Agente Matchmaker para el descubrimiento automático de Servicios Web

Lain Cárdenas, Juan Garzón y José Pérez

Universidad Autónoma de Bucaramanga, Colombia
{lcardenas4, jgarzon, jperez}@unab.edu.co

Resumen

Este artículo presenta un Sistema prototipo basado en la Web Semántica para el descubrimiento automático de Servicios Web. Se describe una arquitectura y un algoritmo del Sistema prototipo que permite realizar el descubrimiento de Servicios Web y que está basado en ontologías de descripciones de conceptos desarrolladas en OWL y en ontologías de descripciones de servicios desarrolladas en OWL-S, los cuales son estándares propuestos por el Consorcio W3C. El sistema prototipo utiliza la API de Protégé OWL como motor de inferencia y sistema razonador de ontologías de conceptos, y la API OWL-S para el manejo de las ontologías de servicios.

Palabras claves: Web Semántica, Servicios Web, Servicios Web Semánticos, Coincidencia Semántica.

Abstract

This paper presents a Web Semantic-based system prototype for Automating Semantic Web Services Discovery. The system prototype proposes and architecture and an algorithm that allow to accomplish Semantic Web Services Automatic Discovering and that is based on concepts description ontologies developed in OWL and services description ontologies developed in OWL-S, which are standards proposed by the W3C Consortium. The system prototype uses Protégé OWL API as an inference engine and a concept ontologies reasoning system, and OWL-S API to handle services ontologies.

1. Introducción

Debido a que en los últimos años los Servicios Web han conseguido una amplia aceptación por parte de las empresas, puede decirse que "los Servicios Web están de moda". Los Servicios Web han permitido la integración e interoperabilidad de sistemas heterogéneos ya que son componentes de aplicaciones que pueden comunicarse con otras aplicaciones sin importar la plataforma en la que corren, aprovechando estándares actualmente disponibles en Internet tales como HTTP y XML. Por consiguiente, las empresas al consumir Servicios Web pueden reducir costos de integración, incrementar la flexibilidad de la empresa para capitalizar las oportunidades de negocio, conectar clientes conocidos, socios y unidades remotas de negocio. Ante esto, surge la necesidad por parte de las empresas de encontrar y consumir Servicios Web que satisfagan sus necesidades.

En un principio, los negocios que querían consumir un Servicio Web específico como parte de sus aplicaciones, tenían que conocer de antemano el proveedor y la ubicación del Servicio Web que ofrece, es decir su URI (Identificador de Recursos Uniforme). Para lograr esto, las empresas tenían que encontrar el proveedor que les ofreciera los Servicios Web de su interés, lo cual les demandaba mucho tiempo y esfuerzo, generando lo que se denomina el problema de descubrir un Servicio Web. Ante este problema, se desarrolló un estándar llamado UDDI (Universal Description, Discovery and Integration). UDDI es un registro que le permite a las compañías describir y registrar sus Servicios Web y también les permite descubrir Servicios Web que se ajusten a sus requerimientos e integrarlos con sus componentes de negocios.

Sin embargo, UDDI tiene limitaciones en su mecanismo de descubrimiento. Su primera limitación es el mecanismo de búsqueda. Las búsquedas en UDDI se limitan a comparar palabras claves y se hacen por categorías de información o esquemas de clasificación[1], pudiendo producir un lote de resultados que pueden no interesar. La segunda deficiencia de UDDI es el uso de XML para describir su modelo de datos, el cual no provee una descripción semántica de su contenido. De tal manera que, para descubrir Servicios Web que coincidan más con las peticiones de los solicitantes, se deben realizar búsquedas de Servicios Web basadas en la descripción semántica de sus capacidades, lo cual superará las limitaciones de UDDI. La descripción semántica de un Servicio Web basado en capacidades permite expresar la funcionalidad del Servicio Web en términos de entradas, precondiciones requeridas, salidas y efectos que produce. Una entrada es lo que un Servicio Web requiere para producir una salida deseada por el solicitante.

[1] Un esquema de clasificación en UDDI es un conjunto de sistemas taxonómicos tales como NAICS (North American Industry Classification System), UNSPSC (The United Nations Standard Products and Services Code), VS Web Service Search Categorization, Geoweb, etc., que se utilizan para mantener clasificados los Servicios Web que los negocios publican para su posterior descubrimiento.

Por otra parte, estándares tales como SOAP, WSDL y UDDI se diseñaron para cumplir los siguientes propósitos: SOAP para proporcionar descripciones de los mecanismos de transporte de mensajes; WSDL para describir la interfaz usada por cada Servicio Web; y UDDI para almacenar información sobre Servicios Web como por ejemplo sus interfaces descritas en WSDL. Pero, ni WSDL ni UDDI permiten el descubrimiento de Servicios Web basado en semántica de capacidades por ser estándares soportados por XML, el cual carece de una semántica explícita, por consiguiente, esto puede hacerse solamente en el nivel semántico.

En el nivel semántico, un Servicio Web puede describirse usando el lenguaje estándar OWL-S que se basa en el lenguaje de marcado semántico OWL y que permite describir semánticamente las capacidades de los Servicios Web, proporcionando a los agentes de Software la capacidad de leer y razonar sobre dichas descripciones semánticas para descubrir automáticamente Servicios Web.

Por lo tanto, la formulación del problema a solucionar es: ¿Cómo descubrir Servicios Web, de un registro de anuncios, en donde la funcionalidad de cada Servicio Web descubierto sea similar o tenga alguna relación semántica con la funcionalidad de un Servicio Web solicitado, mejorando la precisión de los resultados en las búsquedas de sistemas de descubrimiento actuales tales como OWL-S/UDDI Matchmaker [1] y UDDI [8], manteniendo la eficiencia de los tiempos en las búsquedas así como se aplica en [1] el cual mejora los tiempos en las búsquedas aplicados en [2, 4, 5 y 6], y permitiendo la accesibilidad e interoperabilidad a sistemas heterogéneos con el *Mecanismo de Descubrimiento*?.

Entonces, para dar solución a este problema se plantea la siguiente hipótesis: “Con el desarrollo de un Sistema prototipo basado en la Web Semántica, utilizando el lenguaje de ontologías de conceptos OWL y el lenguaje de ontologías de descripción de Servicios Web OWL-S, se dará solución al problema de descubrimiento de Servicios Web en donde la funcionalidad de cada Servicio Web descubierto sea similar o tenga alguna relación semántica con la funcionalidad de un Servicio Web solicitado; y mediante la aplicación de siete niveles de coincidencia entre anuncios y posibles solicitudes se mejorará la precisión de los resultados en las búsquedas respecto a [1 y 8]; además utilizando una técnica de pre-computación se mantendrá la eficiencia de los tiempos en las búsquedas como se realiza en [1]; y por último, con la incorporación de un Servicio Web de Publicación y un Servicio Web de Búsqueda, se permitirá a sistemas heterogéneos el acceso e interoperabilidad con el *Mecanismo de Descubrimiento* del Sistema prototipo”.

Para comprobar la hipótesis planteada, el artículo está organizado de la siguiente manera: en la Sección 2 se muestra un breve marco teórico y una síntesis del estado del arte con respecto al descubrimiento de Servicios Web; en la Sección 3 se describe la arquitectura planteada para el Sistema prototipo denominado agente Matchmaker, junto con el algoritmo de descubrimiento utilizado; en la Sección 4 se realiza un análisis de resultados y el aporte del Sistema prototipo con respecto a otros enfoques; en la Sección 5 se presenta un escenario de aplicación que ilustra el uso del Sistema prototipo; y en la Sección 6 se presentan las conclusiones y se proponen algunos trabajos futuros.

2. Marco teórico y estado del arte

A continuación se presenta un breve marco teórico que reúne los principales conceptos con respecto a los Servicios Web y la Web Semántica junto con una síntesis del estado del arte que describe algunos trabajos relacionados con lo expuesto en este artículo para atacar el problema de descubrimiento de Servicios Web.

2.1. Servicios Web y Web Semántica

Los Servicios Web son aplicaciones que proveen elementos particulares de funcionalidad, tal como la lógica de aplicación, que se pueden acceder desde cualquier sistema usando estándares de Internet tales como XML y HTTP. Los Servicios Web XML proporcionan una solución viable para interoperar entre datos y sistemas. Los Servicios Web XML utilizan mensajería basada en XML como una forma fundamental de comunicación de datos entre sistemas que emplean diferentes modelos de componentes, diferentes sistemas operativos y diferentes lenguajes de programación [11].

Por otro lado, la Web Semántica es una idea creada por Tim Berners-Lee [20] que pretende desarrollar lenguajes que faciliten la inclusión en la Web de contenido legible por las máquinas. Este contenido puede ser representado a través de ontologías. Una ontología es la representación formal y explícita del significado de un dominio. Mediante la utilización de ontologías para los elementos (Páginas Web, Servicios Web, etc.), es posible saber si elementos creados independientemente se refieren o no al mismo contenido semántico. Así, la Web Semántica es una extensión de la Web actual en la cual se le da a la información un significado bien definido [10].

El uso de la Web Semántica permite proveerle a la infraestructura de los Servicios Web la interoperabilidad semántica que necesita. Esta

interoperabilidad semántica le permite a los Servicios Web: (a) representar y razonar acerca de la tarea que un Servicio Web realiza para poder hacer el descubrimiento automático de Servicios Web basado en el anuncio y la descripción de la funcionalidad del servicio, (b) expresar y razonar explícitamente acerca de relaciones y reglas de negocios, (c) representar y razonar acerca del orden de los mensajes intercambiados, (d) entender el significado de los mensajes intercambiados, (e) representar y razonar sobre las precondiciones que se requieren para usar el Servicio Web y los efectos de haber invocado el Servicio Web, y (f) permitir la composición de Servicios Web para lograr obtener un servicio más complejo.

De esta manera, la Web Semántica y los Servicios Web son sinergistas: la Web Semántica transforma la Web en un repositorio de datos legibles por el computador, mientras que los Servicios Web proporcionan las herramientas para el uso automático de esos datos. De la sinergia mencionada anteriormente nace lo que se llama un Servicio Web Semántico.

Los Servicios Web Semánticos son una extensión de los Servicios Web y tienen como objetivo principal extraer significado de los Servicios Web. Los Servicios Web Semánticos se apoyan en especificaciones como OWL-S y OWL, los cuales permiten describir claramente la funcionalidad que brinda un determinado Servicio Web. OWL-S y OWL están basados en la interoperabilidad proporcionada por estándares como SOAP y WSDL de XML. Específicamente, los Servicios Web Semánticos se apoyan en la Web Semántica para describir: (a) el contenido de los mensajes que ellos intercambian, (b) el orden de los mensajes intercambiados y (c) las transiciones de estados que resultan de tales intercambios. El resultado de usar la Web Semántica es una descripción no ambigua de la interfaz del Servicio Web que es entendible por la máquina y proporciona la base para una interoperación libre entre servicios diferentes [4].

2.2. OWL (Ontology Web Language)

El lenguaje de ontologías Web (OWL) está diseñado para ser usado por aplicaciones que requieran procesar el contenido de la información en lugar de sólo presentar información a los humanos. OWL puede ser utilizado para representar explícitamente el significado de las palabras en vocabularios y las relaciones entre esas palabras. Esta representación de palabras y su interrelación es lo que se llama una ontología. OWL tiene más facilidades para expresar el significado y la semántica que XML, RDF, y RDF-S, y así va más allá que estos lenguajes en su habilidad para representar contenido interpretable por la máquina en la Web [9].

2.3 OWL-S (OWL-based Web Service Ontology)

OWL-S (anteriormente DAML-S) suministra a los proveedores de Servicios Web un conjunto de términos de lenguajes de marcas para describir las propiedades y capacidades de sus Servicios Web de forma no ambigua e interpretable por el computador. OWL-S facilita la automatización de tareas de Servicios Web tales como descubrimiento, ejecución, interoperación, composición y monitoreo de ejecución. OWL-S está organizado dentro de tres módulos: (1) un *perfil* que describe las capacidades de un Servicio Web, también como algunas características adicionales que ayudan a describir el servicio, (2) un *modelo de proceso* que describe la actividad del proveedor del Servicio Web, es decir cómo el solicitante puede manejar la información acerca de la invocación del servicio, y (3) un *grounding* que describe cómo el intercambio del resumen de información explicado en el modelo de proceso es mapeado al mensaje que los proveedores y solicitantes intercambian [3]. Por consiguiente, el *perfil del servicio* dice “lo que el servicio hace”, es decir, provee el tipo de información necesitada por un agente que busca un servicio, la cual le permite determinar si el servicio satisface sus necesidades; por otro lado el *modelo del servicio* dice “cómo trabaja el servicio”, es decir, describe qué pasa cuando el servicio es ejecutado; y por último el *grounding* del servicio, el cual especifica los detalles de cómo un agente puede acceder a un servicio [7].

2.4 OWL-S/UDDI Matchmaker

OWL-S/UDDI Matchmaker presentado en [1] es una combinación de OWL-S y UDDI que saca provecho de la proliferación de UDDI en la infraestructura de los Servicios Web y de la adopción de ontologías basadas en OWL-S y OWL las cuales pueden ser usadas para crear un perfil que permita describir las capacidades de los Servicios Web.

Para lograr el OWL-S/UDDI Matchmaker, se necesitó: (a) extender el registro de UDDI para almacenar las descripciones del perfil de OWL-S, específicamente la descripción de capacidades, (b) extender la API de UDDI para agregar la funcionalidad de búsqueda de capacidades, y (c) extender la API de UDDI con la funcionalidad del mapeo OWL-S/UDDI a fin de que el perfil OWL-S

pueda ser convertido dentro de los anuncios UDDI y publicado usando el mismo API.

UDDI consta de dos puertos, el puerto de publicación y el puerto de búsqueda. Al recibir un anuncio a través del puerto de publicación, el componente UDDI del OWL-S/UDDI Matchmaker, lo procesa como cualquier otro anuncio UDDI. Si el anuncio contiene información del perfil de OWL-S, éste envía el anuncio al componente Matchmaker el cual clasifica el anuncio basado en la información semántica presente en el anuncio. Por otro lado, un cliente puede usar la funcionalidad de buscar a través del puerto de búsqueda, sin embargo estas búsquedas no usan la información semántica ni la descripción de capacidades proporcionada por la información del perfil de OWL-S. Por lo tanto se extiende el registro UDDI añadiendo un puerto de capacidad para solucionar el problema anterior. Usando el puerto de capacidad podemos buscar Servicios Web basados en la descripción de sus capacidades que están representadas por las entradas, salidas, precondiciones y efectos. Las búsquedas recibidas a través del puerto de capacidad son procesadas por el componente Matchmaker, por lo tanto las consultas son semánticamente coincidentes basadas en la información del perfil de OWL-S. La respuesta de la búsqueda contiene una lista de servicios de negocios claves que coinciden con la búsqueda del cliente.

Cuando se envía una petición, el algoritmo de coincidencia usado por el componente Matchmaker encuentra un servicio apropiado haciendo coincidir las salidas de la petición contra las salidas de los anuncios publicados, y luego, si cualquier anuncio coincide después de la fase de salida, las entradas de la petición se hacen coincidir contra las entradas de los anuncios que coincidieron durante la fase de salida. En este algoritmo, el grado de equivalencia entre dos salidas y dos entradas depende de la equivalencia entre los conceptos que las representan dentro de su ontología. Los grados de equivalencia planteados aquí son cuatro y establecen los niveles de flexibilidad que soporta el mecanismo de coincidencia del componente Matchmaker. Estos grados de equivalencia se deducen a partir de la relación entre el concepto *P* (para suponer que es de una petición) y el concepto *A* (para suponer que es de un anuncio) y son los siguientes: *Exact*, cuando *P* es igual que *A* o cuando *P* es subclase directa de *A*; *Plugin*, cuando *A* incluye a *P* en su jerarquía y *P* no es subclase directa de *A*; *Subsume*, cuando *P* incluye a *A* en su jerarquía y *A* no es subclase directa de *P*; y por último *Fail*, cuando no hay una relación de inclusión entre *A* y *P*.

2.5. DAML-S/UDDI Matchmaker y la Máquina Virtual DAML-S

Similar al método presentado en [1], la implementación de DAML-S/UDDI Matchmaker presentado en [4] extiende a UDDI proveyendo coincidencia semántica de capacidades. Adicionalmente este método presenta una Máquina Virtual DAML-S que usa el *Modelo de Procesos* de DAML-S para administrar la interacción con el Servicio Web.

La primera tarea de un Servicio Web es publicar sus capacidades con un registro, en este caso el DAML-S/UDDI Matchmaker. El registro de capacidades permite a los Servicios Web ser descubiertos y actuar como un proveedor. Cuando un Servicio Web necesita contactar otro Servicio Web con un conjunto específico de capacidades, compila el perfil del Servicio Web ideal que le gustaría contactar y lo envía como una petición al Matchmaker. La tarea del Matchmaker es seleccionar el proveedor que declaró un conjunto de capacidades que más coincidió con las capacidades esperadas del solicitante. Finalmente, el solicitante conoce sobre el proveedor, y puede iniciar la interacción. La interacción se regula por las especificaciones en el *Modelo de Procesos* y *Grounding*, el cual define la interfaz del Servicio Web del proveedor.

El método presentado aquí permite que un solicitante utilice el *Modelo de Procesos* del proveedor para interactuar con el Servicio Web. Esta interacción se hace a través de la Máquina Virtual de DAML-S. La arquitectura de la Máquina Virtual de DAML-S se basa en tres componentes principales: (a) el invocador del Servicio Web, (b) el procesador de DAML-S, el cual es responsable de manejar la interacción con el proveedor por medio del intercambio de mensajes, y (c) el motor de inferencia de DAML, el cual es responsable de interpretar mensajes recibidos, cargar ontologías adicionales que puedan ayudar al Servicio Web en su interacción, y derivar las consecuencias de la información que él carga.

2.6. Agente Broker

En contraste con el agente Matchmaker, el agente Broker descrito en [3] actúa de un modo un tanto distinto. El agente Broker hace uso de dos protocolos, el de anuncio y el de mediación. En el protocolo de anuncio, el agente Broker primero recolecta una lista de anuncios de Servicios Web que luego usa para seleccionar el mejor proveedor. El protocolo de mediación requiere los siguientes pasos: (a) el solicitante pregunta al agente Broker y espera por una respuesta mientras el agente Broker usa su mecanismo de descubrimiento

basado en capacidades para localizar un proveedor apropiado; (b) una vez que descubra el proveedor, el agente Broker reformula la pregunta para ése Servicio Web específico; (c) sobre la recepción de la pregunta, el proveedor transmite la respuesta al agente Broker; y (d) el agente Broker responde al solicitante. De esta manera, el agente Broker queda implicado a través de la interacción proveedor y solicitante.

2.7. Un Software Framework para Matchmaking

Según lo planteado en [5], se investiga cómo las tecnologías de Servicios Web y Semántica pueden ser utilizadas para soportar publicación y descubrimiento de Servicios Web en e-commerce. En particular, se describe el diseño e implementación de un prototipo de coincidencia de servicios que usa ontologías basadas en DAML-S y un razonador DL (Description Logics) para comparar descripciones de Servicios Web basadas en ontologías. Aquí utilizan Racer [19] como razonador para computar las coincidencias semánticas entre anuncios de Servicios Web y peticiones de Servicios Web. A diferencia de Paolucci en [2], extienden los niveles de coincidencia a: *Exact*, si el anuncio A y la petición P son conceptos equivalentes; *PlugIn*, si la petición P es subconcepto del anuncio A; *Subsume*, si la petición P es súper concepto del anuncio A; *Intersection*, si la intersección del anuncio A y la petición P es satisfacible, es decir, que el anuncio no es incompatible con la petición; y *Disjoint*, significa que ningún ítem puede satisfacer el anuncio, es decir, la petición es considerada como una coincidencia fallida.

2.8. Automatización para el Descubrimiento de Servicios Web

En [6] se enfocan en el asunto del descubrimiento dinámico de Servicios Web basado en sus capacidades. Su objetivo es hacer un proceso de Matchmaking entre una solicitud y una descripción de algún Servicio Web disponible. Se formaliza el enfoque de descubrimiento de Servicios Web en el contexto de la lógica de descripciones (DL). El trabajo apunta a mejorar el potencial de los Servicios Web centrándose en formalismos y aspectos flexibles de su descubrimiento, específicamente, la contribución más importante es: proponen una técnica que va más allá de comparaciones simples de subsunción entre una petición de un Servicio Web y anuncios de Servicios Web. Para cumplir con este requerimiento se propone usar una "operación de diferencia" sobre las descripciones de los Servicios Web. Tal operación permite extraer, de un subconjunto de descripciones de Servicios Web, la parte que es semánticamente común con una petición de Servicio Web dada y la parte que es semánticamente diferente de la petición. Se propone un algoritmo de coincidencia que toma como entrada una petición de un Servicio Web Q y una ontología T del Servicio Web, y encuentra un conjunto de Servicios Web llamado la "mejor cobertura" de Q, cuyas descripciones contienen tanta información común con Q como sea posible y tan poca información extra con respecto a Q como sea posible. Toda esta caracterización de la automatización del descubrimiento de Servicios Web se basa en DAML-S.

2.9. Coincidencia de Especificaciones de Componentes de Software

Zaremsky y Wing en [21], presentan la coincidencia de especificaciones como una manera de comparar dos componentes de software. Entiéndase por coincidencia, que "satisface", "conoce", "es equivalente a", o "que interactúa adecuadamente con". La Coincidencia de especificaciones, en el contexto de reutilización de software y recuperación de librerías, puede ayudar a determinar si un componente puede ser sustituido por otro o cómo un componente puede ser modificado para ajustarse a los requerimientos de otro. En el contexto de la programación orientada a objetos, puede ayudar a determinar cuándo un tipo es un subtipo que se comporta de manera similar a otro. En el contexto de la interoperabilidad de sistemas, puede ayudar a determinar si las interfaces de dos componentes no coinciden. Esta propuesta usa especificaciones formales, que describen el comportamiento de los componentes de software, para determinar si dos componentes coinciden. Los componentes de software en los cuales se centra su interés son las funciones (ej., rutinas en C) y los módulos (conjunto de funciones), que pueden estar almacenados en una librería de programa, en un directorio compartido de archivos, o en un repositorio de software. Básicamente plantean el uso de siete grados de coincidencia que pertenecen a dos grandes grupos, Coincidencia *Pre/Post* y Coincidencia de *Predicados*. Al primer grupo pertenecen los grados coincidencia *Pre/Post Exact*, *Plug-in*, *Plug-in Post* y *Weak Post*. Al segundo grupo pertenecen los grados de coincidencia de predicados *Exact*, *Generalized* y *Specialized*. Las formalizaciones de Zaremsky y Wing constituyen la base fundamental de las especificaciones de los niveles de equivalencia propuestos para los algoritmos de coincidencia actuales, como se plantea por ejemplo en [1] y [5].

3. Sistema Prototipo

El Sistema prototipo, denominado *Agente Matchmaker*, es el encargado de: (a) registrar las descripciones de los Servicios Web publicados por los proveedores o anunciantes, y (b) descubrir del registro de anuncios aquellos Servicios Web que coincidan con la descripción del Servicio Web requerido por parte del cliente o solicitante.

Para lograr este propósito se plantea una arquitectura denominada Matchmaker OWL-S y un algoritmo de descubrimiento basado en coincidencia semántica. A continuación se presenta la arquitectura planteada junto con el algoritmo de coincidencia y posteriormente se presentan casos de prueba para demostrar y validar la funcionalidad del prototipo.

3.1. Arquitectura Matchmaker OWL-S

Está arquitectura está compuesta de dos módulos que son: (a) una Interfaz de Comunicación, y (b) un Mecanismo de Descubrimiento. Enseguida describimos en detalle cada uno de estos módulos junto con el algoritmo de descubrimiento.

3.1.1. Interfaz de Comunicación

Este módulo le permite al proveedor del Servicio Web publicar la descripción del Servicio Web y al solicitante buscar los Servicios Web de su interés. Este módulo de Interfaz de Comunicación está conformado por páginas Web y Servicios Web. Las páginas Web sirven para que cualquier usuario (solicitante o anunciante) pueda, desde un navegador Web, buscar o publicar Servicios Web. Por otro lado, los Servicios Web (de búsqueda y de publicación) sirven para que cualquier aplicación de usuario pueda consumirlos y así automatizar el proceso de búsqueda o publicación de Servicios Web. Ver Figura 1 y Figura 2 para una mejor ilustración.

3.1.2. Mecanismo de descubrimiento

Este módulo está conformado por tres elementos básicos: el Mecanismo de Coincidencia, el Mecanismo de Publicación y un Registro de Anuncios. El *Mecanismo de Descubrimiento* plantea la realización de dos etapas, la etapa de publicación y la etapa de búsqueda. En la etapa de publicación se hace uso del *Mecanismo de Publicación* y en la etapa de búsqueda se hace uso del *Mecanismo de Coincidencia*, ambos mecanismos hacen uso del Registro de Anuncios. Ver Figura 1 y Figura 2 para una mejor ilustración.

Los objetivos del *Mecanismo de Descubrimiento* son: (a) pre-computar los grados de equivalencia entre el anuncio y posibles solicitudes guardando posteriormente en el *Registro de Anuncios* los datos relacionados con la descripción del Servicio Web publicado junto con sus grados de equivalencia pre-computados, esto es realizado por el *Mecanismo de Publicación*; y (b) descubrir del *Registro de Anuncios* aquellos Servicios Web que coincidan con la descripción del Servicio Web solicitado, esto es realizado por el *Mecanismo de Coincidencia*.

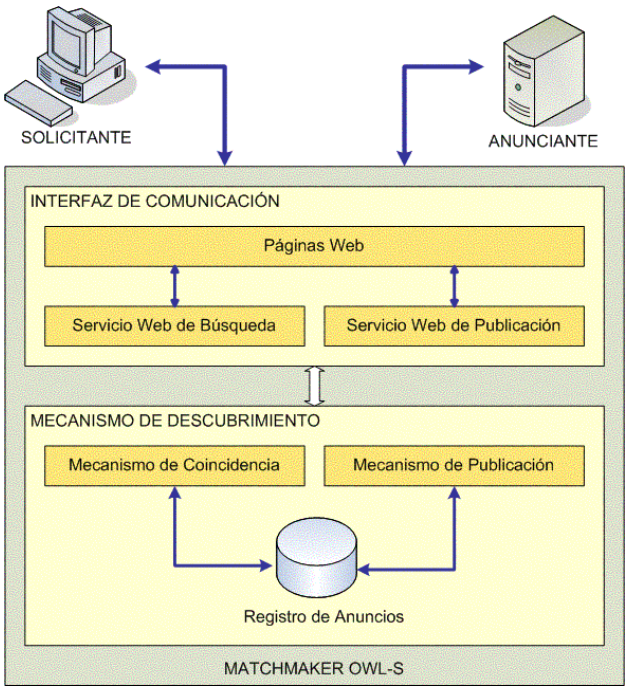

Figura 1. Arquitectura Matchmaker OWL-S

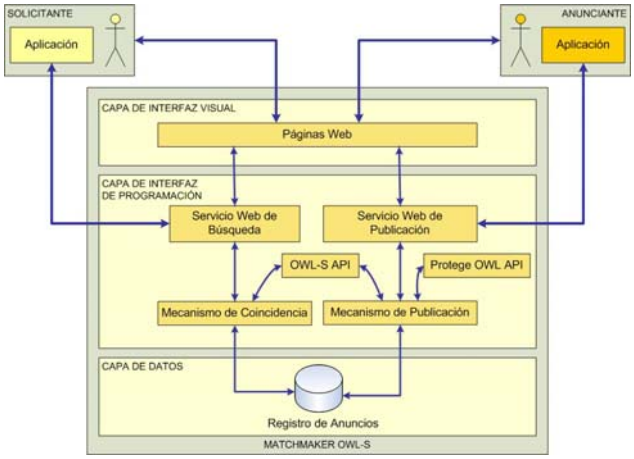

Figura 2. Arquitectura Matchmaker OWL-S basada en capas

Para que el *Mecanismo de Descubrimiento* pueda lograr estos objetivos, nuestra arquitectura hace uso de OWL-S, el cual es una ontología superior basada en OWL que sirve para describir semánticamente Servicios Web. El perfil, que es parte de OWL-S, describe básicamente las capacidades del Servicio Web (entradas, salidas, precondiciones y efectos) que son utilizadas para el descubrimiento de Servicios Web. Por lo tanto, el Sistema prototipo desarrollado aquí interactúa con ontologías de conceptos desarrolladas en OWL y con descripciones de Servicios Web desarrolladas en OWL-S, haciendo uso solamente de las capacidades de entradas y salidas descritas en el perfil de OWL-S del Servicio Web.

Así, en la etapa de publicación el *Agente Matchmaker* realiza la pre-computación valiéndose del perfil del Servicio Web que será publicado y de las ontologías OWL relacionadas con el perfil.

Por otra parte, en la etapa de búsqueda, el *Agente Matchmaker* encuentra anuncios (Servicios Web publicados) que sean equivalentes con la solicitud (Servicio Web solicitado) valiéndose para este caso del perfil de la solicitud. Según como se definió en [2], un anuncio equivale a una solicitud cuando el anuncio describe un Servicio Web que es suficientemente similar a la descripción del Servicio Web solicitado. Por consiguiente, es necesario realizar equivalencias flexibles que reconozcan el grado de similitud entre un anuncio y una solicitud para no restringir a que un anuncio y una solicitud describan exactamente el mismo Servicio Web.

A continuación se argumentan los grados de equivalencia utilizados en el *Mecanismo de Descubrimiento*; posteriormente se describen las etapas de publicación y de búsqueda.

3.1.2.1. Grados de Equivalencia

Los grados de equivalencia que se proponen son siete a diferencia de lo planteado en [1], lo cual hará que la búsqueda sea más flexible y más precisa en el *Proceso de Descubrimiento*. Para etiquetar cada grado de equivalencia se usa la nomenclatura "*A*" (de *Anuncio*), una relación de jerarquía y "*S*" (de *Solicitud*). Los pesos asignados a cada grado de equivalencia varían de 0 a 6 en orden de menor a mayor importancia o nivel de coincidencia, así, el peso 0 corresponde a un nivel de coincidencia fallida y el peso 6 a un nivel de coincidencia exacta.

El grado de equivalencia entre un anuncio y una solicitud depende del nivel de coincidencia entre todas las salidas de la solicitud con las salidas del anuncio, y todas las entradas del anuncio con las entradas de la solicitud. Este nivel de coincidencia entre dos salidas o dos entradas no es sintáctico, sino que está basado en la relación entre los conceptos asociados con esas entradas y salidas en su ontología OWL, es decir se realiza coincidencia semántica.

Por lo tanto, para describir los grados de equivalencia entre dos conceptos se asume que *Salida_A* representa el concepto de una salida de un anuncio y *Salida_S* el de una solicitud; de la misma forma se asume que *Entrada_A* representa el concepto de una entrada de un anuncio y *Entrada_S* el de una solicitud.

A continuación se describe la ocurrencia de cada uno de los grados de equivalencia propuestos, en orden de mayor a menor importancia, en función a las salidas. Estos mismos grados de equivalencia se aplican a las entradas:

- ***A_igual_S*** (peso 6). Este grado ocurre cuando *Salida_A* y *Salida_S* son las mismas y se lee *A* (*Salida_A*) es igual a *S* (*Salida_S*).

- ***A_padre_S*** (peso 5). Este grado ocurre cuando *Salida_A* incluye a *Salida_S*, es decir, *Salida_S* es una subclase directa de *Salida_A* y se lee *A* (*Salida_A*) es padre de *S* (*Salida_S*).

- ***A_hijo_S*** (peso 4). Este grado ocurre cuando *Salida_S* incluye a *Salida_A*, es decir, *Salida_A* es una subclase directa de *Salida_S* y se lee *A* (*Salida_A*) es hijo de *S* (*Salida_S*).

- ***A_abuelo_S*** (peso 3). Este grado ocurre cuando *Salida_A* incluye a *Salida_S*, pero se diferencia del grado *A_padre_S* porque aquí *Salida_S* es una subclase de segundo nivel de *Salida_A* y se lee *A* (*Salida_A*) es abuelo de *S* (*Salida_S*).

- ***A_nieto_S*** (peso 2). Este grado ocurre cuando *Salida_S* incluye a *Salida_A*, pero se diferencia del grado *A_hijo_S* porque aquí *Salida_A* es una subclase de segundo nivel de *Salida_S* y se lee *A* (*Salida_A*) es nieto de *S* (*Salida_S*).

- ***A_hermano_S*** (peso 1). Este grado ocurre cuando *Salida_A* y *Salida_S* son subclases directas del mismo padre y se lee *A* (*Salida_A*) es hermano de *S* (*Salida_S*).

- ***A_noesfamiliar_S*** (peso 0). Este grado ocurre cuando *Salida_A* y *Salida_S* no encuentran ninguna de las relaciones anteriores y se lee *A* (*Salida_A*) no es familiar de *S* (*Salida_S*).

En los grados descritos anteriormente se asume que *A_padre_S* es más importante que *A_hijo_S* porque cuando el concepto de la salida de un Servicio Web anunciado (*Salida_A*) es más

genérico que el de un Servicio Web solicitado (*Salida_S*), el anunciante ofrece un concepto menos restrictivo que el demandado por el solicitante, ya que en este caso puede suceder que también ofrezca lo que requiere el solicitante. En caso contrario, si un anunciante ofrece algo más restrictivo que lo que requiere el solicitante, puede que a éste no le interese. El caso del orden de importancia entre *A_abuelo_S* y *A_nieto_S* es análogo, sólo que considerando una mayor distancia en el árbol de la taxonomía de conceptos especificados en la ontología.

3.1.2.2. Etapa de Publicación

Esta etapa se realiza cuando un proveedor quiere publicar algún Servicio Web. La publicación debe incluir: el URI de la localización de acceso al Servicio Web, el URI de la descripción del Servicio Web en OWL-S y las URI de sus ontologías en OWL correspondientes a las entradas y salidas de las capacidades del Servicio Web definidas en el perfil de OWL-S. Para llevar a cabo esta tarea, el proveedor o anunciante del Servicio Web puede utilizar la página Web de publicación a través de un navegador Web o consumir el Servicio Web de publicación dentro de su aplicación. La página Web de publicación hace uso del Servicio Web de publicación el cual a su vez se encarga de llamar al *Mecanismo de Publicación* (API de Publicación), ver Figura 2.

El *Mecanismo de Publicación* es el encargado de pre-computar los grados de equivalencia entre el anuncio y posibles solicitudes y guardar en el *Registro de Anuncios* los datos relacionados con la descripción del Servicio Web junto con sus grados de equivalencia pre-computados. La razón de aplicar una pre-computación en el proceso de descubrimiento se basa en el enfoque planteado en [1] donde se argumenta la importancia de éste método, la cual radica en que gastar tiempo cargando ontologías (realizando inferencia) para encontrar los grados de equivalencia en la etapa de publicación, en lugar de aplicarlo en la etapa de búsqueda, no es una cuestión critica y además disminuirá notablemente los tiempos en las búsquedas ya que el *Agente Matchmaker* puede extraer el valor correcto mediante una búsqueda exacta sin necesidad de hacer inferencia.

A continuación se describen los pasos que realiza el algoritmo utilizado por el *Mecanismo de Publicación*:

- Al momento de publicar un Servicio Web, el *Mecanismo de Publicación* carga en estructuras de datos en memoria, tanto la descripción OWL-S del Servicio Web, para obtener las entradas y salidas, como las ontologías en OWL que corresponden a las entradas y salidas de la descripción del Servicio Web. Para poder realizar esta tarea, el *Mecanismo de Publicación* se apoya en el uso de APIs tales como: OWL-S API [16] para el manejo de descripciones de Servicios Web desarrolladas en OWL-S, y en Protégé OWL API [17] que incorpora a Jena [18] como razonador para realizar la tarea de inferencia sobre las ontologías de descripciones de conceptos desarrolladas en OWL. Entonces, para explicar mejor el algoritmo, se toma como ejemplo la descripción de un Servicio Web (A1) desarrollado en OWL-S que recibirá como entrada un país y devolverá como salida información económica del país. Por otro lado, las ontologías relacionadas con las salidas y entradas de la descripción del Servicio Web (A1) se ilustran en la Figura 3 y en la Figura 4.

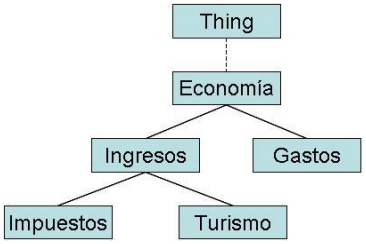

Figura 3. Ontología de las salidas del anuncio

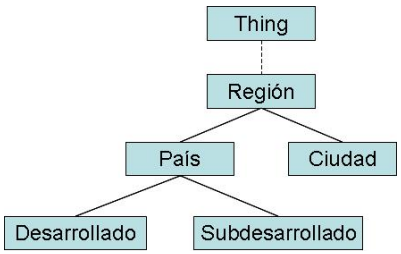

Figura 4. Ontología de las entradas del anuncio

- Luego de cargar las ontologías de conceptos OWL en estructuras de datos en memoria, estas estructuras se utilizan para deducir los grados de equivalencias que existen entre tales conceptos los cuales corresponden a las entradas y a las salidas del Servicio Web.

- Cada grado de equivalencia deducido entre dos conceptos se almacena en otra estructura de datos en memoria denominada *Lista de Equivalencias*. La estructura de esta lista tiene tres atributos y se representa de la siguiente forma: [<Con_A, Con_S, Grado>], donde Con_A y Con_S representan conceptos de la ontología y Grado representa la equivalencia entre ellos. Los conceptos se etiquetan como

Con_A para suponer que corresponde a un anuncio y como Con_S para suponer que corresponde a una solicitud. Esta lista de equivalencias se debe generar para las ontologías de conceptos que corresponden a las salidas y a las entradas del anuncio. Por ejemplo, si cargamos en una estructura de datos en memoria la ontología que corresponde a las salidas mostrada en la Figura 3 y deducimos de esta estructura los grados de equivalencia entre los conceptos, se generará la lista de equivalencias que se muestra en la Tabla 1.

- Una vez generada la lista mostrada en la Tabla 1, se filtra por los elementos de la lista cuyos valores de la columna Con_A coincidan con los valores de las salidas del anuncio. Este mismo proceso se realiza con las entradas del anuncio. Por ejemplo, tomando los elementos de la Tabla 1 y filtrándolos con las salidas del anuncio (A1), la Tabla 1 quedará filtrada con los elementos mostrados en la Tabla 2.

- Por último, de la lista filtrada mostrada en la Tabla 2, por cada valor coincidente en la columna Con_S se suman los pesos asignados a su grado de equivalencia y se almacenan en una estructura de datos igual a la que se muestra en la Tabla 3, en donde el campo Grado indica el grado de equivalencia que existe entre el concepto del campo Con_S y el anuncio. Este mismo proceso se realiza con las entradas del anuncio. Luego los datos de la Tabla 3 y los datos relacionados con el anuncio se harán persistentes en el *Registro de Anuncios* para que sean utilizados en la etapa de búsqueda.

3.1.2.3. Etapa de Búsqueda

Esta etapa se realiza cuando un cliente quiere buscar algún Servicio Web de su interés. La búsqueda debe incluir el URI de la descripción OWL-S del Servicio Web solicitado. Para llevar a cabo esta tarea, el cliente o solicitante del Servicio Web puede utilizar la página Web de búsqueda a través de un navegador Web o consumir el Servicio Web de búsqueda dentro de su aplicación. La página Web de búsqueda hace uso del Servicio Web de búsqueda el cual a su vez se encarga de llamar al *Mecanismo de Coincidencia* (API de Coincidencia), ver Figura 2.

El *Mecanismo de Coincidencia* es el encargado de descubrir en el *Registro de Anuncios* los anuncios que sean equivalentes con la solicitud. La equivalencia entre un anuncio y una solicitud se da cuando todas las salidas de la solicitud tienen algún nivel de coincidencia con las salidas del anuncio, y todas las entradas del anuncio tienen algún nivel de coincidencia con las entradas de la solicitud. Esto garantiza que los servicios encontrados satisfacen la necesidad del solicitante y que el solicitante le provee a los servicios encontrados todas las entradas que éste necesita para operar correctamente.

#	Con_A	Con_S	Grado (peso)
1	Economía	Economía	A_igual_S (6)
2	Economía	Ingresos	A_padre_S (5)
3	Economía	Gastos	A_padre_S (5)
4	Ingresos	Economía	A_hijo_S (4)
5	Gastos	Economía	A_hijo_S (4)
6	Economía	Impuestos	A_abuelo_S (3)
7	Impuestos	Economía	A_nieto_S (2)
8	Economía	Turismo	A_abuelo_S (3)
9	Turismo	Economía	A_nieto_S (2)
10	Ingresos	Ingresos	A_igual_S (6)
11	Ingresos	Impuestos	A_padre_S (5)
12	Ingresos	Turismo	A_padre_S (5)
13	Ingresos	Gastos	A_hermano_S (1)
14	Impuestos	Ingresos	A_hijo_S (4)
15	Turismo	Ingresos	A_hijo_S (4)
16	Gastos	Ingresos	A_hermano_S (1)
17	Gastos	Gastos	A_igual_S (6)
18	Impuestos	Impuestos	A_igual_S (6)
19	Impuestos	Turismo	A_hermano_S (1)
20	Turismo	Impuestos	A_hermano_S (1)
21	Turismo	Turismo	A_igual_S (6)

Tabla 1. Lista de equivalencias de la ontología de salidas

#	Con_A	Con_S	Grado (peso)
1	Economía	Economía	A_igual_S (6)
2	Economía	Ingresos	A_padre_S (5)
3	Economía	Gastos	A_padre_S (5)
4	Economía	Impuestos	A_abuelo_S (3)
5	Economía	Turismo	A_abuelo_S (3)

Tabla 2. Lista de equivalencias filtrada

Anuncio	Con_S	Grado	Tipo
A1	Economía	6	Salida
A1	Ingresos	5	Salida
A1	Gastos	5	Salida
A1	Impuestos	3	Salida
A1	Turismo	3	Salida
A1	Región	4	Entrada
A1	Ciudad	1	Entrada
A1	Desarrollado	5	Entrada
A1	Subdesarrollado	5	Entrada
A1	País	6	Entrada

Tabla 3. Lista de equivalencias entre el anuncio y los conceptos

Puesto que la mayor parte de la información coincidente es pre-computada en la etapa de publicación, en la etapa de búsqueda el *Agente Matchmaker* se limita a realizar una simple búsqueda en el *Registro de Anuncios* basándose en la estructura de la Tabla 3.

A continuación se describen los pasos que realiza el algoritmo utilizado por el *Mecanismo de Coincidencia*:

- Cuando el *Agente Matchmaker* recibe una solicitud, es decir, el URI de la descripción OWL-S del Servicio Web solicitado, el *Mecanismo de Coincidencia* carga la solicitud en memoria y obtiene las salidas y entradas de la solicitud.

- Todas las salidas obtenidas de la solicitud se buscan en el campo Con_S de la estructura mostrada en la Tabla 3, teniendo en cuenta solamente aquellos conceptos de tipo "Salida", con el fin de recuperar grupos de anuncios que tienen un grado de equivalencia con dichas salidas, es decir, un grupo de anuncios por cada salida de la solicitud. Por ejemplo, si la solicitud tiene como conceptos de salida a "Ingresos" y "Gastos", entonces se va a recuperar un grupo de anuncios que tienen equivalencia con la salida "Ingresos" representado por el conjunto: Grupo_Salida1 = {<A1, Ingresos, 5>, <A2, Ingresos, 4>, <A3, Ingresos, 6>, <A4, Ingresos, 3>}, y un grupo de anuncios que tienen equivalencia con la salida "Gastos" representado por el conjunto: Grupo_Salida2 = {<A1, Gastos, 5>, <A2, Gastos, 3>, <A4, Gastos, 4>}.

- Luego, el *Mecanismo de Coincidencia* encuentra los anuncios que son comunes entre los grupos de salidas recuperados sumando los grados de equivalencia por cada anuncio común, es decir Grupo_Salida1 interceptado con Grupo_ Salida2. Si la intercepción no es hallada entonces la búsqueda falla. Por el contrario, si la intercepción es hallada lo llamaremos Grupo_Salida_Equivalente. Por ejemplo, en la intersección de Grupo_Salida1 y Grupo_Salida2 se obtiene un Grupo_Salida_Equivalente = {<A1, 10>, <A2, 7>, <A4, 10>}. En el ejemplo anterior, la intersección garantiza que sólo los anuncios A1, A2 y A4 tienen un grado de equivalencia con todas las salidas de la solicitud.

- De igual forma, todas las entradas obtenidas de la solicitud se buscan en el campo Con_S de la estructura mostrada en la Tabla 3, teniendo en cuenta solamente aquellos conceptos de tipo "Entrada" de los anuncios equivalentes encontrados en la fase anterior, con el fin de recuperar grupos de anuncios que tienen un grado de equivalencia con dichas entradas, es decir, un grupo de anuncios por cada entrada de la solicitud. Por ejemplo, si la solicitud tiene como concepto de entrada a "País", entonces se va a recuperar sólo un grupo de anuncios que tienen equivalencia con la entrada "País" representado por el conjunto: Grupo_Entrada1 = {<A1, País, 6>, <A2, País, 5>}.

- Luego, el *Mecanismo de Coincidencia* encuentra los anuncios que son comunes entre los grupos de entradas recuperados sumando los grados de equivalencia por cada anuncio común, es decir Grupo_Entrada1 interceptado con algún otro grupo. Para este caso no se realiza la intersección ya que sólo existe un grupo de entrada, entonces el Grupo_Entrada1 se convertirá en un nuevo grupo que lo llamaremos Grupo_Entrada_Equivalente. Este nuevo grupo queda conformado así: Grupo_Entrada_Equivalente = {<A1, 6>, <A2, 5>}. En el ejemplo anterior, se garantiza que sólo los anuncios A1 y A2 tienen un grado de equivalencia con todas las salidas y entradas de la solicitud.

- Finalmente, los anuncios encontrados en la fase anterior, para nuestro ejemplo A1 y A2, se ordenan de mayor a menor grado de equivalencia teniendo en cuenta primero los grados de equivalencia con respecto a las salidas y de existir un empate entre sus grados de equivalencia, se toman en cuenta los grados de equivalencia con respecto a las entradas. En el ejemplo, los grados de equivalencia obtenidos con respecto a las salidas son {<A1, 10>, <A2, 7>} y los grados de equivalencia obtenidos con respecto a las entradas son {<A1, 6>, <A2, 5>}, por lo tanto, A1 se ordena primero que A2 porque el grado de equivalencia de A1 es mayor que el de A2 con respecto a las salidas y por consiguiente, para el ordenamiento no se toma en cuenta los grados de equivalencia con respecto a las entradas. Esta lista de anuncios ordenados es la que se retorna al solicitante como respuesta a su petición de búsqueda.

De esta manera, se ha descrito la arquitectura y el algoritmo que utiliza el *Agente Matcmaker* para realizar el *Descubrimiento Automático de Servicios Web*. A continuación se presentan casos de prueba para demostrar y validar la funcionalidad del prototipo.

3.2. Casos de prueba

Los casos de prueba planteados aquí servirán para demostrar la hipótesis, específicamente el descubrimiento de Servicios Web basado en

semántica con tiempos de búsquedas eficientes y con un nivel de precisión satisfactorio en los resultados.

Los casos de prueba consisten en: ejemplos de Servicios Web a publicar que se basan en descripciones de Servicios Web desarrolladas en OWL-S y en ontologías desarrolladas en OWL, las cuales se usan para realizar la publicación en el *Registro de Anuncios* del *Sistema Prototipo*; y en ejemplos de Servicios Web a buscar que se basan en descripciones de Servicios Web desarrolladas en OWL-S para realizar el descubrimiento. De este modo, las pruebas se centran en la realización de publicaciones y búsquedas de los Servicios Web de ejemplo aplicando los algoritmos de publicación y descubrimiento de nuestro *Sistema Prototipo* y de los sistemas OWL-S/UDDI Matchmaker [1] y UDDI [8].

3.2.1 Ejemplos de Servicios Web a Publicar

Los Servicios Web a publicar (anuncios) van a consistir en Servicios Web que brindan información demográfica de algún país. Cada Servicio Web tiene: un conjunto de entradas y salidas cuya descripción está desarrollada en OWL-S; y una ontología desarrollada en OWL asociada al conjunto de entradas y otra asociada al conjunto de salidas. Cada descripción OWL-S de los Servicios Web y cada ontología OWL asociada a los Servicios Web se crearon de manera independiente ya que se asume que serán publicados por distintos proveedores o anunciantes que quieran brindar información demográfica de algún país, por lo tanto, las ontologías de salidas OWL mostradas en las Figuras 6, 7 y 8 reflejan similitudes. Sin embargo, en cada uno de los anuncios se utilizará la ontología de entrada OWL mostrada en la Figura 5 para efectos de simplicidad.

Los Servicios Web a publicar (anuncios) son:

Primer anuncio (A1), llamado "Información de Población", tiene como entrada al concepto "País" y como salida al concepto "Población". Su ontología de entrada se muestra en la Figura 5 y su ontología de salida se muestra en la Figura 6.

Segundo anuncio (A2), llamado "Información Sector Rural Urbano", tiene como entrada al concepto "País" y como salida al concepto "Sector". Su ontología de entrada se muestra en la Figura 5 y su ontología de salida se muestra en la Figura 7.

Tercer anuncio (A3), llamado "Información Demografía", tiene como entrada al concepto "País" y como salida al concepto "Demografía". Su ontología de entrada se muestra en la Figura 5 y su ontología de salida se muestra en la Figura 8.

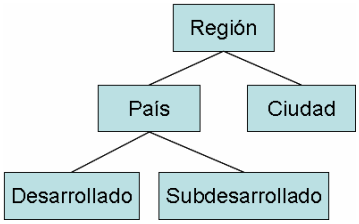

Figura 5. Ontología de entrada para los anuncios

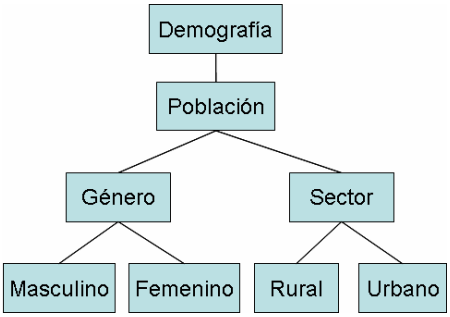

Figura 6. Ontología de salida del anuncio 1 (A1)

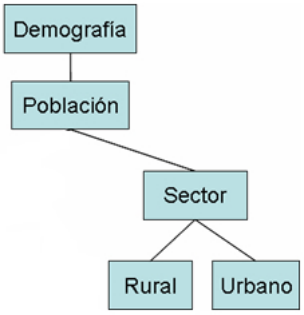

Figura 7. Ontología de salida del anuncio 2 (A2)

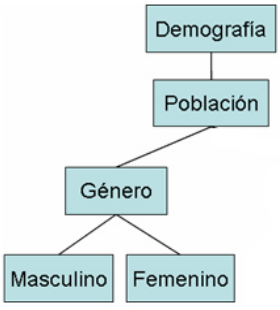

Figura 8. Ontología de salida del anuncio 3 (A3)

3.2.2 Ejemplos de Servicios Web a Buscar

Los ejemplos de búsqueda van a requerir Servicios Web que brinden información demográfica de algún país o región.

El primer ejemplo de búsqueda es una solicitud (S1) que tiene como entrada el concepto "País" y como salidas el concepto "Sector".

El segundo ejemplo de búsqueda es una solicitud (S2) que tiene como entrada el concepto "País" y como salidas el concepto "Población".

El tercer ejemplo de búsqueda es una solicitud (S3) que tiene como entrada el concepto "Región" y como salidas el concepto "Demografía".

3.2.3 Resultados Esperados

Los resultados esperados están en función a los resultados que arroja nuestro sistema, a los resultados que arroja el sistema basado en la arquitectura OWL-S/UDDI Matchmaker, y a los resultados que se obtienen en UDDI.

3.2.3.1 Resultados del Sistema Prototipo

Al realizar la publicación de cada uno de los anuncios se obtienen los grados de equivalencia mostrados en la Tabla 4 aplicando el algoritmo descrito en la sección 3.1.2.2 y teniendo en cuenta los grados de equivalencia descritos en la sección 3.1.2.1.

Después de la etapa de publicación se lleva a cabo el descubrimiento de Servicios Web basado en los ejemplos de búsquedas aplicando el algoritmo descrito en la sección 3.1.2.3 y los resultados obtenidos por cada solicitud son:

- **Resultados para la primera solicitud (S1):**

Grupo_Salida = {<A1, Sector, 5>, <A2, Sector, 6>}

Grupo_Entrada = {<A1, País, 6>, <A2, País, 6>}

Grupo_Resultado = {<A2, 6, 6>, <A1, 5, 6>}

- **Resultados para la segunda solicitud (S2):**

Grupo_Salida = {<A1, Población, 6>, <A2, Población, 4>, <A3, Población, 5>}

Grupo_Entrada = {<A1, País, 6>, <A2, País, 6>, <A3, País, 6>}

Grupo_Resultado = {<A1, 6, 6>, <A3, 5, 6>, <A2, 4, 6>}

- **Resultados para la tercera solicitud (S3):**

Grupo_Salida = {<A1, Demografía, 4>, <A2, Demografía, 2>, <A3, Demografía, 6>}

Grupo_Entrada = {<A1, Región, 4>, <A2, Región, 4>, <A3, Región, 4>}

Grupo_Resultado = {<A3, 6, 4>, <A1, 4, 4>, <A2, 2, 4>}

Anuncio	Con_S	Grado	Tipo
A1	Población	6	Salida
A1	Demografía	4	Salida
A1	Género	5	Salida
A1	Sector	5	Salida
A1	Masculino	3	Salida
A1	Femenino	3	Salida
A1	Rural	3	Salida
A1	Urbano	3	Salida
A2	Sector	6	Salida
A2	Rural	5	Salida
A2	Urbano	5	Salida
A2	Población	4	Salida
A2	Demografía	2	Salida
A3	Demografía	6	Salida
A3	Población	5	Salida
A3	Género	3	Salida
A1	País	6	Entrada
A1	Región	4	Entrada
A1	Desarrollado	5	Entrada
A1	Subdesarrollado	5	Entrada
A1	Ciudad	1	Entrada
A2	País	6	Entrada
A2	Región	4	Entrada
A2	Desarrollado	5	Entrada
A2	Subdesarrollado	5	Entrada
A2	Ciudad	1	Entrada
A3	País	6	Entrada
A3	Región	4	Entrada
A3	Desarrollado	5	Entrada
A3	Subdesarrollado	5	Entrada
A3	Ciudad	1	Entrada

Tabla 4. Lista de equivalencias usando el sistema prototipo

3.2.3.2 Resultados de la Arquitectura OWL-S/UDDI Matchmaker

Al realizar la publicación de cada uno de los anuncios se obtienen los grados de equivalencia mostrados en la Tabla 5 aplicando el algoritmo y los grados de equivalencia descritos en [1]. Para

este caso se asume que los grados de equivalencia toman los siguientes valores: Exact = 3, Plugin = 2, Subsume = 1 y Fail = 0.

Anuncio	Con_S	Grado	Tipo
A1	Población	3	Salida
A1	Demografía	3	Salida
A1	Género	3	Salida
A1	Sector	3	Salida
A1	Masculino	2	Salida
A1	Femenino	2	Salida
A1	Rural	2	Salida
A1	Urbano	2	Salida
A2	Sector	3	Salida
A2	Rural	3	Salida
A2	Urbano	3	Salida
A2	Población	3	Salida
A2	Demografía	1	Salida
A3	Demografía	3	Salida
A3	Población	3	Salida
A3	Género	2	Salida
A1	País	3	Entrada
A1	Región	3	Entrada
A1	Desarrollado	3	Entrada
A1	Subdesarrollado	3	Entrada
A2	País	3	Entrada
A2	Región	3	Entrada
A2	Desarrollado	3	Entrada
A2	Subdesarrollado	3	Entrada
A3	País	3	Entrada
A3	Región	3	Entrada
A3	Desarrollado	3	Entrada
A3	Subdesarrollado	3	Entrada

Tabla 5. Lista de equivalencias usando OWL-S/UDDI Matchmaker

Después de la etapa de publicación se lleva a cabo el descubrimiento de Servicios Web basado en los ejemplos de búsquedas aplicando el algoritmo descrito en [1] y los resultados obtenidos por cada solicitud son:

- **Resultados para la primera solicitud (S1):**

Grupo_Salida = {<A1, Sector, 3>, <A2, Sector, 3>}

Grupo_Entrada = {<A1, País, 3>, <A2, País, 3>}

Grupo_Resultado = {<A1, 3, 3>, <A2, 3, 3>}

- **Resultados para la segunda solicitud (S2):**

Grupo_Salida = {<A1, Población, 3>, <A2, Población, 3>, <A3, Población, 3>}

Grupo_Entrada = {<A1, País, 3>, <A2, País, 3>, <A3, País, 3>}

Grupo_Resultado = {<A1, 3, 3>, <A2, 3, 3>, <A3, 3, 3>}

- **Resultados para la tercera solicitud (S3):**

Grupo_Salida = {<A1, Demografía, 3>, <A2, Demografía, 1>, <A3, Demografía, 3>}

Grupo_Entrada = {<A1, Región, 3>, <A2, Región, 3>, <A3, Región, 3>}

Grupo_Resultado = {<A1, 3, 3>, <A3, 3, 3>, <A2, 1, 3>}

3.2.3.3 Resultados en UDDI

Los anuncios A1, A2 y A3 no pueden publicarse en UDDI [22] con su descripción semántica OWL-S junto con sus ontologías OWL porque UDDI no soporta descripciones semánticas de Servicios Web dentro de su registro de anuncios, por consiguiente, aquí no va a existir una lista de grados de equivalencias. Así, la publicación en UDDI utiliza solamente el proveedor del Servicio Web, el nombre del Servicio Web, la descripción del Servicio Web, el URI de la localización del Servicio Web, y el URI de su interfaz WSDL, tal y como se especifica en [8].

Después de la etapa de publicación se lleva a cabo el descubrimiento de Servicios Web. Para esto es necesario hacer búsquedas por palabras claves y no por la descripción semántica en OWL-S como es realizado en 3.2.3.1 y 3.2.3.2. Estas búsquedas por palabras claves se pueden hacer por nombre del proveedor, por nombre del Servicio Web, por la descripción del Servicio Web, entre otros. Para los casos de prueba, se tuvo en cuenta el nombre del Servicio Web. Entonces, los resultados obtenidos son:

- **Resultados de una solicitud de búsqueda por nombre de Servicio "Información de población":**

Grupo_Resultado = {<A1, Información de Población>, <A4, Información de Población de Enfermos de Cáncer>, <A5, Información de Población Activa Económicamente>, …}

- **Resultados de una solicitud de búsqueda por nombre de Servicio "Rural":**

Grupo_Resultado = {}

4. Discusión

4.1 Análisis de resultados

A continuación presentamos el análisis de los resultados obtenidos en base a los casos de prueba realizados en la sección anterior.

Comparando los resultados obtenidos en el Sistema Prototipo con los resultados obtenidos en la arquitectura OWL-S/UDDI Matchmaker, observamos que el orden de la lista de anuncios encontrados por cada uno es diferente, puesto que cada método asume diferentes grados de equivalencia y diferente interpretación de las relaciones entre los conceptos de una ontología. Para esto, el grado de equivalencia Exact (de OWL-S/UDDI Matchmaker) corresponde a los grados de equivalencia Igual, Padre e Hijo del Sistema Prototipo; el grado de equivalencia PlugIn (de OWL-S/UDDI Matchmaker) corresponde al grado de equivalencia Abuelo del Sistema Prototipo; el grado de equivalencia Subsume (de OWL-S/UDDI Matchmaker) corresponde al grado de equivalencia Nieto del Sistema Prototipo; el grado de equivalencia Hermano del Sistema Prototipo no corresponde con ningún grado de equivalencia en OWL-S/UDDI Matchmaker; y el grado de equivalencia Fail (de OWL-S/UDDI Matchmaker) corresponde al grado de equivalencia NoesFamiliar del Sistema Prototipo. Debido a esto, los resultados del Sistema Prototipo muestran una mejor interpretación del nivel de coincidencia semántica que puede existir entre una solicitud dada y los anuncios, ya que en la arquitectura OWL-S/UDDI Matchmaker, dos anuncios con el mismo nivel de equivalencia pueden significar anuncios con distintos niveles de equivalencia en el Sistema Prototipo; por ejemplo, esto se puede ver en los grupos de resultados obtenidos para la primera solicitud S1 en ambos métodos de descubrimiento, en donde para el Sistema Prototipo A2 tiene un nivel de coincidencia mayor que A1 con respecto a S1, mientras que para OWL-S/UDDI Matchmaker A1 y A2 tienen el mismo nivel de coincidencia con respecto a S1. Igualmente para los grupos de resultados obtenidos con la segunda solicitud S2, el Sistema Prototipo muestra que A1, A3 y A2 tienen distintos niveles de equivalencia con respecto a S2, mientras que para OWL-S/UDDI Matchmaker A1, A2 y A3 tienen el mismo nivel de coincidencia con respecto a S2. De la misma forma, para los grupos de resultados obtenidos con la tercera solicitud S3, el Sistema Prototipo muestra que A3 y A1 tienen distintos niveles de equivalencia con respecto a S3, mientras que para OWL-S/UDDI Matchmaker A1 y A3 tienen el mismo nivel de coincidencia con respecto a S3. Por lo tanto, utilizando siete grados de equivalencia como lo hace el Sistema Prototipo, en lugar de cuatro como lo hace OWL-S/UDDI Matchmaker en [1], se obtiene mayor precisión y mejor diferenciación en los niveles de coincidencia de los anuncios descubiertos, como también un mayor número de resultados que guarden una relación semántica con la solicitud puesto que la coincidencia semántica es memos restrictiva, brindando una mayor flexibilidad al *Mecanismo de Descubrimiento.* Para el Sistema Prototipo se han considerado siete grados de equivalencia porque al agregar más grados se aumentaría el riesgo de obtener servicios de poco interés para el solicitante, ya que la relación semántica entre el concepto que corresponde a un anuncio y el concepto que corresponde a una solicitud dentro de la jerarquía de la ontología puede ser tan distante semánticamente que no cumple con las expectativas del solicitante.

Por otro lado, al comparar los resultados obtenidos en el Sistema Prototipo con los resultados obtenidos en la arquitectura UDDI, se observa que en UDDI se encontraron anuncios que no tenían relación con la primera solicitud y por otro lado no se encontraron anuncios relacionados con la segunda solicitud, dado que el Mecanismo de Descubrimiento en UDDI se limita a hacer comparaciones sintácticas en sus búsquedas basado en palabras claves. Por lo tanto, el Sistema Prototipo arroja resultados que coinciden mas con la solicitud ya que no se limita a hacer búsquedas por palabras claves sino que hace uso de la descripción semántica del Servicio Web.

4.2 Aporte del Sistema Prototipo

El aporte a la solución del problema de descubrimiento de Servicios Web está dado por el desarrollo de nuestro Sistema Prototipo. El enfoque utilizado para el desarrollo del Sistema Prototipo y que marca la diferencia con otros trabajos relacionado expuestos en [1, 2, 4, 5 y 6], aporta básicamente tres particularidades que tienen que ver con la arquitectura, con el mecanismo de descubrimiento y con los grados de equivalencia utilizados.

En la arquitectura planteada aquí se incorporan dos Servicios Web, el de publicar y el de buscar, que son los componentes del sistema que permiten la interoperabilidad con otros sistemas o agentes de software que quieran hacer uso del Mecanismo de Descubrimiento del Sistema Prototipo. Esta interoperabilidad, permitida por los Servicios Web de publicación y de búsqueda, da el valor agregado al Sistema Prototipo y es un aporte importante ya que permite a otros sistemas o agentes de software poder consumir fácilmente estos servicios y así automatizar sus tareas de publicación o búsqueda de Servicios Web. Con respecto al Mecanismo de Descubrimiento, el Sistema

Prototipo realiza una pre-computación para encontrar grados de equivalencia entre el anuncio y posibles solicitudes al momento de publicar y no al momento de buscar un Servicio Web, garantizando una búsqueda rápida. Y por último, los grados de equivalencia utilizados aquí son siete y van a permitir una mayor flexibilidad al proceso de descubrimiento, ya que va a existir una menor restricción en los resultados de búsqueda, puesto que la coincidencia entre un servicio anunciado y uno solicitado no tiene que ser necesariamente exacta.

Además de estos aportes, la arquitectura propuesta está basada en OWL y OWL-S los cuales son estándares de la W3C y versiones mejoradas de DAML+OIL y DAML-S utilizados en [2, 4, 5 y 6]. Otra diferencia de la arquitectura propuesta en comparación a lo planteado en [1, 2 y 4] es que no se utiliza ni se extiende la API del registro UDDI para incorporarle un mecanismo de búsqueda basado en capacidades, en lugar de esto se ha implementado una API desarrollada en Java y se ha diseñado un Registro de Anuncios cuyo modelo de datos ha sido creado en una base de datos relacional para almacenar información de los Servicios Web publicados tales como: nombre del Servicio Web, finalidad del Servicio Web, URI del Servicio Web, URI de la descripción semántica OWL-S del Servicio Web y los grados de equivalencia pre-computados del Servicio Web.

De esta manera se sustenta la contribución del Sistema Prototipo o Agente Matchmaker. Así, para demostrar la utilidad del Sistema Prototipo se presenta en la siguiente sección un escenario de aplicación que ilustra el uso del Agente Matchmaker.

5. Escenario de aplicación

El escenario de aplicación planteado aquí consiste en una empresa denominada “Planeta” la cual está interesada en crear un portal Web donde se muestre información económica, geográfica y demográfica de algunos países del mundo. Dado que esa información no es manejada por la empresa “Planeta”, la empresa debe encontrar Servicios Web de otras empresas que manejen y brinden dicha información. Por consiguiente, la empresa “Planeta” hará uso del Agente Matchmaker para descubrir los Servicios Web que necesita y consumirlos dentro de su portal Web. Por otro lado, existe otra empresa denominada “Atlas” que maneja información sobre datos económicos, geográficos y demográficos de varios países de América. Esta empresa ha desarrollado tres Servicios Web para brindar esa información y también ha utilizado el Agente Matchmaker para publicar las descripciones de sus Servicios Web.

Para ilustrar mejor este escenario de aplicación, en la Figura 9 se muestran todos los elementos que intervienen en este proceso de descubrimiento.

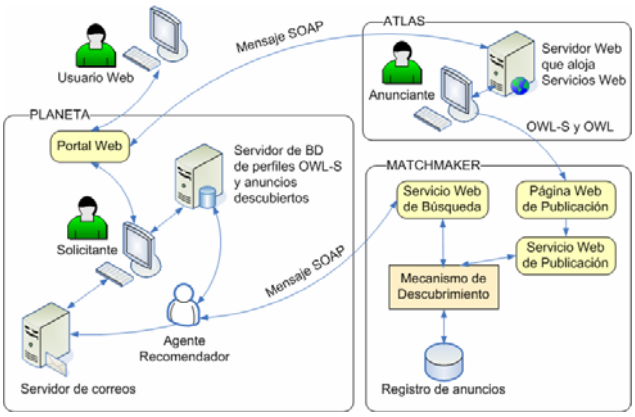

Figura 9. Escenario de aplicación

La empresa “Atlas” utiliza la página Web de publicación que expone el Agente Matchmaker para publicar las descripciones de sus Servicios Web. En esta página Web la empresa “Atlas” debe ingresar: el URI de la localización del Servicio Web, el URI de la descripción OWL-S del Servicio Web, y el URI de las ontologías OWL relacionadas a las entradas y salidas del Servicio Web. Entonces, antes de publicar, la empresa “Atlas” crea las descripciones de sus Servicios Web en OWL-S y las ontologías de conceptos en OWL.

En el caso de la empresa “Planeta”, para poder buscar los Servicios Web de su interés podría de igual forma hacer uso de la página Web de búsqueda en donde ingrese el URI de la descripción del Servicio Web a buscar. Esta descripción también esta basada en OWL-S que la empresa “Planeta” tendrá que crear antes de realizar la búsqueda. Pero, para automatizar la tarea de búsqueda y no tener que entrar periódicamente a la página Web de búsqueda para encontrar Servicios Web, la empresa “Planeta” ha desarrollado un agente de software que realiza dicha tarea de manera automática. Este agente, al que se ha denominado “Agente Recomendador”, es el encargado de consumir el Servicio Web de búsqueda del Agente Matchmaker para encontrar descripciones de Servicios Web publicados que coincidan con las descripciones de los Servicios Web solicitados. Las descripciones de los Servicios Web a solicitar se almacenan en una base de datos que el “Agente Recomendador” utiliza para hacer una petición al Agente Matchmaker y luego de que el Agente Matchmaker le retorne los resultados de la búsqueda, el “Agente Recomendador” toma los resultados para almacenarlos en la base de datos que maneja y a la vez enviar dichos resultados a una cuenta de correo que es utilizada por la

persona encargada de seleccionar e incorporar los Servicios Web descubiertos en el portal Web.

6. Conclusiones y trabajos futuros

En este artículo se expuso la importancia de los Servicios Web y la necesidad de descubrirlos utilizando un mecanismo más eficiente que UDDI basado en los estándares OWL-S y OWL de la W3C. Se presentó una síntesis del estado del arte que muestra otros trabajos relacionados para atacar el problema de descubrimiento de Servicios Web y que sirvieron de modelo para plantear la arquitectura del Sistema Prototipo. También se describió el algoritmo de descubrimiento utilizado en el Sistema Prototipo basado en las entradas y salidas de la descripción semántica del Servicio Web. El desarrollo de este Sistema Prototipo constituye una contribución a la solución del problema de descubrimiento de Servicios Web y se enfatiza que para hacer descubrimiento automático de Servicios Web con mayor precisión es necesario el uso de semántica en la descripción de los Servicios Web.

Por otro lado, como trabajos futuros se propone realizar: (1) el desarrollo de un mecanismo de descubrimiento aún más eficiente haciendo uso no sólo de las entradas y salidas de la descripción de capacidades de los Servicios Web, sino también, de las pre-condiciones y efectos descritos en las capacidades de los Servicios Web; y (2) desarrollar un Agente Broker que no sólo se encargue del descubrimiento (como lo hace el Agente Matchmaker) sino también de la *invocación automática* de Servicios Web.

Referencias

[1] Naveen Srinivasan, Massimo Paolucci and Katia Sycara, "Adding OWL-S to UDDI, implementation and throughput". First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004) June 2004, San Diego, California, USA.

[2] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, Katia Sycara; "Semantic Matching of Web Services Capabilities". In Proceedings of the 1st International Semantic Web Conference (ISWC2002).

[3] Katia Sycara, Massimo Paolucci, Julien Soudry, and Naveen Srinivasan "Dynamic Discovery and Coordination of Agent-Based Semantic Web Services". IEEE Internet Computing May 2004.

[4] Katia Sycara, Massimo Paolucci, Anupriya Ankolekar and Naveen Srinivasan, "Automated Discovery, Interaction and Composition of Semantic Web Services". Journal of Web Semantics, September 2003.

[5] Lei Li, Ian Horrocks, "A Software Framework For Matchmaking Based On Semantic Web Technology". In Proceedings of the Twelfth International World Wide Web Conference (WWW 2003), pages 331-339. ACM, 2003.

[6] Boualem Benatallah, Mohand-Said Hacid, Alain Leger, Christophe Rey, Farouk Toumani, "On automating Web services discovery". The VLDB Journal — The International Journal on Very Large Data Bases. Marzo, 2005.

[7] The OWL Services Coalition "OWL-S, Semantic Markup for Web Services", 2003. http://www.daml.org/services/owl-s/1.1/overview/

[8] Reporte técnico UDDI. http://www.uddi.org/pubs/Iru_UDDI_Technical_White_Paper.pdf

[9] OWL Web Ontology Language Overview. http://www.w3.org/TR/owl-features/

[10] Semantic Web. http://www.w3.org/2001/sw/

[11] Arquitectura de los Servicios Web. http://www.w3.org/TR/ws-arch

[12] Ethan Cerami. "Web Services Essentials. Distributed Applications with XML-RPC, SOAP, UDDI & WSDL". Editorial O'Reilly. Primera edición febrero 2002. ISBN: 0-596-00224-6.

[13] Kim Topley. "Java Web Services in a Nutshell. A Desktop Quick Reference". Editorial O'Reilly. Primera edición junio 2003. ISBN: ISBN : 0-596-00399-4.

[14] Sheila A. McIlraith, Tran Cao Son, and Honglei Zeng, "Semantic Web Services". Stanford University. Marzo 2001. http://www.ksl.stanford.edu/people/sam/ieee01.pdf

[15] Requerimientos de Arquitectura de los Servicios Web Semánticos. http://www.daml.org/services/swsa/swsa-requirements.html

[16] OWL-S API. http://www.mindswap.org/2004/owl-s/index.shtml

[17] Protégé OWL API. http://protege.stanford.edu/plugins/owl/index.html

[18] Jena Semantic Web Framework. http://jena.sourceforge.net/

[19] Racer: Middleware for the Semantic Web. http://www.racer-systems.com/

[20] Tim Berners-Lee. Director del Consorcio World Wide Web. http://www.w3.org/People/Berners-Lee/

[21] Amy Moormann Zaremsky, Jeannette M. Wing. "Specification Matching of Software Components". ACM Transactions on Software Engineering and Methodology (TOSEM), octubre de 1997.

[22] Registro de Negocios UDDI de Microsoft. http://uddi.microsoft.com/