5] Pressman R. Ingeniería de Software. Un enfoque práctico. Ed. McGraw Hill Int. 2001.
[6] Bell, R. Code Generation from Object Models. 1998. Embedded Systems Programming. 74 – 88
[7] Herrington, J. Code Generation in Action. Greenwich: Manning Publications. 2003.
[8] UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado. Craig Larman. Prentice Hall. 2003
[9] Pinter, G., Majzik, I. Program Code Generation Based on UML Statechart Models. 2003. Periodica Polytechnica-Electrical Engineering. 187-204.
[10] Happel, H and Seedorf, S. Applications of Ontologies in Software Engineering. 2008.
[11] Anandaraj, A. ; Dheepak, G. and Raja, K. Study of Ontology in Software Modelling Process and Life Cycle. International Journal of Research and Reviews in Software Engineering Vol. 1, No. 1, March 2011 United Kingdom.[12] Montalvo, J. A multimedia ontology-driven architecture for autonomic quality of service management in home networks. Networking and Internet Architecture. INSA de Toulouse, 2012.
[12] P. Tetlow, J. Pan, D. Oberle, E. Wallace, M. Uschold, and E. Kendall. Ontology Driven Architectures and Potential Uses of the Semantic Web in Systems and Software Engineering. W3C Working Draft, 2006.
[13] Pahl, C. Layered Ontological Modelling for Web Service-oriented Model-Driven Architecture. Dublin City University. 2008.
[14] Achilleos, A ; Kapitsaki, G and Papadopoulos, G. A Model-Driven Framework for Developing Web Service Oriented Applications. Department of Computer Science, University of Cyprus. 2006.
[15] Patil, A; Oundhakar, S ;Sheth, A and Verma, K. METEOR-S: Semantic Web Services and Processes. 2005.
[16] Vega, W. & Umaña H. Diseño de Servicios Web Semánticos utilizando el desarrollo de software dirigido por modelos. En: Ventana Informática No. 30 (ene-jun). Manizales (Colombia): Facultad de Ciencias e Ingeniería, Universidad de Manizales. p. 97-108. ISSN: 0123-9678. 2014.
[17] KALANTARI, Alaeddin; IBRAHIM, Suhaimi & TAHERDOOST, Hamed. A categorization of model-driven approaches for developing Semantic Web Service. ISBN: 978-94-007-2791-5. 2011.
[18] KIM, Il-Woong & LEE, Kyong-Ho. A Model-Driven Approach for Describing Semantic Web Services: From UML to OWL-S. ISSN: 1094-6977. 2009.
[19] TIMM, Jhon T. E. & GANNOD, Gerald C. A model-driven approach for specifying semantic Web services. In: 3rd. IEEE International Conference on Web Services, 2005, ICWS 2005, IEEE. Proceedings of the ICWS 2005, Vol. 1, p. 313-320. ISBN: 0-7695-2409-5.
[20] Burke, R., Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction, 2002. 12(4): p. 331-370.
[21] Bobadilla, J. Recommender systems survey, 2013. España

Jose Aguilar is a System Engineer graduated in 1987 from the Universidad de los Andes, Merida, Venezuela. M. Sc. degree in Computer Sciences in 1991 from the University Paul Sabatier-Toulouse-France. Ph. D degree in Computer Sciences in 1995 from the University Rene Descartes-Paris-France. He completed post-doctorate studies at the Department of Computer Science in the University of Houston, United States, between 1999 and 2000. Titular Professor at the Department of Computer Science in the Universidad de los Andes and researcher at the Microcomputer and Distributed Systems Center (CEMISID) at the same university. Member of the Mérida Science Academy and the International Technical Committee of the IEEE-CIS on Artificial Neural Networks.

Omar Portilla es Ingeniero de Sistemas graduado en 2001 de la Universidad Industrial de Santander, Bucaramanga, Colombia. Candidato a Magister en Computación de la Universidad de los Andes Mérida. Profesor del programa de Ingeniería de Sistemas de la Universidad de Pamplona, Colombia.