AUTOMATIZACIÓN DE LA ANOTACIÓN SEMÁNTICA DE SERVICIOS CONVERGENTES CON BASE EN LA DESAMBIGUACIÓN LINGÜÍSTICA DE SU DESCRIPTOR

Universidad del Cauca

Monografía presentada para optar al título de Ingeniero en Electrónica y Telecomunicaciones

ÁNGELA PAOLA DE LA CRUZ CAICEDO
YONATAN LEIDER BOLAÑOS BASTIDAS

Director: PhD. Ing. Juan Carlos Corrales Muñoz

Universidad del Cauca
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE TELEMÁTICA
POPAYÁN
2013

Tabla de Contenido

Anexos

Índice de Ilustraciones

Índice de Tablas

1. INTRODUCCIÓN

1.1. CONTEXTO

Debido a la reducción en las barreras de acceso a la tecnología, la generalización de las redes de conmutación de paquetes, el auge de la arquitectura IMS que conlleva a un incremento en las capacidades a nivel de infraestructura de red, y el creciente entorno competitivo global, las empresas de telecomunicaciones se han visto en la necesidad de ajustarse a modelos de negocio adaptables y flexibles (User-Centric Future Internet and Telecommunication Services, 2009). Como consecuencia, en el nivel operativo, estas organizaciones están evolucionando hacia un nuevo modelo de creación de servicios, en el cual el desarrollo de los mismos es independiente de la tecnología subyacente y del dispositivo utilizado, permitiendo la creación de servicios convergentes, los cuales surgen a partir de la unión de servicios Web y servicios en el dominio de Telecomunicaciones (User-Centric Future Internet and Telecommunication Services, 2009). Cuando se habla de servicio convergente, se habla de servicio de valor agregado, de reusabilidad de componentes, de disminución en el tiempo de despliegue del servicio en el mercado (*time to market*), y de formación de cadenas de valor entre empresas de telecomunicaciones y empresas proveedoras de servicios en la Web u otros (terceras partes). Un servicio convergente puede definirse como aquel que funciona a través de redes de operadores de telecomunicaciones e internet, interactuando entre diferentes redes y protocolos de comunicación (Omelette, 2011), ventajas claras que facilitan a las empresas de telecomunicaciones, aumentar el número de servicios ofertados a menor costo de producción y de venta, para la satisfacción de sus usuarios, logrando atraerlos y lo que es más importante fidelizarlos. Sin embargo, debido a la gran cantidad de servicios Web existentes, es necesaria la creación de mecanismos de recuperación que faciliten su búsqueda. No obstante, este tipo de mecanismos no se observan en el dominio Telco debido al reducido número de los mismos.

Encontrar el servicio Web adecuado para componer un servicio convergente, requiere que la comunidad de desarrolladores tenga a su disposición herramientas automáticas que faciliten la búsqueda de servicios existentes en la Web. Los descriptores de los servicios, tales como *WSDL* (*Web Service Description Language*) (Christensen, y otros, 2001), *WADL* (*Web Application Description Language*) (Hadley, 2009) y *WSDL* 2.0 (D. Booth, 2007), permiten el descubrimiento de los servicios. Es por esta razón, que en la última década se han propuesto soluciones para encontrar servicios basados en la exploración de descriptores, algunos ejemplos son: *Binding Point*[1]*, Grand Central*[2]*, Salcentral*[3]*,* y *Web Service List*[4] (Y. Zhang, 2010). Estos motores de búsqueda permiten la recuperación de servicios de la Web mediante la comparación de palabras claves, entre el parámetro de consulta del usuario y los atributos de los descriptores, con el fin de hallar alguna coincidencia.

Por lo general, la recuperación de servicios por parte de los motores de búsqueda, depende totalmente de la información provista por los descriptores, la cual no es suficiente, debido a que esta información no captura eficazmente la semántica funcional del servicio. Este problema, ha requerido la creación de métodos y

[1] http://www.bindingpoint.com
[2] http://www.grandcentral.com/directory
[3] http://www.salcentral.com
[4] http://www.Webservicelist.com

mecanismos que permiten adicionar información en los descriptores, con el fin de hacer más eficientes las operaciones de búsqueda y recuperación (Maleshkova, 2009) (J. Domingue, 2009). No obstante, los mecanismos existentes dependen de manera total o parcial de la intervención humana, lo cual genera un elevado consumo de tiempo y dinero, así como calidad limitada.

Teniendo en cuenta la importancia que tiene la recuperación de servicios, debido a que de ella dependen procesos más complejos como la composición de servicios convergentes, el presente trabajo parte del estudio de los diferentes enfoques propuestos para la adición de información semántica en los descriptores, con el fin de determinar la manera en la cual puede prescindirse de la intervención humana en el proceso.

1.2. DECLARACIÓN DEL PROBLEMA

Actualmente, las empresas de telecomunicaciones se enfrentan a un entorno competitivo global, que a su vez ha involucrado una reducción en las barreras de acceso a la tecnología, junto con un incremento en las capacidades de infraestructura de red. Todo esto ha impuesto para estas organizaciones, la necesidad de ajustarse a modelos de negocio flexibles y escalables (User-Centric Future Internet and Telecommunication Services, 2009). Como resultado, en el nivel operativo, estas organizaciones están orientando su trabajo a la creación y oferta de servicios convergentes de valor agregado para satisfacer las necesidades cada vez más demandantes de sus usuarios mediante la composición de servicios, integración de aplicaciones y la adopción de arquitecturas orientadas a servicios.

La creación de servicios convergentes implica la ubicación e integración de componentes reusables (Servicios atómicos) cada uno desplegado sobre su propia red. Estos componentes disponen de interfaces que describen sus capacidades funcionales en forma tal que permita su publicación, descubrimiento, selección, contratación y seguimiento (descriptores WSDL de servicios Web y en el dominio Telco). Sin embargo en la práctica el proceso de composición de servicios convergentes se ha visto restringido, entre otras cosas por la limitada precisión en los resultados obtenidos de los sistemas de búsqueda de servicios disponibles en la Web, debido a la enorme y creciente cantidad de este tipo de recursos (Y. Zhang, 2010).

La cantidad de recursos en la Web crece rápidamente, al igual que el número de usuarios, quienes generalmente enfrentan un proceso dispendioso al ejecutar búsquedas de recursos o servicios acordes a sus necesidades. Existen directorios Web mantenidos por personal técnico especializado (*Yahoo*[5], por ejemplo), los cuales cubren temas o categorías populares, es decir, aquellos consultados con más frecuencia por parte de los usuarios. No obstante, dichos directorios demandan más costos de construcción y mantenimiento, y su cobertura en cuanto a categorías es limitada. Por otro lado, los motores de búsqueda dependen en gran medida de la comparación de palabras claves; en consecuencia, suelen retornar muchos resultados de baja calidad (Y. Zhang, 2010). Una de las alternativas de solución a este problema consiste en la adición de anotaciones semánticas en los descriptores de los servicios con el fin de facilitar su búsqueda. (Maleshkova, 2009)

En la literatura una gran variedad de enfoques propuestos para la adición de anotaciones semánticas requieren de manera parcial o total la intervención humana (Cherifi, 2010); lo anterior implica un esfuerzo considerable por parte de la comunidad

[5] Sitio Web en la dirección *www.yahoo.com,* para Colombia

de desarrolladores y/o consumidores que realizan anotaciones de manera colaborativa, lo cual se traduce en un elevado consumo de recursos de tiempo y dinero. Las aproximaciones actuales relacionadas con la adición semiautomática de componentes semánticos sobre descriptores de servicios, generalmente abordan este proceso de anotación considerando sus atributos de manera aislada, es decir, sin tener en cuenta su significado en relación con el contexto que define el contenido del documento descriptor. Limitando en consecuencia, la calidad de las anotaciones en la medida en que estas pueden no estar alineadas con el sentido de los atributos del servicio. (M. Maleshkova, 2010) (Vitvar, 2008)

En este sentido, el problema central abordado en este proyecto, está relacionado con la optimización del proceso de adición de componentes semánticos sobre descriptores de servicios Web y en el dominio de Telecomunicaciones, que podría mejorar los resultados obtenidos por motores de búsqueda, en términos de la precisión de los mismos. Para esto, se considera necesario el desarrollo de mecanismos que permitan la automatización de operaciones de adición de anotaciones semánticas sobre los descriptores estándar de los servicios Web y en el dominio de Telecomunicaciones, de manera coherente con su contexto. Durante el desarrollo de la presente investigación se ha dado respuesta al siguiente interrogante:

¿Cómo automatizar el proceso de adición de anotaciones semánticas sobre servicios del dominio Web y de Telecomunicaciones teniendo en cuenta el contexto lingüístico de su descriptor?

1.3. ESCENARIO DE MOTIVACIÓN

Considere una plataforma para la composición de servicios convergentes, que se encargue del proceso de descubrimiento. Necesita, como parte de sus procesos, recuperar no solo servicios Web y de telecomunicaciones sino también servicios convergentes. Suponga que la plataforma hace uso de su propio repositorio de servicios, a partir del cual realiza la composición. Se considera entonces necesario que los servicios disponibles en el repositorio, se encuentren enriquecidos semánticamente para que su búsqueda y recuperación se realice de manera precisa y eficiente. Un mecanismo automático de anotación semántica de servicios Web, de Telecomunicaciones y convergentes, permitiría a la plataforma encontrar fácilmente el servicio adecuado para el servicio convergente que se desee construir, lo cual derivaría en la optimización del consumo de recursos en el desarrollo de las actividades de descubrimiento y composición de servicios convergentes.

1.4. ALCANCE

En el presente proyecto se analizaron algunos mecanismos existentes para la extracción de atributos de los descriptores de servicios web como de telecomunicaciones, así como diferentes métodos de desambiguación del sentido de la palabra, los cuales fueron adaptados y aplicados al ámbito de la anotación semántica de servicios convergentes sin dejar de contemplar la anotación de servicios web y de telecomunicaciones. Lo anterior, se constituye en la base para el desarrollo de mecanismos que permitan la automatización de operaciones de adición de anotaciones semánticas sobre los descriptores estándar de los servicios, de manera coherente con su contexto, a partir de la identificación de términos relevantes contenidos en el descriptor, los cuales comprenden la semántica del servicio. Este proyecto solo considera la anotación semántica de servicios convergentes descritos en

WSDL ya que es el lenguaje estándar para la descripción de servicios web y además permite describir servicios del dominio de las telecomunicaciones según la especificación OneApi (Smith, 2009). El proceso de evaluación y pruebas, al cual fue sometido el prototipo de anotación semántica de servicios convergentes desarrollado en este trabajo, permitió estimar la calidad de los resultados en cuanto a la coherencia semántica entre el dominio del conocimiento del servicio y del dominio del conocimiento de las ontologías utilizadas para realizar la anotación.

1.5. CONTRIBUCIONES

- **Anotación semántica de servicios convergentes:** Luego de analizar muchos trabajos relacionados con la anotación semántica de servicios, y encontrar que sus enfoques se centran solamente en servicios del dominio Web, se plantea la definición de técnicas y algoritmos adaptados simultáneamente a las características y requisitos tanto de los servicios Web, como de servicios de Telecomunicaciones. A partir del estudio de diferentes formatos para la descripción de servicios de diferentes dominios, se abstraen características comunes y se plantean soluciones que comprendan la anotación semántica de servicios pertenecientes a los dominios Web y de Telecomunicaciones.

- **Mecanismo para identificar e interpretar los elementos relevantes de un descriptor de servicio en función del contexto del descriptor:** con el fin de obtener parámetros que permitan seleccionar conceptos ontológicos adecuados y coherentes semánticamente con el servicio, se define un mecanismo para identificar e interpretar los elementos relevantes de un descriptor de servicio en función de su contexto lingüístico. Este mecanismo se compone de una serie de algoritmos de extracción de palabras relevantes contenidas en el descriptor de servicio y de métodos de desambiguación del sentido de la palabra basados en conocimiento.

- **Mecanismo de búsqueda de ontologías disponibles en la Web:** En la actualidad la Web cuenta con un sinfín de recursos de alta calidad que pueden ser utilizados por cualquier agente de acuerdo con sus necesidades, ahorrándole así tiempo y esfuerzo al permitirle soportarse en el trabajo desarrollado y publicado por otros. Las ontologías por ejemplo son uno de estos recursos. Apoyados en un motor de búsqueda de recursos en la Web, que garantiza resultados de alta precisión, en este proyecto se propone un mecanismo de búsqueda automática de ontologías OWL disponibles en la Web.

- **Mecanismo para desambiguar conceptos ontológicos:** La definición de un mecanismo para desambiguar conceptos ontológicos en función de sus conceptos padres y conceptos hijos. Este mecanismo se compone de procesos de razonamiento y métodos de desambiguación del sentido de la palabra basados en conocimiento.

- **Prototipo de anotación semántica de servicios convergentes:** Durante el desarrollo de este proyecto se implementó el prototipo funcional "*ASA-CS*" *(Automatic Semantic Annotator of Converged Services)*, el cual constituye una herramienta automática para la anotación semántica de servicios convergentes con base en la desambiguación lingüística de su descriptor, fundamentada en los mecanismos propuestos también en este trabajo.

- El trabajo desarrollado en el presente proyecto, sirve como alternativa para comparar el mecanismo de enriquecimiento semántico de servicios basado en aprendizaje automático, propuesto en el trabajo de grado de maestría **Automatización del Enriquecimiento Semántico de Servicios en la Web**, desarrollado al interior del Grupo de Ingeniería Telemática de la Universidad del Cauca.

1.6. CONTENIDO DE LA MONOGRAFÍA

Capítulo 2. Estado del Arte

En este capítulo, se abordan las definiciones formales de conceptos claves para el entendimiento del proyecto realizado, tales como servicios convergentes, descripción semántica de servicios, desambiguación lingüística, entre otros; y abordando los trabajos relacionados correspondientes.

Capítulo 3. Desambiguación Lingüística y Descriptores de Servicios

Este capítulo define los métodos de desambiguación basados en conocimiento, para determinar el sentido correcto de una palabra ambigua, y expone cuál de ellos se ajusta adecuadamente al enfoque de anotación semántica automática que se formula en el presente proyecto. Así mismo, presenta una descripción de los componentes de los descriptores de servicio Web y de Telecomunicaciones, y la información provista por cada uno de ellos. Por último, define la forma en la que se usó un método de desambiguación semántica para la desambiguación lingüística de atributos de los descriptores de servicio.

Capítulo 4. Estructura ontológica y Anotación semántica

Corresponde a una descripción de los tipos, los lenguajes y la estructura de las ontologías, que pueden utilizarse en el proceso de anotación.
Presenta una descripción detallada de la técnica para la asignación de entidades ontológicas a los descriptores de servicio, propuesta para la automatización de la anotación semántica de servicios Web y del dominio de Telecomunicaciones, que en su unión generan servicios convergentes.

Capítulo 5. Prototipo y Evaluación

Este capítulo está dedicado a la descripción detallada del prototipo desarrollado, a partir del mecanismo de desambiguación lingüística de los descriptores de servicio Web y de Telecomunicaciones (presentado en el Capítulo 3), así como de la técnica para la asignación de entidades ontológicas a los descriptores de servicio (introducida en el Capítulo 4). En este capítulo, se especifica cada uno de los módulos que componen la arquitectura del sistema y la forma como se llevó a cabo su implementación en el prototipo.

Adicionalmente, se aborda la documentación del proceso de evaluación de desempeño aplicado sobre el prototipo implementado, partiendo de la descripción del benchmark de referencia utilizado, para confrontar los resultados arrojados por el mecanismo y determinar la calidad de los mismos. Posteriormente, se identifican las medidas de desempeño empleadas y finalmente se expone el análisis de los resultados obtenidos en la aplicación de dichas medidas.

Capítulo 6. Conclusiones y Trabajo Futuro

Por último, se analizan los resultados del trabajo realizado, se detallan las principales contribuciones obtenidas en la ejecución del proyecto y se expone un conjunto de recomendaciones importantes para el desarrollo de trabajos futuros.

2. ESTADO DEL ARTE

2.1. INTRODUCCIÓN

El presente capítulo tiene como objetivo dar una visión general sobre los conceptos más relevantes, necesarios para ubicar y comprender la temática tratada en este proyecto. Posteriormente se presenta una revisión del avance de diferentes trabajos de investigación desarrollados en torno a esta temática. Finalmente, se presenta un resumen que expone los principales aportes de este capítulo.

2.2. CONCEPTOS GENERALES

2.2.1. Servicios Convergentes

La convergencia de servicios se refiere a la confluencia de los diferentes servicios ofertados dentro de la infraestructura de telecomunicaciones de un mismo proveedor, que hasta hace poco tiempo, se entendían como independientes. Luego un servicio convergente es aquel que opera entre las diferentes redes y protocolos de comunicación (Omelette, 2011), interactuando con varios servicios atómicos, cada uno funcionando dentro de su propia plataforma. Por ejemplo, una llamada VoIP a un teléfono móvil es un servicio convergente. Como se puede ver en la Figura 1, un proveedor ofrece los servicios no convergentes directamente al usuario, mientras que un servicio convergente se ofrece a través de la interacción y organización de servicios no convergentes.

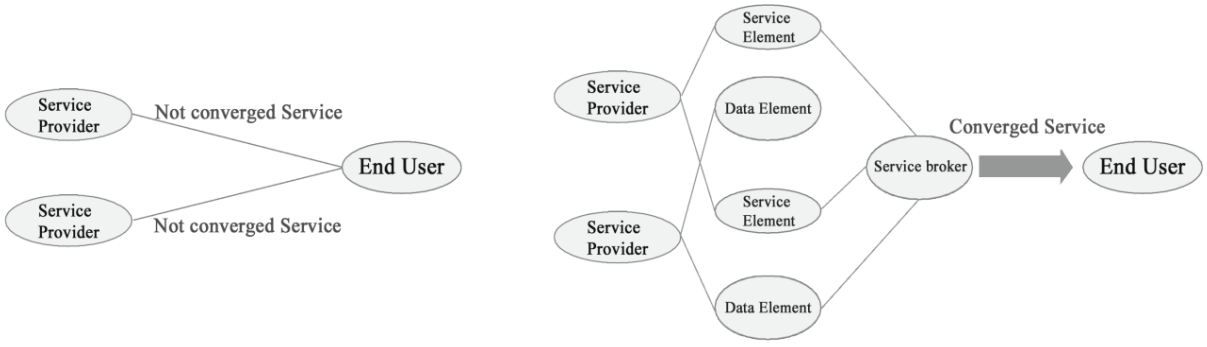


Figura 1. (a) Servicio no convergente, (b) servicio convergente

La convergencia de los servicios que operan en diferentes redes de acceso y de transporte conllevó también a la evolución de las redes tradicionales, hacia lo que hoy se conoce como NGN (Next Generation Network), las cuales tienen la capacidad de desplegar cualquier servicio, permitiendo así pasar de un conjunto de servicios sobre múltiples redes, a una única red que los soporta. La NGN tiene la versatilidad de soportar diferentes tipos de convergencia de servicios que evolucionan de acuerdo con las exigencias de los clientes. Esta convergencia de servicios ha sido posible gracias al Subsistema Multimedia IP (IMS, *IP Multimedia Subsystem*) núcleo de la arquitectura

NGN (Baroncelli, y otros, 2011), el cual permite que cualquier servicio (actual o futuro) pueda desplegarse y transportarse a través de una NGN bajo un protocolo común: IP, eliminando la limitación de las redes tradicionales, en las que el desarrollo de servicios se debe ajustar a la infraestructura de las tecnologías de redes (conmutación de circuitos). La arquitectura para este tipo de redes está conformada por los siguientes niveles (Hayashi, y otros, 2011):

- **Aplicación o Servicios**: Contiene los servidores en donde residen y se ejecutan las aplicaciones que ofrecen los servicios para los clientes. Se proveen funciones, interfaces y APIs estándar para el acceso de las aplicaciones a la NGN.
- **Control**: Asegura el correcto funcionamiento entre la red de transporte, servicios y aplicaciones, mediante la generación, interpretación, distribución y traducción de la señalización correspondiente.
- **Transporte**: En donde se ubican las tecnologías de red que se encargan de las tareas de conmutación, enrutamiento y transmisión de los paquetes IP.

Los actores dentro de la arquitectura de una NGN son: Proveedores de red fijos o móviles, proveedores de servicios y contenido e integradores. Fuera de la NGN, se encuentra como actor principal: el usuario de la misma (Boussard, y otros, 2009). La alta complejidad alrededor del entorno de la red de próxima generación obliga a todos estos actores a cooperar entre sí con el fin de facilitar servicios convergentes a los usuarios.

Esta convergencia de servicios también obliga a los proveedores de dispositivos finales a evolucionar, ofreciendo terminales inteligentes que sean capaces de brindar al usuario una experiencia con variedad, facilidad e interoperabilidad de acceso a servicios convergentes (Lee, y otros, 2012). De esta manera, podría accederse a diferentes ámbitos de comunicación, entretenimiento, información y otros servicios desde un mismo terminal.

Con la convergencia, impulsada por la digitalización, el protocolo de internet (IP) y equipos multimedia, ahora es posible convertir cualquier tipo de contenido en paquetes y transmitirlo digitalmente a través de cualquier plataforma, para entregarlo al usuario en diferentes dispositivos finales. A nivel de comunicaciones alámbricas, se puede ver claramente como ahora servicios de acceso a Internet de banda ancha pueden prestarse tanto por operadores de telefonía como por prestadores del servicio de televisión por cable, pero aun así el usuario cuenta con diferentes receptores como teléfonos móviles, televisores, entre otros, los cuales le permiten interactuar con los diversos servicios.

La implementación de arquitecturas orientadas a servicios (SOA), permite a los operadores de telecomunicaciones replantear el modelo de negocio clásico que durante años han manejado. La aparición de nuevos servicios de comunicaciones basados en Internet como VoIP resultan más rentables y la necesidad de fidelizar a sus clientes convirtiéndose para ellos en una marca socialmente atractiva, son elementos a tener en cuenta para definir las plataformas tecnológicas y métodos de acceso a ellas. Una de las soluciones propuestas en el mundo de las telecomunicaciones ha sido precisamente observar a sus nuevos competidores de Internet, y ofrecer servicios convergentes de valor añadido para el usuario como una nueva y poderosa alternativa de negocio. El objetivo principal es hacer que el operador vaya más allá del beneficio indirecto del transporte de bits. Se fomenta de esta manera que su participación en la cadena de valor se renueve, con una oferta propia y atractiva para los usuarios de servicios convergentes Telco + Internet (Baladrón, y otros, 2009).

2.2.1.1. Servicios Web y de Telecomunicaciones

A nivel de la industria y la academia, existen múltiples definiciones del concepto de *Servicio Web*; posiblemente, cada institución relacionada con el sector maneja una definición propia de este concepto de acuerdo con la naturaleza de sus operaciones y la estructura de su organización. En (Yeganeh, y otros, 2010) por ejemplo, se presentan algunas concepciones alrededor de éste término, existentes al interior de organizaciones como W3C, VLDB Endowment, entre otras.

Una de las definiciones más acertadas acerca de Servicio Web es la concebida en (Muller, 2012), según la cual se establece como:

"Toda aplicación accesible desde otras aplicaciones a través de Internet"

Como se menciona en (Srirama, y otros, 2010) y (Agarwal, y otros, 2009), dentro de una arquitectura de servicios web SOAP, interactúan tres entidades o roles:

- **Proveedor del servicio:** Se encarga de crear servicios y su descripción (interfaz). Esta última se publica de tal manera que el *Registro de servicios* pueda ubicarla. El proveedor publica en el registro un documento WSDL el cual adopta un formato XML para describir servicios como un conjunto de variables que operan sobre mensajes que contienen información orientada a documentos u orientada a procedimientos. Cada servicio ofrecido por el proveedor está asociado a un documento WSDL, publicado en el *Registro de servicios*.
- **Solicitante del servicio:** Es quien consulta al *Registro de servicios* para encontrar una funcionalidad requerida. El registro UDDI provee al Solicitante una descripción del servicio y una *URL* (Uniform Resource Locator) apuntando hacia dicho servicio, a través de la cual el solicitante ejecuta las invocaciones a este último.
- **Registro de servicios:** Actúa como facilitador entre el *Proveedor del servicio* y el *Solicitante*. Se encarga de asistir al solicitante en la búsqueda de los servicios requeridos y le provee la información de enlace hacia los mismos.

UDDI (Universal Description Discovery and Integration) es la especificación estándar adoptada para la interoperabilidad de servicios web (Rajendran, y otros, 2010). UDDI actúa como centro de información público, que permite registrar, encontrar una descripción y acceder de una manera teóricamente automática a un servicio.

Por otro lado, en la estructura de los servicios web REST, las entidades o roles que participan son (L. Richardson, 2007):

- **Cliente:** Como solicitante del servicio, que a través de los métodos HTTP; HEAD, GET, POST, PUT, DELETE, TRACE, OPTIONS Y CONNECT se comunican con el *Servidor Web* contenedor de recursos, término con el que se denomina a las interfaces que transmitan información específica de un dominio sobre el protocolo HTTP.
- **Servidor Web:** Actúa como intermediario entre el *Cliente* y el *Almacenamiento*. Recibe las peticiones de los clientes y las adecua para realizar en el *Almacenamiento* lo que el *Cliente* desea.
- **Almacenamiento:** Contenedor de los recursos, recibe peticiones de los clientes a través del *Servidor* Web para obtener, actualizar, eliminar o insertar un recurso.

Por su parte, un servicio de telecomunicaciones se define genéricamente como la actividad o conjunto de actividades tendientes a satisfacer las necesidades de comunicación de un grupo específico de usuarios, las cuales son planeadas e implementadas por una empresa proveedora de telecomunicaciones con capacidad instalada para soportar sus requerimientos.

De acuerdo con la legislación Colombiana:

"*[…]Se entiende por servicios de telecomunicaciones aquellos que son prestados por personas jurídicas, públicas o privadas, debidamente constituidas en Colombia, con o sin ánimo de lucro, con el fin de satisfacer necesidades específicas de telecomunicaciones a terceros, dentro del territorio nacional o en conexión con el exterior.[…]*" (1993)

En cuanto al despliegue de servicios de telecomunicaciones, las plataformas existentes actualmente son generalmente entornos cerrados basados en soluciones propietarias. Este aspecto condiciona negativamente la integración de servicios entre redes y aumenta el tiempo de oferta de nuevos servicios de telecomunicaciones. Ante esta situación, están surgiendo nuevas tecnologías abiertas, basadas en estándares, para hacer frente a esta limitación. Algunos ejemplos de ellas son OSA (Open Service Platform)/Parlay, Parlay X y OneApi, que permiten comunicar los elementos de red con los servicios ofrecidos a los usuarios. Pero además de esta forma de comunicación, se hace necesario definir una arquitectura única para el entorno de ejecución donde corren los servicios. En ese campo, nuevas tecnologías como SipServlets o JAIN SLEE (Falcarin, y otros, 2008) se adaptación a la perfección. La primera tiene alcance limitado a servicios IMS (IP Multimedia Subsystem) basado en el protocolo SIP (Session Initiation Protocol), mientras que la aplicabilidad de JAIN SLEE puede extenderse a todo tipo de protocolos de red y sistemas externos. Dentro de este entorno, 3GPP estandariza un protocolo y la arquitectura necesaria para el intercambio de mensajes multimedia sobre IP, denominado IMS, que se utiliza para la implementación tanto de servicios multimedia como de los servicios tradicionales de telecomunicaciones dentro de las redes móviles de tercera generación.

Por otra parte, la descripción de servicios de telecomunicaciones no cuenta con un lenguaje o marco de trabajo estándar. Actualmente existe OneAPI, un conjunto de APIs, basado en estándares Web que permiten exponer las capacidades de red del operador para ser accedidas a través de HTTP. Así, cualquier operador de red o proveedor de servicios es capaz de implementar OneAPI para exponer capacidades como SMS, MMS, servicios de localización, pago, entre otros. (Smith, 2009).

2.2.2. Descripción semántica de servicios

Debido a que la cantidad de servicios web existentes es bastante grande y va en aumento, los esfuerzos por parte de la comunidad científica para realizar la descripción semántica de los mismos, son mayores que para los servicios del dominio de Telecomunicaciones; por lo tanto, se hablará únicamente de la descripción semántica sobre servicios Web.

Se ha establecido que los servicios web poseen una descripción sintáctica contenida en su descriptor, y una descripción semántica que define para cada operación el significado de los elementos de entradas y salidas, pre-condiciones y pos-condiciones (Sbodio, y otros, 2012). Gran parte de los motores de búsqueda de servicios disponibles en la web, trabajan con base en la comparación de palabras claves, es decir, realizan una búsqueda sintáctica. Este tipo de búsqueda no entrega resultados

muy precisos, ya que no logra capturar efectivamente las funcionalidades de los servicios. La búsqueda basada en palabras claves no permite realizar una comparación semántica entre el parámetro de solicitud del usuario y la información contenida en el descriptor del servicio. Algunos ejemplos de este tipo de buscadores son: *Binding Point, Grand Central*[2]*, Salcentral*[3]*,* y *Web Service List*[4] (Y. Zhang, 2010).

Una de las posibles soluciones para este problema, es la adición de anotaciones semánticas en los descriptores de los servicios para facilitar su búsqueda (Maleshkova, y otros, 2009). El uso de ontologías de dominio específico para anotar semánticamente descriptores de servicios se ha convertido en una de las soluciones más comunes en la mayoría de enfoques que han sido propuestos. Las ontologías son herramientas que permiten la estructuración conceptual de un ámbito del conocimiento específico por medio de vocabularios controlados. Estas herramientas logran proporcionar una descripción lógica y formal que puede ser interpretada tanto por las personas, como por las máquinas, las cuales consiguen diferenciar términos y referenciarlos de manera exacta, facilitando la entrega, localización e integración de recursos a través de la web, para el uso de los mismos en aplicaciones cada vez más distribuidas. (Piraquive, y otros, 2009)

La descripción semántica de un servicio se forma a partir de una anotación semántica que enriquezca la información sobre los parámetros de dicho servicio. En otras palabras, la anotación semántica consiste en asociar conceptos y relaciones de una ontología con parámetros y operaciones de un servicio web. (Hernández, y otros, 2010)

2.2.2.1. Tecnologías para la descripción semántica de servicios web

Tres de las tecnologías más utilizadas para realizar la descripción semántica de servicios web son:

SAWSDL (Semantic Annotations for WSDL and XML Schema): Define mecanismos por medio de los cuales pueden añadirse anotaciones semánticas a los componentes de los descriptores WSDL. SAWSDL se centra en la anotación semántica de la definición abstracta de un servicio. Para el proceso de adición de información semántica en los elementos *WSDL* y *XML Schema*, establece un conjunto de atributos de extensión, los cuales corresponden a atributos que se añaden al descriptor del servicio para hacer referencia a información semántica. El conjunto de atributos se divide en 2 (Akkiraju, y otros, 2007):

- ***ModelRefence*:** Atributo de extensión utilizado para especificar la asociación de tipos complejos en el *XML Schema*, tipos simples, declaraciones de elementos y de atributos con conceptos de algún modelo semántico. *ModelReference* permite:
    - Clasificar o añadir otras descripciones semánticas de la interfaz en el elemento *wsdl:interface.*
    - Agregar información semántica acerca de las operaciones en los elementos *wsdl:opertation* y *wsdl:fault.*
    - Especificar la semántica de los datos de salida y entrada en las operaciones del descriptor, en el elemento *xs:complexType,xs:element*, *xs:simpleType* y *xs:attribute.*

- ***LiftingSchemaMapping* y *LoweringSchemaMapping*:** Atributos de extensión que se añaden a la definición de tipos complejos y simples para especificar el mapeo entre datos semánticos y elementos en *XML Schema*. El valor de estos dos

atributos corresponden a un conjunto de cero o más URIs que hacen referencia a definiciones de mapeo:

- ***liftingSchemaMapping*:** Especifica un aumento de información. Establece la forma como un documento XML se conforma con un modelo semántico determinado. La entrada de la transformación es un elemento XML que tendrá la referencia al mapeo, mientras que la salida serán datos semánticos.
- ***loweringSchemaMapping:*** La entrada de la transformación será un determinado grupo de datos semánticos y a la salida se obtendrá el elemento XML en el que se encontrará la declaración del mapeo. (Akkiraju, y otros, 2007)

WSMO (*Web Service Modeling Ontology*): Ontología que proporciona una especificación basada en el uso de ontologías para los elementos de los servicios web semánticos que WSMO considera como principales, estos son:

- **Ontologías:** Encargadas de proporcionar la terminología y la semántica necesarias para describir un servicio.
- **Servicios Web:** Proporcionan la descripción de las capacidades: precondiciones, pos condiciones, requisitos y efectos, así como también de las interfaces: proceso, grounding del servicio.
- **Objetivos:** Describen las necesidades de un usuario, lo que espera obtener del servicio a buscar.
- **Mediadores:** Encargados de enlazar diferentes componentes de WSMO, con el fin de mejorar la interoperabilidad entre los servicios. Existen cuatro tipos de mediadores:

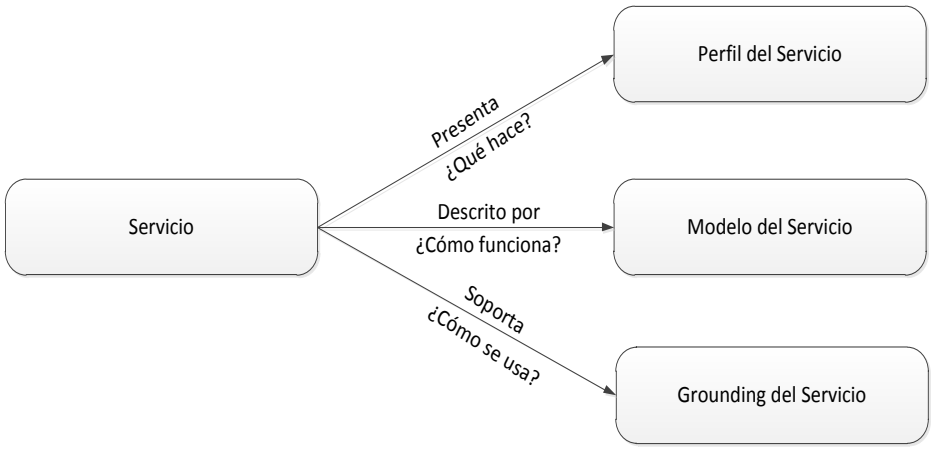
  - **OOMediators:** Importan ontologías relacionadas y solucionan las incompatibilidades que se presenten entre ellas. Permiten la unión entre dos ontologías para resolver posibles errores que puedan ocurrir entre ellas.
  - **GGMediators:** Buscan el perfeccionamiento y compatibilidad de metas (goals), las cuales hacen referencias a diferentes URIs que se utilizan en los descriptores de servicios. GGMediators se utilizan para conectar diferentes metas con el fin de que algunas de ellas definan otras de carácter más general.
  - **WGMediator:** Satisface necesidades entre los servicios web y las metas. Permite la conexión entre metas y servicios web con el fin de resolver errores terminológicos y establecer diferencias funcionales entre ambos.
  - **WWMediator:** Es usado en el proceso de composición de servicios. Permite la conexión de servicios web únicamente con el fin de resolver cualquier dato, protocolo o proceso existente entre los servicios conectados.

    El trabajo de los mediadores inicia una vez se hayan importado las ontologías para el proceso de descripción semántica, a partir de ese momento los mediadores se utilizan para solucionar malas uniones y problemas en general que hayan surgido al realizar las importaciones, un claro ejemplo de estos problemas es la existencia de dos elementos con el mismo nombre, pero pertenecientes a ontologías diferentes.

OWL-S (*Web Ontology Languaje for Services*): Se basa en la definición de varias ontologías representadas en OWL las cuales permiten la descripción de servicios web semánticos en diferentes niveles de abstracción. A diferencia de WSMO, OWL-S

establece como elementos principales para la descripción de servicios web semánticos (ver Figura2) (Hernández, y otros, 2010):

- **Perfil del servicio:** Elemento que explica de forma específica qué hace el servicio.
- **Modelo del servicio:** Este elemento se encarga de definir la manera cómo funciona el servicio
- **Grounding del servicio:** Este último elemento permite comprender la forma cómo se usa el servicio.



Fuente: (Hernández, y otros, 2010)

Figura 2. Ontología de alto nivel de OWL-S

El objetivo principal para realizar anotaciones semánticas en servicios web es facilitar a agentes software la recuperación de servicios, por medio de la comprensión de la semántica de los parámetros funcionales, tales como: Entradas (E), Salidas (S), Precondiciones (P), Efectos (E), y alguna información adicional como el nombre del servicio, el dominio del negocio y el proveedor, y parámetros no funcionales, los cuales describen cómo funciona el servicio y la interacción entre los datos y el flujo de control (Klusch, 2008).

2.2.3. Desambiguación Lingüística

El lenguaje natural contiene un gran número de palabras cuyos sentidos o significados pueden variar enormemente dependiendo del contexto en la que sean utilizadas. Un claro ejemplo de lo anterior es el término *capital*, el cual puede referirse a una población principal y cabeza de un Estado, provincia o distrito, o bien a cierta cantidad de dinero o bienes materiales que se tienen como patrimonio. En la oración *Bogotá es la capital de Colombia*, un lector humano conoce inmediatamente cual es el contexto circundante, y determina que capital se refiere a una ciudad principal y no un número de bienes materiales o dinero. Sin embargo, los sistemas computacionales no cuentan con el extenso conocimiento de los humanos con relación al mundo y el lenguaje. De esta manera, para dichos sistemas, determinar el sentido correcto de una palabra polisémica es un problema difícil. Para abordar dicho problema existe un proceso denominado desambiguación del sentido de las palabras, proveniente del inglés "*Word Sense Disambiguation*" (WSD) (Proposición de un Modelo para la Acentuación Automática de Palabras Ambiguas del Español, Utilizando Etiquedado de Texto, 2010).

La WSD se encarga de asignar el sentido correcto de una palabra con relación al contexto en el que esta se encuentra, entendiendo por contexto un conjunto de

palabras alrededor del término que se desea desambiguar. Esta tarea se puede enmarcar dentro de un conjunto más amplio de técnicas denominado Procesamiento del Lenguaje Natural (PLN), las cuales se ocupan de la formulación y el desarrollo de mecanismos computacionalmente eficaces para la comunicación entre personas o entre personas y máquinas, por medio del lenguaje natural (Leal, y otros, 2009).

La Desambiguación del sentido de las palabras o desambiguación lingüística se considera un proceso necesario para el desarrollo de aplicaciones como (Ide, y otros, 1998) :

- **Traducción automática:** Es preciso el uso de WSD, ya que dependiendo del contexto, la traducción de una palabra varía. Por ejemplo, al traducir rejilla del francés al inglés, esta puede ser traducida como barandilla, puerta, bar, parrilla, escala, u horario.
- **Recuperación de información y navegación hipertextual:** En su gran mayoría, los motores de búsqueda utilizan la comparación de palabras claves para recuperación de información, y se hace necesario el proceso de desambiguación para eliminar los resultados en los que la palabra tiene un sentido inapropiado.
- **Análisis de contenido y temática:** Se analiza la distribución de categorías de palabras predefinidas, es decir, palabras que indican un tema, un concepto, una idea a través del texto. Es importante aplicar WSD para que la categoría que la palabra represente sea acorde con el contenido del texto.
- **Análisis gramático:** La desambiguación del sentido de las palabras es muy útil en el etiquetado de textos, pues muchas de sus frases pueden tener un sentido diferente según como se entiendan las palabras. Por ejemplo, la frase en francés *L'étagère pliesous les livres*, que en español significa *La plataforma se inclina con (el peso de) los libros*, se hace necesario desambiguar la palabra *livres,* ya que puede significar libros o libras (sustantivo masculino en el primer caso, femenino en el segundo), y al no desambiguar correctamente la palabra, el etiquetado podría realizarse como un sustantivo femenino siendo masculino.
- **Procesamiento del discurso:** Es necesario aplicar WSD para la adecuada fonetización de las palabras en la síntesis del discurso. Por ejemplo, la palabra evoca, en inglés *conjure*, es pronunciada de diferente forma en las oraciones *He conjured up animage* y *I conjure you to help me*.
- **Procesamiento de textos:** Se precisa de la desambiguación del sentido de las palabras con el fin de realizar corrección ortográfica en los textos. Por ejemplo, en francés el cambio de *comte* a *comté*, en el primer caso significa contar, en el segundo condado.

2.2.3.1. Clasificación de métodos para WSD

Existe una variedad de métodos empleados para la Desambiguación del sentido de las palabras, actualmente se agrupan en dos categorías (Leal, y otros, 2009), (Grette, 2010):

Métodos basados en conocimiento: Estos métodos utilizan un conocimiento lingüístico previamente adquirido. Aplican técnicas como selección de restricciones, uso de ejemplos para cada significado, traslape en el texto de las definiciones, medidas de similitud semántica y heurísticas. El objetivo de este conjunto de métodos es hacer inferencia semántica usando el conocimiento almacenado en recursos externos como: diccionarios MRD (*Machine Readable Dictionaries*), tesauros (vocabularios controlados que representan las relaciones semánticas con otras palabras y sus significados), textos sin ningún tipo de etiquetado, e incluso recursos de la Web. Los métodos basados en conocimiento permiten la desambiguación de textos

sin importar el tema que traten, gracias a los recursos que estos utilizan, como diccionarios MRD, por ejemplo Longman Dictionary of Contemporary English (LDOCE)[6] o Collins English Dictionary (CED)[7].

Métodos basados en corpus: Los métodos que conforman esta categoría se fundamentan en el uso de técnicas estadísticas y de aprendizaje automático con el fin de inducir modelos del lenguaje a partir de grandes conjuntos de ejemplos textuales. El contexto de la instancia de una palabra se trata de emparejar con información acerca del contexto de otras instancias de la misma palabra, desambiguadas anteriormente, derivados del corpus. Por corpus se entiende una colección de textos, en los cuales se abordan uno o varios temas. El propósito de un corpus es servir de fuente de datos, proporcionando ejemplos de oraciones y ejemplos de uso de varias palabras, para procesarlos con algoritmos de aprendizaje automático.

Esta categoría se subdivide en dos:

- **Métodos supervisados (Corpus etiquetado):** En este tipo de métodos se emplean corpus en los que cada ocurrencia de una palabra ambigua ha sido anotada manualmente con el sentido correcto, de acuerdo con los sentidos obtenidos de un diccionario. Por lo tanto, el corpus anotado sirve como material de entrenamiento para el algoritmo de aprendizaje, el cual se encarga de entregar un modelo aprendido automáticamente que puede ser usado para el proceso de desambiguación.

  Existe un gran número de algoritmos supervisados, se destacan entre ellos los que han provisto mejores resultados, tales como: clasificadores bayesianos, redes neuronales, modelos gráficos, perceptrones, árboles, listas de decisión, entre otros. (R. Mihalcea, 2004)

- **Métodos no supervisados (Corpus no etiquetado):** Los métodos contenidos en esta categoría identifican patrones en conjuntos de datos que no proporcionan mucha información, como corpus no etiquetados que tratan sobre temas específicos. Los patrones se usan para dividir los datos en grupos según unas características en específico. La hipótesis de este enfoque es: "palabras con significados similares tienden a tener contextos similares". Por lo tanto, se realizan agrupaciones de palabras con base a la similitud de su contexto; para esto se hace uso de técnicas de clustering, haciendo innecesario el uso de herramientas de soporte como diccionarios, tesauros o anotaciones manuales.

Se considera a los métodos supervisados como los más exitosos para la desambiguación del sentido de las palabras. Sin embargo, tienen un problema denominado "Obstáculo de adquisición de conocimientos". Los corpus utilizados en los métodos supervisados, necesitan una gran cantidad de ejemplos para cada sentido de la palabra, haciendo que los métodos supervisados sean imprácticos al desambiguar todas las palabras en un texto. (Grette, 2010)

La precisión con la que funcionan los métodos supervisados se debe en gran parte al uso de corpus anotados manualmente. No obstante, la formulación de inventarios de sentidos por parte de los lexicógrafos se ha convertido en una actividad cada vez más costosa, hecho por el cual los métodos no supervisados han recibido mayor atención en los últimos años, ya que obtienen el sentido de la palabra directamente desde el corpus en uso. Pese a esta ventaja, la obtención de resultados puede ser muy pobre,

[6] http://www.ldoceonline.com/
[7] http://www.collinslanguage.com/

debido a que la desambiguación se hace a partir de conjuntos de sentidos no muy bien formados. (Grette, 2010)

Por otra parte, a pesar de presentar mayor complejidad en comparación con los métodos basados en corpus, los métodos basados en conocimiento permiten mayor cobertura en cuanto a dominios, lo que permite que puedan ser aplicados a cualquier tipo de texto. (Pérez, 2009)

2.3. TRABAJOS RELACIONADOS

En torno a la temática de anotación semántica de servicios y la desambiguación lingüística para el etiquetado de información se han desarrollado diferentes trabajos de investigación. A continuación se citan algunos trabajos que definen sus objetivos parcialmente similares a los definidos de este trabajo, se expone de manera general el enfoque planteado, los resultados obtenidos, los aportes y desventajas de cada uno de ellos frente al enfoque planteado de este proyecto.

2.3.1. Anotación semántica de servicios del dominio web y del dominio de las Telecomunicaciones.

La investigación en el área de Web Semántica señala que la anotación de recursos con metadatos puede ayudar a resolver el problema de búsquedas ineficientes basadas en palabras clave. Este concepto de la anotación se puede extender a los servicios para obtener la anotación semántica de servicios, que tiene como fin facilitar el descubrimiento, ejecución, composición e invocación de servicios distribuidos en Internet. Un Servicio descrito semánticamente según (Hernández, y otros, 2010), está formado por una descripción sintáctica y una anotación semántica. La anotación semántica consiste en asociar conceptos y relaciones de una ontología con parámetros y operaciones de un servicio. A continuación se presentan algunas aproximaciones desde las cuales éste concepto ha sido aplicado.

- En (R. Battle, 2008), se propone el uso de una herramienta creada con el fin de facilitar la integración de servicios web existentes dentro de la web semántica. La herramienta se ha denominado SBWS (*Semantic Bridge for Web Services*), la cual trabaja alrededor de un conjunto de operaciones de servicio descritas empleando WADL. Para el proceso de enriquecimiento semántico de los descriptores de servicio, la herramienta provee anotaciones personalizadas para la entrada y salida de los métodos de los servicios REST. Para iniciar el proceso de anotación de determinado servicio el usuario debe proporcionar la ontología que será utilizada. Las anotaciones realizadas por SBWS se encargan de adaptar los descriptores de este tipo de servicios a consultas semánticas. Aunque en (R. Battle, 2008) se propone la anotación de descriptores WADL considerando así servicios REST, en el enfoque planteado en el presente trabajo se consideran servicios tanto REST como SOAP. Por otra parte una desventaja del método presentado en (R. Battle, 2008) frente al que se propone en este proyecto es que las ontologías necesarias para realizar las anotaciones no se buscan automáticamente en la web y el usuario debe preocuparse por proporcionarlas.

- En (Y. Chabeb, 2009), se define un enfoque denominado YASA4WSDL (Yet Another Semantic Annotation for WSDL), que facilita el proceso de adición de componentes semánticos en los descriptores de los servicios web. Esta propuesta permite describir las propiedades semánticas y sintácticas tanto funcionales como no funcionales de los descriptores WSDL mediante el uso de dos tipos de

Ontologías: *Ontologías Técnicas* -que contienen los conceptos que describen atributos funcionales de los servicios, por ejemplo entradas y salidas, y conceptos que describen atributos no funcionales, como calidad del servicio y el contexto de la información-, y *Ontologías de Dominio* -que contienen conceptos que definen la semántica del dominio del negocio del servicio, por ejemplo turismo, salud, etc. Para contener las referencias a los conceptos de las ontologías con las que se está realizando la anotación, este enfoque propone un nuevo atributo denominado "serviceConcept". El primer atributo "*serviceConcept*" contiene un conjunto de URIs (Uniform Resource Identifier) que referencia los conceptos técnicos del descriptor a una o más ontologías técnicas; por su parte, el segundo atributo "serviceConcept" contiene un conjunto de URIs que referencia el/los concepto(s) de dominio del servicio en una o más ontologías de dominio. En YASA4WSDL el proceso de anotación de descriptores de servicios puede realizarse manualmente o a través de una interfaz gráfica que le permite al usuario importar ontologías. El proceso de anotación que se propone en YASA4WSDL diferencia entre las propiedades sintácticas como semánticas de los descriptores de servicio lo que permite realizar una anotación más exacta de cada atributo. Sin embargo, entre las desventajas frente al proceso definido en el presente trabajo son: el proceso de anotación no es automático; tanto los atributos del descriptor a anotar como los conceptos ontológicos necesarios para realizar las anotaciones no se identifican automáticamente, siendo necesaria la intervención del usuario. Por otra parte las ontologías necesarias para realizar las anotaciones no se buscan automáticamente en la web y el usuario debe preocuparse por proporcionarlas.

- En (Ferreira, y otros, 2009), se propone una ontología denominada *RESTfulGrounding*, la cual es una extensión de *OWL-S* (Ontología construida en Lenguaje de Ontologías Web OWL). Esta ontología delega descripciones sintácticas de servicios a los documentos WADL. La ontología *RESTfulGrounding* soporta el mapeo directo, uno a uno, de manera automática entre los parámetros OWL-S y WADL y se basa en protocolos bien definidos. Su tarea de enriquecimiento semántico se realiza con el fin de crear servicios web semánticos de baja complejidad. El proceso realizado con la ontología propuesta permite detallar el acceso a un servicio REST dado, principalmente en lo relacionado con protocolos, formato de mensajes y transporte, socialización de los recursos y direccionalidad. En el enfoque presentado en (Ferreira, y otros, 2009) se anotan descriptores WADL considerando así servicios REST. Sin embargo no se consideran servicios SOAP.

- En (Canturk, y otros, 2011), se propone un método para la anotación semántica de servicios web basados en lo que los autores de este trabajo denominan "*lexicon-based alignment*", el cual considera los diferentes sentidos de una palabra y de esta manera encuentra la asociación entre el servicio y la ontología. La ontología es construida a partir del nombre del servicio y de los nombres de los parámetros de entrada y salida del servicio. Estos nombres son extraídos literalmente, sometidos a un módulo de tokenización y eliminación de palabras que no aporten semántica sobre el servicio, con el fin de obtener las palabras más relevantes del descriptor. Luego, cada palabra final se colocan como un elemento de nivel raíz dentro en un esquema RDF. Posteriormente, con la ayuda de la base de datos léxica *WordNet* (Princeton University) se obtienen sinónimos de cada palabra y sentidos para cada uno de los sinónimos encontrados, se comparan estos sentidos para encontrar un grado de similitud entre ellos y así seleccionar las palabras que harán parte de la ontología. En el enfoque planteado en (Canturk, y otros, 2011) es importante resaltar el proceso de identificación de palabras relevantes, pues es un paso que mejora notablemente la calidad de la

ontología creada. Sin embargo, el hecho de crear la ontología y no hacer uso de ontologías ya existentes, evaluadas, consensuadas y puestas a disposición para su uso, constituye una desventaja frente al enfoque planteado en este trabajo, el cual usa ontologías disponibles en la web.

- En (Aksoy, y otros, 2011), se expone una herramienta denominada *MATAWS* (*Multimodal Automatic Tool for the Annotation of Web Services*), la cual permite realizar un proceso totalmente automático de anotación semántica de grandes colecciones de descriptores WSDL y entregar a su salida una serie de archivos OWL-S. Dentro del proceso de anotación se tiene primeramente un módulo de extracción de parámetros de operación definidos en la colección de archivos WSDL, se recuperan los nombres de los parámetros, los nombres de tipo y de las estructuras de tipo (en el caso de tipos complejos). Estos nombres se descomponen, normalizan y limpian para que puedan ser tratados por el módulo llamado Asociador, el cual se basa en el motor de inferencia Sigma (Pease, 2004) para asociar un concepto ontológico a una palabra. Después que todos los parámetros de un servicio Web se han anotado, un componente de salida genera un archivo OWL-S con información contenida en el archivo original WSDL y los conceptos ontológicos seleccionados. De esta manera, al procesar todos los descriptores WSDL se obtendrá como resultado una lista de archivos OWL-S. Al igual que en el anterior trabajo citado, en (Aksoy, y otros, 2011) también se realiza un proceso de selección de palabras relevantes del descriptor, lo que permite realizar una anotación de mejor calidad. Por otra parte, la herramienta propuesta permite anotar grandes colecciones de descriptores de servicio, y es importante resaltar que el proceso de anotación es totalmente automático. Entre las desventajas que presenta el enfoque planteado en (Aksoy, y otros, 2011) frente al planteado en este trabajo es el proceso de anotación como tal, pues está basado en el motor de inferencia Sigma que considera una palabra individualmente para buscar un concepto ontológico que coincida literalmente sin tener en cuenta la relación semántica con contexto lingüístico del descriptor, como si lo hace el proceso que se plantea en este trabajo.

- En (H. Yoo, 2011), se discute un enfoque acerca de la adición de semántica en descriptores de servicio, usando una aplicación que rastrea información semántica para los atributos del descriptor utilizando el motor de búsqueda *SeekDa* y/o *Programmable Web.* La propuesta documentada en este trabajo ofrece una interfaz de usuario en la cual se define el servicio a anotar, y se escoge entre el motor de búsqueda *SeekDa* o *Programmable Web*. Posteriormente, la aplicación procede a hacer el rastreo de la información semántica para anotar automáticamente el mayor número posible de atributos (funcionales y no funcionales). Los atributos restantes deben anotarse manualmente. Una vez terminado este proceso, el usuario puede publicar el descriptor final en el registro del servicio para que pueda ser descubierto y utilizado en composición. El proceso planteado en (H. Yoo, 2011) es parcialmente automático y hace uso de ontologías disponibles que se buscan a través de motores de búsqueda. Por otra parte, a diferencia del enfoque propuesto en el presente trabajo, la búsqueda de conceptos ontológicos que se propone en (H. Yoo, 2011), se realiza explorando coincidencias sintácticas con los atributos del descriptor sin tener en cuenta la relación semántica que los asocie.

- En (Bouchiha, y otros, 2012), se propone un enfoque para la anotación semiautomática de descriptores de servicios WSDL. Se presenta una herramienta en la cual el usuario ingresa el descriptor WSDL y un conjunto de ontologías. La herramienta ejecuta dos procesos principales para determinar qué ontologías aportan semántica al descriptor WSDL: *Categorización* y *Comparación*. El proceso

de *Categorización* tiene como objetivo clasificar el descriptor WSDL a su dominio correspondiente. Para este fin, el descriptor del servicio se descompone en sus elementos fundamentales (XSD tipos de datos, interfaz, operaciones y mensajes), extrayéndose al mismo tiempo una lista de los conceptos de cada ontología. Luego, se obtiene un grado de similitud entre los elementos extraídos del descriptor y los elementos extraídos de cada ontología para determinar qué conceptos ontológicos se mantendrán para el siguiente proceso. La ontología seleccionada indica el dominio o categoría del WSDL. Por su parte, el proceso de *Comparación* tiene como objetivo asignar elementos de WSDL a conceptos ontológicos. En dicho proceso se calcula la similitud semántica entre elementos del documento WSDL y conceptos de la ontología de dominio. Cada elemento del documento WSDL se anota con el concepto ontológico más similar. Aunque en (Bouchiha, y otros, 2012) se realiza un análisis a nivel semántico entre los atributos del descriptor y los conceptos ontológicos que mejoran la calidad de la anotación, el proceso presenta desventajas como: la no automatización total del proceso, y la búsqueda por parte del usuario de las ontologías necesarias para realizar la anotación.

- En (Falleri, et al., 2010) se presenta un método que extrae automáticamente las etiquetas de descripción de servicios Web. Mediante cinco pasos apoyados por técnicas de minería de texto se identifican un conjunto de términos relevantes extraídos de un descriptor del servicio y deja a los usuarios la tarea de asignar las anotaciones de forma manual. Aunque en (Falleri, et al., 2010) se realiza una eficaz identificación de los términos relevantes de un descriptor de servicio se requiere de un componente manual para llevar a cabo la anotación de servicios.

Análisis

Los trabajos de anotación semántica de servicios presentados anteriormente no son los únicos desarrollados en relación con esta temática; sin embargo se consideran los más oportunos teniendo en cuenta el significativo grado de similitud de su enfoque respecto al que se plantea en este proyecto y los objetivos definidos.

En general los trabajos aquí expuestos, abordan factores puntuales dentro del tema de la anotación de servicios. El objetivo en (Aksoy, y otros, 2011), por ejemplo, se centra en la automatización del proceso, en (Canturk, y otros, 2011) y (Bouchiha, y otros, 2012) su propósito está ligado a asegurar una alta coherencia semántica entre el servicio y los componentes ontológicos. Por otra parte, cada trabajo plantea soluciones para descriptores de servicio específicos WSDL o WADL. Con el fin de descubrir los servicios adecuados entre los diversos servicios disponibles y componerlos entre ellos, es necesaria la anotación tanto de servicios SOAP como REST a través de una misma herramienta.

En trabajos como (R. Battle, 2008), (Aksoy, y otros, 2011) y (H. Yoo, 2011), se puede observar que la búsqueda de ontologías se hace teniendo en cuenta el significado individual de cada elemento extraído del descriptor de servicio y no en relación al contexto formado por todos los elementos extraídos, lo cual puede causar problemas de ambigüedad, considerando las diferentes interpretaciones que podría tener una palabra de acuerdo con el contexto donde se encuentre.

Teniendo en cuenta que dentro del enfoque que se propone en este proyecto para la anotación semántica de servicios web y de telecomunicaciones, es necesario una identificación clara de los elementos a ser anotados dentro del descriptor de servicio, se optó por aprovechar el enfoque presentado en (Canturk, y otros, 2011), para la identificación de las palabras más relevantes de un descriptor de servicio. La simple

extracción literal de los elementos de un descriptor puede provocar en la mayoría de los casos la obtención de palabras que no tengan ningún aporte semántico sobre el servicio. En este sentido, la tokenización de los elementos extraídos y la eliminación de palabras con bajo aporte semántico sobre el servicio disminuye en un alto grado el problema de trabajar con términos que ni siquiera estén definidos dentro del lenguaje natural, o palabras que sean demasiado frecuentes para ser significativas. Por esta razón se decidió que es adecuado aplicar el enfoque de extracción de atributos de un descriptor WSDL propuesto en (Falleri, et al., 2010), aunque realizando algunos ajustes y mejoras. En el trabajo se propone a través de pasos de extracción, toquenización, categorización gramatical y eliminación de palabras irrelevantes, la identificación de los atributos relevantes semánticamente dentro de un descriptor WSDL.

2.3.2. Desambiguación lingüística para etiquetado de información

La necesidad de etiquetar corpus de información radica en la imposibilidad de las máquinas de entender su contexto, complicando así, tareas relativamente sencillas como la traducción de textos, o procesos un poco más complicados, como la recuperación de información. Para el proceso de etiquetado, se hace necesaria una primera actividad, que permita que el etiquetado se haga de manera más eficiente y posiblemente automática; esta primera actividad es la desambiguación lingüística, la cual se considera como una actividad necesaria para el desarrollo de otras actividades, más que como una actividad con un fin en sí misma. (Ide, y otros, 1998)

Existe una variedad de enfoques para el etiquetado de corpus, su principal diferencia reside en el método de desambiguación lingüística que se utilice para hallar el sentido correcto de las palabras, pues existe una gran variedad, como se muestra en la sección 2.2.3.1. A continuación se presentan los trabajos relacionados más representativos dentro de este enfoque:

- En (Mezquita, 2011), se propone un método de desambiguación en el cual son utilizados grandes recursos léxicos tales como tesauros, diferentes tipos de diccionarios morfológicos, diccionarios explicativos, diccionarios de sinónimo, entre otros, con el fin de mejorar la actividad de recuperación de información. El método descrito realiza una normalización morfológica de las palabras del documento a etiquetar, es decir, sustituye cada palabra por su forma normalizada; por ejemplo, los términos *trabajo*, *trabaja*, *trabaje*, se sustituyen por *trabajar*. Luego, se eliminan palabras auxiliares del documento, como preposiciones, artículos, verbos auxiliares y pronombres, lo que permite realizar la desambiguación con respecto a un número limitado de palabras. Para cada palabra con varios sentidos se calculan y se suman valores para cada sentido utilizando diferentes recursos como el algoritmo de Lesk, al cual se le asigna un peso de 1, diccionario de sinónimos con un peso de 0.9, y para otras relaciones léxicas un peso de 0.8. Finalmente la palabra se etiqueta con el sentido que tenga una sumatoria de valores óptimos. A pesar de que este enfoque proporciona un proceso de desambiguación similar al requerido para el presente trabajo, en la medida en que se encuentra basado en diccionarios MRD, tiene como desventaja el hecho de que es mucho más complejo que otros procesos de desambiguación existentes, ya que para hallar el sentido correcto de una sola palabra es necesario realizar cálculos haciendo uso de diferentes recursos, los cuales deben estar disponibles y ser completamente funcionales al momento de la desambiguación. Además, a diferencia del presente trabajo de grado, el proceso de desambiguación es realizado sin tener en cuenta el contexto en el cual se encuentra contenida la palabra.

- El trabajo descrito en (Alonso, Junio 2010), propone un método de desambiguación del sentido de las palabras que logra combinar las ventajas de los métodos supervisados y no supervisados de WSD (*Word Sense Disambiguation*). El método usa información léxica adicional, es decir corpus en los que cada ocurrencia de una palabra ambigua ha sido anotada manualmente con el sentido correcto, como lo hacen los métodos supervisados. No obstante, en lugar de soportarse en corpus externos, lo que conlleva un mayor costo computacional, emplea una traducción a otro(s) idioma(s) del texto que recibe como entrada. Con lo anterior se pretende alcanzar resultados semejantes a los obtenidos con métodos supervisados, pero a costo de los no supervisados. El objetivo de este trabajo es utilizar un lenguaje para desambiguar otro. Se utiliza la traducción del texto que se desee desambiguar para facilitar el proceso de etiquetado, ya que de esta manera es posible encontrar intersecciones en los sentidos de las palabras para el conjunto de lenguajes. El proceso de desambiguación que se propone en este trabajo consume pocos recursos al igual que los métodos no supervisados, permitiendo que el costo computacional sea bajo. Sin embargo, presenta una gran desventaja: la desambiguación se realiza a través de la comparación de cada una de las palabras con su correspondiente traducción de manera individual y no según su contexto lingüístico.

- En (Izquierdo, 2010), se expone un sistema para la desambiguación del sentido de las palabras que utiliza un método de aprendizaje supervisado, y una arquitectura basada en clases semánticas, la cual permite la creación de un clasificador para cada clase semántica contenida en un texto. Por clase semántica puede entenderse cualquier grupo de conceptos con coherencia semántica. El método se basa en la premisa de que, en gran parte, la terminología humana puede agruparse bajo cierto número de clases semánticas. De la misma manera, establece dos ideas importantes: la primera, que palabras iguales que aparecen en contextos diferentes suelen hacer parte de clases semánticas diferentes, y la segunda que sentidos diferentes de una misma palabra por lo general pertenecen a clases semánticas distintas. Por lo tanto, la elección del sentido más adecuado para una palabra se realiza con base en grupos de clases semánticas en lugar de discriminar sentidos palabra por palabra. El sistema para la desambiguación propuesto en este trabajo presenta un bajo costo computacional, ya que se encuentra basado en un método de aprendizaje supervisado que lo permite. Sin embargo, las clases semánticas que utiliza para realizar la desambiguación, se limitan al número de corpus externos de los cuales el método de aprendizaje obtenga información. Otra de las desventajas de este enfoque es el hecho de que la desambiguación se realiza con relación a clases semánticas establecidas, y no con relación al contexto en el que las palabras se encuentren.

- El trabajo presentado en (Word sense disambiguation with multilingual features, 2011) expone un método basado en las ventajas que ofrece un espacio multilingüe, extrayendo las características de varios idiomas para generar una base de representación más sólida y eficaz para llevar a cabo la desambiguación del sentido de una palabra. Este método, identifica la palabra a desambiguar y su contexto en inglés, para luego traducir el contexto a varios idiomas con lo que se aprovecha la característica de un motor de traducción automática, para traducir la palabra ambigua de acuerdo a las palabras que la acompañen. El método propuesto toma como base 3 idiomas (francés, alemán y español). De esta manera el contexto original en inglés se complementa con tres contextos más (francés, alemán y español). A continuación, se generan vectores que se componen de la palabra original en inglés, seguida de la parte multilingüe según la representación formal $C(3, k)$, donde $k = 0…3$. Así por ejemplo, un vector resultante de $C(3,0)$ es el vector monolingüe tradicional, mientras que un vector

construido a partir de la combinación *C(3,3)* contiene las características extraídas de todos los idiomas. A pesar de que el proceso de desambiguación propuesto en este trabajo tiene en cuenta el contexto que acompaña a la palabra ambigua, presenta un costo computacional alto, pues hace uso de un motor de traducción. Además, su conocimiento se encuentra limitado a las traducciones en 3 idiomas de las palabras a desambiguar.

- En (T. Pedersen, 2009) se presenta un trabajo en el cual se propone la adaptación de un algoritmo WSD clásico, para realizar el proceso de desambiguación: el *Algoritmo de Lesk.* Este algoritmo recibe como entrada un texto de ejemplo en el cual se establece una única palabra como palabra a desambiguar (palabra objetivo). Luego, obtiene de *WordNet* un sentido para la palabra objetivo, basado en la información léxica que encuentre de ella y de las palabras que se encuentran a su alrededor. Este conjunto de palabras alrededor de la palabra objetivo se denominan *contexto lingüístico*, definido como una ventana de *n* palabras símbolo a la derecha y *n* palabras símbolo a la izquierda, para un total de *2n* palabras circundantes. La palabra objetivo se considera como parte del contexto también, así el tamaño total de este será *2n+1*. El algoritmo se encarga de comparar glosas (definiciones) de los sentidos, entre cada par de palabras en la ventana del contexto. Si existen *N* palabras en la ventana del contexto, entonces habrán *N(N-1)/2* parejas de palabras para ser comparadas. Una vez realizadas todas las comparaciones para cada par de parejas, se suman todos los puntajes individuales, con el fin de obtener el sentido correcto de una palabra.

  A diferencia de la mayoría de los enfoques encargados de realizar desambiguación del sentido de una palabra, encargándose de buscar superposiciones entre las glosas de las palabras vecinas, el enfoque propuesto en (T. Pedersen, 2009) extiende la comparación para incluir las glosas de los términos relacionados con las palabras que están contenidas en el texto a desambiguar. El sentido asignado a una palabra objetivo se establecerá entonces según el contexto en el que esta se encuentre.

Análisis:

En esta sección, se ha presentado un conjunto de mecanismos para la anotación de corpus de información con base en diferentes métodos para la desambiguación del sentido de las palabras. Se encontró que los enfoques pueden variar considerablemente en cuanto a calidad de resultados, ya que algunos de ellos determinan el sentido de una palabra ambigua sin considerar el contexto en el que se encuentre, además del grado de complejidad computacional que pueden presentar debido a la cantidad y tipo de recursos léxicos utilizados, tal como se observa en los enfoques propuestos por (Word sense disambiguation with multilingual features, 2011) y (Alonso, Junio 2010). Sin embargo, dadas las características del presente proyecto, es de vital importancia la calidad en la asignación del sentido correcto de una palabra, por encima de la complejidad computacional, que esta conlleve. Por lo tanto, se consideró que el enfoque más adecuado para el proceso de desambiguación lingüística, es el que a pesar de su complejidad computacional considere en su totalidad el contexto del corpus a desambiguar. Esto se definió considerando que la anotación semántica de los descriptores debe ser realizada de acuerdo al dominio del servicio para que sea precisa y pertinente. Consecuentemente, la desambiguación lingüística de los descriptores debe realizarse teniendo en cuenta el contexto lingüístico del servicio en su totalidad (M. Maleshkova, 2010) (Vitvar, 2008).

Dentro de este enfoque, se analizó un conjunto de posibilidades que permiten la elección del sentido más adecuado para una palabra, y su posterior etiquetado. En

este conjunto se ha encontrado el mecanismo que considera la superposición de glosas, no solo de las palabras vecinas al término a desambiguar, sino también de los términos relacionados a las palabras contenidas en el texto (T. Pedersen, 2009). Adicionalmente, permite que el sentido asignado a la palabra objetivo sea obtenido con respecto al contexto del corpus en el cual se encuentre contenida. Además, este método se encuentra basado en conocimiento, lo que permite que la desambiguación de textos se realice sin ningún tipo de restricciones en cuanto al dominio que estos traten.

2.4. RESUMEN

Este capítulo, corresponde a una contextualización en torno a la temática de la anotación semántica de servicios convergentes y la temática de la desambiguación lingüística. Se comenzó por definir los conceptos fundamentales necesarios para comprender el campo de aplicación del presente proyecto, para posteriormente, hacer un recorrido sobre los principales trabajos de investigación relacionados con esta propuesta.

La documentación con respecto a la anotación semántica de servicios se ha realizado entorno a documentos descriptores WSDL y WADL, enfocándonos en los primeros ya que existen iniciativas, a nivel de la industria de las telecomunicaciones, que promueven la descripción de los servicios empleando interfaces WSDL. (Smith, 2009) (EMA-323, 2004) (ECMA-348, 2004)

Finalmente, se realizó una selección de los enfoques que contribuyen a la propuesta presentada, y se identificaron métodos para la realización de anotaciones semánticas en los documentos descriptores y procesos para la desambiguación del sentido de las palabras, los cuales constituyen el punto de partida del presente proyecto, siendo adaptados y aplicados, para la automatización del proceso de anotación semántica de descriptores con base a su contexto lingüístico, tal como se describe en los siguientes capítulos.

3. DESAMBIGUACIÓN LINGÜÍSTICA Y DESCRIPTORES DE SERVICIOS

3.1. INTRODUCCIÓN

En este capítulo, se hace énfasis en las técnicas empleadas para realizar la desambiguación lingüística de un descriptor de servicio. Inicialmente, se presenta el estudio de los diferentes descriptores de servicios tanto del dominio web como del dominio de las telecomunicaciones, destacando la estructura de los documentos de cada modelo. Con base en este estudio se selecciona uno de los modelos con el cual se trabajara en el marco de este proyecto. Posteriormente, se contextualiza sobre los métodos de desambiguación basados en conocimiento, y se elige el más adecuado para llevar a cabo la desambiguación del sentido de las palabras extraídas tanto del descriptor de servicio como de las ontologías. Finalmente, se define la metodología que permite realizar la desambiguación lingüística del descriptor de servicio.

3.2. DESCRIPCIÓN DE SERVICIOS WEB Y DEL DOMINIO TELCO

Los servicios Web y Telco han desempeñado, y sin duda seguirán haciéndolo, un papel importante para el desarrollo de sistemas distribuidos, basados en componentes dentro y entre las empresas. Los servicios se consideran diferentes de los productos debido a su naturaleza intangible. La mayoría de los productos se pueden describir con base en sus propiedades físicas notables, tales como el tamaño, color, peso etc. Por otro, lado los servicios carecen de características concretas. Así, los servicios a menudo se deben definir indirectamente en términos de los efectos que tienen sobre los consumidores, por lo tanto los servicios desplegados sobre internet solo pueden alcanzar su máximo potencial cuando se pueden describir en forma tal que permita su publicación, descubrimiento, selección, contratación y seguimiento (A Service Description Language for the Internet of Services, 2009), de esta manera nacen los descriptores de servicio.

Como se mencionó en la sección 2.2.1 la creación de servicios convergentes se basa en la reutilización de servicios existentes desplegados sobre diferentes plataformas de red. Con este fin, tanto en el dominio Web, como en el dominio Telco se definen diferentes modelos de arquitectura que promueven la composición de servicios. Arquitecturas como: REST (REpresentational State Transfer) y SOA (service-oriented architecture) permiten componer aplicaciones Web más complejas. Al mismo tiempo, IMS (IP Multimedia Subsystem) y SIP (Session Initiation Protocol) permiten que la composición de funcionalidades en el dominio de las telecomunicaciones sea posible. A continuación, se presenta una breve descripción de los lenguajes más conocidos, y ampliamente utilizados para la descripción de servicios Web y Telco.

3.2.1. Descripción de servicios web

El mundo de los servicios de la web es ampliamente dominado por el estándar SOA y WSDL. Sin embargo, un paradigma orientado a los recursos y basado en la utilización de los métodos HTTP de manera explícita, conocido como *REST*, ha tenido bastante auge en los últimos años, tanto así que grandes proveedores de la Web 2.0 están migrando a esta tecnología (Hadley, 2009), incluyendo a Yahoo, Google y Facebook entre otros. *RESTful*, se caracteriza por su relativa simplicidad y su exclusivo uso para la Web (Investigating Web APIs on the World Wide Web, 2010).

3.2.1.1. WSDL (Web Service Description Language)

WSDL es un lenguaje basado en XML (Christensen, y otros, 2001) que permite crear una descripción sintáctica de servicios web (Discovering Semantic Web services using SPARQL and intelligent agents, 2012).

WSDL es una recomendación del grupo W3C (World Wide Web Consortium), y se ha convertido en el estándar para la definición de interfaces de servicios Web. En SOA, WSDL se utiliza para describir las interfaces de todos los servicios con independencia de la tecnología subyacente. En junio de 2007, el W3C publico la recomendación WSDL 2.0 (W3C, 2007), en la cual se ha incorporado las mejoras para WSDL 1.1.

WSDL 1.1 describe un servicio web mediante seis componentes principales (Clustering WSDL Documents to Bootstrap the Discovery of Web Services, 2010):

- *<types>* es un elemento definido en XML que describe los tipos de datos utilizados en el intercambio de mensajes, pueden ser tan simple como un único

elemento o tan complejo como un conjunto de elementos. Cada *type* tiene un atributo *"name"* y un atributo "*type"*.

- <*messages*> este elemento es una representación abstracta de la información transmitida. Típicamente, un mensaje contiene una o más partes lógicas (parámetros), como un mensaje que contiene una orden de compra con los artículos de la orden y su factura. Cada parte lógica se define por el esquema correspondiente como un tipo simple XML, un tipo complejo o un elemento, definidos en la sección <*types*>.

- <*portType*> es un componente importante en los documentos WSDL, en este se definen la combinación y secuencia de mensajes para cada una de las operaciones abstractas (funciones) que pueden ser ejecutadas por el servicio web. Cada operación se define mediante un mensaje de entrada, otro de salida y opcionalmente un mensaje de error.

- <*binding*> este componente especifica el protocolo de comunicación y formato de los datos de cada operación y mensaje definido en un elemento *portType* particular.

- <*service*> este elemento es una operación compuesta que agrega múltiples puertos relacionados o funciones.

- <*port*> define un terminal mediante la especificación de una única dirección para un binding.

WSDL 1.1 soporta cuatro tipos de patrones de intercambio de mensajes:

- *One-way*: El servicio recibe un mensaje único de entrada.
- Request-response: El servicio recibe un mensaje de solicitud y responde con un mensaje de salida.
- *Solicit-response*: El servicio envía primero un mensaje de salida y espera un mensaje de entrada como respuesta.
- *Notification*: El servicio envía un mensaje de salida sin esperar ninguna respuesta.

WSDL 2.0 ha simplificado la descripción de la interfaz de un servicio web, define un lenguaje para describir la funcionalidad abstracta de un servicio, así como un marco para describir los detalles concretos del mismo (Chinnici, y otros, 2007). La interfaz se define dentro de la sección <*interface*>. WSDL 2.0 ha prescindido del concepto de mensajes de entrada y salida, especificando elementos XML de entrada y de salida directamente en las operaciones. Los mensajes de falla se pueden especificar en la sección <*interface*>, lo que permite que el mismo mensaje se utilice en diferentes operaciones, simplificando el manejo de errores. Los mensajes se describen independientes de un formato de conexión específico, utilizando una estructura típicamente XML. WSDL 2.0 ha introducido los patrones de intercambio de mensajes, que especifican la secuencia en la que los mensajes asociados han de transmitirse entre el servicio y el cliente. WSDL 2.0 también permite la herencia de interfaces. Los componentes <*binding*> y <*service*> no han cambiado significativamente en esta especificación del lenguaje.

Debido a la alta popularidad del modelo REST, dentro del desarrollo de la recomendación WSDL 2.0 se ha contemplado la necesidad de la compatibilidad con HTTP en las descripciones de aplicaciones Web. Por tanto, los descriptores WSDL 2.0 ofrecen una compatibilidad absoluta con HTTP y SOAP, lo que lo hace muy útil tanto

para aplicaciones Web sencillas, como para aplicaciones de servicios Web que requieran de funcionalidades adicionales.

3.2.1.2. WADL (Web Application Description Language)

WADL es un lenguaje basado en XML, diseñado para permitir la descripción de aplicaciones Web basadas en HTTP (Hadley, 2009). Este lenguaje se crea a partir de la necesidad de describir algunas características comunes de todas las aplicaciones Web como: recursos disponibles, relaciones entre los recursos, métodos HTTP que se pueden aplicar a cada recurso y formatos admitidos de representación de recursos. WADL define un lenguaje que puede ser procesado por una máquina, y facilita procesos como: soporte para el desarrollo de herramientas de modelado y manipulación de recursos, generación automática de código para la manipulación de las representaciones de los recursos, y la configuración de cliente y servidor utilizando un formato portable.

Un documento WADL está compuesto por los siguientes elementos (Marc J. Hadley, 2005):

- <*doc*>: un descriptor WADL puede tener cero o más elementos de este tipo, y a su vez cada elemento <*doc*> puede contener cero o más elementos de tipo <*doc*>. Cada elemento <*doc*> cuenta con los siguientes atributos:
    - *xml:lang*: define el idioma para el valor del atributo *title*
    - *title*: contiene una descripción textual del elemento que se está documentando

- <*grammars*>: es un elemento opcional y contiene la definición de formato de los datos de intercambio durante la ejecución del protocolo especificado por el descriptor WADL. Las definiciones de los formatos se pueden incluir por medio de elementos <*include*>.
    - <*include*>: se pueden incluir uno o más de estos elementos dentro del elemento <*grammars*>. Por medio del atributo *href* se hace referencia al recurso que contiene las definiciones de formato.

- <*resources*>: el descriptor WADL puede tener cero o más elementos de este tipo. Este elemento contiene los recursos que provee la aplicación, también cuenta con un atributo base que indica la URL de la aplicación. Cada recurso es indicado por el elemento <*resource*>.
    - <*resource*>: indica un recurso que provee la aplicación, cuenta con siguientes atributos:
        - *id*: Identifica el elemento
        - *path*: Si está presente, se proporciona una plantilla URI relativa para el identificador del recurso
        - *type*: Atributo opcional cuyo tipo es una lista separada por espacios. Cada valor de la lista es una referencia cruzada que identifica un elemento <*type_resource*> que define un conjunto de métodos admitidos por el recurso.
        - *queryType*: Define el tipo de comunicación para el componente de consulta del recurso URI.

Un elemento <*resource*> contiene cero o más elementos de tipo: <*doc*>, <*param*>, <*metod*>, <*resource*>.

- <*param*>: un elemento de este tipo describe un componente parametrizado de su elemento padre. Un elemento <*param*> contiene cero o más elementos <*doc*>, cero o más elementos <*option*>, y opcionalmente un elemento <*link*>.

- <*option*>: define uno de un conjunto de posibles valores para el parámetro representado por su elemento <*param*> padre.

- <*link*>: se utiliza para identificar los vínculos a los recursos dentro de las representaciones.

- <*method*>: contiene cero o más elementos de tipo <*doc*>, un elemento <*request*> y cero o más elementos <*response*>. A través de este elemento se describe para cada recurso de la aplicación tanto la entrada como la salida de cada uno de los métodos del protocolo HTTP (HEAD, GET, POST, PUT, DELETE, TRACE, OPTIONS Y CONNECT) que son aplicables al recurso.

    - <*request*> parámetros de entrada al método.
    - <*response*> los posibles resultados del método.

- <representation> este elemento describe una representación del estado de un recurso.

3.2.2. Descripción de servicios de Telecomunicaciones

En el dominio de las telecomunicaciones no hay un lenguaje o marco de trabajo estándar para el modelado de servicios. Sin embargo, con la aparición de los conceptos de NGN e IMS; tecnologías que brindan la posibilidad de comunicar servicios de telecomunicaciones con servicios de internet, a partir de los lenguajes para la descripción de servicios web pueden construirse descriptores de servicios de telecomunicaciones, desentendiéndose de los mecanismos de comunicación subyacentes. Dentro de los esfuerzos más representativos para el modelado y descripción de servicios del dominio de las telecomunicaciones están Parlay X, que soporta servicios SOAP, y realiza la descripción de servicios de telecomunicaciones mediante descriptores de servicios WSDL (Jie, y otros, 2009). Sin embargo, el GSMA (*GSM Association*) (GSMA, 2012) en colaboración con la OMA (*Open Mobile Alliance*) (Alliance, 2012) han revisado las especificaciones de Parlay X, para desarrollar la iniciativa OneAPI, un conjunto de APIs mucho más fácil de usar para los desarrolladores web. Las mejoras introducidas mediante esta iniciativa incluyen enlaces RESTful, así como una selección de respuestas de JSON y XML, entre otras, lo cual le permite soportar tanto servicios SOAP como REST.

El conjunto de APIs OneAPI está basado en estándares Web que permiten exponer las capacidades de red del operador para ser accedidas a través de HTTP. Así, cualquier operador de red o proveedor de servicios es capaz de implementar OneAPI para exponer capacidades como SMS, MMS, servicios de localización, pago, entre otros. (Smith, 2009).

De acuerdo con lo anterior, tanto los servicios del dominio web, como los servicios del dominio de las telecomunicaciones pueden especificarse mediante un documento WSDL, teniendo en cuenta que la versión 2.0 de este lenguaje soporta la descripción de servicios REST y SOAP. Por lo tanto, para el desarrollo del presente trabajo de

grado, se ha decidido trabajar con los descriptores de servicio WSDL 1.1 y WSDL 2.0, cubriendo de esta manera la descripción de servicios Web SOAP, REST y de Telecomunicaciones.

3.3. MÉTODOS DE DESAMBIGUACIÓN BASADOS EN CONOCIMIENTO

Los métodos de desambiguación basados en conocimiento emplean un conocimiento lingüístico adquirido con anterioridad, contenido en recursos externos como: diccionarios, tesauros, textos sin etiquetado, entre otros. Entre estos recursos, los más utilizados por este tipo de métodos son los diccionarios MDR (*Machine Readable Dictionaries*) como *WordNet*. (Leal, y otros, 2009)

Según lo mencionado en la sección 2.2.3.1, la complejidad computacional de los métodos basados en conocimiento, en comparación con los métodos basados en corpus, es un poco más alta. No obstante, pueden aplicarse a cualquier tipo de texto y no se restringen únicamente a textos cuyas palabras estén contenidas en corpus anotados (Pérez, 2009). Debido a las necesidades del presente proyecto, el método de desambiguación a utilizar debe permitir la realización de este proceso sin importar el tema que trate el texto de entrada, ya que la anotación semántica de descriptores de servicios Web y de Telecomunicaciones se realizara independientemente de su dominio. Considerando lo anterior, a continuación se exponen diferentes métodos de desambiguación basados en conocimiento.

3.3.1. Método de Lesk

El método de Lesk se basa en una medida de solapamiento contextual entre las definiciones de un diccionario, para identificar los sentidos de las palabras en un contexto determinado. Éste método emplea uno de los primeros algoritmos desarrollados para la desambiguación del sentido de todas las palabras en un texto: el *Algoritmo de Lesk* (Ríos, 2008)

3.3.1.1. Algoritmo de Lesk

En 1986, Michael E. Lesk (Lesk, 1986) creó una base de conocimiento, la cual tiene como único recurso requerido, un conjunto de definiciones de sentidos por cada palabra a desambiguar y sus palabras vecinas. De esta manera, logra un conocimiento del contexto en donde se desambiguará la palabra. El algoritmo determina el sentido más adecuado para una palabra ambigua solapando las definiciones de los sentidos que la palabra pueda tener, con las definiciones de los sentidos de las palabras vecinas, es decir el contexto que rodea a la palabra ambigua. De esta manera, el sentido que tenga un mayor número de coincidencias entre su definición y las definiciones de los sentidos de sus palabras contiguas, será considerado el sentido correcto para la palabra que se está desambiguando.

Formalmente, para dos palabras W_1 y W_2, cada una con sus sentidos respectivos N_{w1} y N_{w2} obtenidos a partir de un diccionario, se establece que para cada par de sentidos se debe calcular el número de palabras que tienen en común realizando solapamiento de sus correspondientes definiciones. Luego, se selecciona el par de sentidos con el mayor solapamiento, para posteriormente asignar un solo sentido a cada palabra del par inicial. A continuación se muestran el procedimiento definido mediante el algoritmo de Lesk (Algoritmo 1):

Algoritmo 1. Algoritmo de Lesk

(1) Para cada sentido i de W_1
(2) Para cada sentido j de W_2
(3) Calculo del número de palabras en común entre las definiciones de los sentidos (i, j). (Solapamiento)
(4) Encontrar i y j tales que el solapamiento (i, j) sea el máximo.
(5) Asignar el sentido i a W_1 y el sentido j a W_2

Para ilustrar la operación de este algoritmo, considere un ejemplo en el cual se desambiguan las palabras "*pine*", en español pino, y "*cone*", en español cono. La entrada al algoritmo corresponde al conjunto "*pine cone*". Se obtienen, en primer lugar las definiciones de los sentidos, tanto de "*pine*" como de "*cone*", tarea realizada empleando el diccionario *Oxford Advanced Learner's*[8], disponible en internet, según el cual se definen cuatro sentidos para "*pine*": (a) "*Tipos de árboles de hojas perenes con forma de agujas*" (b) "*pino*" (c) "*Desgastarse por dolor o enfermedad*" (d) "*fatigarse por algo, fatigarse por hacer algo*", mientras que para "*cone*" define tres sentidos: (a) "*muchacho fuerte que se angosta en un punto*" (b) "*algo de esta forma, ya sea sólido o hueco*" (c) "*frutos de ciertos árboles de hojas perennes (abeto, pino)*". Posteriormente, se realiza el solapamiento entre las definiciones de cada sentido de "*pine*" y cada sentido de "*cone*" (Tabla 1). El paso siguiente consiste en seleccionar los sentidos que presenten mayor solapamiento. La primera definición de "*pine*" y la tercera de "*cone*" cumplen con esta característica, debido a que en sus definiciones cuentan con dos palabras en común: "*evergreen*" y "*tree*", por lo tanto estos dos son los sentidos seleccionados por el algoritmo de Lesk.

Tabla 1. Valores de solapamiento según la ontología de dominio

Solapamiento de sentidos	Valor solapamiento
$\boldsymbol{Pine\#1 \cap Cone\#1}$	0
$\boldsymbol{Pine\#2 \cap Cone\#1}$	0
$\boldsymbol{Pine\#3 \cap Cone\#1}$	0
$\boldsymbol{Pine\#4 \cap Cone\#1}$	0
$\boldsymbol{Pine\#1 \cap Cone\#2}$	0
$\boldsymbol{Pine\#2 \cap Cone\#2}$	0
$\boldsymbol{Pine\#3 \cap Cone\#2}$	1
$\boldsymbol{Pine\#4 \cap Cone\#2}$	0
$\boldsymbol{Pine\#1 \cap Cone\#3}$	2
$\boldsymbol{Pine\#2 \cap Cone\#3}$	1
$\boldsymbol{Pine\#3 \cap Cone\#3}$	0
$\boldsymbol{Pine\#4 \cap Cone\#3}$	1

A lo largo del tiempo el Algoritmo de Lesk ha sido modificado, adquiriendo algunas variaciones que se explican a continuación:

- **Simulated Annealing:** Aplicar el Algoritmo de Lesk original para desambiguar las palabras de un texto puede convertirse en una tarea muy tediosa: si se considera una oración sencilla de nueve palabras, cada una de ellas con sus correspondientes sentidos (palabra1 (26), palabra2 (11), palabra3 (4), palabra4

[8] http://oald8.oxfordlearnersdictionaries.com/

(8), palabra5 (5), palabra6 (4), palabra7 (10), palabra8 (8), palabra9 (3)), puede obtenerse un total de 43929600 combinaciones.

Para solucionar este inconveniente, se ha propuesto una modificación al Algoritmo de Lesk, en la cual se define una función E que representa la combinación correcta de sentidos (el solapamiento entre las definiciones de los sentidos es máxima), o incorrecta de sentidos (el solapamiento entre las definiciones de los sentidos es mínima). Cuando la función E toma su valor mínimo corresponde a la combinación correcta de sentidos. Dada una combinación de sentidos, se procede a extraer las definiciones correspondientes de un diccionario. Posteriormente se suma el número de ocurrencias de cada palabra en estas definiciones, asignándole a cada una de ellas ese valor. Luego se unen estos valores y se obtiene así la "redundancia" del texto. De esta manera, la función E se define como la inversa de la redundancia, y ya que un valor mínimo de la función representa la combinación correcta de sentidos, el objetivo del algoritmo será buscar la combinación de sentidos cuyo valor de redundancia sea lo suficientemente alto como para lograr el objetivo. Para la combinación inicial de sentidos se toman los más frecuentes para cada palabra, y se realizan iteraciones con los sentidos restantes, en las cuales el sentido de una palabra en la combinación se reemplaza sucesivamente con otro sentido, recalculando el valor de la función E, y determinando si la nueva combinación es correcta o no, con base en la disminución o aumento del valor de la función con respecto a las demás combinaciones. Las iteraciones finalizan cuando ya no hay más combinaciones de sentidos que puedan realizarse. (Cowie, y otros, 1992).

- **Algoritmo de Lesk simplificado:** Esta modificación al Algoritmo de Lesk trata de solucionar el mismo problema que la variante *Simulated Annealing*: la explosión combinatoria que resulta de intentar desambiguar más de dos palabras con el Algoritmo de Lesk original. La propuesta de este algoritmo simplificado es determinar el significado de cada palabra en un texto de manera individual. El objetivo es encontrar el sentido que presente el mayor solapamiento entre las definiciones de un diccionario y el contexto actual (palabras que rodean a la palabra a desambiguar). Por lo tanto, para el proceso de desambiguación se tienen en cuenta las palabras contiguas como tal, y no la definición de sus sentidos (Vasilescu, y otros, 2004). A continuación se muestran los principales procesos del Algoritmo de Lesk simplificado (Algoritmo 2):

Algoritmo 2. Algoritmo de Lesk simplificado

(1) Para cada sentido i de W
(2) Calculo del número de palabras en común entre la definición del sentido i y el contexto donde aparece la palabra. (Solapamiento)
(3) Encontrar el sentido i con el mayor solapamiento.
(4) Asignar el sentido i a W

Por ejemplo, para desambiguar la palabra "*pino*" en la oración "*Conos de pino cuelgan de árboles*", se obtienen en primer lugar las definiciones de los sentidos de "pine": (a) "*Tipos de árboles de hojas perenes con forma de agujas*" (b) "*Desgastarse por dolor o enfermedad*". Posteriormente, se calcula para cada definición, el número de palabras que tiene en común con la oración que contiene la palabra a desambiguar. Como puede observarse la primera

definición contiene una palabra en común con la oración, mientras que la segunda definición no contiene ninguna. Según lo anterior, primer sentido es el asignado a la palabra “pino”.

Espacios semánticos aumentados: Este enfoque propone la desambiguación de una palabra mediante el uso de las definiciones tanto de la palabra ambigua como de palabras relacionadas con esta. Empleando la jerarquía de *WordNet* (hiperónimos, hipónimos, holónimos, merónimos, tropónimos), se obtiene una lista de conceptos relacionados y sus correspondientes definiciones, construyendo así un contexto más amplio para la desambiguación. Se realiza combinación de sentidos y se trata de identificar la combinación que lleva al valor de solapamiento más alto. Este método presupone la comparación de glosas (definiciones de los sentidos) de las palabras a ser desambiguadas con las glosas de palabras que se encuentran conectadas a ellas por medio de varias relaciones semánticas. Se ha demostrado que este enfoque, propuesto por (Banerjee, y otros, 2002), tiene un gran desempeño cuando las palabras que recibe como entrada se encuentran altamente vinculadas (Vasilescu, y otros, 2004).

3.3.2. Método de Similitud semántica

Este método se basa en la premisa de que las palabras en un texto deben encontrarse relacionadas en sus sentidos con el fin de obtener coherencia en el discurso. Por lo tanto, se puede afirmar que las palabras que se encuentran contenidas en un contexto similar están relacionadas, permitiendo la selección de sus sentidos correctos a partir de la distancia semántica. Sin embargo, a pesar de que la premisa es una propiedad del lenguaje natural, se encuentra restringida a un grupo limitado de palabras del contexto más cercano a la palabra a desambiguar, o a un grupo pequeño de las palabras relacionadas sintácticamente. Este método extrae el contexto local sin introducir información contextual adicional. (Pérez, 2009)

Este tipo de métodos son afectados por el mismo problema que el Algoritmo de Lesk original, al tratar de desambiguar más de dos palabras. Para solucionar este problema pueden aplicarse las mismas soluciones propuestas para Lesk.

3.3.2.1. Medidas de similitud semántica

Las medidas de similitud semántica se utilizan en la cuantificación del grado en el que dos palaras se relacionan semánticamente. A continuación se presentan algunas medidas de similitud aplicadas sobre la jerarquía de *WordNet*, para su funcionamiento reciben como entrada un par de conceptos y entregan como salida el grado de similitud semántica entre ambas palabras:

- La primera de las medidas a presentar es la introducida en (Jiang, y otros, 1997). En esta propuesta, la fórmula presentada (Ecuación 1) para el cálculo del grado de similitud semántica, emplea la diferencia que existe en el contenido de información de los dos conceptos que recibe como entrada. En la ecuación 1, C_1, C_2 representan los conceptos 1 y 2 recibidos como entrada, $IC(C)$ se define como su probabilidad de ocurrencia en un corpus de gran tamaño, mientras que *LCS* (*Lowest Common Subsumer*), es el primer concepto de la red semántica que contiene a los dos conceptos (C_1, C_2), en otras palabras, es el nodo común para el cual se encuentra definido un camino entre los dos conceptos.

$$Similitud(C_1, C_2) = 2\ x\ IC\big(LCS(C_1, C_2)\big) - (IC(C_1) + IC(C_2)) \qquad \textbf{Ec. 1.}$$

- Otra propuesta para la medida de similitud es la expuesta en (Using corpus statistics and WordNet relations for sense identification, 1998). Este enfoque propone la normalización del valor que se obtiene como salida, al determinar el camino mínimo entre los dos conceptos que recibe como entrada. Siendo C_1 el concepto 1, C_2 el concepto 2 y D la profundidad, la ecuación 2 representa el número de arcos en la red semántica que deben ser atravesados para llegar del concepto 1 al concepto 2, es decir, entrega el valor de la longitud del camino que conecta ambos conceptos:

$$Similitud(C_1, C_2) = -\log\left(\frac{Camino(C_1,C_2)}{2D}\right) \quad \textbf{Ec. 2.}$$

- El siguiente enfoque para la medida de similitud semántica es el propuesto en (Hirst, 1998), en el cual a la medida de similitud le añaden la dirección de los enlaces que forman el camino. La razón para realizar esto es la premisa de que, además de la longitud, el camino no debería cambiar de dirección frecuentemente (cambios entre los arcos de la red semántica). Siendo C y k constantes, C_1 y C_2 los conceptos a los cuales se les calculara el valor de similitud semántica y d el número de cambios de dirección, la ecuación 3 expresa la medida de similitud para este enfoque:

$$Similitud(C_1, C_2) = C - Camino(C_1, C_2) - kd \quad \textbf{Ec. 3.}$$

- Hasta ahora, las medidas de similitud presentadas pueden aplicarse exclusivamente a conceptos que se encuentran conectados de manera explícita por algún arco en la red semántica. Sin embargo, en (A Method for word sense disambiguation of unrestricted text, 1999) se propone un método que permite medir el grado de similitud semántica entre jerarquías independientes, pertenezcan o no a la misma categoría léxica, a través de la creación de caminos virtuales entre diferentes jerarquías por medio de las definiciones de las glosas de *WordNet*. En la ecuación 4, $|CD_{12}|$ representa el número de palabras comunes a las definiciones en la jerarquía de los conceptos 1 y 2. Por otra parte, W_k determina la profundidad de cada concepto dentro de la jerarquía, como un peso que se asocia con cada concepto. Finalmente, $descendientes(C_2)$ representa el número de conceptos en la jerarquía del concepto 2.

$$Similitud(C_1, C_2) = \frac{\sum_{k=1}^{|CD_{12}|} W_k}{\log(descendientes(C_2))} \quad \textbf{Ec. 4.}$$

- La última de las medidas de similitud presentada, es la propuesta por Wu &Palmer (WUP) (Wu, y otros, 1994). Esta propuesta se encuentra basada en la estructura y contenido de *WordNet*, lo cual le permite establecer relaciones de similitud entre conceptos. Esta medida de similitud semántica mide la profundidad de dos conceptos en la taxonomía de *WordNet*, al igual que la profundidad de *LCS*, que como se dijo anteriormente, representa el primer concepto de la red semántica que contiene a los dos conceptos, y combina estos valores en un valor de similitud. La medida WUP se encuentra definida por la ecuación 5, donde $N1$ y $N2$ se definen como la distancia que existe entre los conceptos 1 y 2 que se desean desambiguar, respecto al concepto raíz que contiene a ambos, y N se define como la distancia entre el nodo antecesor, más cercano a los conceptos 1 y 2, y el nodo raíz.

$$Sim_{wp} = \frac{2N}{N1+N2} \quad \textbf{Ec. 5.}$$

3.3.3. Método de Preferencias de Selección

Para la desambiguación del sentido correcto de las palabras, el método de preferencias de selección realiza una captura de información acerca de las posibles relaciones entre clases semánticas, de esta manera logra restringir los posibles sentidos que las palabras puedan tener en un contexto determinado. Por ejemplo, *comer-comida*, *beber-líquidos*, es una muestra clara de la forma en que se asocian los sentidos a categorías de palabras que se han construido con base en el sentido común. En la oración "*Peter drunk burgundy*", *burgundy* no puede ser traducido como un color, ya que el verbo que acompaña la palabra requiere que el sentido de esta se encuentre relacionado con un líquido (Ríos, 2008).

3.3.3.1. Adquisición de preferencias de selección

Una de las opciones existentes para obtener preferencias de selección es el uso de corpus anotados semánticamente, sobre los cuales sea posible aplicar técnicas estadísticas que permitan establecer relaciones entre palabras, como la que se presenta a continuación (Pérez, 2009). La ecuación 6 (*Contador de frecuencia*), permite determinar el número de veces que una palabra (W_1) co-ocurre con otra palabra (W_2), mediante la relación R.

$$Cont\_frec(W_1, W_2, R) \qquad \textbf{Ec. 6.}$$

Por su parte, la ecuación 7, haciendo uso del contador de frecuencia, determina la probabilidad de que la relación R aparezca entre las dos palabras (W_1, W_2), con respecto a la probabilidad de que la relación aparezca entre una de las dos palabras y el corpus en su totalidad.

$$P(W_1|W_2, R) = \frac{Cont\_frec(W_1,W_2,R)}{Cont\_frec(W_2,R)} \qquad \textbf{Ec. 7.}$$

Otro de los algoritmos existentes para la adquisición de preferencias de selección es el propuesto por (Agirre, y otros, 2001), en el cual se muestra el uso de relaciones clase-a-clase. Un ejemplo de esto es la relación clase a clase existente entre "*ingerir comida*" y "*comer carne"*, las clases a las que pertenecen se encuentran semánticamente relacionadas, restringiendo los posibles sentidos que puedan ser asignados a las palabras.

3.3.4. Método de Heurísticas para WSD

Este método aprende, a través de textos, heurísticas basadas en propiedades lingüísticas. Tres de las heurísticas más utilizadas son:

- Sentido más frecuente
- Sentido por discurso
- Sentido por colocación

Las dos últimas heurísticas se basan en la suposición de que una palabra siempre tiene el mismo sentido en un caso en particular, un mismo discurso o una misma colocación. Finalmente, existe una cuarta heurística que establece "un sentido por dominio". A continuación una breve explicación de las heurísticas expresadas anteriormente (Pérez, 2009).

- **Sentido más frecuente:** Normalmente, de los posibles sentidos que una palabra puede tomar, existe uno que se presenta con mayor frecuencia. En un

proceso sencillo y rápido de desambiguación, se establecería el sentido frecuente como el sentido correcto para la palabra ambigua.

De acuerdo al corpus *Semcor*[9], creado por la universidad de Princeton y obtenido a partir de las últimas versiones de *WordNet*, el corpus *Brown*[10] y la novela *The Red Badge of Courage*[11]. El sentido que se presenta con mayor frecuencia para todas las categorías léxicas (nombre, verbo, adjetivo, adverbio) es el primero. Sin embargo, existe un inconveniente en el desarrollo de este método: no siempre se dispone de la distribución de las ocurrencias de los sentidos en todos los lenguajes, limitando la aplicación del método a lenguajes en los que exista una cantidad suficiente de textos de los cuales puedan ser extraídas las distribuciones. Lo anterior se debe a que un cambio en el dominio general de un texto altera la distribución que puedan tener los sentidos, por lo tanto la eficiencia del método disminuye notablemente.

- **Sentido por discurso:** Esta heurística permite que el sentido correcto de una palabra y el de todas sus ocurrencias en un mismo texto sea determinado identificándolo una sola vez. Establece que las palabras tienden a preservar su sentido a lo largo de un texto. Sin embargo, palabras polisémicas que presentan sentidos similares pueden tener más de un sentido dentro de un mismo texto, es por esto que la heurística "*sentido por discurso*", solo funciona bien en el caso que se aplique en textos en los cuales las palabras tengan sentidos bien diferenciados, pues aquellos textos en los que los sentidos de las palabras presentan diferencias sutiles pueden generar error en el momento de la desambiguación. Esta heurística fue introducida por (Gale, 1992).

- **Sentido por colocación:** El sentido correcto de una palabra ambigua se establece con relación a sus palabras adyacentes. Esta heurística propone que independientemente del contexto, palabras ubicadas en la misma colocación tendrán el mismo sentido. Por ejemplo, la palabra "*plant*", en la colocación "*industrial plant*" tendrá el mismo sentido para todas sus ocurrencias en cualquier texto. El sentido para una palabra se mantiene dependiendo de la distancia entre las palabras a su alrededor. Al igual que en el caso de la heurística anterior, los resultados con la heurística de *sentido por colocación* son erróneos al utilizarla con palabras cuyos sentidos presentan diferencias sutiles (Martínez, y otros, 2000).

3.3.5. Selección del método para desambiguación lingüística

Uno de los principales procesos en el marco de este trabajo es la desambiguación lingüística de los descriptores de servicio. Con el fin de realizar anotaciones semánticas teniendo en cuenta, no solo el atributo a etiquetar, sino también el dominio del servicio, la desambiguación de los atributos debe ser realizada con relación al contexto lingüístico del servicio (M. Maleshkova, 2010) (Vitvar, 2008). En virtud de lo anterior, y a pesar del buen funcionamiento y baja complejidad computacional que otros métodos de desambiguación puedan presentar, los métodos basados en conocimiento poseen una amplia cobertura que permite que sean aplicados a la desambiguación de cualquier tipo de texto, sin limitarse a la desambiguación de textos cuyos temas hayan sido aprendidos con anterioridad a partir de corpus anotados manualmente (Pérez, 2009). Realizando un análisis de los métodos basados en conocimiento, se ha encontrado que la variación del algoritmo de Lesk: Espacios

[9] Corpus disponible en http://www.cse.unt.edu/~rada/downloads.html#semcor
[10] Corpus disponible en http://icame.uib.no/brown/bcm.html
[11] (Crane, 2002)

semánticos aumentados, alcanza una precisión global del 32% en comparación al 16% y 23% de precisión alcanzados por las otras variaciones del algoritmo de Lesk (Banerjee, y otros, 2002). Además, gracias al proceso que realiza para la desambiguación, el método Espacios semánticos aumentados, a diferencia de los demás métodos presentados en esta sección, permite encontrar el sentido de las palabras con relación al contexto en el que estas se encuentran, ya que tiene en cuenta tanto el término ambiguo y sus vecinos, como las palabras relacionadas con la palabra objetivo. Por esta razón, y según lo que se concluyó en el análisis de la sección 2.3.2, se seleccionó este enfoque, que permite la desambiguación de cualquier texto independientemente de su dominio, y tiene en cuenta el contexto en el cual se encuentra la palabra ambigua.

3.4. DESAMBIGUACIÓN LINGÜÍSTICA DE DESCRIPTORES WEB Y TELCO

El proceso para la desambiguación lingüista de descriptores se divide en dos procedimientos importantes. El primero de ellos consiste en realizar una extracción de los atributos que se desean anotar semánticamente. Este procedimiento se compone de procesos secundarios como la eliminación de palabras cuyo significado es irrelevante para la desambiguación, la clasificación léxica de las palabras a desambiguar, entre otros que serán explicados más adelante en esta misma sección. El segundo de los procesos corresponde a la desambiguación de los sentidos de los atributos del descriptor con relación a su contexto. Para esto en la sección anterior se ha seleccionado un método de *WSD* basado en conocimiento, que permite la asignación del sentido correcto a una palabra ambigua teniendo en cuenta el sentido global del contexto en el que se encuentra: *Método espacios semánticos aumentados*.

3.4.1. Mecanismo para la desambiguación semántica de descriptores Web y Telecomunicaciones.

A continuación, se presenta una descripción detallada de los dos procesos en los que se divide la desambiguación lingüística de descriptores: extracción y desambiguación de atributos, basados en los trabajos de investigación mencionados en el capítulo 2.

3.4.1.1. Extracción de atributos a anotar semánticamente

Este módulo tiene como objetivo identificar automáticamente las palabras relevantes de un descriptor de servicio WSDL. Para esto, con base en varias técnicas de minería de texto se extraen literalmente las palabras que tengan un aporte semántico significativo en la descripción del servicio. A partir de los métodos propuestos en (Falleri, et al., 2010) y agregando algunas mejoras para ajustar los resultados a nuestros requerimientos se obtienen las palabras relevantes del descriptor de servicio que pasaran a ser desambiguadas.

Como se explica en la sección 3.2.1.1 un archivo WSDL se compone de seis tipos de elementos. El primer paso de la operación de este módulo consiste en identificar un conjunto de parejas (*tipo*, *identificador*), donde el campo *tipo* indica el elemento al cual pertenece la pareja (*Service*, *Port*, *PortType*, *Message*, *Type*, ó *Binding*) y el campo *identificador* un atributo extraído del documento WSDL. En la Figura 3 se resume este proceso.

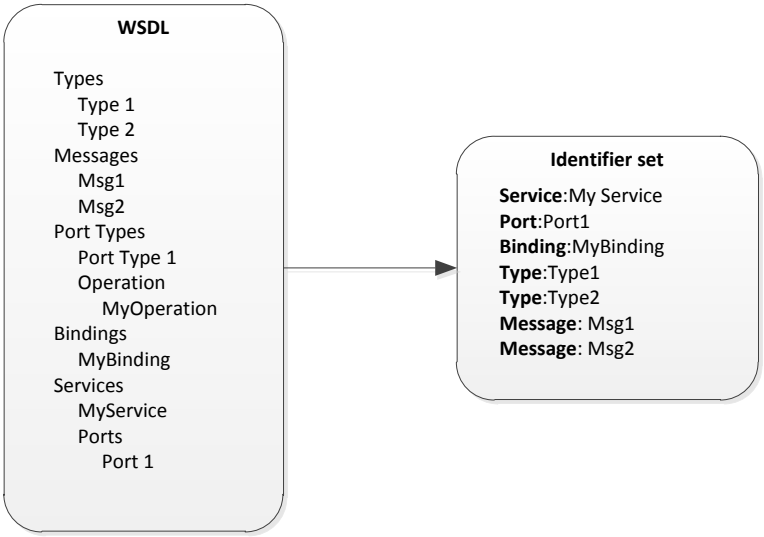


Figura 3. Identificador de conjunto de parejas

Aunque ya se tienen extraídos, y clasificados los atributos del descriptor de servicio, estos datos son por lo general palabras compuestas, por ejemplo: *MyWeatherService*. Además, contienen una gran cantidad de palabras y/o caracteres irrelevantes que no tienen ningún aporte semántico sobre la descripción del servicio. Por este motivo, el proceso de extracción de los atributos relevantes se soporta en un conjunto de técnicas de minería de texto, descritas a continuación:

1. *Filtro de Tipos de elementos*: En la ejecución de este proceso, se descartan las parejas *(tipo, identificador)* que correspondan a elementos *PortType*, *Message* o *Binding*. Lo anterior se justifica en que generalmente, los identificadores de los elementos de estas categorías se duplican a partir de los identificadores en las categorías restantes.

2. *Tokenización*: En este proceso, cada pareja *(tipo, identificador)* se sustituye por una pareja *(tipo, tokens).* Donde *tokens* es el conjunto de palabras que aparecen en el campo *identificador*. Por ejemplo, la pareja *(Servicio, MyWeatherService)* se sustituye por *(Service, [my, weather, service])*.

3. *Etiquetado POS (Part of Speech)*: en esta fase, cada par *(tipo, tokens)* obtenido previamente, se sustituye por un *par (tipo, ptokens). ptokens* es un conjunto de parejas de la forma [*(token$_1$, posición$_1$), (token$_2$, posición$_2$),… (token$_i$, posición$_i$),... (token$_n$, posición$_n$)*], donde para cada pareja, el elemento *token* es la palabra obtenida en el paso anterior y *posición* por su parte es la categoría gramatical correspondiente. Los posibles valores *POS* para una palabra son:

   NN: sustantivo
   NNS: sustantivo plural
   NP: sustantivo propio
   PP: pronombre
   NPS: sustantivo propio plural
   JJ: adjetivo
   JJS: adjetivo plural
   VV: verbo
   VVG: verbo gerundio
   VVD: verbo pretérito

SYM: símbolo

Teniendo en cuenta nuevamente el ejemplo mencionado, la pareja *(Service, [my, weather, service])* se sustituye por *(Service, [(my, PP), (weather, NN), (service, NN)])*.

4. *Eliminación de palabras vacías:* En este paso, se eliminan los *token* irrelevantes. Un *token* irrelevante es un término demasiado frecuente para ser significativa. Se ha establecido manualmente una lista de palabras vacías para cada tipo de elemento del descriptor WSDL como se indica a continuación:

   Service: *service*
   Port: *port*, *response*, *request*
   Type: *type*

5. *Filtro POS*: Durante este paso, se eliminan los conjuntos *(token, posición)* donde *posición* sea diferente de *(NN, JJ, VV, SYM).* Luego, continuando con el ejemplo, la pareja *(Service, [(my, PP), (weather, NN)])* se sustituye por *(Service, [(weather, NN)]).*

Finalmente después de todo el proceso como salida se obtiene una lista de palabras con su respectivo tipo de elemento al que pertenece. Según el ejemplo entregaría *(weather, Service).*

3.4.1.2. Desambiguación de los atributos del descriptor del servicio

El propósito de este proceso es asignar el sentido más adecuado a cada uno de los atributos recibidos a través del proceso previo de extracción. Este sentido asignado debe cumplir una característica especial: corresponder con el dominio del servicio, en otras palabras, la desambiguación de los atributos debe realizarse teniendo en cuenta el contexto del descriptor.

Como se mencionó al inicio de esta sección, y en la sección 3.3.5, el método seleccionado para el proceso de desambiguación del sentido de las palabras es el denominado *Espacios semánticos aumentados*, ya que, siendo un método basado en conocimiento, permite la desambiguación con relación al contexto en el que se encuentra contenida la palabra ambigua, y presenta una mayor precisión en la asignación del sentido adecuado, con relación a los demás métodos basados en conocimiento.

Para ejecutar el proceso de desambiguación, se ha configurado una herramienta libre denominada *WordNet*::SenseRelate::AllWords (SR-AW) (T. Pedersen, 2009), desarrollada en Perl, lenguaje de programación libre creado por Larry Wall. Perl es un lenguaje práctico y sencillo, basado en características de otros lenguajes como C, AWK, sed, Lips, entre otros. Adicionalmente, Perl es un lenguaje ampliamente adoptado por ser portable y robusto y por su capacidad para el procesamiento de textos, archivos y expresiones regulares. (Parlante, 2002)

El funcionamiento básico de SR-AW, se basa en el uso del conocimiento disponible en *WordNet* para la asignación del sentido más adecuado a cada palabra contenida en un corpus con relación al contexto. Para esto emplea uno de los diez métodos de desambiguación lingüística disponibles en la herramienta, procesa el corpus que recibe oración por oración y en cada una de ellas analiza cada palabra. Para el proceso de desambiguación, SR-AW permite el establecimiento de una ventana de contexto que puede ser configurada por el usuario, la cual representa el número de

palabras a desambiguar. Las palabras dentro de la ventana de contexto son desambiguadas por medio del cálculo de similitud semántica, el cual depende del método de desambiguación que se seleccione dentro de las diez posibilidades.

En general, la herramienta realiza el cálculo de similitud a los posibles sentidos de la palabra que se encuentre en el centro de la ventana, con los posibles sentidos de cada una de las palabras contenidas en la ventana de contexto, formando parejas. El sentido de la palabra en el centro de la ventana, que contenga el valor más alto después de haber calculado todos los puntajes en las parejas que se formaron, será considerado como el sentido apropiado para esa palabra. Una vez obtenido el sentido, SR-AW mueve el centro de la ventana de contexto hacia la derecha para repetir el proceso con una nueva palabra. (T. Pedersen, 2009)

El proceso de desambiguación de SR-AW se basa en la estimación de la similitud entre los sentidos de la palabra a desambiguar y los sentidos de las palabras que forman su contexto (palabras a su alrededor). Esta medida de similitud se obtiene calculando el número de palabras que se superponen en la definición de los sentidos de las palabras. Así por ejemplo, suponga que se tiene la oración "*Two tax revisión bills were passed"*, que la ventana de contexto es de tamaño cinco y se encuentra formada de la siguiente manera: [*tax revision bills were passed*]. Para la palabra *bills*, las palabras *tax*, *revision*, *were* y *passed*, forman su contexto.

El puntaje de similitud de un sentido S_i, será calculado por la ecuación 8, en donde $S_{j,k}$ corresponde al *k-esimo* sentido de la *j-esima* palabra (Kikas, y otros, 2007):

$$\text{score}_{S_i} = \sum_{j,k} max_k \; relatedness(S_i, S_{j,k}) \qquad \textbf{Ec. 8.}$$

El algoritmo 3 describe el proceso de desambiguación realizado por SR-AW. En él, cada uno de los atributos a anotar se denota como *token*. El *token* a ser desambiguado se denomina *token objetivo* (*target token*). De esta manera, en primer lugar, el puntaje de cada sentido del *target token* se iguala a cero (*score_i=0*) y se determina que mientras un *token* es definido como *target token*, los demás se conviertan en contexto de este. Para obtener el sentido más adecuado para cada uno de los atributos, el algoritmo estima la relación semántica entre el sentido s_{ti} del *target token*, y cada sentido s_{jk}, para cada t*oken* t_j se calcula la relación $s_{ti} \times s_{jk} \rightarrow R$, donde s_{ti} and s_{jk} representan dos sentidos de una pareja de *tokens* en la ventana de contexto, y *R* representa el conjunto de números reales. Es decir, la medida de relación semántica recibe dos sentidos y entrega como salida un número real, el cual indica el grado de relación semántica entre los sentidos s_{ti} y s_{jk}. Este valor de relación semántica se asigna a *tem-score*$_k$, luego se selecciona el puntaje de relación más alto (*best-score*). El algoritmo entonces añade el *best-score* de cada uno de los *tokens* en la ventana, y este se convierte en el puntaje para el sentido s_{ti} (*score*$_i$) del *target token*. Este sentido con el más alto valor de *score*$_i$ es retornado como el sentido apropiado para el *target token.*

Algoritmo 3 Desambiguación de descriptores Web y Telco

1: **Input**: *token*$_i$, token of the attribute to disambiguate
2: **Output**: s_{ti}, sense more related to the token
3: **for** each token *T* in set
4: disambiguate-single-token (w)
5: **for** each sense s_{ti} of target token t, where *i=0..N*
6: let *score*$_i$ = 0
7: **for** each token w_j in context_window
8: next if *j = t*
9: for each sense s_{jk} of w_j

10*: temp-score*$_k$ = *relatedness* (s_{ti}, s_{jk})
11: *best-score = max temp-score*
12: **if** *best-score > pairScore*
13*: score*$_i$ = *score*$_i$ + *best-score*
14: **return** s_{ti} s.t. *score*$_i$ > *score*$_j$ for all j in
15: {s_{t0},..., s_{tN}} and *score*$_i$ > *contextScore*

A continuación se describe la forma en que se ha configurado la herramienta para la desambiguación de los descriptores de servicio Web y Telco:

- **Entrada:** La información recibida por la herramienta puede encontrarse en tres tipos de formatos:

  - ***Raw*:** Se refiere a texto plano no etiquetado. Por ejemplo: *The astronomer is married a movie star*.
  - ***Tagged*:** Texto etiquetado con etiquetas del corpus Penn Treebank[12]. Por ejemplo: *The/DT astronomer/NN married/VBD a/DT movie_star/NN*.
  - ***Wntagged*:** Texto etiquetado con etiquetas de *WordNet*. Por ejemplo: *The astronomer#n married#v a movie_star#n*.

  Ya que los atributos que SR-AW recibe a partir de la extracción que se realiza del descriptor se encuentran sin etiquetar, la herramienta se configura para que entienda el texto de entrada en formato *raw*. Solo para este tipo de formato sin etiquetas, la herramienta busca e identifica, a través de *WordNet*, los términos compuestos que recibe, ya que al configurar un formato con etiquetas, la herramienta asume que el usuario ha identificado y unido este tipo de términos con anterioridad. Estos términos son generalmente nombres que presentan un solo sentido al analizarse juntos, y que por separado cada uno adquiere un significado diferente. Por ejemplo: *White House*, al analizar los sentidos de cada palabra por separado es imposible llegar a desambiguarlas con un sentido que las identifique como la residencia del presidente de los Estados Unidos, el cual es el único sentido que el termino compuesto *White_House* puede tener. Otra característica de SR-AW cuando se encuentra configurada en *raw*, es que eliminará toda puntuación del texto de entrada, siempre y cuando esta no haga parte de términos compuestos, como por ejemplo: john_f**.**_kennedy.

  Una vez identificados los términos compuestos, la herramienta revisa si debe o no examinar una lista de palabras con significados irrelevantes o ambiguos (*stoplist*). Dentro de la configuración se ha activado esta opción y se ha utilizado la lista por defecto contenida en *WordNet*.

- **Opciones de configuración:** Entre las opciones que se pueden configurar para controlar el proceso de desambiguación, se encuentra el tamaño de la ventana de contexto, el cual establece el número de palabras involucradas en el proceso de desambiguación. Una ventana de contexto de tamaño N, incluyendo la palabra central, se extiende, a partir de esta, $N/2$ palabras hacia la izquierda y hacia la derecha, a menos que la palabra central se encuentre al final de la ventana. Si el tamaño de la ventana es impar, el número de palabras con el que se extenderá tanto a la derecha como a la izquierda será $(N-1)/2$, pero sí N es par, se extenderá $(N/2)-1$ palabras a la derecha, y $N/2$ palabras a la izquierda. Para el uso de esta herramienta en el presente trabajo,

[12] Proyecto disponible en http://www.cis.upenn.edu/~treebank/

la ventana de contexto se establece en un número igual al número de atributos recibidos, de esta manera la desambiguación de cada atributo será realizada con relación a todos los atributos restantes.

La segunda configuración que se realizó está relacionada con el método de desambiguación. Entre los diez posibles métodos, se ha elegido una de las modificaciones del Algoritmo de Lesk: superposición extendida de glosas (Extended Gloss Overlaps as a Measure of Semantic Relatedness, 2003). Este método fue seleccionado debido a que su comportamiento es igual al del método de desambiguación seleccionado en la sección 3.3.5: *Espacios semánticos aumentados*, ya que la propuesta del enfoque de superposición extendida de glosas fue realizada con base en la propuesta espacios semánticos aumentados. El funcionamiento del enfoque seleccionado fue explicado en la sección 3.3.1.1.

- **Salida:** La herramienta entrega como salida, el mismo texto de entrada desambiguado, es decir con sus respectivas etiquetas obtenidas de la base léxica *WordNet*, la definición del sentido y un ejemplo que contenga la palabra y se relacione con el sentido asignado. Las etiquetas representan la definición del sentido que se encuentra más relacionado con el dominio del texto. Estas etiquetas se encuentran formadas por tres partes, la primera corresponde a la palabra como tal, la segunda a su clasificación léxica dentro del texto, y finalmente el número del sentido que le correspondió. Por ejemplo, si se desambigua la palabra "*chess*" con respecto al contexto formado por las palabras "*game*, *checkers*", el resultado que se obtiene es: "chess#n#2: "*a board game for two players who move their 16 pieces according to specific rules; the object is to checkmate the opponent's King*", el cual presenta al segundo sentido del sustantivo (noun) "*chess*".

3.5. RESUMEN

Para facilitar la comprensión del dominio del servicio por parte de sistemas de recuperación de información, es necesario que la información ofrecida por los descriptores de servicio sea suficientemente clara. La cantidad de información que actualmente ofrecen los descriptores de servicio no satisface las necesidades de los motores de búsqueda. Una solución ampliamente adoptada para abordar este inconveniente consiste en la adición de anotaciones semánticas en los descriptores de servicio que permitan comprender automáticamente sus capacidades. Estas anotaciones semánticas se obtienen a partir de recursos como ontologías de dominio, que proporcionan un lenguaje común por medio de la representación formal del conocimiento. Procesos manuales o semiautomáticos de anotación semántica, se llevan a cabo por parte de personal calificado quienes se encargan de analizar los atributos de los descriptores para la asociación correcta de entidades semánticas. Sin embargo, esto involucra un proceso dispendioso, en el que invierten cantidades de recursos y tiempo considerables. El presente proyecto propone eliminar el componente manual en el proceso de adición de anotaciones semánticas, para el que se hace necesario que el análisis de los atributos del descriptor sea realizado de manera automática. La desambiguación lingüística permite realizar este análisis sin intervención humana, esto se constituye en un requisito indispensable para el proceso de automatización de la anotación semántica. El estudio de los diferentes métodos de WSD permitió establecer cuál de ellos es el que mejor se adecua a los requerimientos del proceso de desambiguación de los atributos de los servicios, a partir de ello fue

posible diseñar el mecanismo de extracción y desambiguación de atributos descrito en este capítulo.

4. ESTRUCTURA ONTOLÓGICA Y ANOTACIÓN SEMÁNTICA

4.1. INTRODUCCIÓN

El presente capitulo aborda, en primer lugar, el tema de las ontologías con el fin de entender su estructura y funcionamiento. Posteriormente, se define el mecanismo para la búsqueda de ontologías disponibles en la web, a través de una herramienta que ofrezca rapidez y precisión en los resultados. Seguidamente, se propone el mecanismo para identificar las ontologías con mayor grado de similitud semántico con el servicio. Por último, se presenta el mecanismo seleccionado para realizar la relación entre las entidades ontológicas y los elementos del servicio anotados semánticamente.

4.2. ONTOLOGÍAS

Existen numerosas definiciones acerca del concepto de ontología. Estas definiciones se encuentran divididas en dos grupos, el primero de ellos relaciona a las ontologías con la filosofía, mientras que el segundo contiene definiciones relacionadas con la inteligencia artificial (IA). Para el presente trabajo solo interesan las definiciones dentro del campo de la (IA). Una definición ampliamente aceptada es la establecida por (Neches, y otros, 1991):

"Una ontología define los términos básicos y relaciones que conforman el vocabulario de un área específica, así como las reglas para combinar dichos términos y las relaciones para definir extensiones de vocabularios"

Una ontología puede ser entendida entonces como una representación formal del conocimiento.

Para la organización del conocimiento de un dominio específico, las ontologías emplean un modelo que contiene un conjunto de elementos que conforman su estructura.

4.2.1. Estructura ontológica

En las ontologías, La representación del conocimiento se formaliza con el uso de cinco componentes: clases, relaciones, funciones, axiomas e instancias (A translation approach to portable ontology specifications, 1993). A continuación se realiza una breve descripción de cada uno de ellos (Breis, 2003):

- **Clases:** Los términos clase y concepto se utilizan indistintamente en las ontologías, debido a que un concepto puede referirse a algo acerca de lo cual se dice alguna cosa, por lo tanto puede hacer referencia a una acción, estrategia, proceso de razonamiento, función, etc. Su organización dentro de las ontologías suele ser realizada en taxonomías.

- **Relaciones:** Se utilizan para representar la interacción entre los conceptos del dominio. Formalmente, una relación se define como $R: C1 \times C2 \times \ldots \times Cn$, donde R corresponde a cualquier subconjunto de un producto de n conjuntos. Algunos ejemplos de relaciones son: "*subclase de*" y "*conectado a*".

- **Funciones:** Formalmente, una función se define como $F: C1 \times C2 \times \ldots \times Cn - 1 \rightarrow Cn$. Son un tipo especial de relaciones en las que el n-ésimo elemento de la relación es único para los n-1 precedentes. Por ejemplo, existen las funciones "*madre de*" y "*precio de un carro usado*".

- **Axiomas:** Pueden cumplir varios propósitos al ser incluidas en una ontología, entre ellos está el definir el significado de los componentes ontológicos, definir argumentos de relaciones, restricciones complejas sobre los valores de los atributos, entre otros. Los axiomas son expresiones verídicas que pueden o no ser incluidas en las ontologías. Aquellas que los incluyen son denominadas ontologías pesadas, mientras que las que no lo hacen se denominan ontologías ligeras. La combinación del uso de axiomas y la herencia de conceptos permite realizar inferencia de conocimiento que no se encuentre explícitamente indicado en la taxonomía de conceptos.

- **Instancias:** Son utilizadas para representar elementos específicos de una clase.

4.2.2. Tipos de ontologías

Los tipos de ontologías se dividen generalmente en dos, según el tipo de conocimiento contenido y la motivación de la ontología (Breis, 2003):

- Según el conocimiento contenido las ontologías se clasifican en (Van Heijst et al, 1997):

    - **Ontologías de información:** La información contenida es la especificación de los registros de una base de datos. Su estructura se forma a partir de estos registros. Por ejemplo, información sobre productos ofertado en un almacén.

    - **Ontologías terminológicas:** La información que este tipo de ontologías contiene es conocimiento de un dominio específico. Por ejemplo, la red semántica UMLS (*Unified Medical Language System*).

    - **Ontologías para modelar conocimiento:** Son utilizadas para especificar conceptualizaciones de conocimiento. Una ontología de este tipo define los términos y las relaciones básicas para la compresión de un área del conocimiento, así como las reglas para poder combinar los términos para definir las extensiones de este tipo de vocabulario controlado. Por ejemplo, una ontología descrita en OWL.

- Según la motivación las ontologías se clasifican en (Poli, 2000):

    - **Ontologías generales:** Son ontologías estructuradas a partir de la definición de categorías fundamentales del conocimiento y conexiones de dependencia entre las mismas. Con el fin de facilitar el manejo de estas categorías se ha empleado una organización de categorías principales que sea lo más transparente posible. Por ejemplo, conceptos generales como espacio, tiempo, materia, objeto, etc.

    - **Ontologías genéricas:** Permiten la clasificación de términos en una serie de niveles, de esta manera cada termino es accesible sólo en su sentido genérico y sus significados especializados lo serán para cuando

se active una ontología de domino especifico. Las ontologías genérales a diferencia de las ontologías genéricas solo definen términos de categorías fundamentales. Por ejemplo categorías como la cuantificación, los procesos o los tipos de objetos.

- **Ontologías del dominio:** Permiten estructurar de manera detallada un contexto de análisis con respecto a sus subdominios. Por ejemplo, dominios como la medicina, las aplicaciones militares, la cardiología etc.

- **Ontología aplicada:** Se utilizan cuando el entorno ontológico es aplicado a un objeto específico. Por ejemplo, un hospital.

4.2.3. Lenguajes de descripción de ontologías

La generación de estructuras ontológicas es posible gracias a lenguajes avanzados que permiten explotar las características de la web de una forma más eficiente, y que facilitan los procesos de razonamiento, inferencia y evaluación automática sobre las descripciones de los recursos. Actualmente existen varios lenguajes de representación de ontologías, entre los más conocidos, y ampliamente utilizados se encuentran: RDF (Resource Description Framework), DAML (DARPA Agent Markup Language), y OWL (Web Ontology Language). A continuación, se presenta una breve descripción de cada uno de ellos.

- **RDF (Resource Description Framework):** Este lenguaje representó el primer paso hacia una web basada en lenguajes de ontologías. Es un marco diseñado para representar información de la web de una forma muy flexible y mínimamente restrictiva (W3C, 2004). RDF fue desarrollado por la W3C. Este lenguaje está basado en tripletas de la forma *sujeto-predicado-objeto* (Pérez, y otros, 2011):

  - Recursos (sujeto): Son aquellos recursos de la web que pueden ser referenciados por un identificador único de recursos (URI) tales como: páginas web, documentos, etc.

    Ejemplo: http://localhost/default#HardDisk

  - Propiedades (predicado): permite conectar varios recursos.

    Ejemplo: Predicado: rdf:SubClassOf

  - Expresión (objeto): Es el valor que se da a la propiedad del recurso.

    Ejemplo: http://localhost/default#Hardware

  Además de esta estructura básica basada en tripletas, la especificación RDF incluye una serie de características incorporadas por medio de vocabularios especializados (Muñoz, y otros, 2009), que permiten la herencia de clases y propiedades, entre otras características.

- **DAML (DARPA Agent Markup Language):** Define una especificación de lenguaje basado en XML para la estructuración de ontologías, la cual incluye la definición de tipos de datos, una expresión coherente de las enumeraciones, entre otras ventajas. DAML está diseñado para tener una capacidad mayor a RDF para describir objetos y las relaciones entre los mismos, para expresar la

semántica, y para crear un nivel más alto de interoperabilidad entre los sitios Web. DAML permite crear instancias de las clases con el fin de definir recursos del correspondiente tipo RDF, y luego relacionar propiedades relevantes a dicha instancia. (Ouellet, y otros, 2002)

- **OWL (Web Ontology Language):** Lenguaje principalmente ideado para representar el conocimiento del mundo real, creando representaciones de objetos y cómo estos se relacionan entre sí. La semántica de OWL se basa en descripciones lógicas, cuya ventaja es que el conocimiento puede ser automáticamente comprobado con base en la estructuración de clases y subclases, la inconsistencia de los conceptos y las relaciones de vinculación. OWL ofrece la definición de componentes mediante una sintaxis basada en RDF/XML, por lo que OWL es fácilmente legible y comprensible, incluso para los desarrolladores de ontologías no expertos en el lenguaje. En cuanto al poder expresivo del lenguaje, OWL es una extensión de RDFS, por lo que incluye toda la capacidad de representación de RDFS. Al ser más potente que RDFS, con OWL se puede estructurar combinaciones lógicas entre las clases, restricciones sobre propiedades y características de propiedades, además de los constructores básicos de RDFS. Por ejemplo, se puede indicar que una clase es la combinación lógica (unión, intersección, complementación) de otras clases, o algunas clases son inconexas de otras. OWL también permite definir restricciones sobre el dominio y rango de una propiedad, restricción en la cardinalidad de los rangos y dominios. Así mismo, a través de OWL es posible especificar el intervalo de valores que una propiedad puede tomar exactamente (mínimo, máximo). Por otra parte, OWL puede describir si una propiedad es transitiva, simétrica, funcional o es la inversa de otra propiedad. (Yang, y otros, 2009)

4.3. ASIGNACIÓN DE ENTIDADES ONTOLÓGICAS

Este capítulo presente iniciante el análisis de los resultados de trabajos que enfocan sus objetivos en evaluar diferentes motores de búsqueda, esto con el fin de seleccionar el motor de búsqueda más eficiente y oportuno para la encontrar ontologías en la web para cada atributo relevante extraído del descriptor de servicio. Posteriormente, se describe el enfoque definido para desambiguar las entidades ontológicas contenidas en una ontología y a partir de dicha desambiguación seleccionar la entidad ontológica con mayor grado de similitud semántica con el atributo extraído del descriptor.

4.3.1. Búsqueda de Ontologías

Como se mencionó en la sección 4.2.3 existen varios lenguajes para la construcción de ontologías. Partiendo del hecho de que la anotación semántica de servicios exige una precisa relación de los elementos sintácticos del servicio con los datos semánticos utilizados para realizar la anotación semántica, además de requerimientos específicos del enfoque propuesto en este trabajo para llevar a cabo la anotación semántica de servicios como: el manejo de razonadores automáticos que permitan ubicar los conceptos ontológicos dentro de una estructura ontología; implica tener dichos recursos semánticos estructurados en modelos conceptuales que hagan uso de lenguajes bien definidos. Por estas razones OWL constituye una buena opción para trabajar en este proyecto, ya que además de incluir toda la capacidad expresiva de RDFS, la extiende con la posibilidad de utilizar expresiones lógicas. OWL permite

abarcar más exhaustivamente un dominio particular del conocimiento y a su vez trabajar de forma más eficiente con razonadores automáticos sobre las estructuras obtenidas. Adicionalmente, este lenguaje proporciona anotaciones útiles para llevar sus modelos de datos al mundo real (Tester, 2012). Por otra parte OWL se constituyó en un estándar avalado por el W3C, por lo cual es el lenguaje de representación de ontologías más utilizado en el contexto de la web semántica; razón que apoya la elección de OWL como el lenguaje más adecuado para trabajar en el marco de este proyecto.

Las ontologías por lo general, se crean para diferentes dominios del conocimiento. Sin embargo, cada ontología define una terminología consensuada para un dominio determinado; luego una ontología existente puede ser reutilizada independiente del propósito, evitando así esfuerzos para crear ontologías que aborden una misma área del conocimiento. Muchas de estas ontologías están disponibles en la web, pero el proceso para seleccionar una de estas y reutilizarla es bastante complicado: el usuario antes debe examinar literalmente el contenido que la ontología proporciona, con el fin de asegurar que elige la adecuada. Frente a este problema, se hace necesario el apoyo de sistemas informáticos, que a partir de parámetros indicados, busquen de forma automática y eficiente archivos almacenados en los servidores web, distribuidos en diferentes partes del mundo. Actualmente existen varios buscadores que ofrecen innumerables servicios, entre los que podemos encontrar a *Yahoo* (Yahoo!), *Google* (Google), *Msn* (Microsoft), *Hakia* (hakia, Inc), Ask (Ask), *Seekport* (Seekport), *Exalead* (Exalead), entre otros. En (Tümer, y otros, 2009) se presenta un estudio para evaluar el nivel de precisión y los coeficientes de *recall*[13] normalizados de los resultados obtenidos por cuatro de los buscadores más populares de la web (*Yahoo*, *Google*, *Msn* y *Hakia*). Yahoo mostró el mejor desempeño en términos de p*recisión*[14]*,* mientras que Google fue el mejor motor de búsqueda en términos de razón de *exhaustividad normalizada*[15] (Figura 4 y Figura 5).

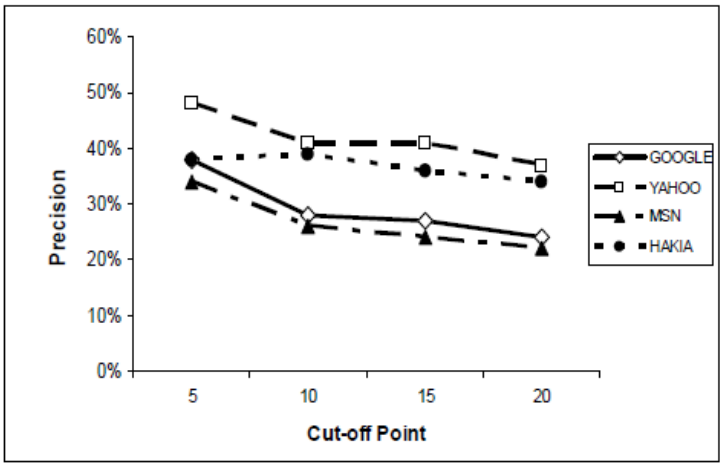


Figura 4. Medida de índices de precisión de los motores de búsqueda (Tomado de (Tümer, y otros, 2009))

[13] Recall: entendida como la fracción de los resultados relevantes que son recuperados por los buscadores.

[14] Precisión: se define como la relación entre el número de documentos relevantes recuperados frente al número total de documentos recuperados.

[15] Exhaustividad Normalizada: porcentaje de resultados recuperados válidos para el usuario comparado con el total de resultados interesantes para el usuario disponibles en la web.

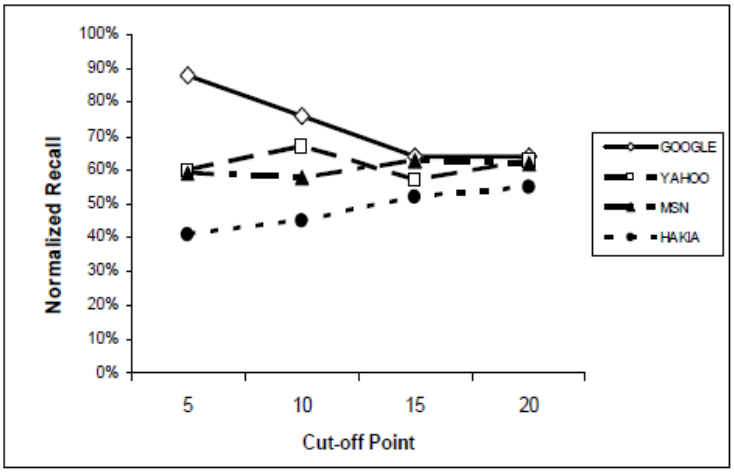


Figura 5. Medida de coeficientes normalizados de recall de los motores de búsqueda (Tomado de (Tümer, y otros, 2009))

En (Lewandowski, 2010) también se muestra una comparación entre 6 de los principales motores de búsqueda en internet (Google, Yahoo, MSN, Ask, Seekport, and Exalead) para evaluar la efectividad de cada uno, teniendo en cuenta no sólo los resultados, sino también la capacidad de listar en los primeros lugares los resultados que el usuario espera. El criterio de selección de los motores de búsqueda a evaluar en este trabajo, consiste en que cada motor debe proporcionar su propio índice de direcciones web. Muchos portales de búsqueda más populares, como *AOL* (Aol), utilizan los resultados de los grandes proveedores de búsqueda, por esta razón, este buscador, por ejemplo, fue excluido de este estudio. El resultado de esta evaluación indica que Google y Yahoo son los motores de búsqueda con el mejor desempeño, y que cuando se consideran únicamente los tres primeros resultados, se puede encontrar una ligera ventaja para *Google*. Cuando se consideran los cuatro o cinco primeros resultados, el desempeño de *Yahoo* se acerca al de *Google*. Sin embargo *Google* se mantiene como líder en términos de consultas contestadas correctamente para cualquier número de resultados considerados, como se muestra en la figura 6.

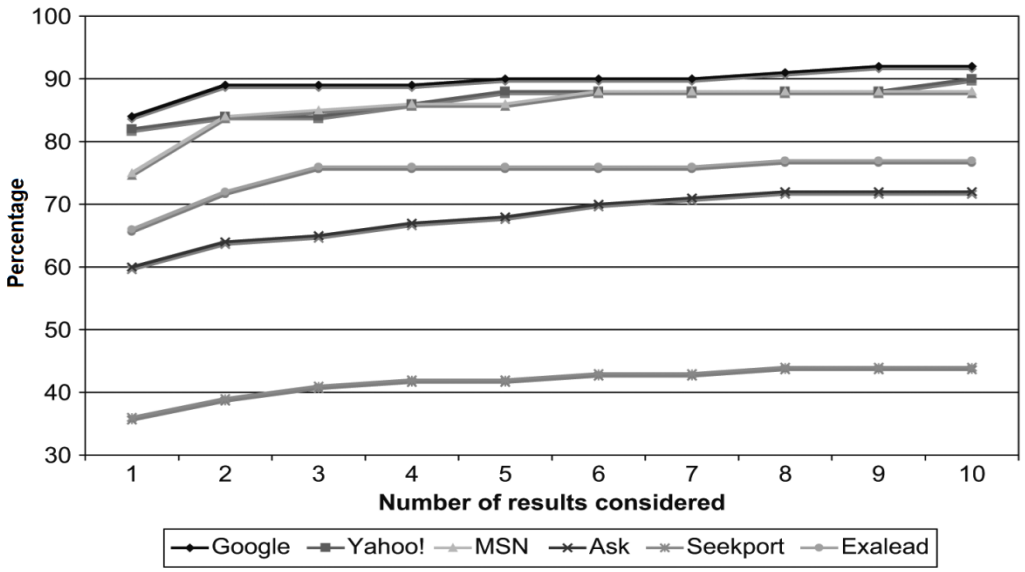


Figura 6. Precisión en consultas contestadas correctamente por cada motor de búsqueda considerando los 10 primeros resultados (Tomado de En (Lewandowski, 2010))

Teniendo en cuenta los resultados obtenidos en los trabajos anteriormente citados, se puede concluir que tanto *Google* como *Yahoo* son los motores de búsqueda eficiente para apoyar la búsqueda de ontologías disponibles en la web. Sin embargo, teniendo en cuenta requerimientos concretos de este proyecto como: encontrar ontologías descritas en OWL y que los primeros resultados obtenidos sean los más relevantes, se optó por utilizar el motor de búsqueda *Google*. Adicionalmente, desde el punto de vista de implementación *Google* ofrece mayores facilidades para especificar el tipo de resultado que se desea obtener (documentos OWL en este caso). Este buscador permite realizar una búsqueda más precisa a partir de la utilización de un conjunto de operadores asociados a la consulta (Google).

4.3.1.1. Buscador de ontologías OWL disponibles en la Web

Luego de definir el tipo de ontologías y el motor de búsqueda a utilizar, se ha implementado un módulo software que integra la herramienta *Google Custom Search* (Google, 2012) *que proporciona Google para acceder a todas las funcionalidades de búsqueda a través del protocolo HTTP desde una aplicación remota*. Para cada atributo obtenido en el proceso de *Extracción de atributos a anotar semánticamente* expuesto en la sección 3.4.1.1, se buscan automáticamente ontologías donde el identificador de alguna clase de la ontología coincida con el atributo extraído. Para esto se especifica el atributo a buscar y se indica que restrinja la búsqueda solo a archivos con extensión OWL (empleando el operador *filetype:owl*), luego a través del método GET se envía la petición al servicio de búsqueda, el cual retorna un objeto JSON con las URL para acceder a cada resultado.

4.3.2. Desambiguación de entidades ontológicas

Como se mencionó anteriormente, para cada atributo extraído del descriptor de servicio se buscan ontologías disponibles en la web. Luego a cada atributo le corresponde un grupo de ontologías, tal como se ilustra en la Figura 7.

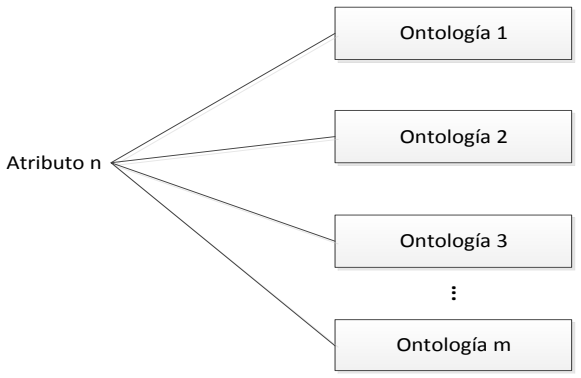


Figura 7. Correspondencia Atributo - Ontologías

La búsqueda y asociación de estas ontologías a un determinado atributo se realiza con el fin de encontrar en alguna de ellas la entidad ontológica apropiada para realizar la anotación semántica del atributo en consideración. En primer lugar, se realiza una búsqueda sintáctica en cada ontología, de la entidad o entidades más similares respecto al atributo a anotar (entidad(es) coincidente(s)). De estas entidades coincidentes se extraen adicionalmente las entidades que derivan de las mismas (entidades de nivel inferior), y las entidades de las cuales ellas se derivan (entidades

de nivel superior), de acuerdo con la jerarquía de clases definida en la ontología. Seguidamente, la(s) entidad(es) ontológicas se desambiguan lingüísticamente para encontrar sus sentidos con relación a la ontología que las contiene, esto con el fin de determinar si las entidades que se han relacionado sintácticamente con el atributo a anotar se relacionan también en un nivel semántico. El orden que se ha propuesto para hallar una entidad ontológica adecuada para la anotación semántica de un determinado atributo de un descriptor es: en primer lugar la búsqueda de una entidad ontológica que coincida de forma sintáctica con el atributo del descriptor a anotar, esto partiendo de la premisa de que palabras sintácticamente iguales tienen una mayor posibilidad de tener sentidos iguales. Sin embargo, en este punto del proceso, no se pude asegurar que la entidad ontológica encontrada se relacione con el atributo, por lo tanto, se propone en segundo lugar, realizar el proceso de desambiguación lingüística de la entidad ontológica que coincidió sintácticamente con el atributo, con el ánimo de buscar la entidad que se relacione, no solo sintácticamente, sino también semánticamente con el atributo a anotar.

Para el proceso de búsqueda sintáctica de entidades ontológicas asociadas al atributo a anotar, se emplea un razonador de ontologías, que permite la búsqueda y extracción de estas entidades. Existe una serie de herramientas que permiten razonar sobre las ontologías. Sin embargo, teniendo en cuenta que en el marco de este proyecto se optó por trabajar con ontologías OWL, son de particular interés los motores de inferencia que soportan este lenguaje específico. En este sentido, se presentan a continuación los razonadores que permiten la evaluación de este tipo de ontologías. (Comparison of reasoners for large ontologies in the OWL 2 EL profile, 2011)

- **CB (Consequence-based reasoner):** Permite que las ontologías *SHIQ* puedan traducirse al fragmento *Horn* de primer orden lógico. Este razonador es teóricamente óptimo para ontologías *SHIQ*, *Horn*, así como también para el fragmento común de EL^{++} y *SHIQ.*

- **CEL (Classifier for *EL*):** Permite procesar grandes ontologías EL^{+} en un tiempo razonable gracias a la implementación de un algoritmo de tiempo polinomial.

- **FaCT++ (Fast Classification of Terminologies):** Este razonador procesa OWL DL y un subconjunto de OWL 2. Esta implementado en C++ y se basa en los algoritmos optimizados *tableaux*.

- **HermiT:** Permite determinar la consistencia de una ontología, identificar relaciones entre conceptos, entre otras características. Se encuentra basado en un cálculo “hypertableau”.

- **Pellet:** Este razonador soporta OWL DL (*SHOIN (D)*), OWL 2 (*SROIQ (D)*), soporta perfiles OWL 2 incluyendo OWL 2 EL. Fue uno de los primeros razonadores en soportar OWL DL.

- **RacerPro (Renamed ABox and Concept Expression Reasoner):** Este razonador implementa la descripción lógica SHIQ, y contiene optimizaciones para OLW 2 EL, lo cual le permite razonar ontologías muy largas como SNOMED CT[16].

[16] Ontología disponible en http://krono.act.uji.es/people/Ernesto/umlsassessment/SNOMED-ontology.zip/view

A continuación se presenta el análisis realizado con base en un conjunto de características funcionales en cada razonador, tales como el uso de OWL API[17], una API en Java que permite acceder al razonador para manipular y serializar ontologías OWL; Protégé plugin[18], el cual permite acceder a las diferentes características de Protégé como la edición de ontologías; Jena API, que permite la lectura, escritura, y procesamiento de ontologías RDF y OWL; el lenguaje en el que los razonadores han sido desarrollados, entre otras características.

- **OWL API:** A excepción del razonador CB, todos los razonadores son accesibles a través de OWL API, permitiendo que aplicaciones que necesitan usar a múltiples razonadores por medio de una misma interfaz puedan hacerlo sin mayores complicaciones.

- **Protégé Plugin:** Con excepción de los razonadores CB y RacerPro, todos los razonadores tienen un plugin para conectarse con las características de Protégé.

- **Open source:** A diferencia de la mayoría de los razonadores RacerPro y Snorocket no son de código abierto.

- **Lenguaje:** Los razonadores que hacen uso del lenguaje java son HermiT, Pellet y Snorocket.

- **Plataforma:** El razonador CEL es el único que está restringido a una plataforma Linux. El resto de los razonadores están soportados por todos los sistemas operativos comerciales.

- **Jena:** El único razonador que tiene una interfaz nativa de Jena es Pellet.

En consecuencia, teniendo en cuenta las características de los motores de inferencia mencionadas, el razonador seleccionado es Pellet por los motivos que se enumeran a continuación: (*i*) Soporta OWL de forma nativa, incluyendo un subconjunto de OWL Full, (*ii*) tiene un soporte amplio para tipos de datos XML Schema, y (*iii*) es una herramienta de código abierto que se encuentra en un continuo proceso de mejoramiento (Parsia, y otros, 2005). Adicionalmente, Pellet es el único que posee una interfaz nativa de Jena, framework que facilita el manejo de ontologías OWL y RDF (Sirin, y otros, 2007). Por último, Pellet es uno de los tres razonadores desarrollados en java y puede trabajar en cualquier tipo de plataforma.

Para la desambiguación lingüística de las entidades ontológicas se considera la entidad y las entidades de nivel superior e inferior asociadas a ella. El proceso se realiza mediante el uso de una herramienta de código abierto denominada *WordNet::SenseRelate::WordToSet (SR-WTS)* (Michelizzi, y otros, 2008), la cual recibe como entrada una palabra a desambiguar y un conjunto de términos (contexto de la palabra ambigua), en función de las cuales hace la desambiguación, entregando como salida el sentido más adecuado para la palabra ambigua de acuerdo con términos que se establecieron como contexto. Para los propósitos del presente trabajo, la herramienta se utiliza de la siguiente manera:

- **Entrada:** Del conjunto de palabras que la herramienta recibe como entrada, la primera es a la palabra a desambiguar, la cual corresponde a la entidad ontológica coincidente, mientras que las palabras que la suceden,

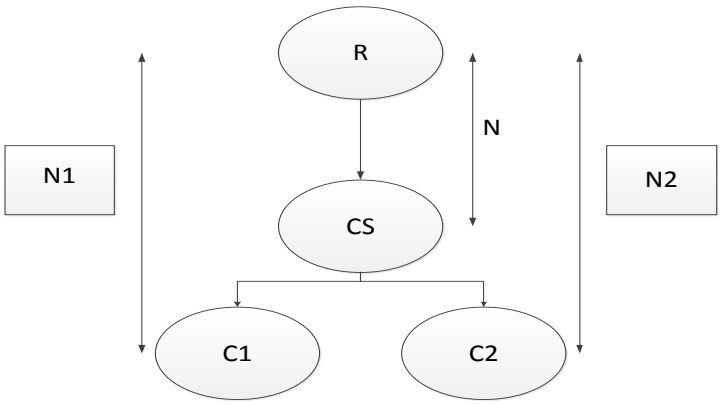
[17] http://owlapi.sourceforge.net/
[18] http://protege.stanford.edu/download/plugins.html

correspondientes a las entidades ontológicas de nivel superior e inferior de la entidad coincidente, las cuales se establecen como el contexto de la primera. La herramienta determina el sentido de la palabra mediante el uso de la jerarquía de *WordNet* y una medida de relación semántica que puede ser elegida por el usuario dentro del conjunto que la herramienta tiene implementadas.

- **Opciones de configuración:** De las medidas de similitud semántica presentadas en la sección 3.3.2.1, la medida de similitud seleccionada es la propuesta por Wu &Palmer (*WUP*) (Wu, y otros, 1994), la cual se encuentra como opción dentro del conjunto de medidas de similitud semántica ofrecidas por la herramienta *SR-WTS*; tiene como ventajas su fácil implementación y presenta mejor desempeño que otras medidas de similitud (Lin, 1998). Esta medida de similitud se basa en la estructura y contenido de *WordNet*, lo que le permite utilizar su jerarquía de conceptos para establecer relaciones de similitud, como establecer, por ejemplo, que un *carro* se relaciona en mayor medida con un *bote* que con un *árbol*, ya que tanto la palabra *bote* como la palabra *carro* tienen como antecesor la palabra *vehículo* en la jerarquía de *WordNet*.

  El método WUP para el cálculo de la similitud, se basa en el método de recuento de borde, de la siguiente manera (Slimani, y otros, 2006):

  Considere una ontología Ω, formada por un nodo raíz (R) y un conjunto de nodos (ver Figura 8). $C1$ y $C2$ representan dos elementos ontológicos a los cuales se les calculara la similitud. $N1$ y $N2$ se definen como la distancia que separa los nodos $C1$ y $C2$ del concepto raíz R. La distancia que separa al concepto antecesor (CS) más cercano, común para $C1$ y $C2$, del nodo raíz R, se identifica con N.



Fuente: (Slimani, y otros, 2006)
Figura 8. Ejemplo del extracto de una ontología

Según lo anterior, la medida de similitud WUP se define como se presenta en la ecuación 9:

$$Sim_{wp} = \frac{2N}{N1+N2} \qquad \textbf{Ec. 9.}$$

Para comprender mejor la forma en que WUP trabaja puede considerarse el siguiente ejemplo (Ganesan, y otros, 2012):

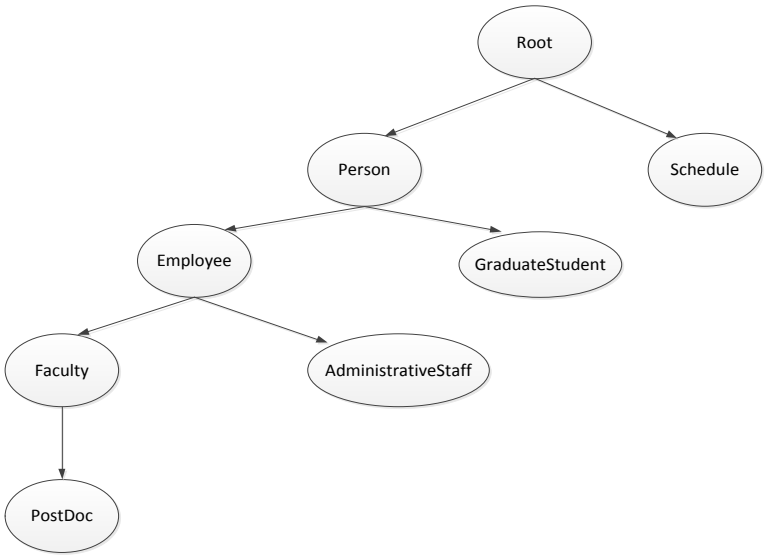


Figura 9. Ontología de ejemplo

Considerando la ontología presentada en la Figura 9, en la cual los conceptos $C1$ y $C2$ corresponden a "*PostDoc*" y "*AdministrativeStaff*" respectivamente. Aplicando el cálculo de la similitud semántica con WUP (ecuación 10), se obtiene como resultado:

$$Sim_{wp} = \frac{2*2}{4+3} = 0.57 \qquad \textbf{Ec. 10.}$$

Gracias a la organización de nombres y verbos dentro de jerarquías en *WordNet* , así como las relaciones entre las mismas, se considera esta base de datos léxica como una de las más adecuadas para determinar el grado de similitud entre conceptos (Pedersen, y otros, 2004).

- **Salida:** La herramienta *SR-WTS* recibe como parámetros un conjunto de palabras: la primera, se define como palabra ambigua y las siguientes como su contexto. Así mismo, recibe la especificación de la medida de similitud semántica que se desea utilizar. Por ejemplo, si se deseara desambiguar la palabra "*game*" (*juego*), con relación a las palabras "*monopoly*" (*monopolio*), "*chess*" (*ajedrez*), "*checkers*" (*damas*), y con la medida de similitud semántica *WUP*, lo que se obtendría a través de la herramienta sería:

  game#n#1 : 2.52631578947368 : un concurso con reglas para determinar un ganador; "tú necesitas cuatro personas para jugar este juego "

  game#n#10 : 2.17777777777778 : tu ocupación o línea de trabajo; "él está en el juego de la plomería "; "ella está en el mundo del espectáculo"

  game#n#3 : 2.17777777777778 : diversión o pasatiempo; "ellos jugaron juegos de palabras"; "él pensó en su pintura como un juego que llenó su tiempo libre"; "su vida era toda diversión y juegos"

Donde el primer numeral que acompaña a la palabra (#n) indica que la palabra ha sido desambiguada como un *sustantivo* (*noun*), mientras que el segundo establece el número del sentido que le ha sido asignado; por ejemplo, entre un conjunto de posibles sentidos a la palabra "game", en el último de los ejemplos, se le ha asignado el sentido número 3. A continuación, se muestra un número

decimal que muestra el grado de similitud semántica entre el sentido de la palabra ambigua y su contexto lingüístico. Finalmente, se encuentra la definición del sentido que le ha sido asignado a la palabra y uno o más ejemplos relacionados. De las definiciones de los sentidos obtenidos por *SR-WTS*, se decidió seleccionar el primero en todos los casos, ya que es el sentido que presenta mayor relación semántica con su contexto lingüístico.

4.3.3. Filtro de ontologías

Para la asignación de entidades ontológicas, el filtrado de ontologías es uno de los procesos más importantes, pues es el que se encarga de seleccionar las entidades más apropiadas según el atributo a anotar. Su funcionamiento se basa en realizar una comparación de definiciones de sentidos y analizar el grado de similitud semántica que presentan. El filtro recibe como entrada tanto la definición del sentido del atributo a anotar, como la definición del sentido de una de las entidades ontológicas que se presentan como posible opción para la anotación del atributo. De esta manera, el filtro realiza iteraciones analizando una a una las entidades asociadas a cada atributo a anotar. Finalmente, el filtro entrega a la salida un valor correspondiente al grado en el cual el atributo y la entidad ontológica se relacionan semánticamente, y adicionalmente, se realiza un análisis de este valor para asignar al atributo la entidad ontológica que presente el mejor grado de similitud semántica.

La definición del sentido del atributo a anotar se obtiene a partir del proceso de desambiguación de atributos del descriptor introducido en la sección 3.4.1, mientras que la definición de cada entidad ontológica se consigue del proceso de desambiguación de ontologías (sección 4.3.2). Para obtener el grado de similitud semántica, existe una herramienta de código abierto que permite estimar este cálculo, a partir de la elección de una medida de similitud semántica de un conjunto de diez medidas posibles. Esta herramienta se denomina *WordNet::Similarity* (Pedersen, y otros, 2004), y se ha configurado de la siguiente manera para que su funcionamiento se adecue a las necesidades del presente trabajo:

- **Entrada:** Las entradas a *WordNet::Similarity* son la salida del proceso de desambiguación de atributos del descriptor, y la salida del proceso de desambiguación de ontologías, las cuales se encuentran etiquetadas por *WordNet*, lo que significa que las etiquetas de los atributos y las entidades ontológicas presentan tres partes como se explicó: la palabra, la categoría léxica a la cual pertenece y el número del sentido que le corresponde. Por ejemplo: *car#n#2*, corresponde al segundo sentido del sustantivo (*noun*) *car*.

- **Opciones de Configuración:** Como se mencionó (sección 4.3.2) la medida de similitud semántica elegida fue la propuesta por Wu &Palmer (*WUP*), cuyo funcionamiento y desempeño con relación a las demás medidas se describió en la sección anterior: *Desambiguación de entidades ontológicas.*

  Además de la selección de la medida de similitud semántica, la herramienta presenta otra opción configurable que corresponde a dos nodos raíz hipotéticos, que pueden o no utilizarse al calcular la similitud entre conceptos. Si se usan, la herramienta permitirá que el cálculo de similitud semántica pueda realizarse entre cualquier par de sustantivos o verbos, de lo contrario, los conceptos deberán encontrarse dentro de la misma jerarquía (Pedersen, y otros, 2004). Para el presente trabajo se decidió emplear estos nodos, con el fin de facilitar el cálculo de la similitud entre conceptos.

Debido a que los conceptos para la entrada se encuentran etiquetados por *WordNet*, la herramienta *WordNet::Similarity* se configuró para que aplique la medida de similitud semántica teniendo en cuenta la información que ofrecen las etiquetas de este MRD(Machine Readable Dictionary), reduciendo así la complejidad del proceso, ya que solo se evalúa el cálculo de la similitud semántica entre la definición del sentido del atributo que fue seleccionado como el más adecuado, de acuerdo con el contexto del descriptor, y la definición del sentido de la entidad ontológica que más se relaciona con el atributo en consideración.

Salida: La herramienta entrega como salida un valor correspondiente al grado de similitud existente entre las dos palabras que recibe como entrada. Si dicho valor es igual a uno, la similitud entre ambos conceptos será la máxima, mientras que un valor igual a cero denota una total de disimilitud entre los términos evaluados.
A manera de resumen, la herramienta se comporta de la siguiente manera: suponiendo que la palabra "*car*" corresponde a un atributo de un descriptor de servicio; la palabra "*bus*" corresponde a una entidad ontológica; y que ambas ya se han desambiguado según un contexto cualquiera, el resultado obtenido, con las etiquetas asignadas por WordNet, sería: *car#n#3* (*cable car*), *bus#n#1* (*motor coach*). Se procede entonces a realizar el cálculo de similitud semántica entre ellos con las consideraciones realizadas, obteniendo como resultado de la medida WUP un valor de 0.5714. El proceso debe repetirse cambiando la palabra "*bus*" por la siguiente entidad ontológica, dentro del conjunto de posibles entidades que podrían usarse para anotar el atributo. Posteriormente, se selecciona la entidad que presente la mejor similitud semántica con el atributo a anotar, en este caso la palabra "*car*".

4.3.4. Calidad de las ontologías

Una vez filtradas las entidades ontológicas a utilizar para el proceso de anotación, es de vital importancia analizar la calidad de las ontologías a las cuales estas entidades pertenecen. Con base en este análisis se debe realizar un segundo proceso de filtrado, descartando ciertas ontologías –y por ende sus entidades–, con el fin de evitar anotaciones irrelevantes que no enriquezcan la información contenida en el descriptor del servicio. Este análisis de calidad se basa en la evaluación del contenido de las ontologías, es decir, en una determinación de su profundidad, y en la búsqueda de conceptos inconsistentes. Este último es un error que hace parte de un grupo de errores más grande denominado inconsistencias semánticas. (Gangemi, y otros, 2005)

Teniendo en cuenta que las ontologías son desarrolladas por uno o varios ingenieros de conocimiento, y que como seres humanos son propensos a errores, debe considerarse el hecho de que pueden surgir inconsistencias en las ontologías (Gómez, y otros, 2008). Existen enfoques que proponen la reparación de ontologías con este tipo de problemas, así como otros que proponen procesar estas inconsistencias y continuar trabajando con ellas a pesar de los errores. Sin embargo, ambos enfoques corresponden a propuestas fuera del alcance del presente proyecto, por lo tanto se ha establecido el proceso de descarte de ontologías en las que los conceptos inconsistentes predominen.

Antes de diagnosticar un error en una ontología este debe ser detectado. Existe un gran número de errores frecuentes cometidos en el diseño de una ontología, los cuales van desde fallos en la taxonomía y diseño, hasta errores en la postura

estructural en los axiomas que se declaran, y en la semántica conceptual. A continuación se expone la relación de los errores más comunes (Senso, y otros, 2011):

- **Errores de Distribución:** Debido a este error se presentan clases que están mutuamente relacionadas a subclases disjuntas, y la fragmentación de clases en muchas subclases. Este error se presenta cuando la estructuración del conocimiento se realiza a partir de una base clasificatoria, generando excesiva dependencia ("tipo de" y "subtipo de") entre clases y subclases.

- **Errores de Ubicación:** Se presenta cuando la disposición correcta de clases no se tiene en cuenta, generando solapamiento entre ellas. Este error se genera cuando una misma clase se define como subclase, y superclase al mismo tiempo en diferentes niveles de jerarquía. Para prevenir este error debe realizarse un análisis lingüístico de los conceptos, para obtener una estructuración coherente del conocimiento.

- **Errores de inconsistencia semántica:** Este error abarca una serie de errores, entre los que se encuentran: taxonomías mal formadas, errores en la ubicación de las clases, axiomas mal declarados, descripciones simplistas de las clases por el uso incorrecto de anotaciones, entre otros. Se producen cuando se desarrollan jerarquías de nodos para conceptos erróneos, los cuales corresponden a conceptos que no pertenecen a la clase principal.

- **Clases y clasificaciones incompletas:** Este error se presenta debido al poco cuidado que se tiene con los elementos utilizados en la conceptualización, anotación y descripción de la ontología, generando ambigüedad y una construcción errónea de herramientas de razonamiento.

- **Errores de redundancia:** Surgen a partir de la repetición de conceptos, debida a la falta de un plan en la fabricación de las taxonomías de base del sistema.

- **Errores en la poca especificación o delimitación de las propiedades de los componentes del sistema:** Consiste en asignar una misma definición a todo, produciendo que el razonamiento se vea poco desarrollado, desde el punto de vista formal y conceptual.

En consecuencia, para determinar el nombre de los conceptos inconsistentes en las ontologías, estas deben poderse procesar en su totalidad por el razonador Pellet. Esta es una de las primeras características para analizar su calidad, ya que en caso de que el razonador no pueda procesarlas, se concluye que su construcción tiene numerosos problemas de inconsistencia y por tanto se descarta. En caso de que la ontología pueda procesarse totalmente, el razonador reorganiza el árbol de conceptos para un mejor manejo y revisa la consistencia de cada uno de ellos. Al de detectar conceptos inconsistentes, Pellet los ubica al final del árbol de conceptos presentándolos como subclases del concepto vacío *owl:Nothing*. De esta manera, se descartan todas aquellas ontologías que presenten conceptos inconsistentes. Finalmente, Pellet calcula y entrega el número de conceptos contenidos en cada ontología, se realiza el análisis y se descartan aquellas ontologías cuyo árbol de conceptos es relativamente pobre.

4.3.5. Anotación de descriptores de servicio con SAWSDL

Debido a que SAWSDL corresponde a una recomendación del W3C, como mecanismo para la adición de información semántica en los componentes de los

descriptores WSDL, se ha seleccionado como la tecnología para realizar la anotación semántica de los servicios en el presente trabajo. Como se presentó en la sección 1.2.2.1, SAWSDL define mecanismos que permiten la adición de anotaciones semánticas en los componentes de los descriptores WSDL. Para esto presenta un conjunto de atributos de extensión, formado por: *modelReference*, *LiftingSchemaMapping* y *LoweringSchemaMapping*. En el desarrollo del presente trabajo de grado, el atributo de extensión que se utiliza es *ModelReference*, ya que permite especificar la asociación entre un componente del descriptor WSDL y una entidad ontológica, mientras que los atributos *LiftingSchemaMapping* y *LoweringSchemaMapping* permiten la especificación del mapeo entre datos semánticos y XML, proceso que no está asociado directamente con la anotación semántica del descriptor de servicio.

A continuación se presentan algunos ejemplos de la forma en la que se realizan las anotaciones SAWSDL:

- *Anotación modelReference en wsdl:operation* (ver Figura 10):

Para los elementos de tipo *operation* la anotación semántica se realiza por medio del atributo de extensión *modelRefence*, el cual permite la asociación de este tipo de elementos con componentes semánticos.

SAWSDL

```
…
<wsdl:operation name=”order”patern= “http://www.w3.org/2006/01/wsdl/in-out”
        sawsdl:modelReference=”http://www.w3.org/2002/ws/sawsdl/spec/ontology/ ⟶ Anotación Semántica
purchaseorder#RequestPurchaseOrder”>
        <wsdl:input element=“OrderRequest”/>
        <wsdl:output element=”OrderResponse”/>
</wsdl operation>…
```

```
<rdf:RDF xmlns:rdf=”http://www.w3.org/1999/02/22-rdf-syntax-ns#” xmlns:rdfs=”http://www.w3.org/2000/01/rdf-schema#”
xmlns:owl=”http://www.w3.org/2002/07/owl#” xmlns=”http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#”
xml:base=”http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#”>
<owl:Ontology/>
…
…
<owl:ObjectProperty rdf:ID=”hasIdentifier”>
<rdfs:domain rdf:resource=”http://www.w3.org/2002/07/owl#Thing”/>
<rdfs:range rdf:resource=”#Identifier”/>
</owl:ObjectProperty>
<owl:Class rdf:ID=”RequestPurchaseOrder”/> ⟶ Entidad Ontológica
</rdf:RDF>
```

Ontología

Figura 10. Anotación semántica modelReference en wsdl:operation

A continuación se muestra la anotación de un atributo de tipo simple, el cual es anotado semánticamente con la entidad ontológica *OrderConfirmation*, perteneciente al modelo semántico que se referencia en la anotación. Al igual que en la anotación de elementos operación, para la anotación de tipos simples también se utiliza el atributo *modelReference*.

SAWSDL

```
…
<xs:simpleType name=”Confirmation”
sawsdl:modelReference=”http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#OrderConfirmation”>
…
</xs:simpleType>
…
```

↓ **Entidad Ontológica**

Figura 11. Anotación semántica modelReference en xs:simpleType

- *Anotación modelReference en xs:element* (ver Figura 12):

Al igual que los elementos de tipo *operation*, los elementos de tipo *element* también son anotados por medio del atributo por extensión *modelReference* el cual permite que el descriptor del servicio sea enriquecido con información semántica fácilmente. A continuación se muestra la forma en que el atributo “*OrderRequest*” de tipo *element* es anotado semánticamente por la entidad ontológica “*OrderRequest*”, que se encuentra en el modelo semántico que se referencia en la anotación realizada.

SAWSDL

```
…
<xs:element name=”OrderRequest”
        sawsdl:modelReference=”http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#OrderRequest”>
 <xs:complexType>
  <xs:sequence>
    <xs:element name=”customerNo” type=”xs:integer”/>
    <xs:element name=”orderItem” type=”item” maxOccurs=”unbounded”/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
…
```

Figura 12. Anotación semántica modelReference en xs:element

4.4. RESUMEN

En este capítulo se realizó una descripción de la estructura ontológica y los elementos que la componen, así como de los tipos de ontologías que existen. Se describieron los principales procesos que deben realizarse para una exitosa asignación de entidades ontológicas disponibles en la web, a los descriptores de servicios Web y Telco. El trabajo en conjunto de los procesos descritos en este capítulo permite la asignación de entidades ontológicas sin requerir intervención humana. De esta manera, la aplicación de cada uno de los procesos permite que la asignación de entidades ontológicas se realice con precisión y de manera automática.

5. PROTOTIPO Y EVALUACIÓN

5.1. INTRODUCCIÓN

En este capítulo se aborda la descripción detallada del trabajo realizado alrededor del desarrollo de un sistema software, que implementa el mecanismo propuesto para la desambiguación lingüística de los atributos de descriptores de servicio Web y de Telecomunicaciones (sección 3.3), y la técnica planteada para la asignación de entidades ontológicas disponibles en la Web a los descriptores de servicio Web y de Telecomunicaciones (sección 4.3). Esta descripción abarcará desde la definición de la arquitectura diseñada para soportar la construcción del prototipo, hasta la especificación del plan de pruebas realizadas sobre el mismo, junto con los respectivos resultados obtenidos.

5.2. DESCRIPCIÓN DEL PROTOTIPO

El prototipo construido, para evaluar el mecanismo propuesto, consiste en una aplicación Java, implementada a partir de la adopción del Modelo de Construcción de Soluciones (Serrano, 2005), como referencia metodológica para soportar el proceso de desarrollo.

Las secciones de este capítulo, se dedican a describir la aplicación de dos macro-componentes fundamentales de dicho modelo: la Estructura para la Descripción del Sistema y el Modelo del Proceso de Desarrollo. El primer macro-componente, involucra la especificación de artefactos esenciales, para el entendimiento adecuado de la funcionalidad y el comportamiento esperados del sistema/solución (Modelo de Casos de Uso, Arquitectura de Referencia, Modelo de Diseño (Anexo A), Modelo de Despliegue (Anexo A) y Plan de Pruebas); mientras que el segundo macro-componente define cuatro fases de referencia, las cuales, ejecutadas en conjunto de forma iterativa/incremental, orientan el proceso de desarrollo de la solución propuesta.

5.2.1. Modelo de Casos de Uso del Sistema

La definición de los casos de uso del sistema, parte de la identificación de los requisitos funcionales del mismo. A continuación se detalla el único requisito funcional abstraído para el sistema implementado.

i. **Asociar entidades ontológicas a descriptores de servicios Web y de Telecomunicaciones.** Este requisito funcional se refiere a la capacidad que debe tener el sistema de asociar anotaciones semánticas a los descriptores de servicios Web y de Telecomunicaciones.

A partir del análisis del requisito funcional del sistema, se definió un caso de uso el cual se describe a continuación:

Caso de Uso No. 1: ***Anotar Servicio***
Iniciador: Usuario (Agente Externo: Persona, Aplicación)
Propósito: Anotar semánticamente descriptores de servicio Web y de Telecomunicaciones.

Resumen: Este caso de uso satisface el requisito funcional (i). Realiza una serie de procesos, que en conjunto con la búsqueda de ontologías mediante las funcionalidades ofrecidas por el buscador Web *Google*, permiten la asociación de entidades semánticas a descriptores de servicio, de manera automática.

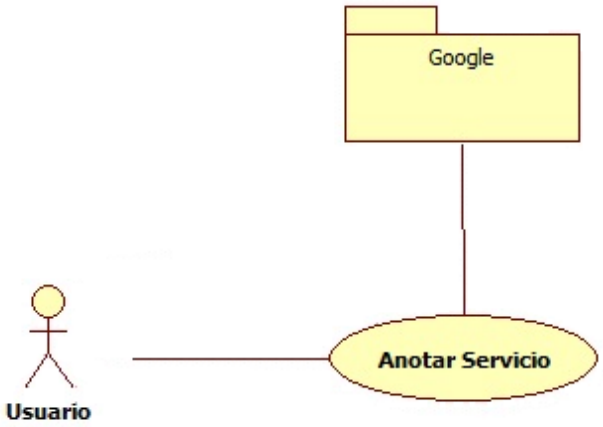


Figura 13. Diagrama de Casos de Uso del Sistema

Junto con los casos de uso, es prioritario definir una arquitectura de referencia, que proporcione una visión de los componentes más significativos del sistema y provea soporte a las actividades definidas para su desarrollo. En la siguiente sección, se describe la arquitectura establecida para el sistema implementado.

5.2.2. Descripción de la Arquitectura del Sistema

El sistema desarrollado para evaluar el mecanismo para la desambiguación lingüística de descriptores de servicio propuesto en el Capítulo 3, y la técnica de asignación de entidades ontológicas a descriptores de servicios propuesta en el Capítulo 4 de este documento, adopta una arquitectura de aplicación en tres capas: la capa de Lógica de Presentación, la capa de Lógica de Negocio y una capa de Soporte. La Figura 18, presenta un diagrama de la arquitectura del sistema, en el cual, se observa como la capa de Lógica de Presentación, implementada mediante una aplicación Web, recibe la petición de anotación del usuario, representada mediante la URI de un servicio Web. La petición es tramitada en la capa de Lógica de Negocio, donde se llevan a cabo una serie de procedimientos necesarios para ejecutar la anotación de descriptores de servicios:

- Extracción de atributos relevantes
- Desambiguación de atributos relevantes
- Búsqueda de ontologías
- Desambiguación de entidades ontológicas
- Selección de ontologías

Posteriormente, las entidades ontológicas que superan el proceso de filtrado (realizado mediante el análisis de la medida de similitud semántica y la calidad de las ontologías) aplicado en el último de los procedimientos (Selección de ontologías), son asignadas a los atributos correspondientes por medio de los atributos de extensión para la anotación definidos por *SAWSDL*. De esta manera, se obtiene un documento que especifica la relación entre las entidades ontológicas de un descriptor y las entidades

ontológicas de un modelo semántico, el cual es dirigido hacia la aplicación Web donde se procesa y se despliega para el usuario.

Todo el comportamiento definido para las capas superiores de la aplicación, se sustenta en un conjunto de API de soporte.

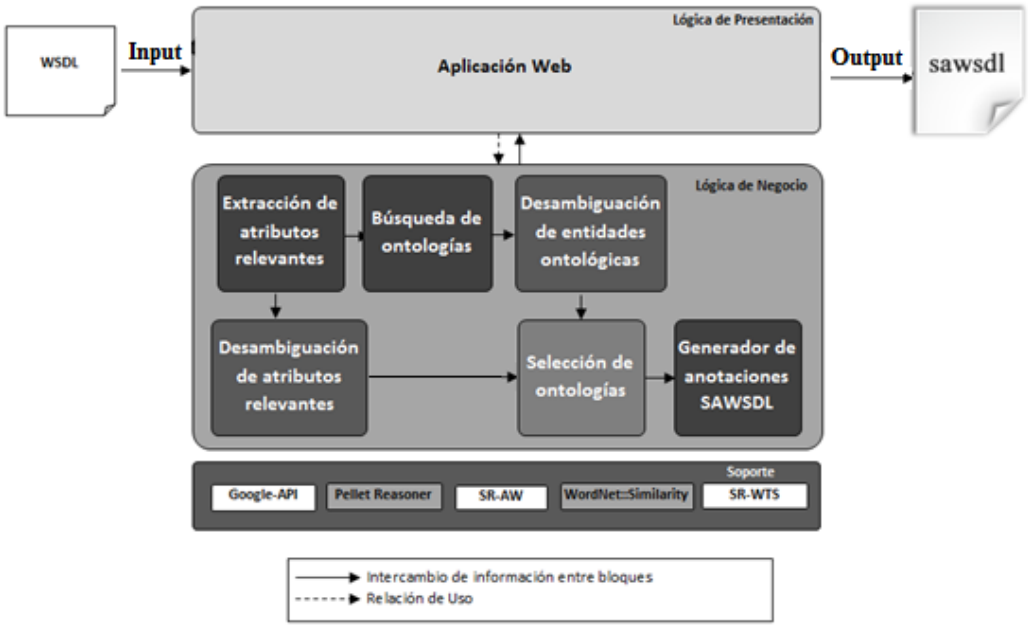


Figura 14. Arquitectura del Sistema

A continuación, se aborda una descripción detallada de los componentes de la arquitectura de referencia del sistema.

5.2.2.1. SOPORTE

Esta capa está constituida por un conjunto de herramientas software, que hacen posible el desarrollo de los procesos de: desambiguación de atributos del descriptor del servicio y de entidades ontológicas; obtención de las ontologías disponibles en la Web, asociadas a un determinado atributo; y análisis de la calidad de las ontologías. Las API que hacen parte de esta capa son:

- ***Google API:*** La implementación de esta API permite el funcionamiento del módulo para la búsqueda de ontologías OWL disponibles en la Web.

- ***Pellet Reasoner:*** Provee una herramienta para razonamiento sobre las ontologías, permitiendo la extracción de entidades ontológicas y sus relaciones, así como también el análisis de la calidad, por medio de la observación de conceptos inconsistentes y la profundidad en las ontologías. Una de las grandes ventajas de este razonador, es ser el único que posee una interfaz nativa de Jena, un framework que facilita el manejo de ontologías OWL y RDF. (Sirin, y otros, 2007)

- ***WordNet::SenseRelate::WordToSet (SR-WTS), SR-AW, WordNet::Similarity:*** Herramientas que proveen el soporte necesario para la desambiguación lingüística de las entidades ontológicas, de los atributos del descriptor de servicio, y para el proceso de filtrado de entidades ontológicas mediante el cálculo de la similitud semántica entre definiciones de sentidos de los atributos del descriptor y las ontologías, respectivamente.

5.2.2.2. LOGICA DE NEGOCIO

En esta capa se implementan y articulan, todos los mecanismos necesarios para manejar las peticiones del usuario y satisfacer los casos de uso definidos para el sistema. Este nivel de la arquitectura, se descompone en un conjunto de unidades funcionales, las cuales interactúan con la capa de Lógica de Presentación y la capa de Soporte.

- ***Extracción de atributos relevantes***

Este módulo recibe la URL de un descriptor de servicio, obtiene su estructura y a partir de ella extrae los atributos que se han definido como relevantes para el desarrollo del presente proyecto. Para el proceso de extracción, el módulo cuenta con la librería predic8, la cual permite recorrer el descriptor del servicio en búsqueda de los atributos del mismo, para luego analizar las palabras que los conforman. Luego, realiza una clasificación gramatical de estos términos para eliminar palabras irrelevantes. Finalmente, entrega como resultado una lista de atributos relevantes con las palabras que los definen.

- ***Desambiguación de atributos relevantes***

En este módulo se implementa el mecanismo para la desambiguación lingüística de descriptores de servicios Web y de Telecomunicaciones, descrito en el Capítulo 3 del presente documento. Este módulo realiza la desambiguación de los atributos del descriptor con relación al contexto del mismo, mediante el uso de la herramienta SR-AW de la capa de soporte. De esta manera, recibiendo los atributos extraídos por el módulo anterior, este módulo entrega al módulo de selección de ontologías uno a uno los atributos desambiguados con sus definiciones correspondientes.

- ***Búsqueda de ontologías***

Este módulo se encarga de la búsqueda de ontologías OWL disponibles en la Web, que se encuentren relacionadas con el atributo a anotar, utilizando para esto el API de Google de la capa de soporte. Recibe del módulo de extracción de atributos, el atributo en consideración, y entrega al módulo de desambiguación de entidades ontológicas, una lista de las ontologías OWL mas relacionadas con el atributo a desambiguar.

- ***Desambiguación de entidades ontológicas***

Este bloque desambigua cada una de las entidades ontológicas recibidas del módulo de búsqueda de ontologías. En primer lugar extrae de cada ontología la entidad que más se relacione con el atributo a anotar y sus clases derivadas, con la ayuda del razonador Pellet de la capa de soporte. Posteriormente, la herramienta para la desambiguación lingüística SR-WTS, también de la capa de soporte, recibe las entidades extraídas de la ontología y desambigua la entidad ontológica con respecto a sus entidades derivadas. Finalmente, entrega al módulo de selección de ontologías la definición del sentido de la entidad ontológica más relacionada con el atributo a anotar.

- ***Selección de ontologías***

Este es el bloque más importante de la arquitectura de referencia del sistema. En él se implementa la técnica para la asignación de entidades ontológicas a los descriptores de servicio, descrita en el Capítulo 4 de este documento. Este módulo, para la selección de ontologías, se compone de dos procesos importantes que permiten filtrar las entidades ontológicas que no produzcan una anotación relevante para el atributo: el primero de ellos es el cálculo de la medida de similitud semántica entre la definición del sentido de la entidad ontológica -obtenida del módulo de desambiguación de ontologías- y la definición del sentido del atributo a anotar -obtenida del módulo de desambiguación de atributos. Este primer proceso se realiza empleando la herramienta *WordNet::Similarity*, de la capa de soporte. El segundo proceso corresponde a un análisis de calidad, aplicado a las ontologías que han sido seleccionadas en el primer proceso, esto con la ayuda del razonador Pellet, herramienta de la capa de soporte.

- ***Generador de anotaciones SAWSDL***

Este módulo permite exponer las entidades ontológicas que han superado el proceso de selección y finalmente han sido asignadas a los atributos de un descriptor determinado. Basado en los atributos de extensión establecidos por SAWSDL, el generador de anotaciones produce un documento en el cual se encuentra la asociación entre los atributos y sus entidades ontológicas correspondientes. En la Figura 15 se observa un ejemplo de un documento WSDL, que contiene algunas etiquetas para ser anotadas semánticamente, la Figura 16 contiene el documento SAWSDL con dichas anotaciones.

```
<wsdl:types>
  <s:schema elementFormDefault="qualified" targetNamespace="http://www.webserviceX.NET">
    <s:element name="GetWeather" >
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="CityName" type="s:string"/>
          <s:element minOccurs="0" maxOccurs="1" name="CountryName" type="s:string"/>
        </s:sequence>
      </s:complexType>
    </s:element>
    .
    .
    .
  </s:schema>
</wsdl:types>
.
.
.
<wsdl:portType name="GlobalWeatherHttpGet">
  .
  .
  .
  <wsdl:operation name="GetCitiesByCountry">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
    <wsdl:input message="tns:GetCitiesByCountryHttpGetIn"/>
    <wsdl:output message="tns:GetCitiesByCountryHttpGetOut"/>
  </wsdl:operation>
</wsdl:portType>
.
.
.
```

Figura 15. Documento WSDL

```
<xsd:service name="GlobalWeather"
        sawsdl:modelReference="http://zaltys.net/ontology/AKTiveSAOntology.owl#WeatherPhenomenon">
</xsd:service>
</xsd:port>
<xsd:type name="GetWeather"
        sawsdl:modelReference="http://zaltys.net/ontology/AKTiveSAOntology.owl#WeatherPhenomenon">
</xsd:type>
<xsd:type name="GetCitiesByCountry"
        sawsdl:modelReference="http://www.w3.org/Consorti/RDFTutorial/rdfs/Countries.owl#Country">
</xsd:type>
.
.
.
```

Figura 16. Documento SAWSDL

5.2.2.3. LÓGICA DE PRESENTACIÓN

En esta capa se presentan las interfaces graficas contenidas en una aplicación web que permiten al usuario llevar acabo la anotación semántica de un servicio. La aplicación cuenta con una interfaz que presenta al usuario un menú a través del cual puede seleccionar una de dos opciones: "*Introducción*" y "*ASA-CS*" *(Automatic Semantic Annotator of Converged Services).* Al acceder a la aplicación se despliega automáticamente la interfaz de introducción (Figura 15), la cual suministra al usuario información relacionada con el sistema. En la opción *ASA-CS* (Figura 16) se presenta la interfaz de anotación como tal.

La interfaz de la opción *ASA-CS* permite al usuario indicar la URL del descriptor de servicio, y realizar la anotación del mismo, para finalmente recibir el resultado que genera el procesamiento de su solicitud por parte del sistema. Este resultado se muestra en la misma interfaz de anotación (Figura 16) en un campo editable donde se puede modificar si se desea, el contenido del documento *SAWSDL*, el cual puede adicionalmente descargarse como un archivo independiente.

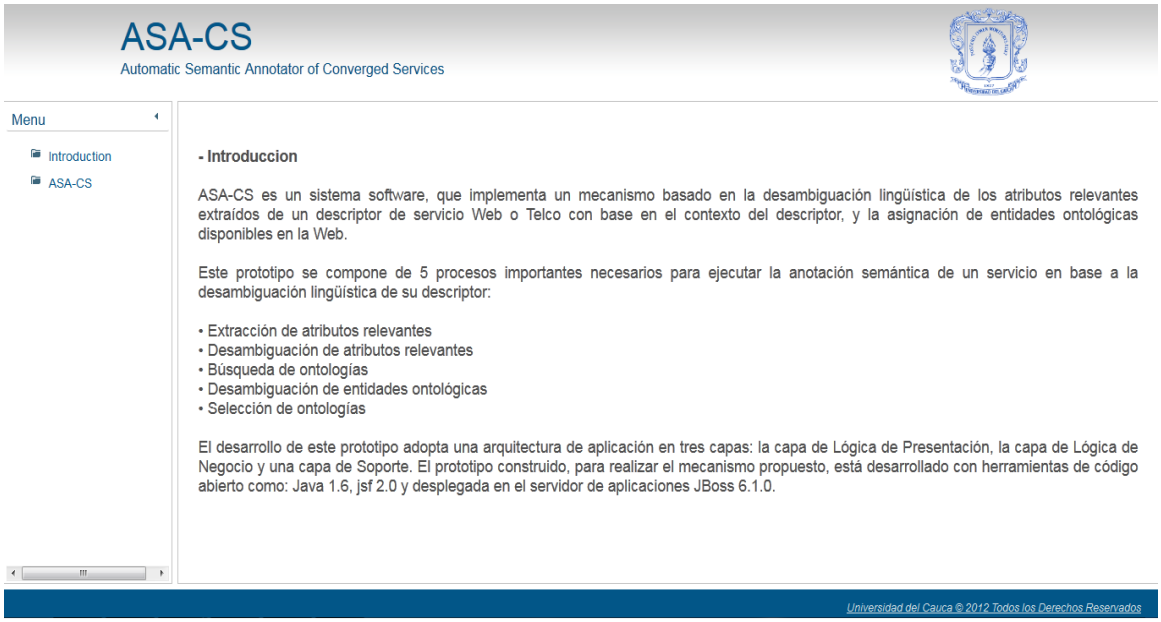


Figura 15. Interfaz de Introducción al sistema ASA-CS

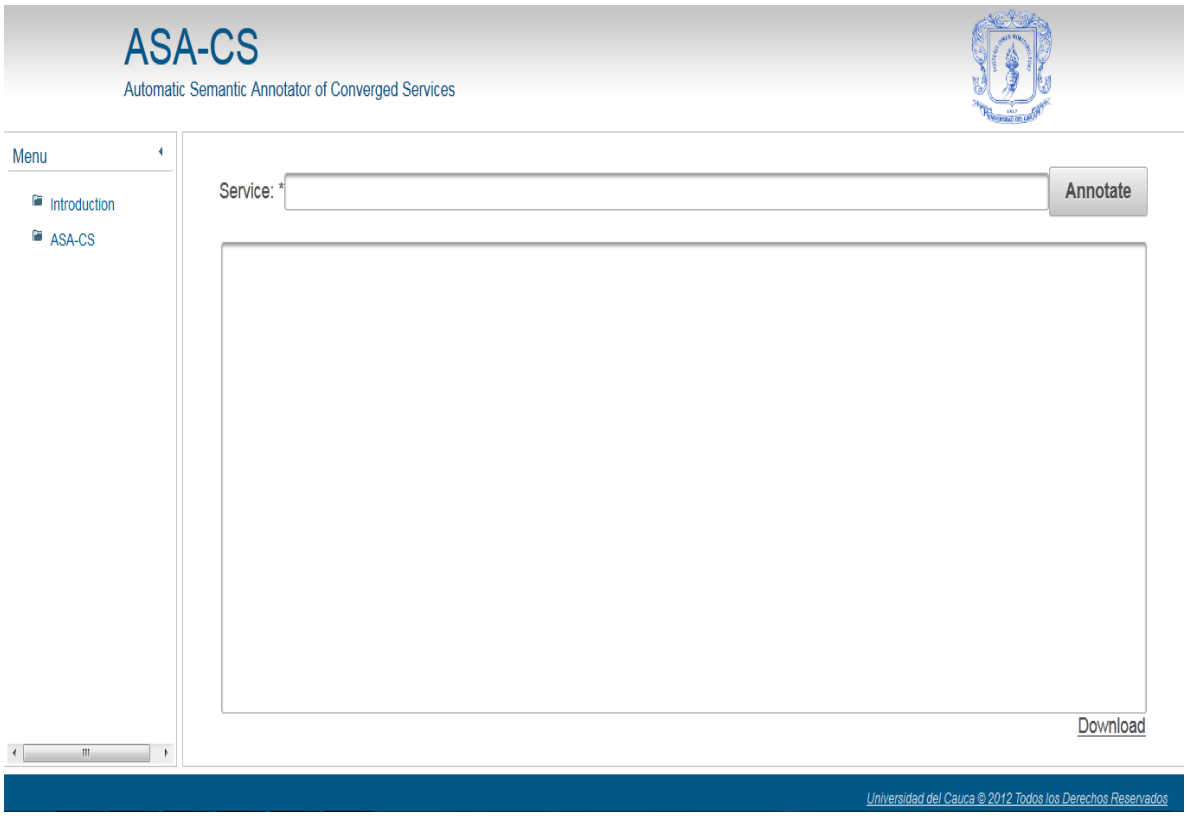


Figura 16. Interfaz para la anotación semántica de servicios Web y Telco

El trabajo realizado alrededor de esta aplicación web, se abordó como parte de la tercera iteración del proceso de desarrollo del sistema. Para su implementación, se empleó la tecnología Java Server Faces, la cual permite establecer una clara separación entre la Lógica de Presentación y la Lógica de Negocio, de forma similar a la alcanzada aplicando el patrón de diseño MVC, en el desarrollo de una aplicación de escritorio. Como contenedor web, para desplegar la aplicación diseñada se empleó el servidor JBoss v6.1.0., el cual implementa las especificaciones *Java Servlet* y *Java Server Pages* (JSP) de *Sun Microsystems* y provee un potente entorno de ejecución. Vale la pena señalar que la capa de Lógica de Presentación, fue desarrollada con fines puramente ilustrativos, es decir, no constituye una parte relevante de la funcionalidad del sistema, concentrada fundamentalmente en la capa de Lógica de Negocio. En consecuencia, es posible separar y empaquetar los componentes de la capa central de la arquitectura, como una aplicación independiente, la cual puede importarse y utilizarse en cualquier otra aplicación que lo requiera.

5.3. PRUEBAS DEL PROTOTIPO

Una vez concluida la implementación del prototipo operacional, es necesario someterlo a un proceso de evaluación y puesta a punto que permita identificar y corregir deficiencias en la operación del sistema y garantizar su óptimo desempeño.

Con el fin de llevar a cabo actividades de evaluación sobre el prototipo desarrollado, se emplea un corpus de servicios anotados semánticamente de forma manual, el cual se ha utilizado como una base de referencia para la comparación de la calidad de las

anotaciones semánticas realizadas por el prototipo. Este corpus se ha denominado *benchmark de referencia*.

Benchmark puede definirse como un punto de referencia, a partir del cual es posible evaluar comparativamente el rendimiento de un sistema respecto de otro, que por lo general representa las mejores prácticas de su dominio (Chillarege, 1999). Para el caso de referencia, el benchmark está integrado por servicios enriquecidos semánticamente, cuyas anotaciones han sido realizadas de manera manual por personal calificado, formado por PhD. Dr. Matthias Klusch[19] y PhD. Dr. Patrick Kapahnke[20], investigadores Senior en el Centro Alemán de Investigación de Inteligencia Artificial. Este benchmark está conformado por 1088 descriptores de servicios Web, y 4 descriptores de servicios de telecomunicaciones, para un total de 1092 descriptores, ellos se encuentran divididos en 13 dominios, de ellos 9 corresponden a servicios Web: Comunicaciones (58 descriptores), Economía (358), Educación (285), Comida (34), Geografía (60), Medicina (73), Simulación (16), Viajes (164), Armamento (40), y 4 a dominios de servicios de telecomunicaciones: MMS (1), Pago (1), SMS (1), Localización de terminal (1).

Para la evaluación del desempeño se tomaron los 10 primeros descriptores de servicios de cada una de las 9 categorías de servicios Web y todos los descriptores de servicios de las 4 categorías de telecomunicaciones. De esta manera, para evaluar la calidad del prototipo determinando qué tan correcta es la desambiguación de los descriptores de servicio, y qué tan adecuadas son las anotaciones realizadas por la técnica de asignación de entidades ontológicas, se requiere precisamente conocer, cuáles deberían ser las anotaciones que el prototipo debería asignar a cada atributo, entendiendo como resultados apropiados aquellos que se ajustan al criterio de un experto. En este sentido, la comparación de las anotaciones realizadas por el prototipo, con las anotaciones realizadas manualmente para un determinado atributo se realiza de forma manual, con el fin de encontrar el número de atributos cuyas anotaciones coincidieran con las anotaciones en el benchmark de referencia. Con el objetivo de realizar esta comparación, se anotaron semánticamente con el prototipo, los descriptores originales (sin anotación manual) de los servicios definidos como referencia.

Para este propósito de evaluación del prototipo, se analizaron 94 descriptores de servicios, y se realizaron en total 2418 comparaciones entre anotaciones, garantizando que cada una de las anotaciones realizadas por el prototipo, fuera comparada con cada una de las anotaciones realizadas manualmente para cada descriptor. En consecuencia, se contaba al final del proceso de evaluación con un resultado que establecía el número de anotaciones coincidentes por cada atributo para un determinado descriptor. Para estimar el nivel de calidad del sistema, se realizó un análisis estadístico basado en la aplicación de un conjunto de medidas empleadas en la evaluación del desempeño de sistemas de recuperación de información. Dichas medidas se describen en la siguiente sección.

5.3.1. Medidas de Desempeño

Una funcionalidad básica del sistema desarrollado en el presente proyecto, consiste en generar un documento contenedor de las anotaciones *SAWSDL* que la herramienta realiza en los descriptores de servicio. En este sentido, es posible evaluar la calidad de los resultados obtenidos en el proceso automático de anotación semántica, mediante el análisis de este documento, a partir de la aplicación de medidas estadísticas

[19] http://projects.semwebcentral.org/users/klusch/
[20] http://projects.semwebcentral.org/users/pkapahnke/

ampliamente empleadas en la caracterización del desempeño de sistemas de recuperación de información (p.ej., motores de búsqueda en internet): *precisión (p)*, *recall (r)*, y *f-measure (f)* (Yatskevich, 2003).

Estas medidas (*precision, recall* y *f-measure*), operan sobre la cardinalidad de tres conjuntos conformados por las anotaciones semánticas generadas por el prototipo: el conjunto de *verdaderas positivas(TP)* (anotaciones semánticas realizadas de forma correcta (manual) de acuerdo con los resultados registrados en el benchmark de referencia), el conjunto de *falsas positivas(FP)* (anotaciones semánticas no coincidentes con la información suministrada en el bechmark de referencia) y el conjunto de *falsas negativas(FN)* (anotaciones semánticas que a pesar de ser relevantes de acuerdo con el benchmark de referencia, no son realizadas por el prototipo). De esta manera, las medidas *precision, recall* y *f-measure* se definen como:

***Precision(p)*:** se refiere a la proporción de anotaciones semánticas clasificadas como *verdaderas positivas,* respecto al número total de anotaciones realizadas el prototipo, esto es:

$$p = \frac{|TP|}{|TP|+|FP|} \qquad \textbf{Ec. 11.}$$

***Recall(r)*:** identifica la proporción de anotaciones semánticas clasificadas como *verdaderas positivas,* respecto al número total de anotaciones consideradas como relevantes (es decir, todas aquellas anotaciones que deben ser realizadas correctamente de acuerdo con el benchmark de referencia):

$$r = \frac{|TP|}{|TP|+|FN|} \qquad \textbf{Ec. 12.}$$

***F-measure(f)*:** relaciona las medidas de *precision* y *recall*, y se emplea para determinar la calidad del proceso de anotación semántica de servicios. Matemáticamente se calcula:

$$f = (2 * r * p)/(r + p) \qquad \textbf{Ec. 13.}$$

De acuerdo con las dos primeras definiciones, la medida de *precision* determina el porcentaje de relevancia del conjunto de anotaciones semánticas realizadas por el sistema, mientras que *recall* estima que porcentaje del total de anotaciones relevantes fue recuperado.

5.3.2. Plan de Pruebas y Resultados Obtenidos

Todas las pruebas a las cuales se hace referencia a continuación, se llevaron a cabo sobre el prototipo generado, al concluir la segunda iteración del proceso de desarrollo de la solución propuesta. De esta manera, se realizó la evaluación del desempeño de las funcionalidades implementadas en la Capa de Lógica de Negocio de la Arquitectura descrita en secciones previas. Asimismo, la ejecución iterativa y la continua verificación de los resultados de este proceso de evaluación, soportaron la puesta a punto del sistema.

A continuación se describen las pruebas de calidad y eficiencia ejecutadas sobre el prototipo desarrollado:

5.3.2.1. Plan de Pruebas

Debido a que la herramienta propuesta para el presente trabajo se desarrolló con el fin de automatizar el proceso de anotación semántica de descriptores de servicio, es de vital importancia la calidad con la que estas anotaciones sean realizadas, por encima del tiempo que le tome a la herramienta realizarlas, ya que como se expone más adelante, el tiempo requerido por ingenieros de conocimiento para la anotación de descriptores de servicio, es mucho mayor que el tiempo que puede tomarle a un sistema automático realizarlo.

Tabla 2. Plan de Pruebas

Asignación de entidades Ontológicas
Prueba de Calidad: permite evaluar la calidad de los resultados de las anotaciones realizadas por el sistema, al comparar las anotaciones de los servicios enriquecidos de manera manual con las anotaciones de los servicios enriquecidos semánticamente por el prototipo.
Anotación semántica de servicios
Prueba de Rendimiento: permite medir el tiempo que le toma al sistema llevar a cabo la anotación semántica de los descriptores de servicios.

Los resultados de las pruebas están condicionados por las prestaciones del equipo donde se ejecuta el software del prototipo. En este caso, el equipo empleado para desplegar el plan de pruebas descrito cuenta con las siguientes especificaciones técnicas:

Tabla 3. Especificaciones Técnicas del Equipo empleado para Pruebas del Prototipo

Microprocesador	Intel Pentium Xeon 3,20GHz
Caché del microprocesador	8MB de caché de nivel 3
Memoria	8192MB
Disco duro	320GB (5400 RPM)
Sistema Operativo	Debian Squeeze

5.3.2.2. Resultados

A partir de la información obtenida en la ejecución del plan de pruebas descrito, se generó el conjunto de gráficas que presenta esta sección, las cuales permiten ilustrar y analizar el desempeño general del sistema desarrollado.

- **Asignación de entidades ontológicas**

Prueba de Calidad: Es la prueba más importante dentro del análisis del prototipo desarrollado. La ejecución de esta prueba, permite evaluar la calidad de los resultados obtenidos por el mecanismo propuesto, en la actividad de asignación de entidades ontológicas a los atributos del descriptor del servicio que se anota.

La gráfica de la Figura 17 presenta los resultados obtenidos en la aplicación de las medidas de *precision*, *recall* y *f-measure* para cada uno de los dominios de servicios Web que se muestran en la imagen. Esta gráfica fue obtenida a partir de la aplicación de las medidas anteriormente mencionadas, para cada uno de los descriptores de servicios que conforman cada uno de los dominios que se evaluaron. Finalmente se aplicó macro promedio por cada dominio.

De acuerdo con estos resultados, el desempeño del sistema es en general adecuado, en la medida en que gran parte de las anotaciones semánticas realizadas coinciden con el benchmark de referencia. Esto se refleja en el valor de la medida de *precision*, el cual es en promedio superior al 88%. El mejor desempeño del sistema se obtiene para el dominio de viajes, el cual contiene servicios con un menor número de atributos a anotar que el resto de dominios (entre 2 y 14 atributos). Lo anterior se concluye de acuerdo con la medida de *f-measure* (superior al 92%), la cual determina que para este dominio, el sistema anota apropiadamente los atributos de los servicios que lo conforman.

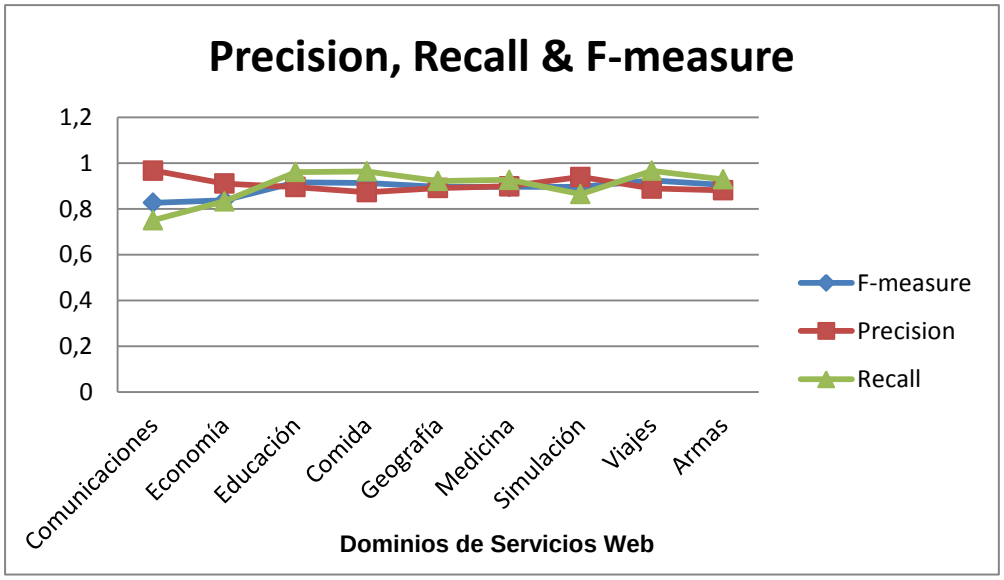


Figura 17. Medidas de desempeño: Precision, Recall y F-measure para el Sistema (Servicios Web)

Por otra parte, el desempeño más pobre del sistema se presenta en los dominios de comunicaciones, geografía y medicina, cuyos servicios contienen la mayor cantidad de atributos a anotar (entre 57 y 66 atributos). En ellos se manifiesta la dificultad que presenta el sistema, para la asignación de entidades semánticas cuando el número de entidades a anotar crece. A pesar de que la mayor parte de los atributos se anotan correctamente (*precision* aceptable), puede observarse que el desempeño del sistema, para estos dominios de servicios con un mayor número de atributos, es menor.

En la Figura 18 se presentan los resultados obtenidos de la aplicación de las medidas de *precision*, *recall* y *f-measure* para diferentes dominios de servicios de Telecomunicaciones.

Conforme con estos resultados, el desempeño del sistema es en general adecuado, de acuerdo con el valor de la medida de *precision*, que es en promedio superior al 94%. Para el dominio MMS se obtiene el mejor desempeño del sistema, el cual contiene un servicio con 6 atributos a anotar, un número menor comparado con el número de atributos contenidos en los descriptores de los servicios de los demás dominios de telecomunicaciones que se anotaron semánticamente (entre 13 y 18 atributos). Lo anterior se refleja en la medida de *f-measure* (superior al 88%), que permite concluir que para el dominio MMS, el sistema anota apropiadamente los atributos del servicio que lo conforma.

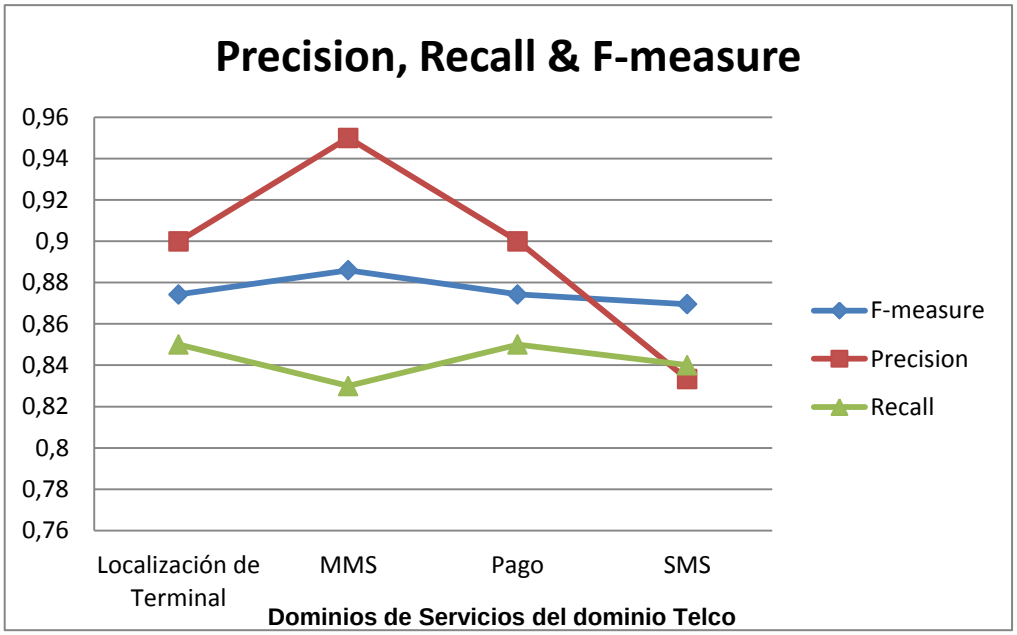


Figura 18. Medidas de desempeño: Precision, Recall y F-measure para el Sistema (Servicios Telco)

Por otra parte, el desempeño más pobre del sistema se presenta en el dominio de SMS, el cual contiene la mayor cantidad de atributos a anotar (18 en total), dentro de los dominios de telecomunicaciones que se utilizaron.

En general, el sistema presenta un buen desempeño en la anotación de descriptores cuyo número de atributos se encuentra entre 2 y 34, presentando una medida promedio de *f-measure* entre el 87% y 92%, mientras que para descriptores con un número de atributos entre 35 y 66, el sistema reduce un poco su desempeño, con una medida promedio de *f-measure* entre el 82% y 87%. Esta conclusión es obtenida a partir del cálculo del promedio entre el número de atributos analizados por cada dominio, y la medida de *f-measure* obtenida para cada uno de ellos.

El bajo desempeño del sistema al anotar semánticamente descriptores de servicio con un mayor número de atributos, se debe a que el mecanismo para la desambiguación de atributos aumenta su probabilidad de errar en la asignación de los sentidos, debido a la ambigüedad que involucra un contexto lingüístico más grande. Además, la complejidad de este proceso aumenta con relación al tamaño del contexto lingüístico con el cual se trabaje. Por lo tanto, a mayor número de atributos a anotar, habrá un mayor número de atributos a desambiguar, y el contexto lingüístico también será más amplio. Este proceso podría optimizarse estableciendo un límite de atributos a desambiguar, es decir al recibir un descriptor de servicio con un número de atributos entre 35 y 66, estos podrían dividirse en dos conjuntos para realizar el proceso de desambiguación. Así, en caso de procesar un descriptor de servicio con 66 atributos, la desambiguación lingüística se realizaría en un principio para los 33 primeros atributos, y luego para los 33 restantes, disminuyendo el contexto de desambiguación a la mitad. Sin embargo, esto podría provocar un cambio en el contexto lingüístico, disminuyendo así la calidad del proceso de anotación semántica.

Sin embargo, a pesar del inconveniente previamente mencionado, la calidad de los resultados obtenidos mediante la aplicación del mecanismo para la desambiguación lingüística de los atributos del descriptor, y la técnica para la asignación de entidades ontológicas a los atributos de un descriptor es, en general, sustancialmente mejor respecto al desempeño de sistemas semiautomáticos de anotación de servicios.

La Figura 19, ilustra la relación existente entre las medidas de *precision* y *recall*. En esta gráfica se visualiza como el desempeño del sistema está en general ubicado en una zona de alta *precision* (superior al 95%) y *recall* considerable (superior al 87%), lo cual implica que el mecanismo implementado es significativamente bueno en la asociación de entidades semánticas a los atributos de descriptores de servicios.

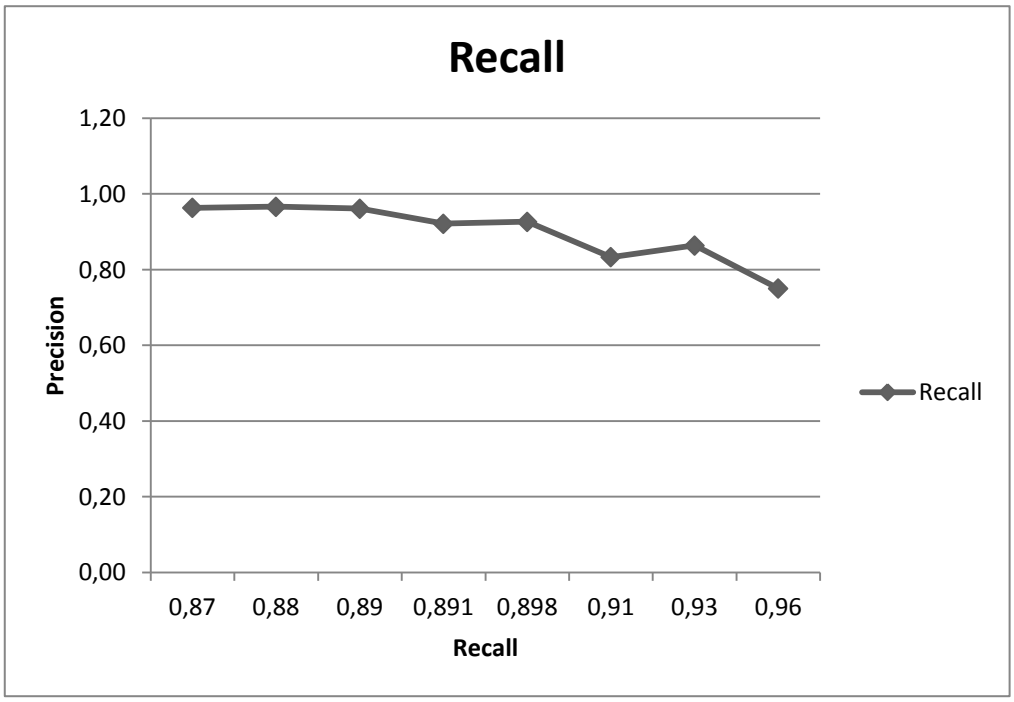


Figura 19. Gráfica de Precision vs Recall

- **Anotación semántica de servicios**

Prueba de Rendimiento: a partir de la ejecución de esta prueba, se determinó el tiempo que tarda el sistema en anotar semánticamente el descriptor de un servicio. A pesar de que no existen valores determinados del tiempo que le toma a un anotador humano realizar el proceso, según la experiencia en el desarrollo de este trabajo, se puede decir que el tiempo para la realización de esta tarea es aproximadamente 4 horas para la anotación de un servicio con un número de atributos entre 10 y 15. Por otro lado, los desarrolladores de herramientas semiautomáticas de anotación concuerdan en una conclusión: La anotación manual es un proceso costoso, propenso a errores y lento debido a dos razones principales: la primera es el análisis de múltiples fuentes de información necesarias para comprender la funcionalidad de los servicios web, y la segunda es la búsqueda de modelos semánticos relacionados con los descriptores de servicio a anotar (Aksoy, y otros, 2011) (Bouchiha, y otros, 2012). La plataforma propuesta en el presente trabajo de grado, aporta a la solución del anterior problema en la medida en que prescinde del componente manual para la anotación semántica de servicios, convirtiéndolo en un proceso completamente automático que permite la disminución del tiempo de anotación y la existencia de posibles errores.

De acuerdo con el comportamiento de la curva que se ilustra en la Figura 21, se observa como existe una relación de linealidad entre el número de atributos a anotar, y el tiempo que toma el sistema para anotar semánticamente dichos atributos. Por otra parte, se considera que el tiempo empleado en esta operación es aceptable, teniendo

en cuenta la complejidad del procedimiento de desambiguación lingüística tanto de los atributos a anotar, como de las entidades ontológicas, el filtrado de las entidades ontológicas que se usaron para la anotación semántica, y la gran cantidad de transacciones llevadas a cabo en la capa de *Soporte*.

El razonamiento de ontologías, necesario para el proceso de desambiguación de entidades ontológicas (sección 4.3.2), es sin lugar a dudas el proceso que consume la mayor cantidad de tiempo (aproximadamente 40% del 100% del tiempo que toma llevar acabo todos los procesos), debido a que el razonador utilizado (Pellet) debe procesar, analizar y reorganizar la ontología de manera que el prototipo pueda detectar las entidades ontológicas coincidentes de manera sintáctica con los atributos. Este proceso podría optimizarse analizando la calidad de las ontologías antes de procesarlas, de esta forma el número de ontologías a analizar sería menor, y el sistema podría tardar menos tiempo en obtener las ontologías listas para su análisis sintáctico.

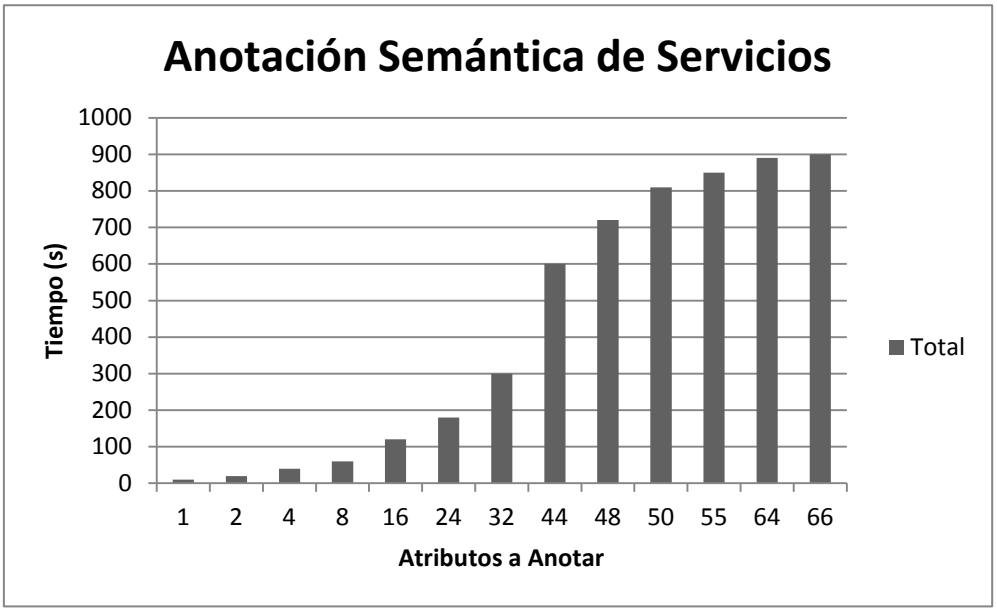


Figura 20. Gráfica de rendimiento del sistema en la anotación semántica de servicios

5.4. RESUMEN

En este capítulo, se abordó una descripción detallada del proceso de desarrollo llevado a cabo alrededor de la construcción de un sistema software, que implementa el mecanismo propuesto para la desambiguación lingüística de los atributos de descriptores de servicio, así como la técnica planteada para la asignación de entidades ontológicas disponibles en la Web a dichos descriptores. De esta manera, se identificaron inicialmente los requisitos funcionales, a partir de los cuales fue posible definir los casos de uso del sistema y la arquitectura de referencia que permitió soportar el desarrollo del mismo.

Asimismo, se realizó una descripción del benchmark de referencia utilizado para soportar el proceso de evaluación y puesta a punto del sistema. Posteriormente, se definieron las medidas de desempeño, empleadas para determinar la calidad del procedimiento automático de anotación semántica de servicios Web y de Telecomunicaciones.

De acuerdo con estas medidas, el desempeño general del sistema es satisfactorio en cuanto a su capacidad para asociar anotaciones semánticas a un descriptor, de acuerdo con el dominio del mismo, ajustándose de manera aproximada al criterio de un evaluador con conocimiento del dominio de los servicios Web y de Telecomunicaciones. De esta manera, se demostró que el mecanismo para la desambiguación lingüística de los atributos del descriptor, efectivamente permite la asociación de entidades ontológicas a los atributos contenidos en los descriptores de servicios, al mismo tiempo que permite optimizar el consumo de tiempo y dinero empleados en la anotación manual de descriptores, lo cual evidencia la viabilidad y favorabilidad de la propuesta presentada en el presente trabajo de grado.

6. CONCLUSIONES Y TRABAJO FUTURO

6.1 CONTRIBUCIONES

Entre las principales contribuciones de éste proyecto de grado se destacan las siguientes:

- La definición de técnicas y algoritmos que permiten abstraer características comunes entre servicios Web y de Telecomunicaciones con el fin de realizar una anotación semántica de descriptores de servicio para cualquiera de los dos dominios.

- La definición de un mecanismo para identificar e interpretar los elementos relevantes de un descriptor de servicio, en función de su contexto lingüístico. Este mecanismo se compone de una serie de algoritmos, que comprenden procesos de extracción de palabras relevantes contenidas en el descriptor de servicio y de métodos de desambiguación del sentido de la palabra basados en conocimiento, los cuales se propusieron a partir del estudio de diferentes técnicas para la desambiguación del sentido de la palabra.

- La definición de un mecanismo para la búsqueda automática de ontologías disponibles en la Web en formato OWL, apoyados en un motor de búsqueda que garantiza resultados de alta precisión. Con esto se contribuye a fomentar la reutilización de ontologías previamente especificadas bajo una terminología consensuada, y disponibles para su reutilización por cualquier agente de acuerdo a sus necesidades.

- La definición de un mecanismo para desambiguar conceptos ontológicos en función de sus conceptos padres y conceptos hijos. Este mecanismo se compone de procesos de razonamiento e identificación de los conceptos dentro de la estructura de una ontología y de métodos de desambiguación del sentido de la palabra basados en conocimiento.

- El prototipo funcional *ASA-CS*, que constituye una herramienta automática para la anotación semántica de servicios convergentes con base en la desambiguación lingüística de su descriptor. Esta herramienta permite llevar a cabo la adición de componentes semánticos a servicios Web y de Telecomunicaciones (que cuenten con una interfaz WSDL), mediante la interpretación del contexto de su descriptor, empleando para esto ontologías de dominio disponibles en la Web y descritas en *OWL*. Los resultados de esta herramienta permite disminuir el esfuerzo y el consumo de recursos necesarios para realizar anotaciones semánticas de manera manual.

- Como contribución final se destaca el aporte de la plataforma *ASA-CS*, al trabajo de grado de maestría “Automatización del Enriquecimiento Semántico De Servicios en la Web” del Ing. Leandro Ordoñez Ante, desarrollada al interior de la Universidad del Cauca. Esta herramienta sirve como alternativa para comparar el mecanismo de enriquecimiento semántico basado en aprendizaje automático, que trata de inferir la semántica latente en los descriptores de servicio, propuesto en este trabajo de maestría. De esta manera se contribuye a la investigación en torno a la temática de la anotación semántica de servicios, incentivando futuros trabajos dentro del grupo Ingeniería Telemática (GIT) de la

Universidad del Cauca, los cuales integren las líneas de investigación relacionadas con aplicaciones y servicios en internet y servicios avanzados de telecomunicaciones.

6.2 CONCLUSIONES

Dentro del proyecto realizado, se ha abordado una propuesta de anotación semántica de servicios, la cual tiene en cuenta no solo el atributo del servicio a anotar, como usualmente se realiza en los mecanismos semiautomáticos de anotación, sino también el contexto lingüístico del documento descriptor. Para el análisis del atributo con relación al contexto del descriptor, se propuso un mecanismo de desambiguación lingüística que permitió plantear una técnica que asigna entidades ontológicas disponibles en la Web a los descriptores de servicios de manera automática. Esta técnica se soporta principalmente en un filtro de ontologías, el cual, de acuerdo con desambiguación lingüística del atributo a anotar y un análisis de las entidades ontológicas, permite seleccionar la más adecuada para el proceso de anotación.

Al finalizar este proyecto se entregó una herramienta denominada *ASA-CS*, la cual se basa en el mecanismo de desambiguación y en la técnica de asignación de entidades que se mencionó anteriormente. La herramienta permite realizar la desambiguación lingüística de atributos del descriptor según el contexto del mismo, así como la desambiguación de las entidades ontológicas a utilizar para el proceso de anotación. La herramienta permite la anotación semántica de descriptores de servicio Web y de Telecomunicaciones de manera automática, ya que el filtro de ontologías que contiene permite la selección de la entidad o entidades ontológicas adecuadas para el atributo según el tipo de servicio al cual pertenece. Finalmente, la herramienta entrega un archivo SAWSDL por cada descriptor, el cual presenta la relación entre los atributos y sus correspondientes entidades ontológicas.

Lo anterior permite concluir que el trabajo realizado, facilita el proceso de anotación semántica de servicios, a partir de la desambiguación lingüística de los atributos de los descriptores a anotar, y teniendo en cuenta criterios de similitud entre las definiciones de los sentidos, tanto de los atributos, como de las entidades ontológicas a usar, prescindiendo así de la intervención manual a lo largo del proceso.

A continuación se describen las principales conclusiones obtenidas a partir de la ejecución del presente proyecto:

- La aplicación de un mecanismo basado en conocimiento para la desambiguación lingüística de los descriptores de servicio, permite la adecuada asignación de sentidos a los atributos con relación al contexto lingüístico que los contiene (como se describe en la sección 3.3), facilitando que las anotaciones semánticas asociadas al atributo sean coherentes con el dominio del servicio, a diferencia de lo que sucede en algunos de los sistemas de anotación semántica (como el presentado en (H. Yoo, 2011)), que consideran los atributos de manera individual, lo cual provoca que la selección de la entidad semántica pueda no corresponder con el dominio del servicio al que pertenece el servicio.

- A partir del estudio de los mecanismos para la desambiguación lingüística (sección 2.2.3.1), se lograron identificar dos grupos de métodos: uno basado en conocimiento, los cuales aprovechan recursos externos que les permiten utilizar conocimiento lingüístico previamente adquirido, y otro basado en

corpus, los cuales emplean técnicas estadísticas y algoritmos de aprendizaje automático. Dadas las características del presente proyecto, se consideró que el enfoque más adecuado para la desambiguación lingüística de los descriptores de servicios, es aquel cuyo conocimiento se basa en recursos léxicos externos, en tanto que el conocimiento que manejan es mucho más extenso que el tratado por los métodos basados en corpus, permitiendo la desambiguación de textos que traten cualquier tipo de tema. Dentro de este enfoque, el mejor mecanismo para la desambiguación lingüística, se trata de aquel que considera la palabra ambigua, sus palabras vecinas, y las palabras relacionadas con la palabra objetivo, cubriendo de esta manera un contexto lingüístico completo.

- A pesar de los errores que la intervención manual pueda introducir en la anotación semántica de servicios, reemplazar el análisis y procedimientos realizados por personal calificado es un proceso complicado. Sin embargo, el uso de la medida de similitud semántica adecuada en el presente trabajo, permite tomar la decisión de que entidad ontológica usar para la anotación de un determinado atributo.

- En cuanto a la medida de similitud semántica empleada para el filtrado de ontologías, se identificaron 5 enfoques (descritos en la sección 3.3.2), de los cuales, para el presente proyecto se consideró adecuado aplicar aquel que se basa en la estructura y contenido de *WordNet*, ya que permite establecer relaciones de similitud entre conceptos, además presenta una fácil implementación y mejor desempeño que las otras medidas de similitud presentadas.

- Gracias a que los servicios de telecomunicaciones pueden describirse por medio de un documento *WSDL*, el mecanismo propuesto para la desambiguación de atributos y la técnica para la asignación de entidades ontológicas en los descriptores, pueden utilizarse sin mayor inconveniente con los descriptores de servicios de Telecomunicaciones. Por lo tanto la herramienta *ASA-CS*, desarrollada en el presente trabajo de grado, puede ser utilizada para la anotación de servicios tanto Web como Telco.

- A pesar de la complejidad computacional y de consumir la mayoría del tiempo requerido por la herramienta *ASA-CS* para el proceso de anotación, la implementación de un razonador (Pellet) dentro del prototipo es realmente necesaria, debido a que la técnica de *Asignación de entidades ontológicas* (sección 4.3) requiere de un proceso de razonamiento de ontologías con el fin de encontrar la entidad ontológica coincidente con el atributo que se desea anotar, de lo contrario el proceso debería ser realizado de forma manual. Además, si bien puede ser seleccionada una entidad ontológica semánticamente adecuada para el atributo a anotar, puede que la ontología a la que pertenece, y por ende sus conceptos derivados, no se encuentren formados correctamente, lo que produciría que la anotación no ofrezcan enriquecimiento alguno.

- Gracias a las pruebas realizadas sobre la plataforma *ASA-CS*, y el análisis de los resultados obtenidos, puede concluirse que el estudio del contexto lingüístico de los descriptores de servicio, es realmente una técnica efectiva para la adición de componentes semánticos sobre los mismos, la cual permite junto con otros procedimientos, automatizar el proceso de anotación, prescindiendo de la intervención manual, y reduciendo por tanto el consumo de

recursos (dinero, tiempo) necesarios para la operación de sistemas manuales o semiautomáticos.

- Las pruebas de desempeño realizadas sobre la plataforma *ASA-CS* permiten concluir que: el tiempo que toma realizar la anotación semántica de manera automática es considerable, debido a que dicho procedimiento involucra un gran número de procesos complejos y dispendiosos, relacionados principalmente con el razonamiento sobre las ontologías usadas para la anotación, y la desambiguación lingüística tanto de los atributos relevantes a anotar, como de las entidades ontológicas. Sin embargo, en contraste con el tiempo que toma realizar dicho procedimiento de forma manual, los resultados de estas pruebas son muy favorables, lo cual puede constituirse en una disminución del tiempo de despliegue de nuevas soluciones en las empresas del sector.

- Los resultados entregados por la plataforma *ASA-CS* permiten concluir (de acuerdo con las medidas de *Precision*, *Recall y F-measure*) que el mecanismo para la desambiguación lingüística de los atributos de descriptores de servicios Web y de Telecomunicaciones, así como la técnica propuesta para la asignación de entidades ontológicas a los atributos de descriptores de servicios, se ajustan de manera aproximada al criterio de un evaluador con conocimiento de este dominio. Lo anterior evidencia la viabilidad y favorabilidad de la propuesta presentada, en la desambiguación lingüística de descriptores de servicio, y por ende en la anotación semántica de servicios de forma automatizada.

6.3 TRABAJOS FUTUROS

Esta tesis puede considerarse como un estudio inicial de mecanismos para la automatización de la anotación semántica de servicios convergentes, que contribuye en la solución del problema de recuperación de servicios empleando motores de búsqueda. De esta manera, en el campo de investigación del presente trabajo de grado, se proponen los siguientes trabajos futuros:

El conjunto de algoritmos propuestos dentro del presente proyecto, pueden ser evaluados y adaptados a otros dominios de aplicación, con el fin de proponer un mecanismo de anotación de servicios, que puedan ser aplicados en distintos campos del conocimiento aparte de los dominios Web y de Telecomunicaciones.

Adaptación de la plataforma Automatic Semantic Annotator of Converged Services (ASA-CS) para la anotación semántica de grandes colecciones de descriptores WSDL

La secuencia de procedimientos aplicados por la plataforma ASA-CS para la anotación semántica de un descriptor de servicio WSDL, puede ser aplicada cíclicamente a un conjunto amplio de descriptores de servicio, con el fin de automatizar la anotación semántica de una gran colección de descriptores. A través de un módulo que permita acceder al repositorio de descriptores de servicio y someter uno a uno al procesamiento de la herramienta ASA-CS, para obtener finalmente un conjunto de documentos SAWSDL.

Soporte de la plataforma Automatic Semantic Annotator of Converged Services (ASA-CS) para descriptores de servicio WADL y WSDL 2.0

La plataforma propuesta en este trabajo de grado, se basa en la anotación semántica de descriptores de servicio Web y de Telecomunicaciones especificados en los lenguajes WSDL y WSDL 2.0. Sin embargo, podría considerarse la posibilidad de realizar modificaciones en la herramienta, de modo que descriptores de servicio WADL, puedan también ser anotados por el mecanismo propuesto en el presente proyecto.

Extensión de la técnica de asignación de entidades propuesta hacia el uso de múltiples tipos de ontologías

La técnica propuesta realiza la anotación semántica de servicios mediante la asociación de entidades de ontologías OWL, a los atributos de los descriptores de servicio considerados. No obstante, podría considerarse adicionar el uso de otros lenguajes de descripción de ontologías (DAML, OWL, RFD, entre otros), aumentando las posibilidades de encontrar entidades ontológicas apropiadas para el proceso de anotación.

Experimentación de la plataforma Automatic Semantic Annotator of Converged Services (ASA-CS) en un ambiente real

Se propone realizar la experimentación de la plataforma en un ambiente real, para determinar su desempeño, y validar su utilidad en actividades de recuperación de servicios por medio de motores de búsqueda.

Construcción de mecanismos de descubrimiento y composición de los servicios anotados por la plataforma (ASA-CS)

Se plantea la construcción mecanismos de descubrimiento y composición para los servicios Web y de Telecomunicaciones anotados por la plataforma desarrollada en el presente trabajo.

Extensión del mecanismo de anotación a otro tipo de recursos

La plataforma propuesta (ASA-CS) realiza la anotación de servicios Web y de Telecomunicaciones, se propone extenderla para ser aplicada a otro tipo de recursos, como contenido web por ejemplo.

Optimización de procesos realizados por la plataforma (ASA-CS)

Debido a que se ha determinado los procesos realizados por la plataforma que consumen la mayor cantidad de tiempo (Razonamiento de ontologías), y que causan una disminución en el desempeño del sistema (Desambiguación lingüística de atributos del descriptor), se propone como trabajo futuro optimizar estos proceso para obtener mejores resultados que los logrados en el presente trabajo de grado.

Referencias

A Method forword sense disambiguation of unrestricted text. **Rada, M. y D., Moldovan. 1999.** Maryland : s.n., 1999.

A Service Description Language for the Internet of Services. **Cardoso, J.; Winkler, M.; Voigt, K. 2009.** Dresden, Germany : s.n., 2009.

A translation approach to portable ontology specifications. **Gruber, T. 1993.** 2, s.l. : Academic Press Ltd., 1993, Vol. 5.

Agirre, Eneko y Martinez, D. 2001. Learning class-to-class selectional preferences. Toulouse : s.n., 2001.

Akkiraju, R. y Bournez, C. 2007. Semantic Annotations for WSDL and XML Schema — Usage Guide. [En línea] 29 de Agosto de 2007. http://www.w3.org/TR/sawsdl-guide/#invocation.

Alliance, Open Mobile. 2012. OMA. *Open Mobile Alliance.* [En línea] 2012. [Citado el: 28 de 10 de 2012.] http://www.openmobilealliance.org/AboutOMA/Default.aspx.

Alonso, Grettel Barceló. Junio 2010. Desambiguación de los sentidos de las palabras en español usando textos paralelos. Junio 2010.

An adapted lesk algorithm for word sense disambiguation using WordNet. **Banerjee, Satanjeev y Pedersen, T. 2002.** s.l. : CICLing, 2002.

An Efficient WS-QoS Broker Based Architecture for Web Services Selection. **T.Rajendran and P.Balasubramanie and Resmi Cherian. 2010.** 9, 2010, International Journal of Computer Applications, Vol. 1, págs. 79-84.

An Empirical Evaluation on Semantic Search Performance of Keyword-Based and Semantic Search Engines: Google, Yahoo, Msn and Hakia. **Tümer, D., Shah, M. A. y BitirimY. 2009.** [ed.] IEEE Computer Society. 5, Washington : s.n., 2009, págs. 51-55.

Aol. Aol. [En línea] [Citado el: 02 de 11 de 2012.] http://www.aol.com/.

Applying recommender system based mashup to Web-telecom hybrid service creation. **Jie, Guo and Bo, Cheng and Junliang, Chen and Xiangtao, Lin. 2009.** [ed.] IEEE Press. Piscataway, NJ, USA : s.n., 2009, págs. 3321-3325.

Ask. Ask. [En línea] [Citado el: 02 de 11 de 2012.] http://www.ask.com/.

Automatic Tag Identification in Web Service. **J. R. Falleri, Z. Azmeh, M. Huchard, C. Tibermacine. 2010.** [ed.] Joaquim and Cordeiro, José Filipe. 5 de 11 de 2010, The International Conference on Web Information Systems and Technology (WEBIST'10), Spain, págs. 40-47.

AUTOMATIC TAG IDENTIFICATION IN WEB SERVICE. **J. R. Falleri, Z. Azmeh, M. Huchard, C. Tibermacine. 2010.** [ed.] Joaquim and Cordeiro, José Filipe. 5 de 11 de 2010, The International Conference on Web Information Systems and Technology (WEBIST'10), Spain, págs. 40-47.

Breis, J. 2003. Un Entorno de Integración de Ontologías para el Desarrollo de Sistemas de Gestión de Conocimiento. 2003.

Búsqueda sobre catálogos basada en ontologías. **Pérez Sosa, A. y Proenza Arias, Y. E. 2011.** Ciudad de la Habana. Cuba : s.n., 27 de 04 de 2011, Revista Digital Sociedad de la Información.

Cherifi, Chantal, Labatut, Vincent, Santucci, J. François. 2010. Benefits of Semantics on Web Service Composition from a Complex Network Perspective. 2010.

Chillarege, R. 1999. *Software Testing Best Practices.* Center for Software Engineering, IBM Research. 1999. IBM Technical Report.

Chinnici, R.; Moreau, J. J.; Ryman, A.; Weerawarana, S. 2007. Web Services Description Language (WSDL). *W3C.* [En línea] 2007. http://www.w3.org/TR/wsdl20/.

Christensen, E.; Curbera, F.; Meredith, G.; Weerawarana, S. 2001. Web Services Description Language (WSDL) 1.1. [En línea] w3c, 2001. [Citado el: 23 de 10 de 2012.] http://www.w3.org/TR/wsdl.

Clustering WSDL Documents to Bootstrap the Discovery of Web Services. **Elgazzar, Khalid and Hassan, Ahmed E. and Martin, Patrick. 2010.** [ed.] IEEE Computer Society. 8, Washington, DC, USA : s.n., 2010, págs. 147-154.

Comparison of reasoners for large ontologies in the OWL 2 EL profile. **Dentler, K., y otros. 2011.** 2, s.l. : IOS Press, 2011, Vol. 2.

Convergence of Application Services in Next-generation Networks. **Anne Y. Lee, Abdi Modarressi, Seshadri Mohan. 2012.** s.l. : GUEST, 2012.

Cowie, J., Guthrie, J. y Guthrie, L. 1992. Lexical disambiguation using simulated annealing. *Proceedings of the 14th conference on Computational linguistics.* Nantes, France : s.n., 1992.

D. Booth, C. Liu. 2007. *Web Services Description Languaje (WSDL) Version 2.0 Part 0: Primer.* [En línea] Junio de 2007. http://www.w3.org/TR/wsdl20-primer/.

Debugging OWL ontologies. **Parsia, B., Sirin, E. y Kalyanpur, A. 2005.** s.l. : ACM, 2005.

Design of Network Resource Federation towards Future Open Access Networking. **Michiaki Hayashi, Nobutaka Matsumoto, Kosuke Nishimura, Hideaki Tanaka. 2011.** Jun de 2011, Proceedings of The Seventh Advanced International Conference on Telecommunications (AICT), págs. 130 - 134.

Discovering Semantic Web services using SPARQL and intelligent agents. **M. Sbodio, D. Martin, C. Moulin. 2012.** 14, 2012, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 8.

—. **Sbodio, M., Martin, D. y Moulin, C. 2012.** 14, 2012, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 8.

E. Christensen, F. Curbera, G. Meredith, S. Weerawarana. 2001. *Web Service Description Languaje.* [En línea] Marzo de 2001. http://www.w3.org/TR/wsdl.

Enabling technology for knowledge sharing. **Neches, R., y otros. 1991.** 3, 1991, Vol. 12.

Exalead. Exalead. [En línea] [Citado el: 02 de 11 de 2012.] http://www.exalead.com/search/.

Extended Gloss Overlaps as a Measure of Semantic Relatedness. **Banerjee, S. y Pedersen, T. 2003.** Acapulco, Mexico : Morgan Kaufmann Publishers Inc., 2003.

Extending next generation network (NGN) architecture for connection-oriented transport. **F. Baroncelli a, B. Martini a, V. Martini b, P. Castoldi b. 2011.** 9, 2011, Computer Communications, Vol. 34, págs. 1100-1111.

Gale, W., K. Church y D. Yarowsky. 1992. Estimating upper and lower bounds on the performance or word-sense disambiguation programs. 1992.

Ganesan, V., Swaminathan, R. y Thenmozhi, M. 2012. Similarity Measure Based On Edge Counting Using Ontology. 2012. Vol. 3.

Gangemi, A., y otros. 2005. Ontology evaluation and validation. 2005.

GESFOR, y otros. 2011. *State of the art in the field of Mashup concepts.* ICT OMELETTE. 2011. Reporte.

Gómez, S., Chesñevar, C. y Simari, G. 2008. Hacia la solución del problema de chequeo de instancia en ontologías inconsistentes usando argumentación rebatible. 2008.

Google. 2012. Custom Search. [En línea] 13 de 06 de 2012. [Citado el: 02 de 11 de 2012.] https://developers.google.com/custom-search/docs/start?hl=en#overview.

—. Google. [En línea] [Citado el: 02 de 11 de 2012.] http://www.google.com.

—. Google Guide. *Marking searching even easier.* [En línea] [Citado el: 02 de 11 de 2012.] http://www.googleguide.com/advanced_operators.html.

Grette, A. 2010. Desambiguación de los sentidos de las palabras en español usando textos paralelos. México D.F : s.n., 2010.

GSMA. 2012. GSMA. [En línea] 2012. [Citado el: 28 de 10 de 2012.] http://www.gsma.com/.

H. Yoo, Y. Park, H. Bae. 2011. Semi-atutomatic Semantic Service Annotation for SOAP and REST Web Services. 2011.

Hadley, M. 2009. *Web Application Description Language.* [En línea] Agosto de 2009. http://www.w3.org/Submission/wadl/.

Hadley, Marc. 2009. Web Application Description Language. [En línea] 31 de 08 de 2009. http://www.w3.org/Submission/wadl/.

hakia, Inc. Hakia. [En línea] [Citado el: 02 de 11 de 2012.] http://www.hakia.com/.

Hernández y Calderón, L. 2010. *AUTOMATIZACION DE PROCESOS DE NEGOCIO USANDO SERVICIOS WEB SEMANTICOS.* UNIVERSIDAD DE PAMPLONA. 2010. Tesis pregrado.

Hirst, G. and St-Onge, D. 1998. Lexical Chains as representation of context for the detection and correction. [aut. libro] Christiane Fellbaum. *In WordNet: An electronic lexical database.* 1998.

Ide, N. y Véronis, J. 1998. Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art. *Computational Linguistics.* 1998.

Investigating Web APIs on the World Wide Web. **Maleshkova, M.; Pedrinaci, C.; Domingue, J. 2010.** 2010, ECOWS'10, págs. 107-114.

Izquierdo, R. 2010. Una aproximación a la desambiguación del sentido de las palabras basada en clases semánticas y aprendizaje automático. Alicante : s.n., 2010.

J. Domingue, M. Maleshkova, C. Pedrinaci. 2009. Supporting the Creation of Semantic RESTful Service Descriptions. 2009.

Jiang, J. y D., Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. 1997.

Kikas, T. y Treumuth, M. 2007. Word Sense Disambiguation *WordNet*::SenseRelate::AllWords. 2007.

Klusch, M. 2008. Intelligent Service Coordination in the Semantic Web. s.l. : Birkhäuser Verlag, 2008.

L. Richardson, S. Ruby. 2007. *RESTful Web Services.* [ed.] Mike Loukides. s.l. : O'Reilly , 2007.

La Desambiguación del Sentido de las Palabras: revisión metodológica. **Leal y Tello, E. 2009.** Abril de 2009, No Solo Usabilidad, Vol. VIII.

Lesk, M. 1986. Automated Word Sense Disambiguation using Machine-Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. 1986.

1993. Ley 80 de 1993. *Por la cual se expide el Estatuto General de Contración de la Administración Pública.* Bogota : s.n., 1993.

Lin, D. 1998. An Information-Theoretic Definition of Similarity. 1998.

M. Maleshkova, C. Pedrinaci, J. Domingue. 2010. Semantic Annotation of Web APIs with SWEET. 2010.

—. 2009. Supporting the Creation of Semantic RESTful Service Descriptions. 2009.

Making Web services tradable: A policy-based approach for specifying preferences on Web service properties. **Agarwal, S., Lamparter, S. y Studer, R. 2009.** 1, Jan de 2009, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 7, págs. 11-20.

Maleshkova, M., Pedrinaci, C. y Domingue, J. 2009. Supporting the Creation of Semantic RESTful Service Descriptions. 2009.

Maleshkova, Maria, Kopecký, Jacek, Pedrinaci, Carlos. 2009. Adapting SAWSDL for Semantic Annotations of RESTful Services. 2009.

—. 2009. Adapting SAWSDL for Semantic Annotations of RESTful Services. 2009.

Maleshkova, y otros. 2009. Adapting SAWSDL for Semantic Annotations of RESTful Services. 2009.

Marc J. Hadley. 2005. *Web Application Description Language (WADL).* 2005.

Martínez, D. y E., Agirre. 2000. One sense per collocation and genre/topic variations. Hong kong : s.n., 2000.

MATAWS: A Multimodal Approach for Automatic WS Semantic Annotation. **Aksoy, Cihan and Labatut, Vincent and Cherifi, Chantal and Santucci, Jean-François. 2011.** 2011, Springer Berlin Heidelberg, Vol. 136, págs. 319-333.

Mezquita, Y. 2011. Information Retrieval with Word Sense Disambiguation for Spanish. s.l. : Computación y Sistemas, 2011.

Michelizzi, J. y Pedersen, T. 2008. CPAN. *WordNet-SenseRelate-WordToSet.* [En línea] Abril de 2008. http://search.cpan.org/dist/*WordNet*-SenseRelate-WordToSet/doc/README.pod.

Microsoft. Msn. [En línea] [Citado el: 02 de 11 de 2012.] http://co.msn.com/.

Modelo para la evaluación de ontologías. Aplicación en Onto-Satcol. **Senso, J., Mederos, A. y S., Domínguez. 2011.** 3, 2011, Vol. 34. ISSN: 0210-0614.

Muller, Dr. Claudia. 2012. *Service oriented Architecture and Web Services.* Institute for computer Science, Networked Information Systems - Freie Universitat Berlin. Berlin : s.n., 2012.

Omelette. 2011. D2.1 – State-of-the-art in the field of. 2011.

Ouellet, R. y Ogbuji, U. 2002. Introduction to DAML. [En línea] 30 de 02 de 2002. [Citado el: 05 de 11 de 2012.] http://www.xml.com/pub/a/2002/01/30/daml1.html.

Parlante, N. 2002. Essential Perl. 2002.

Pease, Adam. 2004. *Sigma Knowledge Engineering Environment.* 2004.

Pedersen, T. y Patwardhan, S. 2004. *WordNet*::similarity - measuring the relatedness of concepts. 2004.

Pellet: A Practical OWL-DL Reasoner. **Sirin, E., y otros. 2007.** 2007, Vol. 5.

Pérez, S. 2009. Resolución de la ambiguedad semántica mediante métodos basados en conocimiento y su aportación a tareas de PLNSonia V´azquez P´erez. Alicante : s.n., 2009.

Poli, R. 2000. Dependence and Dynamic Categories. 2000.

Probabilistic Part-of-Speech Tagging Using Decision Trees. **Schmid, H. 1994.** 1994, Vol. 12, págs. 44–49.

Product configuration knowledge modeling using ontology Web language. **Yang, D., y otros. 2009.** 3, Part 1, 2009, Expert Systems with Applications, Vol. 36, págs. 4399 - 4411.

Proposición de un Modelo para la Acentuación Automática de Palabras Ambiguas del Español, Utilizando Etiquedado de Texto. **Montiel, R., y otros. 2010.** 1, 2010, Programación Matemática y Software, Vol. 2, págs. 31-43. ISBN: 3-540-43219-1.

R. Battle, E. Benson. 2008. Bridging the semantic Web and Web 2.0 with Representational State Transfer (REST). s.l. : Elsevier Science Publishers B. V., 2008. Vol. 6, 1, págs. 61-69.

R. Mihalcea, T. Pedersen. 2004. Advances in word sense disambiguation. 2004.

Ríos, M. 2008. Desambiguación de sentidos de palabras usando sinónimos . 2008.

Security Aware Mobile Web Service Provisioning. **Srirama, S., y otros. 2010.** Riverton, NJ, USA : IBM Corp., Jul de 2010, CoRR, Vol. abs/1007.3640.

Seekport. http://www.seekport.es/. [En línea] [Citado el: 02 de 11 de 2012.] http://www.seekport.es/.

Semantic Annotation of Web Services. **D Bouchiha, M Malki. 2012.** Sidi Bel Abbes : s.n., 2012.

Semantic Annotation of Web Services with Lexicon-Based Alignment. **Canturk, Deniz and Senkul, Pinar. 2011.** [ed.] IEEE Computer Society. Washington, DC, USA : s.n., 2011, Proceedings of the 2011 IEEE World Congress on Services, págs. 355-362.

Semantic Web service composition testbed. **Yeganeh, y otros. 2010.** 5, September de 2010, Comput. Electr. Eng., Vol. 36, págs. 805-817.

Semantic Web Services : Restful Approach. **Ferreira, O. y Varella, M. 2009.** 2009, IADIS International Conference.

SERVERY: the Web-Telco marketplace. **Mathieu BOUSSARD, Vincent HIRIBARREN, Jean Pierre LE ROUZIC, Stéphanie FODOR, Ivan BEDINI, Noel CRESPI, Gabor MARTON, David MORO, Manuel MACIAS, Oscar Lorenzo DUEÑAS, Benjamin MOLINA. 2009.** Jun de 2009, ICT-Mobile, pág. Summit.

Simple and Efficient Minimal RDFS. **Muñoz, S., Pérez, J. y Gutierrez, C. 2009.** 3, 2009, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 7, págs. 220-234.

Slimani, T., Yaghlane, B. y Mellouli, K. 2006. A New Similarity Measure based on Edge Counting. 2006. Vol. 17.

Smith, K. 2009. OneApi: A standard to simplify cross-network development. 2009.

—. 2009. *OneAPI: A standard to simplify cross-network development,.* 2009.

SRCluster: Web Clustering Engine based on Wikipedia. **Meiyappan, Y., Iyengar, N. y Kannan, A. 2012.** s.l. : IJAST Journal, 2012, Vol. 39.

T. Pedersen, V. Kolhatkar. 2009. *WordNet*::SenseRelate::AllWords - A Broad Coverage Word Sense Tagger that Maximizes Semantic Relatedness. 2009.

Taxonomía, ontología y folksonomía, ¿qué son y qué beneficios u oportunidades presentan para los usuarios de la Web? **Piraquive, D., y otros. 2009.** 16, 04 de 2009, Universidad y Empresa, Vol. VIII, págs. 242-261.

Tester, D. 2012. Cambridge Semantics. *RDFS vs. OWL.* [En línea] 16 de 05 de 2012. [Citado el: 01 de 11 de 2012.] http://www.cambridgesemantics.com/semantic-university/rdfs-vs.-owl.

The retrieval effectiveness of Web search engines: considering results descriptions. **Lewandowski, Dirk. 2008.** [ed.] Emerald. 2008, Journal of Documentation, Vol. 64.

Townsend, ICT KTN Co. Ltd - Eddie. 2011. *FUTURE INTERNET.* London : s.n., 2011.

User-Centric Future Internet and Telecommunication Services. **Baladrón, Carlos Aguiar, Javier Carro, Belén Goix, Laurent Walter Martin, Alberto León Falcarin, Paolo Sienel, Jürgen. 2009.** 2009, IOS Press, págs. 217-226.

—. **Baladrón, y otros. 2009.** 2009, IOS Press, págs. 217-226.

Using corpus statistics and WordNet relations for sense identification. **C., Leacock, Chodorow, M. y A., Miller. 1998.** 1, 1998, Vol. 24.

Vasilescu, F., Langlais, P. y G., Lapalme. 2004. Evaluating Variants of the Lesk Approach for Disambiguating Words. *LREC* . s.l. : European Language Resources Association, 2004.

Vitvar, Tomas, Kopecky, Jacek, Viskova, Jana, Fensel, Dieter. 2008. WSMO-Lite Annotations for Web Services. 2008.

W3C. 2004. Resource Description Framework (RDF): Concepts and Abstract Syntax. *W3C Recommendation 10 February 2004.* [En línea] 10 de 02 de 2004. [Citado el: 04 de 11 de 2012.] http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

Word sense disambiguation with multilingual features. **Banea, Carmen and Mihalcea, Rada. 2011.** 10, Stroudsburg, PA, USA : Association for Computational Linguistics, 2011, Association for Computational Linguistics, págs. 25-34. DOI: 10.3115/1073083.1073126.

Wu, Z. y Palmer, M. 1994. Verb Semantics And Lexical Selection. 1994.

Y. Chabeb, S. Tata, D. Belaïd. 2009. Toward an integrated ontology for Web services. 2009.

Y. Zhang, Z. Zheng, M. Lyu. 2010. WSExpress: A QoS-Aware Search Engine for Web Services. 2010.

Yahoo! Yahoo. [En línea] [Citado el: 02 de 11 de 2012.] http://www.yahoo.com.

Yatskevich, M. 2003. *Preliminary Evaluation of Schema Matching Systems.* Department of Iinformation and Communication Technology, University of Trento. 2003.

A Method forword sense disambiguation of unrestricted text. **Rada, M. y D., Moldovan. 1999.** Maryland : s.n., 1999.

A Service Description Language for the Internet of Services. **Cardoso, J.; Winkler, M.; Voigt, K. 2009.** Dresden, Germany : s.n., 2009.

A translation approach to portable ontology specifications. **Gruber, T. 1993.** 2, s.l. : Academic Press Ltd., 1993, Vol. 5.

Agirre, Eneko y Martinez, D. 2001. Learning class-to-class selectional preferences. Toulouse : s.n., 2001.

Akkiraju, R. y Bournez, C. 2007. Semantic Annotations for WSDL and XML Schema — Usage Guide. [En línea] 29 de Agosto de 2007. http://www.w3.org/TR/sawsdl-guide/#invocation.

Alliance, Open Mobile. 2012. OMA. *Open Mobile Alliance.* [En línea] 2012. [Citado el: 28 de 10 de 2012.] http://www.openmobilealliance.org/AboutOMA/Default.aspx.

Alonso, Grettel Barceló. Junio 2010. Desambiguación de los sentidos de las palabras en español usando textos paralelos. Junio 2010.

An adapted lesk algorithm for word sense disambiguation using WordNet. **Banerjee, Satanjeev y Pedersen, T. 2002.** s.l. : CICLing, 2002.

An Efficient WS-QoS Broker Based Architecture for Web Services Selection. **T.Rajendran and P.Balasubramanie and Resmi Cherian. 2010.** 9, 2010, International Journal of Computer Applications, Vol. 1, págs. 79-84.

An Empirical Evaluation on Semantic Search Performance of Keyword-Based and Semantic Search Engines: Google, Yahoo, Msn and Hakia. **Tümer, D., Shah, M. A. y BitirimY. 2009.** [ed.] IEEE Computer Society. 5, Washington : s.n., 2009, págs. 51-55.

Aol. Aol. [En línea] [Citado el: 02 de 11 de 2012.] http://www.aol.com/.

Applying recommender system based mashup to Web-telecom hybrid service creation. **Jie, Guo and Bo, Cheng and Junliang, Chen and Xiangtao, Lin. 2009.** [ed.] IEEE Press. Piscataway, NJ, USA : s.n., 2009, págs. 3321-3325.

Ask. Ask. [En línea] [Citado el: 02 de 11 de 2012.] http://www.ask.com/.

Automatic Tag Identification in Web Service. **J. R. Falleri, Z. Azmeh, M. Huchard, C. Tibermacine. 2010.** [ed.] Joaquim and Cordeiro, José Filipe. 5 de 11 de 2010, The International Conference on Web Information Systems and Technology (WEBIST'10), Spain, págs. 40-47.

AUTOMATIC TAG IDENTIFICATION IN WEB SERVICE. **J. R. Falleri, Z. Azmeh, M. Huchard, C. Tibermacine. 2010.** [ed.] Joaquim and Cordeiro, José Filipe. 5 de 11 de 2010, The International Conference on Web Information Systems and Technology (WEBIST'10), Spain, págs. 40-47.

Breis, J. 2003. Un Entorno de Integración de Ontologías para el Desarrollo de Sistemas de Gestión de Conocimiento. 2003.

Búsqueda sobre catálogos basada en ontologías. **Pérez Sosa, A. y Proenza Arias, Y. E. 2011.** Ciudad de la Habana. Cuba : s.n., 27 de 04 de 2011, Revista Digital Sociedad de la Información.

Cherifi, Chantal, Labatut, Vincent, Santucci, J. François. 2010. Benefits of Semantics on Web Service Composition from a Complex Network Perspective. 2010.

Chillarege, R. 1999. *Software Testing Best Practices.* Center for Software Engineering, IBM Research. 1999. IBM Technical Report.

Chinnici, R.; Moreau, J. J.; Ryman, A.; Weerawarana, S. 2007. Web Services Description Language (WSDL). *W3C.* [En línea] 2007. http://www.w3.org/TR/wsdl20/.

Christensen, E.; Curbera, F.; Meredith, G.; Weerawarana, S. 2001. Web Services Description Language (WSDL) 1.1. [En línea] w3c, 2001. [Citado el: 23 de 10 de 2012.] http://www.w3.org/TR/wsdl.

Clustering WSDL Documents to Bootstrap the Discovery of Web Services. **Elgazzar, Khalid and Hassan, Ahmed E. and Martin, Patrick. 2010.** [ed.] IEEE Computer Society. 8, Washington, DC, USA : s.n., 2010, págs. 147-154.

Comparison of reasoners for large ontologies in the OWL 2 EL profile. **Dentler, K., y otros. 2011.** 2, s.l. : IOS Press, 2011, Vol. 2.

Convergence of Application Services in Next-generation Networks. **Anne Y. Lee, Abdi Modarressi, Seshadri Mohan. 2012.** s.l. : GUEST, 2012.

Cowie, J., Guthrie, J. y Guthrie, L. 1992. Lexical disambiguation using simulated annealing. *Proceedings of the 14th conference on Computational linguistics.* Nantes, France : s.n., 1992.

D. Booth, C. Liu. 2007. *Web Services Description Languaje (WSDL) Version 2.0 Part 0: Primer.* [En línea] Junio de 2007. http://www.w3.org/TR/wsdl20-primer/.

Debugging OWL ontologies. **Parsia, B., Sirin, E. y Kalyanpur, A. 2005.** s.l. : ACM, 2005.

Design of Network Resource Federation towards Future Open Access Networking. **Michiaki Hayashi, Nobutaka Matsumoto, Kosuke Nishimura, Hideaki Tanaka. 2011.** Jun de 2011, Proceedings of The Seventh Advanced International Conference on Telecommunications (AICT), págs. 130 - 134.

Discovering Semantic Web services using SPARQL and intelligent agents. **M. Sbodio, D. Martin, C. Moulin. 2012.** 14, 2012, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 8.

—. **Sbodio, M., Martin, D. y Moulin, C. 2012.** 14, 2012, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 8.

E. Christensen, F. Curbera, G. Meredith, S. Weerawarana. 2001. *Web Service Description Languaje.* [En línea] Marzo de 2001. http://www.w3.org/TR/wsdl.

Enabling technology for knowledge sharing. **Neches, R., y otros. 1991.** 3, 1991, Vol. 12.

Exalead. Exalead. [En línea] [Citado el: 02 de 11 de 2012.] http://www.exalead.com/search/.

Extended Gloss Overlaps as a Measure of Semantic Relatedness. **Banerjee, S. y Pedersen, T. 2003.** Acapulco, Mexico : Morgan Kaufmann Publishers Inc., 2003.

Extending next generation network (NGN) architecture for connection-oriented transport. **F. Baroncelli a, B. Martini a, V. Martini b, P. Castoldi b. 2011.** 9, 2011, Computer Communications, Vol. 34, págs. 1100-1111.

Gale, W., K. Church y D. Yarowsky. 1992. Estimating upper and lower bounds on the performance or word-sense disambiguation programs. 1992.

Ganesan, V., Swaminathan, R. y Thenmozhi, M. 2012. Similarity Measure Based On Edge Counting Using Ontology. 2012. Vol. 3.

Gangemi, A., y otros. 2005. Ontology evaluation and validation. 2005.

GESFOR, y otros. 2011. *State of the art in the field of Mashup concepts.* ICT OMELETTE. 2011. Reporte.

Gómez, S., Chesñevar, C. y Simari, G. 2008. Hacia la solución del problema de chequeo de instancia en ontologías inconsistentes usando argumentación rebatible. 2008.

Google. 2012. Custom Search. [En línea] 13 de 06 de 2012. [Citado el: 02 de 11 de 2012.] https://developers.google.com/custom-search/docs/start?hl=en#overview.

—. Google. [En línea] [Citado el: 02 de 11 de 2012.] http://www.google.com.

—. Google Guide. *Marking searching even easier.* [En línea] [Citado el: 02 de 11 de 2012.] http://www.googleguide.com/advanced_operators.html.

Grette, A. 2010. Desambiguación de los sentidos de las palabras en español usando textos paralelos. México D.F : s.n., 2010.

GSMA. 2012. GSMA. [En línea] 2012. [Citado el: 28 de 10 de 2012.] http://www.gsma.com/.

H. Yoo, Y. Park, H. Bae. 2011. Semi-atutomatic Semantic Service Annotation for SOAP and REST Web Services. 2011.

Hadley, M. 2009. *Web Application Description Language.* [En línea] Agosto de 2009. http://www.w3.org/Submission/wadl/.

Hadley, Marc. 2009. Web Application Description Language. [En línea] 31 de 08 de 2009. http://www.w3.org/Submission/wadl/.

hakia, Inc. Hakia. [En línea] [Citado el: 02 de 11 de 2012.] http://www.hakia.com/.

Hernández y Calderón, L. 2010. *AUTOMATIZACION DE PROCESOS DE NEGOCIO USANDO SERVICIOS WEB SEMANTICOS.* UNIVERSIDAD DE PAMPLONA. 2010. Tesis pregrado.

Hirst, G. and St-Onge, D. 1998. Lexical Chains as representation of context for the detection and correction. [aut. libro] Christiane Fellbaum. *In WordNet: An electronic lexical database.* 1998.

Ide, N. y Véronis, J. 1998. Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art. *Computational Linguistics.* 1998.

Investigating Web APIs on the World Wide Web. **Maleshkova, M.; Pedrinaci, C.; Domingue, J. 2010.** 2010, ECOWS'10, págs. 107-114.

Izquierdo, R. 2010. Una aproximación a la desambiguación del sentido de las palabras basada en clases semánticas y aprendizaje automático. Alicante : s.n., 2010.

J. Domingue, M. Maleshkova, C. Pedrinaci. 2009. Supporting the Creation of Semantic RESTful Service Descriptions. 2009.

Jiang, J. y D., Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. 1997.

Kikas, T. y Treumuth, M. 2007. Word Sense Disambiguation *WordNet*::SenseRelate::AllWords. 2007.

Klusch, M. 2008. Intelligent Service Coordination in the Semantic Web. s.l. : Birkhäuser Verlag, 2008.

L. Richardson, S. Ruby. 2007. *RESTful Web Services.* [ed.] Mike Loukides. s.l. : O'Reilly , 2007.

La Desambiguación del Sentido de las Palabras: revisión metodológica. **Leal y Tello, E. 2009.** Abril de 2009, No Solo Usabilidad, Vol. VIII.

Lesk, M. 1986. Automated Word Sense Disambiguation using Machine-Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. 1986.

1993. Ley 80 de 1993. *Por la cual se expide el Estatuto General de Contración de la Administración Pública.* Bogota : s.n., 1993.

Lin, D. 1998. An Information-Theoretic Definition of Similarity. 1998.

M. Maleshkova, C. Pedrinaci, J. Domingue. 2010. Semantic Annotation of Web APIs with SWEET. 2010.

—. 2009. Supporting the Creation of Semantic RESTful Service Descriptions. 2009.

Making Web services tradable: A policy-based approach for specifying preferences on Web service properties. **Agarwal, S., Lamparter, S. y Studer, R. 2009.** 1, Jan de 2009, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 7, págs. 11-20.

Maleshkova, M., Pedrinaci, C. y Domingue, J. 2009. Supporting the Creation of Semantic RESTful Service Descriptions. 2009.

Maleshkova, Maria, Kopecký, Jacek, Pedrinaci, Carlos. 2009. Adapting SAWSDL for Semantic Annotations of RESTful Services. 2009.

—. 2009. Adapting SAWSDL for Semantic Annotations of RESTful Services. 2009.

Maleshkova, y otros. 2009. Adapting SAWSDL for Semantic Annotations of RESTful Services. 2009.

Marc J. Hadley. 2005. *Web Application Description Language (WADL).* 2005.

Martínez, D. y E., Agirre. 2000. One sense per collocation and genre/topic variations. Hong kong : s.n., 2000.

MATAWS: A Multimodal Approach for Automatic WS Semantic Annotation. **Aksoy, Cihan and Labatut, Vincent and Cherifi, Chantal and Santucci, Jean-François. 2011.** 2011, Springer Berlin Heidelberg, Vol. 136, págs. 319-333.

Mezquita, Y. 2011. Information Retrieval with Word Sense Disambiguation for Spanish. s.l. : Computación y Sistemas, 2011.

Michelizzi, J. y Pedersen, T. 2008. CPAN. *WordNet-SenseRelate-WordToSet.* [En línea] Abril de 2008. http://search.cpan.org/dist/*WordNet*-SenseRelate-WordToSet/doc/README.pod.

Microsoft. Msn. [En línea] [Citado el: 02 de 11 de 2012.] http://co.msn.com/.

Modelo para la evaluación de ontologías. Aplicación en Onto-Satcol. **Senso, J., Mederos, A. y S., Domínguez. 2011.** 3, 2011, Vol. 34. ISSN: 0210-0614.

Muller, Dr. Claudia. 2012. *Service oriented Architecture and Web Services.* Institute for computer Science, Networked Information Systems - Freie Universitat Berlin. Berlin : s.n., 2012.

Omelette. 2011. D2.1 – State-of-the-art in the field of. 2011.

Ouellet, R. y Ogbuji, U. 2002. Introduction to DAML. [En línea] 30 de 02 de 2002. [Citado el: 05 de 11 de 2012.] http://www.xml.com/pub/a/2002/01/30/daml1.html.

Parlante, N. 2002. Essential Perl. 2002.

Pease, Adam. 2004. *Sigma Knowledge Engineering Environment.* 2004.

Pedersen, T. y Patwardhan, S. 2004. *WordNet*::similarity - measuring the relatedness of concepts. 2004.

Pellet: A Practical OWL-DL Reasoner. **Sirin, E., y otros. 2007.** 2007, Vol. 5.

Pérez, S. 2009. Resolución de la ambiguedad semántica mediante métodos basados en conocimiento y su aportación a tareas de PLNSonia V´azquez P´erez. Alicante : s.n., 2009.

Poli, R. 2000. Dependence and Dynamic Categories. 2000.

Probabilistic Part-of-Speech Tagging Using Decision Trees. **Schmid, H. 1994.** 1994, Vol. 12, págs. 44–49.

Product configuration knowledge modeling using ontology Web language. **Yang, D., y otros. 2009.** 3, Part 1, 2009, Expert Systems with Applications, Vol. 36, págs. 4399 - 4411.

Proposición de un Modelo para la Acentuación Automática de Palabras Ambiguas del Español, Utilizando Etiquedado de Texto. **Montiel, R., y otros. 2010.** 1, 2010, Programación Matemática y Software, Vol. 2, págs. 31-43. ISBN: 3-540-43219-1.

R. Battle, E. Benson. 2008. Bridging the semantic Web and Web 2.0 with Representational State Transfer (REST). s.l. : Elsevier Science Publishers B. V., 2008. Vol. 6, 1, págs. 61-69.

R. Mihalcea, T. Pedersen. 2004. Advances in word sense disambiguation. 2004.

Ríos, M. 2008. Desambiguación de sentidos de palabras usando sinónimos . 2008.

Security Aware Mobile Web Service Provisioning. **Srirama, S., y otros. 2010.** Riverton, NJ, USA : IBM Corp., Jul de 2010, CoRR, Vol. abs/1007.3640.

Seekport. http://www.seekport.es/. [En línea] [Citado el: 02 de 11 de 2012.] http://www.seekport.es/.

Semantic Annotation of Web Services. **D Bouchiha, M Malki. 2012.** Sidi Bel Abbes : s.n., 2012.

Semantic Annotation of Web Services with Lexicon-Based Alignment. **Canturk, Deniz and Senkul, Pinar. 2011.** [ed.] IEEE Computer Society. Washington, DC, USA : s.n., 2011, Proceedings of the 2011 IEEE World Congress on Services, págs. 355-362.

Semantic Web service composition testbed. **Yeganeh, y otros. 2010.** 5, September de 2010, Comput. Electr. Eng., Vol. 36, págs. 805-817.

Semantic Web Services : Restful Approach. **Ferreira, O. y Varella, M. 2009.** 2009, IADIS International Conference.

SERVERY: the Web-Telco marketplace. **Mathieu BOUSSARD, Vincent HIRIBARREN, Jean Pierre LE ROUZIC, Stéphanie FODOR, Ivan BEDINI, Noel CRESPI, Gabor MARTON, David MORO, Manuel MACIAS, Oscar Lorenzo DUEÑAS, Benjamin MOLINA. 2009.** Jun de 2009, ICT-Mobile, pág. Summit.

Simple and Efficient Minimal RDFS. **Muñoz, S., Pérez, J. y Gutierrez, C. 2009.** 3, 2009, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 7, págs. 220-234.

Slimani, T., Yaghlane, B. y Mellouli, K. 2006. A New Similarity Measure based on Edge Counting. 2006. Vol. 17.

Smith, K. 2009. OneApi: A standard to simplify cross-network development. 2009.

—. 2009. *OneAPI: A standard to simplify cross-network development,.* 2009.

SRCluster: Web Clustering Engine based on Wikipedia. **Meiyappan, Y., Iyengar, N. y Kannan, A. 2012.** s.l. : IJAST Journal, 2012, Vol. 39.

T. Pedersen, V. Kolhatkar. 2009. *WordNet*::SenseRelate::AllWords - A Broad Coverage Word Sense Tagger that Maximizes Semantic Relatedness. 2009.

Taxonomía, ontología y folksonomía, ¿qué son y qué beneficios u oportunidades presentan para los usuarios de la Web? **Piraquive, D., y otros. 2009.** 16, 04 de 2009, Universidad y Empresa, Vol. VIII, págs. 242-261.

Tester, D. 2012. Cambridge Semantics. *RDFS vs. OWL.* [En línea] 16 de 05 de 2012. [Citado el: 01 de 11 de 2012.] http://www.cambridgesemantics.com/semantic-university/rdfs-vs.-owl.

The retrieval effectiveness of Web search engines: considering results descriptions. **Lewandowski, Dirk. 2008.** [ed.] Emerald. 2008, Journal of Documentation, Vol. 64.

Townsend, ICT KTN Co. Ltd - Eddie. 2011. *FUTURE INTERNET.* London : s.n., 2011.

User-Centric Future Internet and Telecommunication Services. **Baladrón, Carlos Aguiar, Javier Carro, Belén Goix, Laurent Walter Martin, Alberto León Falcarin, Paolo Sienel, Jürgen. 2009.** 2009, IOS Press, págs. 217-226.

—. **Baladrón, y otros. 2009.** 2009, IOS Press, págs. 217-226.

Using corpus statistics and WordNet relations for sense identification. **C., Leacock, Chodorow, M. y A., Miller. 1998.** 1, 1998, Vol. 24.

Vasilescu, F., Langlais, P. y G., Lapalme. 2004. Evaluating Variants of the Lesk Approach for Disambiguating Words. *LREC* . s.l. : European Language Resources Association, 2004.

Vitvar, Tomas, Kopecky, Jacek, Viskova, Jana, Fensel, Dieter. 2008. WSMO-Lite Annotations for Web Services. 2008.

W3C. 2004. Resource Description Framework (RDF): Concepts and Abstract Syntax. *W3C Recommendation 10 February 2004.* [En línea] 10 de 02 de 2004. [Citado el: 04 de 11 de 2012.] http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

Word sense disambiguation with multilingual features. **Banea, Carmen and Mihalcea, Rada. 2011.** 10, Stroudsburg, PA, USA : Association for Computational Linguistics, 2011, Association for Computational Linguistics, págs. 25-34. DOI: 10.3115/1073083.1073126.

Wu, Z. y Palmer, M. 1994. Verb Semantics And Lexical Selection. 1994.

Y. Chabeb, S. Tata, D. Belaïd. 2009. Toward an integrated ontology for Web services. 2009.

Y. Zhang, Z. Zheng, M. Lyu. 2010. WSExpress: A QoS-Aware Search Engine for Web Services. 2010.

Yahoo! Yahoo. [En línea] [Citado el: 02 de 11 de 2012.] http://www.yahoo.com.

Yatskevich, M. 2003. *Preliminary Evaluation of Schema Matching Systems.* Department of Iinformation and Communication Technology, University of Trento. 2003.

ANEXO A

ANEXO A

MODELO DE DESPLIEGUE Y MODELO DE DISEÑO DEL SISTEMA

Este anexo aborda la documentación correspondiente a la definición y descripción de un conjunto de diagramas que especifican la Vista de Diseño y Despliegue del Sistema. De acuerdo con la referencia metodológica adoptada, este documento forma parte del macro componente denominado *Estructura para la Descripción del Sistema*, el cual involucra la especificación de artefactos esenciales (Modelo de Casos de Uso, Arquitectura de Referencia, Modelo de Diseño, Modelo de Despliegue y Plan de Pruebas) para el entendimiento adecuado de la funcionalidad y el comportamiento esperados del sistema/solución.

A.1. Diagrama de Paquetes del Sistema

La Figura 19 define el conjunto de paquetes en los cuales se distribuye el software del sistema desarrollado. Tal como se observa en la gráfica, dichos paquetes se encuentran dispuestos de acuerdo con la separación establecida en la arquitectura de referencia descrita en el Capítulo 5 de la monografía. La definición de cada uno de los paquetes que conforman la aplicación del sistema se aborda a continuación:

A.1.1. Capa de Presentación.

Esta capa de la arquitectura de referencia se implementa mediante los siguientes paquetes:

- *Web_pages*: implementa la interfaz gráfica de usuario que compone la aplicación Web. Fundamentalmente, consiste de una página *jsf* y dos archivos xml de configuración: el archivo descriptor de despliegue de la aplicación (*Web.xml*) y el archivo que especifica el control de la navegación (*faces-config.xml*).

- *co.edu.unicauca.control*: constituye un elemento mediador entre la Lógica de Presentación y la Lógica de Negocio, el cual garantiza la captura adecuada de los datos suministrados por el usuario en su interacción con la aplicación Web, así como también de que la información que se despliega ante él corresponda apropiadamente con sus peticiones. De la misma manera, controla la información manejada en los paquetes, por medio de la recepción de la información suministrada por las salidas de cada paquete y entrega de la misma a los paquetes que la requieren para su funcionamiento.

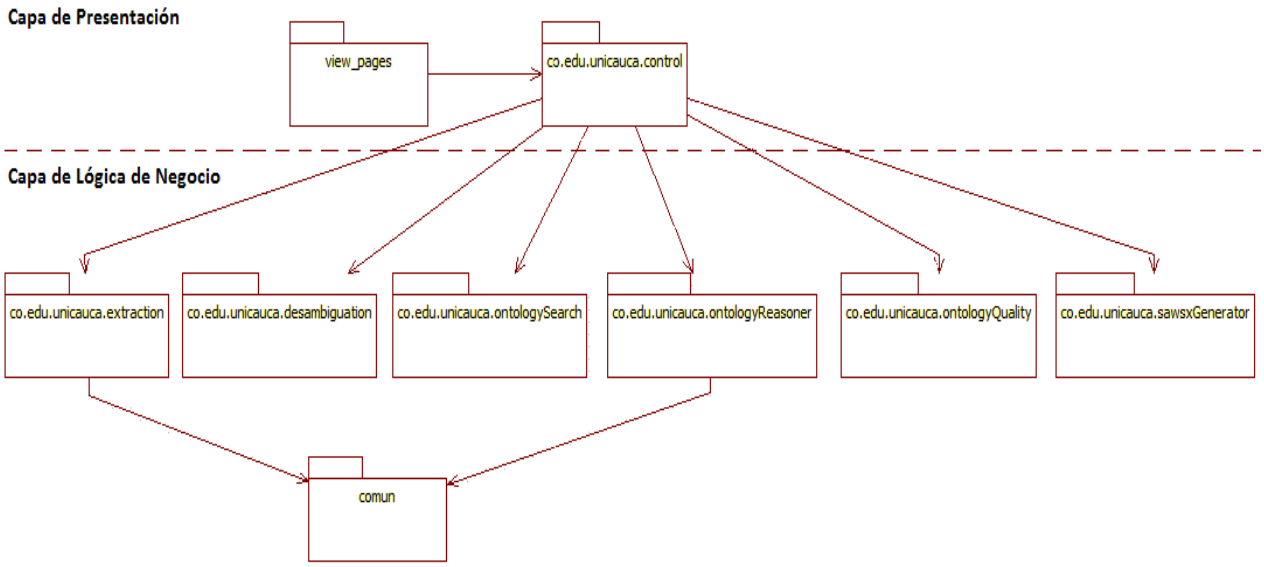


Figura 21. Diagrama de Paquetes del Sistema

A.1.2. Capa de Lógica de Negocio.

Los paquetes que implementan esta capa de la arquitectura de referencia, representan las funcionalidades más importantes del sistema propuesto. En esta sección se realiza una descripción de cada uno de ellos:

- *co.edu.unicauca.extraction*: este paquete contiene el software encargado de llevar a cabo la extracción de los atributos relevantes del descriptor de servicio que el usuario desee anotar semánticamente.

- *co.edu.unicauca.desambiguation*: este componente contiene el software que permite la desambiguación de los atributos que han sido extraídos del descriptor, la desambiguación de las entidades ontológicas y el filtrado de entidades ontológicas partiendo de la estimación del valor de similitud semántica entre entidades ontológicas y atributos a anotar.

- *co.edu.unicauca.ontologySearch*: en este paquete se implementa el componente que permite la búsqueda de ontologías OWL disponibles en la red, mediante el uso del API de Google que permite el uso de su motor de búsqueda y la configuración de un parámetro de filtrado que permite la obtención ontologías descritas en OWL.

- *co.edu.unicauca.ontologyReasoner*: este paquete contiene la implementación del razonador Pellet, utilizado para la extracción de la entidad ontológica que más se relacione con el atributo que se desea anotar, junto con las entidades ontológicas que se derivan de ella.

- co.edu.unicauca.ontologyQuality: el software que integra esta aplicación permite el análisis de entidades inconsistentes en las ontologías, con el fin de descartar aquellas ontologías en las que el número de estas entidades sea muy alto.

- *co.edu.unicauca.sawsdlGenerator*: este componente permite la generación de un documento contenedor de los atributos de extensión propuestos por SAWSDL, con sus entidades ontológicas correspondientes.

- *co.edu.unicauca.comun:* contiene el software utilizado para realizar el proceso de tokenización en dos paquetes: *co.edu.unicauca.extraction* y *co.edu.unicauca.ontologyReasoner*. Este proceso permite la separación de palabras contiguas sin espacios entre ellas, mediante la detección de cambios de minúsculas a mayúsculas, o símbolos entre ellas como puntos, guiones, entre otros.

A.2. Diagrama de Clases del Sistema

La Figura 20 presenta el Diagrama de Clases del sistema desarrollado. Este diagrama provee la vista estática de la aplicación, definiendo además las relaciones existentes entre las entidades que la componen. En esta sección se realiza la descripción de cada una de las clases ilustradas, junto con sus atributos y métodos más representativos.

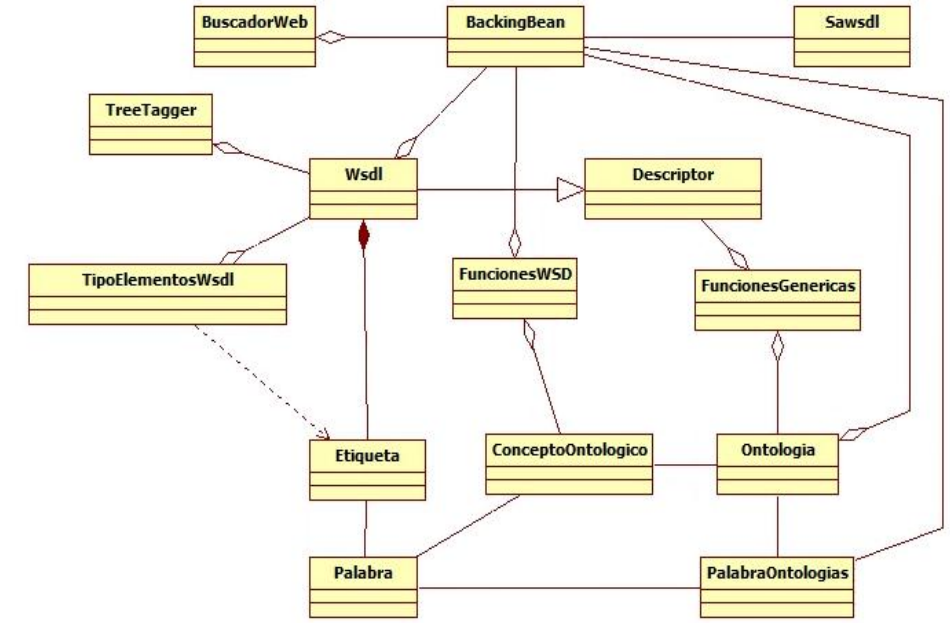


Figura 22. Diagrama de Clases del Sistema

Tabla 4. Descripción de la Clase BackingBean

Nombre
BackingBean (paquete: co.edu.unicauca.control)
Descripción
Esta clase implementa un método que consume de manera secuencial los métodos implementados en las clases de cada paquete del cual BackingBean necesita recibir información para controlar el funcionamiento de la lógica del negocio.
Atributos
• *wsdl*: **Instancia de la clase Wsdl, encargado de permitir el uso de las funciones de dicha clase.**

• *buscadorWeb:* **Instancia de la clase BuscadorWeb, encargado de permitir el uso de las funciones de dicha clase.** • *funcioneswsd:* **Instancia de la clase FuncionesWSD, encargado de permitir el uso de las funciones de dicha clase.** • *sawsdl:* **Instancia de la clase Sawsdl, encargado de permitir el uso de las funciones de dicha clase.** • *palabrasRelevantesDescriptor:* **Contiene una lista de la clase Palabra, la cual contiene las palabras relevantes extraídas del descriptor.** • *palabraOntologiaList:* **Contiene una lista de la clase PalabraOntologias, la cual contiene un atributo y una lista de ontologías encontradas para ese atributo.** • *rutaArchivoSawsdl:* **Contiene la ruta en donde se encontrara almacenado el archivo SAWSDL generado.**
Métodos
• *anotarServicio()*: **a partir de una URL obtiene la estructura del descriptor del servicio al cual corresponde e inicia el consumo de los métodos en las clases de los paquetes contenidos en la lógica del negocio que se mencionan a continuación.** • obtenerPalabrasRelevantesDescriptor(): **Razona el descriptor y obtiene los atributos relevantes del mismo.** • *palabrasDescriptor():* **concatena los atributos extraídos del descriptor y a través del módulo de desambiguación obtiene sus sentidos.** • *buscarOntologías():* **Para cada atributo encuentra una lista de ontologías que se relacionan con él.** • extraerEntidadesOntológicas(): **En cada ontología asociada a un atributo se extraen las entidades ontológicas relacionadas con mismo.** • *desambiguarEntidadesOntologicas():* **A través del módulo de desambiguación encuentra un sentido para cada entidad ontológica.** • *calcularGradoSimilitud():* **Calcula el grado de similitud entre el sentido del atributo y el sentido de cada una de las entidades ontológicas asociadas a ese atributo, y selecciona la entidad ontológica con el mayor puntaje.** • *Resultado():* **Genera el archivo SAWSDL.**

Tabla 5. Descripción de la Clase Palabra

Nombre
Palabra (paquete: co.edu.unicauca.comun)
Descripción
Esta Clase permite almacenar información relevante acerca de los atributos del descriptor y las entidades ontológicas que la plataforma maneja. De esta manera, la información puede ser manipulada y entregada a otras clases.
Atributos
• nombre: **Contiene la palabra que conforma un atributo o una entidad ontológica, en caso de que estén conformadas por más palabras se crean instancias de la clase palabra.** • *etiquetaList*: **Contiene la lista de atributos en las que se encuentran palabras similares que los conforman.** • *pos*: **Contiene la clasificación gramatical de las palabras que conforman los atributos y las entidades ontológicas.** • *sentido*: **Encargado de contener el sentido de las palabras que conforman los atributos y las entidades ontológicas.** • *anotación*: **contiene la URL de la ontología y la referencia a la entidad ontológica con la que se anota un atributo.**

Métodos

Tabla 6. Descripción de la Clase Etiqueta

Nombre
Etiqueta (paquete: co.edu.unicauca.comun)
Descripción
Esta clase contiene información relevante acerca de los atributos de un descriptor, tales como: nombre, palabras que el atributo contiene y el tipo de atributo.
Atributos
• *nombre*: **contiene todo el texto que conforma el atributo.** • *list:* **Contiene una lista de la clase palabra conformada por las palabras que componen el atributo.** • *tipoEtiqueta:* **Describe el tipo de atributo, como: portType, Type, Service, entre otros.**
Métodos

Tabla 7. Descripción de la Clase FuncionesGenericas

Nombre
FuncionesGenericas (paquete: co.edu.unicauca.comun)
Descripción
Esta clase contiene funciones reutilizables desde otros paquetes.
Atributos
Métodos
• *obtenerInstancia()*: **Obtener una instancia de la misma clase.** • *Tokenizar():* **Tokeniza una palabra de acuerdo a los caracteres que se le indique.**

Tabla 8. Descripción de la Clase TreeTagger

Nombre
TreeTagger (paquete: co.edu.unicauca.comun)
Descripción
Contiene métodos para realizar la clasificación gramatical de palabras.
Atributos
Métodos
• *obtenerInstancia()*: **Obtiene una instancia de la misma clase.** • *asignarPosPalabras()*: **Recibe una lista de palabras y realiza la clasificación gramatical de cada una.**

Tabla 9. Descripción de la clase BuscadorWeb

Nombre
BuscadorWeb (paquete: co.edu.unicauca. ontologySearch)
Descripción
Se conecta a través de HTTP a las funciones de un motor de búsqueda personalizado de Google que permite realizar búsquedas en la Web.
Atributos
• apiKey: **contiene la clave proporcionada por Google para el uso del API que permite la creación de un motor de búsqueda personalizado.** • *cx:* **Identificador de contexto que proporciona Google.**
Métodos
• *buscarOntologias()*: **Realiza la búsqueda de ontologías OWL por medio del motor de búsqueda personalizado. A partir de una palabra clave obtiene una lista de URLs.**

Tabla 10. Descripción de la Clase Descriptor

Nombre
Descriptor (paquete: co.edu.unicauca.extraction)
Descripción
Esta clase contiene información relevante acerca de un descriptor de servicio, tales como: URL, y la lista de atributos que lo componen.
Atributos
• *url:* **Contiene la URL del descriptor del servicio**
Métodos

Tabla 11. Descripción de la Clase TipoElementosWsdl

Nombre
TipoElementosWsdl (paquete: co.edu.unicauca.extraction)
Descripción
Esta clase permite obtener los diferentes atributos de un descriptor WSDL.
Atributos
Métodos
• *obtenerInstancia()*: **obtiene una instancia de la clase.** • *leerService():* **obtiene los atributos de tipo service.** • *leerPuertos():***obtiene los atributos de tipo port.** • leerPortType():**obtiene los atributos de tipo Port Type.** • leerOperation():**obtiene los atributos de tipo operation.** • leerTypes():**obtiene los atributos de tipo type.** • leerMenssage():**obtiene los atributos de tipo message.**

Tabla 12. Descripción de la Clase Wsdl

Nombre
Wsdl (paquete: co.edu.unicauca.extraction)
Descripción
Contiene información relevante de un descriptor WSDL.
Atributos
• *etiquetaList:* **Contiene una lista de la clase etiqueta, que representa la lista de atributos que componen el descriptor del servicio.** • *palabraList:* **Contiene una lista de la clase palabra, que representa todas las palabras contenidas en los atributos de los descriptores WSDL.**
Métodos
• *razonarWSDL()*: **separa y clasifica los atributos según el tipo al que pertenecen.** • *obtenerPalabrasRelevantes():* **obtiene las palabras que conforman los atributos relevantes del descriptor.**

Tabla 13. Descripción de la Clase ConceptoOntologico

Nombre
ConceptoOntologico (paquete: co.edu.unicauca.OntologyReasoner)
Descripción
Esta clase contiene información relevante de un concepto ontológico.
Atributos
• *nombre:* **contiene el nombre de la entidad ontológica.** • *list:* **Es una lista de la clase palabra, representa las palabras que conforman el concepto ontológico.** • *conceptosInferiores:* **Es una lista de la clase ConceptoOntologico que representa los conceptos ontológicos de nivel inferior.** • *conceptosSuperiores:* **Es una lista de la clase ConceptoOntologico que representa los conceptos ontológicos de nivel superior.** • sentido: **Contiene el sentido asignado a las entidades ontológicas por la base de datos léxica *WordNet*.** • *grado:* **corresponde a un número decimal asignado por *WordNet*, representa el grado de similitud entre la entidad ontológica y sus entidades superiores e inferiores asociadas.**
Métodos
• *tokenizarNombre()*: **separar el nombre de la entidad ontológica en las palabras que lo conforman.** • *encontrarSentido():* **a través del módulo de desambiguación encuentra el sentido del concepto ontológico, además del grado de similitud con sus entidades ontológicas asociadas.**

Tabla 14. Descripción de la Clase Ontologia

Nombre
Ontologia (paquete: co.edu.unicauca.OntologyReasoner)
Descripción
Contiene información relevante de una ontología.
Atributos
• *url:* **contiene la URL de una ontología**

• *conceptosOntologicosEncontrados:* **Lista de la clase ConceptoOntologico, contiene las entidades ontológicas asociadas a una palabra determinada que se estableció como parámetro para la búsqueda de esa ontología.** • *owlModel:* **contiene la representación en objetos de la ontología, lo realiza a través de Pellet.**
Métodos
• *buscarConceptosOntologicos()*: **realiza la búsqueda de entidades ontológicas relacionadas con una palabra especifica.**

Tabla 15. Descripción de la Clase Palabra

Nombre
FuncionesWsd (paquete: co.edu.unicauca.desambiguation)
Descripción
Contiene las funciones que permite realizar los procesos para la desambiguación lingüística.
Atributos
Métodos
• *obtenerInstancia()*: **obtiene una instancia de esta clase.** • *sentidoTodasLasPalabras():* **obtiene el sentido de todas las palabras que conforman los atributos relevante y las entidades ontológicas, cada una con su respectivo método de desambiguación.** • *sentidoPrimeraPalabra():* **Encuentra el sentido para una sola palabra en función de otras palabras que conforman su contexto.** • *gradoSimilitud():* **obtiene el grado de similitud semántica entre dos definiciones de sentidos.**

Tabla 16. Descripción de la Clase PalabraOntologias

Nombre
PalabraOntologias (paquete: co.edu.unicauca.desambiguation)
Descripción
Esta clase mantiene la relación entre los atributos del descriptor y las ontologías encontradas para cada uno de ellos.
Atributos
• *palabra:* **Instancia de la clase Palabra, contiene una de las palabras que conforman un atributo relevante del descriptor** • *ontologiaList:* **Lista de la clase Ontología, contiene la lista de las ontologías encontradas para un determinado atributo**
Métodos

Tabla 17. Descripción de la Clase Sawsdl

Nombre
Sawsdl (paquete: co.edu.unicauca.desambiguation)
Descripción
Genera el archivo contenedor de las anotaciones SAWSDL.

Atributos
Métodos
• *obtenerInstancia()*: **obtiene una instancia de esta clase.** • *generarDocuemento():* **Genera el documento SAWSDL que contiene las anotaciones semánticas realizadas por la plataforma.**

A.3. Diagrama de Despliegue del Sistema

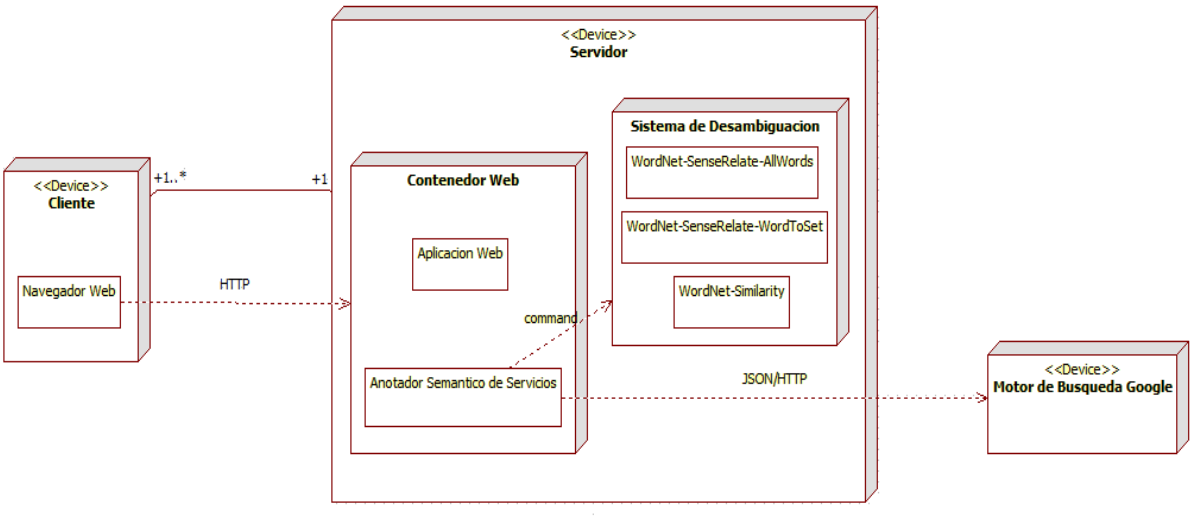


Figura 23. Diagrama de Despliegue del Sistema

El diagrama de despliegue del sistema, presentado en la Figura 21, ilustra el conjunto de nodos en los cuales se distribuyen físicamente los artefactos implementados para soportar su operación. Tal como se muestra en el diagrama, el sistema está dispuesto en una configuración típica de Cliente/Servidor. De esta manera, tanto la lógica de negocio como la lógica de presentación están desplegadas dentro del Contenedor Web presente en el nodo correspondiente al equipo Servidor. En este nodo también se encuentra disponible el sistema de Desambiguación, el cual corresponde a los componentes que permiten la desambiguación de atributos y ontologías y el cálculo de similitud entre entidades ontológicas y atributos del descriptor. Finalmente, un nodo denominado motor de búsqueda Google, el cual representa la conexión que necesita el anotador semántico con Google para la búsqueda de ontologías disponibles en la Web. En cuanto al equipo Cliente, es necesario que esté en red con el Servidor y que cuente con un navegador Web, por medio del cual el usuario pueda acceder al portal del sistema.