Recuperación de Información Bilingüe en la Web Semántica

Informe Final

Tutor
Msc. Juan José Prada

Integrantes
Silvana Castro
Marina Larraud

Instituto de Computación
Facultad de Ingeniería – UDELAR
2006 - 2007

Resumen

Internet se ha transformado en los últimos años en un agente protagónico de difusión del conocimiento. Su crecimiento se ha producido de manera desordenada, sin una estructura que la organice, lo que ocasiona que muchas veces la información no pueda ser recuperada satisfactoriamente.

Se ha planteado la idea de transformar la Web actual en una Web Semántica, donde documentos y aplicaciones estén teñidos de significado. Esto es una tarea de dimensiones mayúsculas, que involucra llegar a consensos respecto a la semántica de los distintos objetos, así como la creación de la metadata que relacione los documentos publicados en Internet con los conceptos definidos consensuadamente.

El trabajo en la Recuperación de Información Bilingüe en la Web Semántica plantea desafíos de diversa índole. El enfoque realizado en este proyecto fue considerar un dominio de conceptos acotado y buscar un esquema de representación adecuado, como una ontología multilingüe, la que se construyó tomando en consideración trabajos previos. El dominio elegido dio lugar a un gran número de instancias de la ontología, haciendo imprescindible efectuar una carga masiva automática.

Para efectuar recuperación de información no basta con construir y poblar la ontología bilingüe, sino que es necesario encontrar una manera de vincular las entidades definidas en la ontología con el conjunto de documentos sobre los que se efectuarán las búsquedas. Esta actividad recibe el nombre de anotación semántica, y puede ser realizada con distintos grados de automatización. Debido a la gran cantidad de páginas a anotar, en este trabajo se efectuó anotación semántica automática. En la misma, se observó que era necesaria cierta corrección manual posterior sobre las anotaciones, así que una vez finalizado el proceso de anotación se analizó la manera de revisar los metadatos generados y corregirlos.

Finalmente, se construyó una aplicación que permite hacer búsquedas semánticas sobre el dominio de conceptos elegido. Dicha aplicación devuelve documentos en español e inglés relevantes a la consulta, marcando en el texto las entidades identificadas. Cabe acotar que se trata de documentos distintos, no de versiones en español e inglés del mismo documento.

Se hizo una comparación entre los resultados obtenidos por el buscador semántico construido y un buscador por palabra clave común. Considerando los indicadores de *Recall* y *Precision*, los valores alcanzados afianzan la idea de continuar el trabajo en el enfoque adoptado.

Palabras clave: Web Semántica, Metadata, Ontologías Multilingües, Anotación Semántica, Recuperación de Información.

Tabla de Contenidos

1 Introducción

1.1. El problema

Internet ha registrado un crecimiento muy importante en los últimos años, tanto en el volumen de documentos como en la diversidad de idiomas y formatos, teniendo como consecuencia, entre otras, la dificultad para encontrar la información que se busca.

La Web actual es un repositorio enorme de documentos, el cual podría ser mucho mejor explotado si contara con una estructura que lo organice. Hay una tendencia corriente a transformar la Web actual en una Web Semántica, donde documentos y aplicaciones estén teñidos de significado. Esto es una tarea de dimensiones mayúsculas, que involucra desde llegar a consensos respecto a la semántica de los distintos objetos existentes, hasta la creación de la metadata que relacione la enorme cantidad de documentos publicados en Internet con los conceptos definidos consensuadamente.

Las aplicaciones multilingües son aquellas con habilidades para procesar documentos en más de un idioma. Ante una consulta de un término en un idioma, devuelven los documentos que lo contienen, con independencia del idioma en que se encuentra el documento y del idioma del término de la consulta.

Un enfoque para la construcción de estas aplicaciones, es la creación de corpus paralelos de los documentos en los que se va a realizar la búsqueda, esto es, cada documento está traducido a los otros idiomas de interés. Ante una consulta en un idioma, la aplicación puede devolver documentos en más de un idioma, aunque se trata de los mismos documentos traducidos.

Otro enfoque a considerar es la utilización de ontologías multilingües. Las ontologías son un concepto relativamente nuevo en la informática, y pueden ser utilizadas para representar el dominio sobre el que trata la colección de documentos. En el ámbito del Procesamiento del Lenguaje Natural, las ontologías se están empleando, entre otros usos, para construir representaciones independientes de la lengua que puedan servir de punto de encuentro entre dos o más lenguas naturales.

Si bien alcanzar la Web Semántica es por el momento una utopía, se ha estado trabajando en la recuperación de información bilingüe a pequeña escala, creando para ciertos dominios en particular, la estructura semántica requerida.

1.2. Motivación

En los últimos años, se ha investigado mucho en el área de la Web Semántica y las ontologías. No está dicha la última palabra en cuanto al desarrollo e implementación de las ontologías. Por el contrario, la metodología y los lenguajes para expresarlas son muy recientes. Permanentemente surgen nuevos lenguajes que extienden a los ya existentes y que incluyen mejoras a los mismos, lo que muestra un proceso de innovación constante en el área.

Hasta el momento, las ontologías multilingües han sido muy poco explotadas, existiendo escasa documentación y proyectos que vayan en esta dirección. Es por esto que resulta muy interesante introducirse en este tema en constante crecimiento, del cual se puede aprender mucho, así como quizás brindar algún aporte en determinadas áreas o herramientas que aún no han sido demasiado estudiadas o analizadas.

OntoChoike [3], un proyecto de grado de la Facultad de Ingeniería de la UdelaR, desarrolló una ontología sobre un dominio específico, para realizar búsquedas basadas en la misma.

El dominio de trabajo de OntoChoike comprende documentos en español e inglés, lo que sumado al desarrollo previo de una ontología en español sobre el mismo, lo transforma en buen punto de partida para un trabajo nuevo. Están dadas las condiciones para aplicar los conceptos de la Web Semántica a un caso de estudio de pequeña escala, ver qué nuevas dificultades aparecen y qué resultados se obtienen.

1.3. Objetivo

El objetivo del proyecto es efectuar recuperación de información bilingüe en un dominio en particular. Para ello, se pretende construir una ontología bilingüe (español e inglés) sobre el dominio elegido, generar semiautomáticamente metadata referida a las páginas Web del sitio y construir un buscador que permita recuperar información del mismo, realizando la consulta tanto en uno como en otro idioma.

Se espera obtener un producto que, ante el requerimiento de un usuario, procese la solicitud en ambos idiomas y recupere los documentos relevantes también en ambos idiomas, donde no necesariamente se traten de versiones del mismo documento.

1.4. Antecedentes

Como se expresó anteriormente, existen pocos proyectos que involucren ontologías multilingües. En esta sección se mencionan algunos trabajos que fueron encontrados durante el estudio inicial, que se relacionan con recuperación de información multilingüe y ontologías.

El proyecto e-COURT [1] es un proyecto europeo que reúne a la Universidad de Amsterdam, y participantes gubernamentales e industriales. Pretende introducir las tecnologías de la información en las cortes criminales europeas, recuperando información multilingüe contenida en las grabaciones de audio, video y otros formatos generados durante las sesiones.

En este proyecto, las ontologías son utilizadas principalmente para la recuperación de información y especialización o extensión de las consultas, tarea facilitada por la jerarquía de clases de la ontología. Son utilizadas en la traducción de las consultas para obtener documentos en otros idiomas. Las ontologías se utilizan también para especificar y generar metadata, mediante la anotación de los documentos (tagging).

El dominio comprende información muy diversificada, por lo que no puede ser capturada en una única ontología. Por tal motivo, los autores generaron ontologías especializadas, y para prevenir inconsistencias, desarrollaron una ontología central que sirviera de nexo entre las mismas.

El proyecto MuchMore [2] tiene entre sus principales objetivos desarrollar métodos para el uso efectivo de ontologías y tesauros para anotación semántica en textos médicos en inglés y alemán y evaluar el impacto de dicha información semántica en la recuperación multilingüe.

En el marco del proyecto MuchMore se ha desarrollado un prototipo de sistema de recuperación de información multilingüe para efectuar recuperación de documentos (en inglés y alemán) a partir de un documento de consulta (en inglés o alemán).

La tarea de recuperación de información multilingüe se ha resuelto mediante una combinación de métodos basados en conceptos (ontologías) y en corpus paralelos. La colección de documentos es un corpus paralelo de reportes médicos en inglés y

alemán y ha sufrido varios procesos de normalización para producir una salida en texto plano.

El Proyecto ITEM [69] exploró la viabilidad de integrar diferentes técnicas de procesamiento del lenguaje natural en un motor de búsqueda de información en un entorno multilingüe. Empleando el Índice Interlingua de EuroWordnet para indexar los documentos de los diferentes idiomas y las consultas, se realizó una búsqueda a nivel conceptual y de forma independiente del idioma.

La experiencia con este motor de búsqueda indicó que la indexación conceptual tiene ciertas ventajas respecto a las aproximaciones basadas en traducción. Por ejemplo, la expansión automática de las consultas empleando las relaciones de EuroWordnet permite traducir conceptos que no tienen una representación directa en el otro idioma. Por ejemplo "grand jury" en inglés no tiene un concepto equivalente en castellano. Sin embargo un hiperónimo suyo es "jury" que tiene una traducción directa como "jurado". Así se puede paliar la pérdida de información que implica el proceso de traducción.

Sin embargo, se comprobaron también los problemas del enfoque: entre otros, que las técnicas de desambiguación automática no han alcanzado todavía un grado suficiente de madurez para un enfoque tan ambicioso.

1.5. Organización del documento

Este documento se compone de nueve capítulos. En el Capítulo 2 se presenta un marco teórico, que incluye los conceptos principales abarcados para el desarrollo de este trabajo. En el Capítulo 3, se hace un resumen de la investigación inicial, que es un resumen del documento Apéndice A – Estudio del Estado del Arte. El Capítulo 4 describe el trabajo realizado, profundizando en los aspectos más relevantes. El Capítulo 5 describe diseño e implementación y el Capítulo 6 las pruebas llevadas a cabo y los resultados obtenidos. En el Capítulo 7 se presentan las conclusiones y el trabajo a futuro. Por último, se encuentran un glosario y las referencias bibliográficas asociadas a este documento.

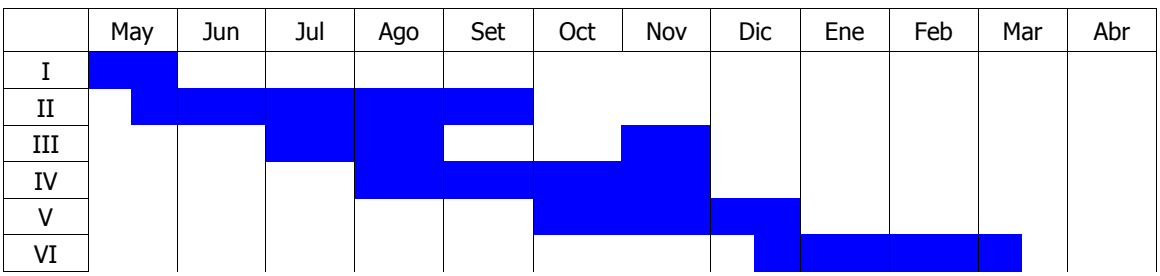
1.6. Plan de Trabajo

A grandes rasgos, el proyecto ha cursado varias etapas: una etapa inicial de investigación, seguida de varios ciclos de análisis y toma de decisiones y por último un fuerte componente de implementación. La documentación se fue confeccionando a lo largo de todo el proyecto, incrementándose en las etapas de investigación y luego de finalizado el diseño.

Se comenzó con un estudio del sitio Web y las páginas del dominio sobre el cual se iba a trabajar, así como también de trabajos anteriores realizados con dicha organización. Seguidamente, se investigaron diferentes técnicas de construcción semiautomática de ontologías multilingües y la generación de metadata.

Posteriormente, se estudiaron técnicas de recuperación de información sobre la Web Semántica.

Finalmente, se desarrolló una aplicación para recuperar documentos (en español e inglés) mediante técnicas de la Web Semántica, basados en la ontología construida.



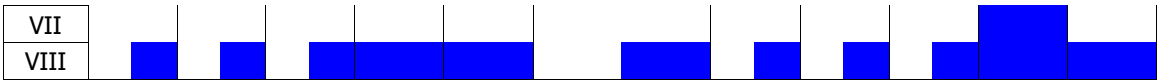


Tabla 1: Cronograma.

I. Análisis del Proyecto "Generación de descripciones de Web Semántica en una red de ONG"

II. Investigación de temáticas de la Web semántica

III. Diseño y construcción de la Ontología

IV. Creación de diccionarios y población de la ontología en base a éstos

V. Profundización en la investigación de técnicas de anotación y recuperación de documentos en la Web semántica

VI. Diseño e implementación de consultas

VII. Diseño e implementación de la interfaz para recuperar los documentos

VIII. Elaboración de informes

A continuación se detallan las etapas mencionadas:

En la etapa I se realizó un estudio del Proyecto mencionado, analizando principalmente la ontología que iba a servir de base para la ontología a construir

En la etapa II se hizo una investigación de todos los temas que abarca el proyecto: Web semántica, ontologías, metadata y las herramientas relacionadas. Esta investigación tuvo como resultado el documento Apéndice A – Estudio del Estado del Arte.

En la etapa III se procedió a diseñar la ontología en base a OntoChoike y a las necesidades particulares del presente proyecto.

En la etapa IV se continuó con la creación de diccionarios XML conteniendo la información de las entidades con datos en ambos idiomas. A partir de estos diccionarios se comenzó con la carga o población de la ontología recientemente creada.

En la etapa V se procedió a hacer un estudio exhaustivo de los anotadores semánticos disponibles, que cumplieran con las necesidades del proyecto y a la elección del mismo.

En la etapa VI se realizó el diseño e implementación de las consultas semánticas posibles a partir de la herramienta de anotación elegida.

En la etapa VII se definió y construyó la interfaz necesaria para que el usuario pueda explotar las consultas fácilmente.

Los informes fueron elaborados durante todo el proyecto, con especial énfasis en el período previo a la presentación de avance (efectuada a principios del mes de octubre), en la construcción del informe del Estudio del Estado del Arte y la documentación final.

2 Marco Teórico

Se realizó una puesta a punto en las temáticas relacionadas al proyecto:

- Web Semántica
- Metadata
- Ontologías
- Anotación Semántica
- Recuperación de Información

A continuación se presenta un resumen del conocimiento relevado. El documento Apéndice A – Estudio del Estado del Arte extiende esta información.

2.1. La Web Semántica

"La Web Semántica es una extensión de la Web actual en la cual la información recibe un significado bien definido, permitiendo a las computadoras y las personas trabajar en cooperación de mejor forma." (Tim Berners-Lee, James Hendler, Ora Lassila).

Se basa en la idea de añadir información extra a la World Wide Web, que permita definir conceptos y relaciones en un dominio específico. Dicha información, que sirve para describir el contenido, significado y relación de los datos, debe estar representada formalmente para permitir que las computadoras "comprendan" tal información y puedan procesarla inteligentemente. Se trata de una Internet en la que las computadoras no sólo son capaces de presentar la información contenida en una página Web, como hacen actualmente, sino que además pueden "entender" dicha información.

Al dotar a la Web de más significado, o sea, de más semántica, se pueden obtener soluciones a problemas habituales en la búsqueda de información, gracias a la utilización de una infraestructura común, mediante la cual es posible compartir, procesar y transferir información de forma sencilla [4].

La Web Semántica no trata de links entre páginas Web, sino que describe las relaciones entre las cosas y las propiedades de las mismas [5].

Los usuarios de Internet podrán encontrar respuestas a sus preguntas de forma más rápida y sencilla gracias a una información mejor definida.

La Web Semántica está basada en dos conceptos fundamentales [6]:

- La descripción del significado que tienen los contenidos en la Web
- La manipulación automática de estos significados

La descripción del significado requiere conceptos ligados a:

- La Semántica, entendida como significado procesable por máquinas
- Los Metadatos, como contenedores de información semántica sobre los datos

- Las Ontologías, conjunto de términos y relaciones entre ellos que describen un dominio de aplicación concreto

La manipulación automática de los contenidos se hace a través de:

- La lógica matemática, que permite establecer reglas para tratar el contenido semántico
- Los motores de inferencia, que permiten combinar conocimientos conocidos para elaborar nuevos conocimientos

Básicamente, la Web Semántica pretende una forma universal de representar las relaciones entre los datos y sus significados para que un sistema automático sea capaz de seguir la estructura de las relaciones y saque sus propias conclusiones respecto a la búsqueda que se está realizando.

En el Apéndice B se hace una descripción más detallada de las características, arquitectura y futuro de la Web Semántica.

2.2. Metadata

En el sentido genérico se define Metadata como datos acerca de los datos. En el contexto de este trabajo, se define como información estructurada (objeto de información) que es procesable automáticamente, que puede ser usada para dar soporte a distintas operaciones.

Los objetos de información constan de tres características básicas:

- Contexto: para qué, cómo, quién, cuándo fueron creados
- Contenido: sobre qué es el objeto o qué contiene
- Estructura: conjunto de relaciones dentro del objeto o con otros.

Se ha clasificado la metadata según diversos criterios:

- Estructura: estructurada, semi-estructurada, no estructurada
- Relación con los datos: externa, embebida, asociada
- Características del objeto de información: administrativa (o técnica), descriptiva y estructural cuando refieren al contexto, contenido y estructura respectivamente.
- Uso: del dominio (descriptiva), del sistema (operativa)
- Origen o forma de creación: manual, automática, consorcios (estándar), individuos (no estándar).

Las características deseables para la metadata son: completitud, consistencia y accesibilidad. La estructura de la información que es sintetizada en la metadata, puede ser muy variada, y comprende sistemas basados en texto (diccionario, índice y catálogo) y basados en grafos (taxonomía, tesauro, red semántica, ontología). Cada sistema tiene características particulares y permite expresar diferentes grados de semántica.

El esquema de metadatos más adecuado para este trabajo por su expresividad y capacidad de realizar inferencias, es la ontología.

La Figura 1 muestra los diferentes tipos de metadata reconocidos, ordenados de acuerdo al grado de expresividad que ofrecen.

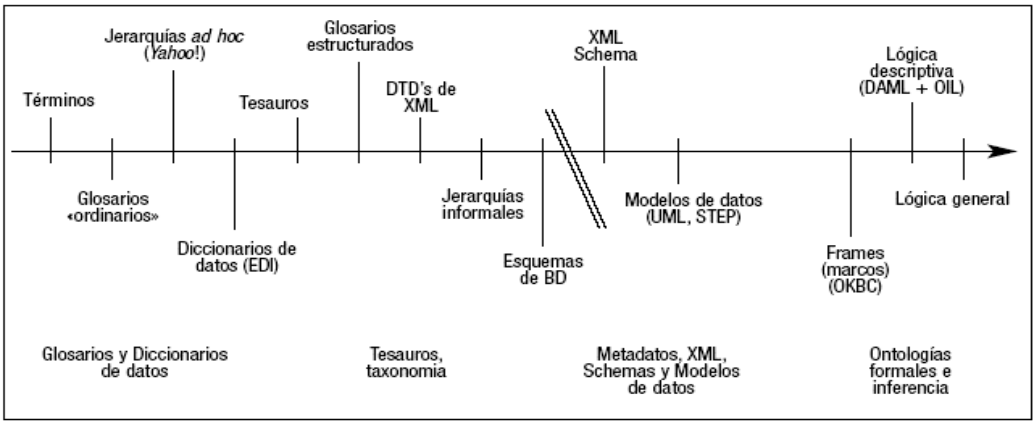


Figura 1: Tipos de metadata/ontologías según Gruninger y Uschold (2002)

2.3. Ontologías

2.3.1. Definición

Chantal Pérez, en su trabajo Estudios de Lingüística Española [7], presenta dos enfoques diferentes para definir las ontologías. Citando a Mahesh, una definición de diccionario típica la identifica como "la rama de la metafísica que estudia la naturaleza de la existencia". En las aplicaciones reales, sin embargo, una ontología es una entidad computacional, y no ha de ser considerada como una entidad natural que se descubre, sino como recurso artificial que se crea. Una ontología puede interpretarse como un entendimiento común y compartido de un dominio, que puede comunicarse entre científicos y sistemas computacionales. El hecho de que puedan compartirse y reutilizarse en aplicaciones diferentes, explica en parte el gran interés suscitado en los últimos años en la creación e integración de ontologías.

El sinónimo más usual de ontología es *Conceptualización*. Según la definición de Gruber [8], una ontología constituye una especificación explícita y formal de una conceptualización compartida.

- Conceptualización: se refiere a un modelo abstracto de algún fenómeno del mundo, del que se identifican los conceptos que son relevantes
- Explícita: hace referencia a la necesidad de especificar de forma consciente los distintos conceptos que conforman una ontología
- Formal: indica que la especificación debe representarse por medio de un lenguaje de representación formalizado
- Compartida: refleja que una ontología debe, en el mejor de los casos, dar cuenta de conocimiento aceptado (como mínimo, por el grupo de personas que deben usarla).

2.3.2. Componentes

En el artículo *Translation Approach to Portable Ontology Specifications* [9], Gruber menciona los componentes de las ontologías que sirven para representar el conocimiento de un dominio. A continuación se mencionan y detallan cada uno de ellos.

2. 3. 2. 1. Conceptos

Los Conceptos, también llamados Clases, son las ideas básicas que se intentan formalizar. Describen y representan conceptos relevantes al dominio. Se utilizan para estructurar el conocimiento y percepción del mundo circundante.

Los conceptos pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento [10]. Se suelen organizar en una jerarquía.

2. 3. 2. 2. Propiedades

Están asociadas a los conceptos y describen sus características y atributos.

- Atributos: expresan cualidades de un concepto, y el valor de dichas cualidades puede ser tanto escalar (numérico o cuantitativo) como literal.

- Relaciones: propiedades que conectan dos conceptos entre sí de manera biunívoca.
  Representan la interacción y enlace entre los conceptos de un dominio. Ejemplos de relaciones serían: subclase-de (taxonomía), parte-de (mereología), parte-exhaustiva-de, conectado-a, etc.

2. 3. 2. 3. Funciones

Son un tipo concreto de relación donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología. Pueden aparecer funciones como categorizar-clase, asignar-fecha, etc.

2. 3. 2. 4. Axiomas

Los Axiomas, o reglas de restricción, son teoremas que se declaran sobre relaciones que deben cumplir los elementos de la ontología. Modelan sentencias lógicas que se verifican siempre.

Se utilizan habitualmente para modelar conocimiento que no puede ser representado utilizando los otros componentes.

Se pueden utilizar axiomas para tres tipos de propósitos:

- Creación de conocimiento: sirven para obtener nuevas sentencias lógicas a partir de la información almacenada en la base de conocimiento.

- Restricciones: indican propiedades que el modelo debe cumplir. Sirven para detectar inconsistencias.

- Reglas reactivas: determinan acciones a tomar por un sistema basado en conocimiento como consecuencia de que se cumplan ciertas condiciones.

Los axiomas, junto con la herencia de conceptos, permiten inferir conocimiento que no está indicado explícitamente en la taxonomía de conceptos.

La decisión de utilizar o no reglas, debe tomarse cuidadosamente, ya que las reglas proporcionan una gran capacidad expresiva pero también una mayor complejidad al razonar con la ontología, lo que en la actualidad, puede ocasionar problemas de escalabilidad.

2. 3. 2. 5. Instancias

Las instancias son entidades que pertenecen a una determinada clase. Se utilizan para representar objetos determinados de un concepto.

2.3.3. Principios de Diseño

Para guiar y evaluar el diseño de ontologías, Gruber [9] propone un conjunto de criterios, con el fin de compartir datos y alcanzar la interoperabilidad entre programas basados en una conceptualización compartida.

- Claridad: una ontología debería comunicar efectivamente el significado pretendido de los términos definidos y las definiciones ser objetivas e independientes del contexto. La formalización es el medio para alcanzar este objetivo.

- Coherencia: una ontología debería ser coherente, realizar inferencias que sean consistentes con las definiciones.

- Extensibilidad: una ontología debería ser diseñada para anticipar los usos del vocabulario compartido. Debería poder definirse nuevos términos para usos especiales basados en el vocabulario existente, de tal forma que no requiera modificar las definiciones previas.

- Sesgo de codificación mínimo: la conceptualización debería ser especificada sin depender de una codificación en particular a nivel de símbolos.

- Compromiso ontológico mínimo: una ontología debería sostener el compromiso mínimo que permita soportar las actividades de compartir el conocimiento.

2.3.4. Clasificación

Guarino [25] reconoce dos dimensiones de clasificación: según el nivel de detalle y según el nivel de dependencia respecto a una tarea o punto de vista particular.

La primera dimensión comprende desde ontologías con alto nivel de detalle, que se aproximan a la realidad, hasta ontologías muy simples que pueden ser construidas por un grupo de usuarios menor.

Tipos de Ontologías:

- Livianas (LightWeight): incluye conceptos, taxonomía de conceptos, relaciones entre los mismos y propiedades que los describen.
- Pesadas (HeavyWeight): se agregan axiomas y restricciones a las ontologías livianas.

La segunda dimensión distingue cuatro tipos de ontologías: de alto nivel, de dominio, de tareas y de aplicación. Ontologías de alto nivel describen conceptos generales, independientes de un dominio en particular, como espacio, tiempo, entre otros. Ontologías de dominio y de tareas describen el vocabulario relacionado con un dominio genérico y con tareas, respectivamente, especializando los términos introducidos en las ontologías de alto nivel. Ontologías de aplicación describen conceptos relacionados con un dominio y una tarea o actividad en particular.

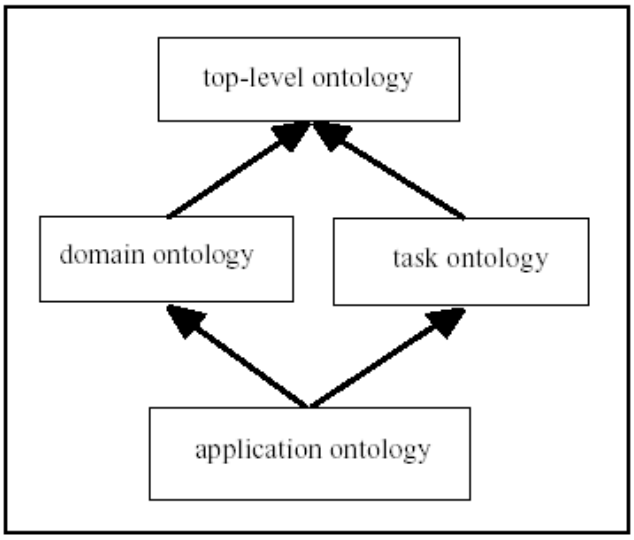


Figura 2: Clasificación de Ontologías según nivel dependencia (Guarino)

2.3.5. Representación Formal del Conocimiento

La Inteligencia Artificial ha desarrollado varios formalismos para la representación del conocimiento: Redes Semánticas, Marcos y Lógica Descriptiva entre otros. Las redes semánticas y los marcos pueden ser considerados como los ancestros de la Lógica Descriptiva, y la Lógica de Primer Orden el formalismo que constituye el punto de partida. A continuación se presenta un resumen del trabajo realizado por Duque [11], que muestra la evolución de los lenguajes para la construcción de la Web Semántica.

2. 3. 5. 1. Redes Semánticas

Es un mecanismo para representar conceptos mediante grafos etiquetados dirigidos, compuestos de un conjunto de nodos y un conjunto de enlaces unidireccionales. Los nodos representan conceptos, instancias de conceptos y valores de propiedad. Los enlaces representan propiedades de conceptos o relaciones entre los mismos. Estas relaciones pueden describir jerarquías clase/subclase, relaciones sujeto/objeto y relaciones basadas en operadores y/o.

La Figura 3 presenta una red semántica, que describe la relación entre un concepto y su instancia y ejemplifica una relación.

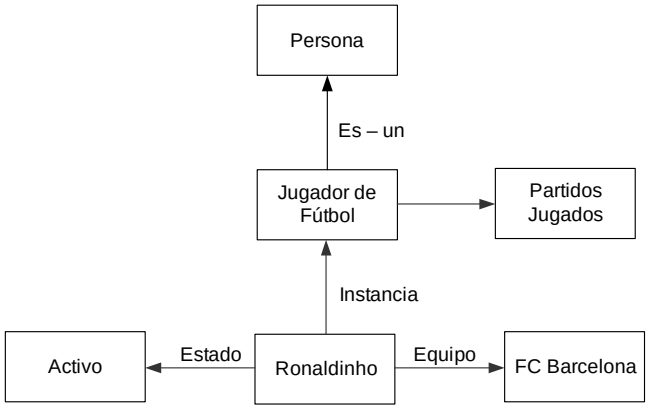


Figura 3: Red Semántica

2. 3. 5. 2. Marcos (Frames)

Los marcos están estrechamente relacionados a las redes semánticas, que a su vez están basadas en la idea de memoria asociativa humana. Pueden entenderse como estructuras de datos de nodos (conceptos) y enlaces (asociaciones) entre ellos. Permiten efectuar razonamientos mediante comparación de valores de propiedades. Tienen más expresividad que las redes semánticas, ya que pueden representar valores por defecto y restricciones de cardinalidad sobre los atributos, por ejemplo.

2. 3. 5. 3. Lógica Descriptiva

La lógica descriptiva parte de la existencia de un conocimiento intencional y uno extensional. El primero de ellos, conocido como TBOX, representa las jerarquías conceptuales del dominio de aplicación. Su estructura básica es el concepto, donde la definición de un concepto nuevo se realiza en términos de uno o varios existentes. Deben cumplirse dos restricciones: no puede existir más de una definición por término y sólo se permiten definiciones acíclicas. El conocimiento extensional se denomina ABOX y contiene el conocimiento de los individuos que participan en el dominio de interés. Aquí redistingue entre aserciones de contexto y aserciones de rol. Las primeras hacen referencia a las relaciones que se establecen entre conceptos e individuos y las segundas a las establecidas entre individuos.

2.3.6. Metodologías

La construcción de ontologías es un proceso que requiere de una metodología y herramientas. Se entiende por metodología al conjunto de métodos y técnicas destinadas a promover un cierto enfoque para la resolución de un problema.

En el Capítulo 3, se nombran las metodologías más conocidas y se profundiza en Enterprise Ontology, la utilizada en este trabajo.

2.3.7. Herramientas de desarrollo de Ontologías

El proceso de desarrollo de ontologías ha dado lugar a la creación de una gran cantidad de herramientas que ayudan a simplificar las tareas. Así, editores de ontologías dan soporte a la creación, unión, desarrollo distribuido y población de las mismas. Otras herramientas ofrecen capacidades para convertirlas a diferentes lenguajes o diferentes formatos dentro de un mismo lenguaje. El Capítulo 3 reseña algunas de las herramientas más conocidas y las que han sido de mayor utilidad en este trabajo.

2.3.8. Razonadores

Por *Razonamiento* se entiende tareas de inferencia de conocimiento nuevo (no especificado explícitamente) a partir de un conjunto de sentencias. En la Web Semántica, se distinguen dos campos de aplicación principales:

- Diseño de ontologías. Se facilitan las tareas de construcción y modelado a partir de la verificación de consistencia entre las clases y el descubrimiento de relaciones implícitas no deseadas.
- El desarrollo de ontologías a partir de bases de conocimiento aprovecha las características de verificación de conjuntos de datos y consultas.

Se espera que un razonador provea los siguientes servicios de inferencia [26]:

- Verificación de consistencia, lo que asegura que una ontología no contiene hechos contradictorios.

- Satisfactibilidad de conceptos, que verifica si es posible que una clase contenga instancias. Si una clase no puede satisfacerse, definir una instancia de dicha clase causará que toda la ontología sea inconsistente.
- Clasificación, lo cual computa las relaciones de subclase entre todas las clases definidas para crear la jerarquía de clases completa. Dicha jerarquía puede ser usada para responder consultas, tales como obtener las subclases de una clase.
- Realización, lo cual encuentra la clase más específica a la que un individuo pertenece; en otras palabras, obtiene el tipo directo de cada individuo.

El Capítulo 3 incluye en detalle, algunos de los razonadores disponibles y sus características principales.

2.4. Ontologías Multilingües

En esta sección, se presenta el enfoque ofrecido por Guyot en su artículo *Recuperación de Información basada en Ontologías Multilingües* [12].

Una ontología multilingüe está definida por una ontología y un conjunto de diccionarios (un diccionario por cada lenguaje). La misma contiene un conjunto de conceptos identificados C, relacionados por un conjunto de relaciones R.

Un diccionario D_L es una asociación de conceptos C con un conjunto de términos T_L pertenecientes a un lenguaje L. Se denota:

$$D_L: C \rightarrow T_L$$

El concepto c es etiquetado por un conjunto de términos $t_1, t_2,\ldots, t_n$ en el lenguaje L. Se denota:

$$D_L(c) = \{t_1, t_2,\ldots, t_n\}$$

También se define la relación recíproca:

$$S_L: T_L \rightarrow C \text{ por } S_L(t) = \{c \in C \mid t \in D_L(c)\}$$

El término t indica los conceptos $c_1, c_2,\ldots, c_m$. Se denota:

$$S_L(t) = \{c_1, c_2,\ldots, c_m\}$$

A continuación se presenta un ejemplo de un concepto extraído de la ontología del autor:

Figura 4: Ejemplo de definición de concepto

8612 - Unidad de medida (en Estados Unidos y Gran Bretaña) igual a un doceavo de un pie.

Asociaciones entre términos y conceptos:

- $D_{EN}(8612) = \{inch\}$
- $S_{FR}(pouce) = \{8612, 28845\}$

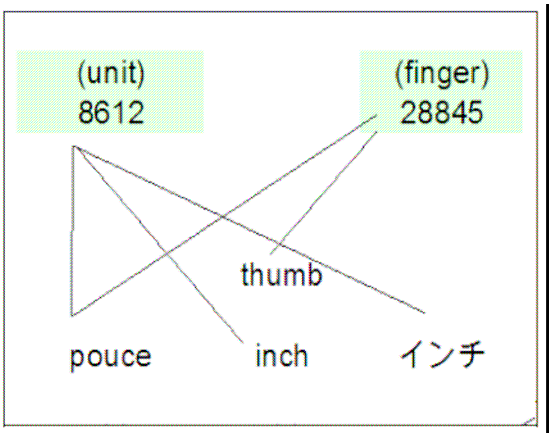


Figura 5: Ejemplo de asociación término-concepto

Para cada documento de la colección, se usa la ontología multilingüe para mapear cada término a su concepto correspondiente.

El documento $d_L = <t_1, t_2,..., t_n>$ es una secuencia de términos del conjunto T_L del lenguaje L. Para llevar a cabo el mapeo término-concepto, se aplica la función S_L en cada término t_i del documento d_L:

$$S_L(t) = \{c \in C \mid t \in D_L(c)\}$$

Así se obtiene la representación conceptual del documento que se denota:

$$CR(d_L) = <S(t_1), S(t_2),..., S(t_n)>$$

Finalmente, $CR(d_L)$ es una secuencia de conjuntos de conceptos.

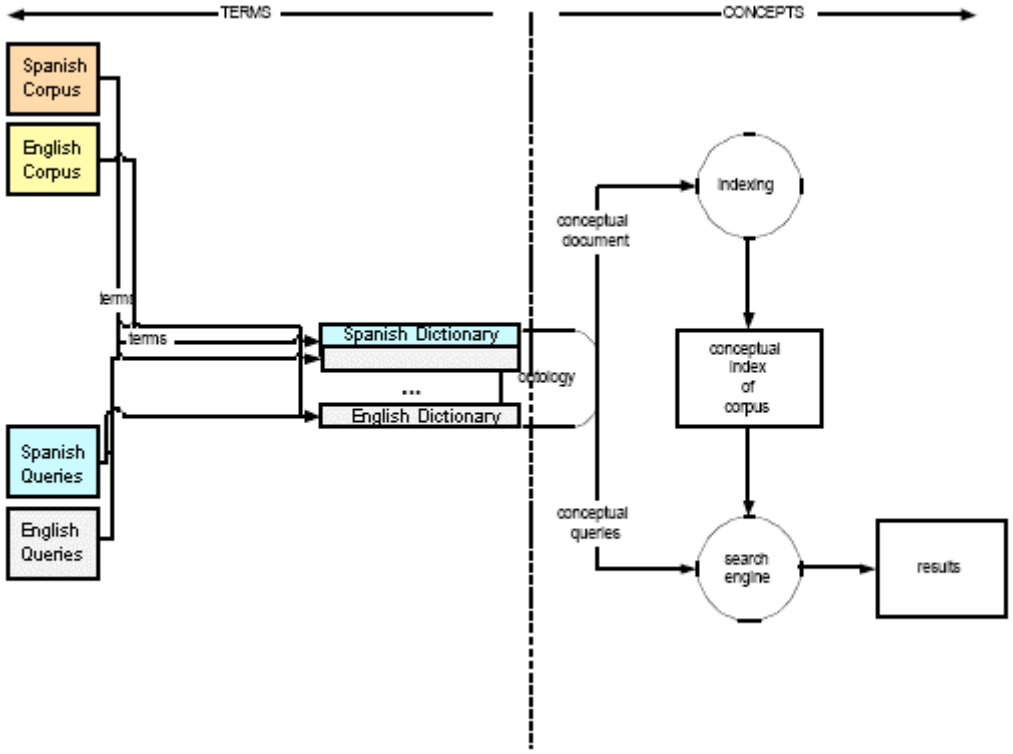


Figura 6: Proceso de indexación y consulta

Este enfoque no efectúa una traducción de los documentos. Se lleva a cabo una representación de los documentos basada en la ontología multilingüe. Además, no hay dependencia de traductores automáticos entre los diferentes lenguajes. Cuando se agrega un nuevo lenguaje, sólo se agrega en la ontología, un nuevo diccionario de mapeo.

2.5. Bases de Conocimiento

En el contexto de las tecnologías de la información, una Base de Conocimiento (Knowledge Base o KB) es un recurso formal, entendible por las máquinas y generalmente online, para la diseminación de la información.

A los efectos del proyecto, una vez construida la ontología, se debe considerar la descripción de las entidades o instancias. Debería ser posible identificar, describir e interconectar entidades de forma flexible y estándar.

La ontología se puede ver como una especie de esquema para la KB, por tanto ambos deberían ser mantenidos en un almacenamiento semántico. Se requiere de un sistema para administración y razonamiento de conocimiento formal que provea las operaciones básicas: almacenamiento y recuperación de acuerdo a la sintaxis y semántica del formalismo elegido. El almacenamiento podría o no proveer inferencia, e implementar diferentes estrategias de razonamiento.

La KB podría contener entidades clasificadas según su origen:

- pre-poblada: entidades importadas o adquiridas de fuentes confiables
- extraídas automáticamente: descubiertas en el proceso de anotación via mecanismos de extracción de información

Dado que las técnicas de extracción de información actuales permiten en reconocimiento de entidades previamente desconocidas y relaciones entre ellas, es

razonable usar dichas técnicas para el enriquecimiento de la KB. Estos métodos son aún imprecisos, por lo que el conocimiento adquirido por este medio debería poder distinguirse de aquel obtenido mediante pre-población. Las entidades reconocidas podrían transformarse en confiables en algún momento mediante una validación semi-automática [13].

2.6. Anotación Semántica

Para aportar significado a las páginas Web, éstas se pueden anotar con información relativa a la ontología, mediante el agregado de metadatos semánticos. Esta información no es visible cuando la página se muestra con un navegador, pero sí lo es para agentes o aplicaciones que buscan información en la Web.

El conocimiento contenido en las páginas se puede anotar usando las ontologías como un esquema para expresar metadatos. La forma de añadir esta meta-información depende del lenguaje que se utilice para ello.

En general la anotación semántica de un recurso Web consiste en "mapear" o relacionar su contenido entero o una parte de él (una palabra concreta) con cierto identificador. Tal identificador (generalmente una URI) determina inequívocamente el concepto mencionado en el contenido del recurso.

Este mapeo es efectuado por el anotador al encontrar apariciones de conceptos e instancias en los textos, en cuyo caso se crea una anotación (un enlace bidireccional entre el concepto y el documento).

Las anotaciones son la base del procesamiento automático de las páginas Web, y por lo tanto la anotación semántica tiene una importancia crítica.

Si las páginas Web tuvieran asociadas anotaciones formales que identificaran los conceptos y entidades contenidos en las páginas, un buscador semántico podría devolver únicamente los documentos relacionados con la información de interés.

Las aplicaciones desarrolladas sobre la Web Semántica podrán explotar tal conocimiento para asistir en la automatización de tareas que en la actualidad se realizan en interacción con el usuario. Estos sistemas tienen dos componentes básicos: una Base de Conocimiento de los hechos conocidos y un motor de inferencia.

Las anotaciones no están restringidas a un vocabulario fijo. Este requisito ha causado que el W3C (World Wide Web Consortium) [44] haya definido una recomendación para representación del conocimiento en la Web Semántica: RDF.

Disponer de anotaciones semánticas puede no ser suficiente para las aplicaciones que utilicen la Web Semántica. La mayoría de estas aplicaciones necesitarán un modelo del dominio en el que operen, que incluya el vocabulario de los conceptos relevantes a ese dominio, y sus interrelaciones. De esta forma, el sistema será capaz de obtener conclusiones y tomar decisiones procesando las anotaciones extraídas de páginas Web. Tales modelos son definidos por medio del segundo ingrediente fundamental de la Web Semántica, las ontologías.

La anotación semántica se puede clasificar por el grado de automatización de las tareas de anotación. Se distingue entre sistemas manuales, semiautomáticos y automáticos, cada uno con las características que se describe a continuación:

- Sistemas de Anotación Manuales: proporcionan típicamente una interfaz de usuario que permite a los anotadores humanos ver y navegar simultáneamente las ontologías y los recursos Web, usando el conocimiento modelado en las ontologías para agregar anotaciones a recursos Web.

  Tienen como ventaja que permiten especificar anotaciones de grano muy fino pero como contrapartida, dichos sistemas se apoyan muy fuertemente en la experiencia del usuario. La anotación manual es muy costosa en

términos de tiempo por lo que es prácticamente imposible la anotación de grandes cantidades de documentos.

- Sistemas de Anotación Semiautomáticos: los agentes Web pueden ser diseñados para tratar la información situada en las páginas Web de manera semiautomática. Se trata de convertir la información en conocimiento, referenciando datos dentro de las páginas Web a metadatos con un esquema común consensuado sobre algún dominio. Mediante técnicas de PLN (Procesamiento de Lenguaje Natural) se puede extraer referencias en el texto a conceptos descritos en ontologías.

  Estos sistemas generalmente requieren de una cierta cantidad de documentos manualmente anotados, a partir de los cuales el sistema pueda ser entrenado. Si bien el nivel de automatización alcanzado es mayor, la tarea sigue siendo ardua, compleja y pasible de errores.

- Sistemas de Anotación Automáticos: son herramientas que aplican técnicas de extracción de información de lenguaje natural para generar automáticamente anotaciones en páginas Web. Evitan el trabajo de entrenar el sistema. Sin embargo, no todas las anotaciones producidas son confiables [14].

En el Capítulo 3 se mencionan los anotadores encontrados y se describen brevemente los más importantes.

2.7. Recuperación de Información

Baeza-Yates [15] define el problema de Recuperación de Información (RI) como que, dada una necesidad de información a través de una consulta, para un perfil de usuario, y un conjunto de documentos, ordenar los documentos de más a menos relevantes para esa necesidad, y presentar un subconjunto de los más relevantes.

Propone dos etapas para abordar el problema:

- elegir un modelo que permita calcular la relevancia de un documento frente a una consulta
- diseñar algoritmos y estructuras de datos que lo implementen eficientemente (índices)

Tomando como base el proyecto de grado MegúSearch [16], a continuación se describen brevemente los modelos clásicos de recuperación de información más populares. Estos modelos estipulan que cada documento puede ser especificado mediante un conjunto de palabras representativas denominadas términos índice. Un término índice es una palabra del documento cuyo significado ayuda a recordar los temas principales tratados en el mismo. Así, los términos índice sirven para indexar y resumir los documentos que los contienen y se pueden utilizar para obtener acceso rápido su información. Cada término índice podría tener distinta relevancia según el documento, por lo que puede ser ponderado con algún valor que represente esta información.

- Modelo Booleano

  En el modelo booleano, los documentos son representados con vectores de pesos binarios que muestran la aparición (o no) de cada término. Para consultas de una palabra, el documento es relevante si y sólo si la contiene. Es posible generar consultas de más de una palabra utilizando los operadores AND, OR y NOT, las cuales devolverán documentos que incluyan todas, alguna o ninguna de las palabras consultadas respectivamente.

  Así, este modelo sólo puede afirmar si un documento es relevante o no frente a una consulta, pero no considera si existe un calce parcial de un documento (por ejemplo, que cumpla con casi todas las cláusulas de un

AND). Tampoco es capaz de comparar la relevancia entre documentos, discriminando entre los que contienen una o todas las cláusulas de un OR, o la cantidad de apariciones de las palabras en el documento.

- Modelo Vectorial

  El Modelo Vectorial intenta medir la concordancia parcial entre documentos y consultas. Se asigna un peso no binario a los términos que aparecen en los documentos y las consultas, en general relacionado con su frecuencia de aparición. Se considera que si un término aparece repetidamente en un documento, el término es relevante en él. Sin embargo, si aparece en muchos documentos, entonces no es útil para distinguir entre documentos. También se considera hacer operaciones sobre las frecuencias de aparición para no favorecer a los documentos más largos.

Los modelos clásicos tienen ciertas falencias comunes, la más notoria de ellas es la incapacidad de capturar las relaciones entre los términos. Por ejemplo, si se busca "guerra fría" podría ser de interés recupera documentos que habla sobre "la crisis de los misiles cubanos". Sin embargo, el sistema no puede decir que existe relación entre ambos y la intersección de vocabulario puede ser nula.

La solución puede pasar por el análisis lingüístico:

- lematizar (para no perder variantes de la misma palabra)
- etiquetar (para distinguir verbos de sustantivos)
- detectar frases comunes (para no recuperar información sobre "heladeras" cuando se pregunte por "guerra fría")

Otro elemento es el uso de tesauros y sinónimos para expandir la consulta, de modo de poder recuperar "vendo camioneta usada" frente a la consulta "coches de segunda mano". Sin embargo, mantener un buen tesauro es costoso en términos de trabajo manual y no suele ser posible mantenerlo al día con las expresiones que van apareciendo.

2.7.1. Motores de Búsqueda

Los Motores de Búsqueda o buscadores son la manera más usada de encontrar la información en la Web. Un buscador permite introducir algunas palabras en un cuadro de texto y proporciona como resultado un conjunto de páginas Web.

2. 7. 1. 1. Búsqueda Sintáctica o Terminológica

La Búsqueda Terminológica es un procedimiento de búsqueda en el cual el resultado de la búsqueda es el conjunto de documentos que contienen los términos introducidos en la consulta. La mayoría de los motores de búsqueda existentes realizan búsquedas sintácticas, incluido Google[1].

Con la búsqueda sintáctica se pueden producir tres situaciones que podrían llevar a errores:

- Polisemia: Se efectúa una consulta en el buscador, introduciendo un término, y se encuentran páginas Web que lo contienen, pero con un significado distinto del de interés.
- Sinonimia: Se realiza una consulta en el buscador, introduciendo un término, y no se devuelven resultados relacionados puesto que las páginas correspondientes contienen un sinónimo del término (pero no el propio término).
- Multilingüismo: Se efectúa una consulta en el buscador, introduciendo términos en un idioma, y no se despliegan ciertas páginas Web relacionadas por estar escritas en otra lengua.

[1] http://www.google.com

En todos estos casos, el problema es que la consulta no identifica en forma precisa el concepto a buscar, sino que solamente identifica un término que en cierto idioma tiene entre sus significados aquello que se está buscando.

2. 7. 1. 2. Búsqueda Semántica

Un motor de búsqueda semántica puede verse como una herramienta que recibe consultas basadas en ontologías, las ejecuta contra una base de conocimiento y devuelve los documentos que satisfacen la consulta. Los conceptos e instancias de la KB se asocian a los documentos mediante las anotaciones.

Mejoras con respecto a la búsqueda sintáctica:

- Mayor recuperación mediante la utilización de jerarquías de clases y reglas.
- A pesar de la separación del espacio de contenidos (documentos) y el espacio de conceptos, es posible combinar condiciones sobre conceptos y contenidos.
- Las mejoras con respecto a la búsqueda por palabra clave crecen con el número de condiciones.
- El grado de mejora depende de la completitud y calidad de la ontología, la KB, y el mapeo de conceptos a palabras.

2.7.2. Principales medidas de evaluación

Los resultados de una búsqueda se pueden valorar cuantitativamente, utilizando indicadores básicos, con el objetivo de que la recuperación llevada a cabo sea de calidad.

Se entiende por Relevancia a la utilidad, o potencial uso de los documentos recuperados. Es la medida de cómo una pregunta se ajusta a un documento [75].

Para calcular la Relevancia, lo más habitual es establecer valores binarios: si un documento es relevante (sirve como respuesta a nuestra pregunta) se valora con 1, si no sirve se valora con 0. También se puede fijar una graduación, y establecer una escala ordinal para medir la relevancia de los documentos.

Una vez definido el concepto de Relevancia, se pueden establecer una serie de medidas que sirven para evaluar los sistemas de recuperación.

Los indicadores más utilizados en la evaluación de sistemas de recuperación son *Recall* y *Precision*. Éstos permiten establecer la fiabilidad y el nivel de éxito de los procesos de recuperación de información.

El conjunto de documentos recuperados se divide en dos grupos: los documentos relevantes (recuperados correctamente) y los no relevantes (recuperados erróneamente) que provocan "ruido" en la salida. Los documentos no recuperados se dividen en los relevantes, rechazados por el sistema de manera errónea, y los no relevantes, rechazados de manera correcta. En la Figura 7 se visualizan los grupos de documentos sobre la totalidad de la colección.

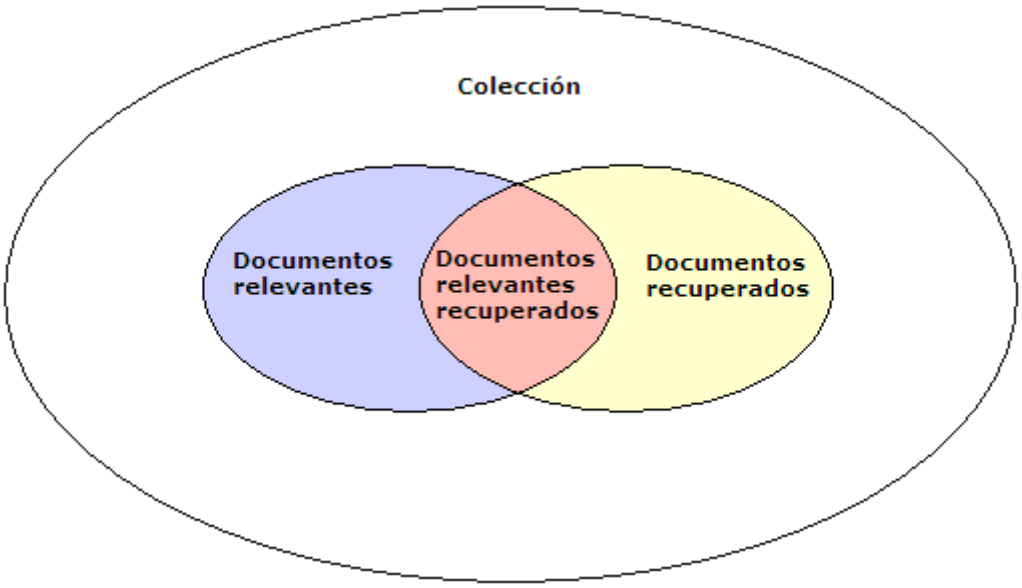


Figura 7: Conjunto de documentos en la recuperación de información

Para obtener los indicadores de *Recall* y *Precision* se combinan el número de documentos relevantes y no relevantes, y recuperados y no recuperados.

Recall (o Exhaustividad): es la cantidad de documentos relevantes recuperados sobre el total de documentos relevantes. En otras palabras, es la proporción de material relevante recuperado, del total de los documentos que son relevantes en la colección [74].

$$Recall = \frac{Nº\ de\ documentos\ relevantes\ recuperados}{Nº\ de\ documentos\ relevantes\ en\ la\ colección}$$

Este indicador mide cuán bien el buscador cubre el espacio de documentos relevantes; es una medida de rendimiento.

El valor de este indicador se encuentra entre 0 y 1. Si el indicador vale 1, se tiene *Recall* máximo, es decir que se ha encontrado todo lo relevante que había en la colección de documentos y por lo tanto no hay ruido.

Cualquier sistema de recuperación de información podría conseguir un *Recall* de 1, simplemente devolviendo todos los documentos de la colección. Por ello se utiliza también la métrica de *Precision.*

Precision: es la cantidad de documentos relevantes recuperados sobre el total de documentos obtenidos. En otras palabras, es la proporción de material recuperado realmente relevante, del total de los documentos recuperados [74].

$$Precisión = \frac{Nº\ de\ documentos\ relevantes\ recuperados}{Nº\ total\ de\ documentos\ recuperados}$$

Este indicador mide qué tan ajustados son los resultados obtenidos con respecto a la información requerida por el usuario que generó la consulta.

El valor de este indicador se encuentra entre 0 y 1. La recuperación perfecta es en la que únicamente se recuperan los documentos relevantes y por lo tanto tiene un

valor de 1. Cuanto más se acerca a 0, mayor es el número de documentos recuperados no relevantes y por lo tanto el ruido es mayor.

Para un mismo sistema y una misma consulta, *Recall* y *Precision* varían inversamente. Cuando se trata de aumentar *Recall*, disminuye *Precision*, y viceversa.

Un sistema de recuperación de información con *Recall* muy alto pero con baja *Precision*, o viceversa, no es adecuado. Un buen sistema de recuperación de información debe intentar maximizar la cantidad de documentos relevantes recuperados y minimizar la cantidad de documentos no relevantes recuperados.

3 Estado del Arte

El presente trabajo ha requerido un esfuerzo importante desde el punto de vista de la investigación, la cual tuvo lugar fundamentalmente en la primera etapa del mismo, y en menor medida en las etapas siguientes.

Para trabajar en la dirección de la recuperación de información en la Web Semántica, fue necesario reunir información referente a la misma, especialmente en lo que se considera las bases para su construcción.

Dado que se pretende dotar a la Web de significado, fue necesario estudiar modelos que permitan representar los distintos conceptos, sus relaciones y atributos; la tendencia actual es utilizar ontologías como esquema de representación de la realidad.

La construcción de ontologías involucra el estudio de los distintos lenguajes en los que se pueden expresar, las metodologías de construcción y los razonadores que se utilizan para verificar su consistencia.

También se observó la necesidad de generar una metadata relacionada con la ontología. La anotación semántica es un proceso complejo, que acarrea muchas dificultades, especialmente cuando se trabaja a mayor escala.

El hecho de que el proyecto incluya áreas de conocimiento relativamente nuevas, tiene como contra cara que las mismas aún no están maduras, y hace que sea difícil muchas veces, encontrar herramientas que hagan las ideas practicables.

Debido al interés que la Web Semántica ha despertado, es muy frecuente ver nuevo material disponible en Internet. Se debió realizar una selección del material a incluir en el estudio, debido al tiempo acotado del que se dispone para la realización de este trabajo.

El documento Apéndice A - Estudio del Estado del Arte es el resultado del trabajo de investigación realizado, y del cual se extrae un resumen que se presenta en este capítulo.

3.1. Lenguajes para representación de Ontologías

Para poder modelar una realidad mediante una ontología, es necesario conocer los lenguajes y herramientas que se dispone para su representación.

En el relativamente corto tiempo transcurrido desde la aparición de las ontologías, han surgido una gran cantidad de lenguajes, cada uno con sus características y poder expresivo. A continuación se mencionan los más representativos.

3.1.3. Primeros Lenguajes

Los lenguajes basados en redes semánticas, marcos y lógica de primer orden que surgen en la década del 90 pueden ser considerados los precursores de los lenguajes para la construcción de ontologías actuales.

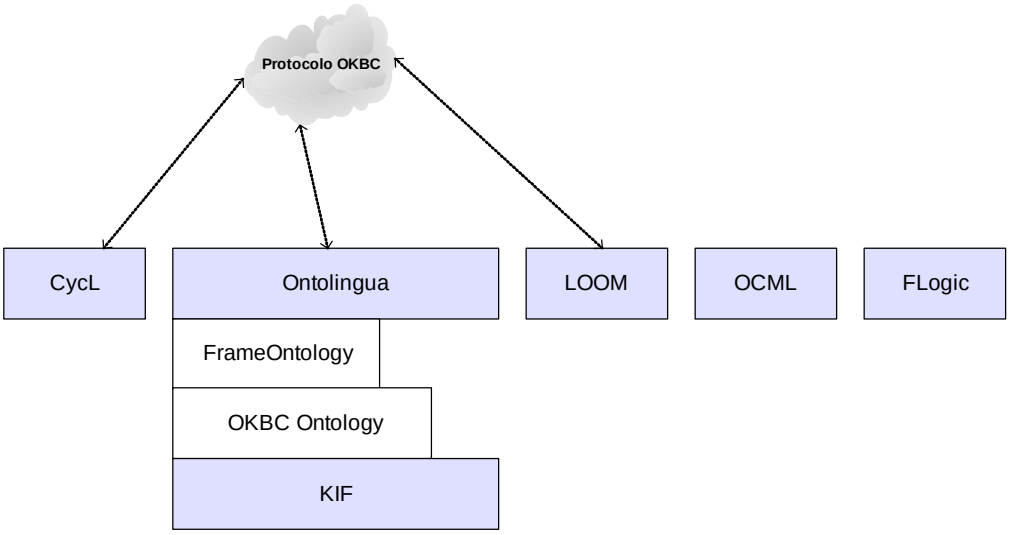


Figura 8: Primeros lenguajes de ontologías

3. 1. 3. 1. CycL

El primer lenguaje en ser creado fue CycL. Está basado en marcos y lógica de primer orden, y fue utilizado para construir la ontología de propósito general CYC. El lenguaje consta de un vocabulario de aproximadamente 160 términos.

El conjunto de términos puede ser clasificado en constantes, términos no atómicos, variables y otro tipo de objetos. Los términos se combinan en expresiones, las cuales son utilizadas para hacer aseveraciones en la base de conocimiento CYC [40].

3. 1. 3. 2. KIF

KIF (Knowledge Interchange Format) está basado en lógica de primer orden y marcos. Es de bajo nivel pero de mayor poder expresivo que algunos de sus sucesores, como OKBC y Ontolingua.

KIF provee definiciones para objetos, funciones, relaciones y constantes lógicas. Es un lenguaje para intercambio de conocimiento, fue diseñado para permitir traducciones entre lenguajes de representación más especializados. Por esta razón es tedioso utilizar esta herramienta para el desarrollo de ontologías [45].

3. 1. 3. 3. LOOM

LOOM es un lenguaje de alto nivel proveniente de la familia de lenguajes KL-ONE, la cual se caracteriza por sus clasificadores automáticos eficientes. Está basado en lógica descriptiva y provee clasificación automática de conceptos. Alcanza una fuerte integración entre los paradigmas basados en marcos y en reglas.

LOOM fue creado para construir bases de conocimiento en general, no específicamente ontologías. Permite representar conceptos, taxonomías de conceptos, relaciones n-arias, funciones, axiomas y reglas de producción.

Este lenguaje fue desarrollado simultáneamente con Ontolingua en el Instituto de Ciencia de la Información (ISI por sus siglas en inglés) en la Universidad de California del Sur [46].

3. 1. 3. 4. Ontolingua

Ontolingua [47] es un lenguaje que permite construir, publicar y compartir ontologías, con una clara semántica basada en KIF. Las ontologías pueden

traducirse automáticamente a distintos lenguajes, como KIF, LOOM, Prolog, y CLIPS. Tiene el inconveniente que provee un gran poder expresivo sin ningún medio para controlarlo.

Todos estos lenguajes guardan relación con OKBC (Protocolo para la Conectividad de Bases de Conocimiento Abierto). Este protocolo permite acceder bases de conocimiento almacenadas en diferentes sistemas de representación y que pueden estar basadas en diferentes paradigmas de representación de conocimiento.

3. 1. 3. 5. OCML

OCML (Operational Conceptual Modelling Language) [48] es un lenguaje de modelado que soporta la construcción de modelos de conocimiento por medio de diferentes tipos de estructuras.

Permite la especificación y la utilización de funciones, relaciones, clases, instancias y reglas. También incluye mecanismos para definir ontologías y métodos que resuelven los problemas.

Así, mientras es posible especificar y prototipar modelos de conocimiento en OCML, el lenguaje no apunta a dar soporte para lograr aplicaciones eficientes.

3.1.4. Lenguajes de ontologías basados en Web

La utilización masiva de Internet originó una gran cantidad de lenguajes para explotar las características de la Web. En la Figura 9 se muestra la relación que existe con los lenguajes de marcado. Se puede observar que todos ellos están basados en la capacidad sintáctica de XML.

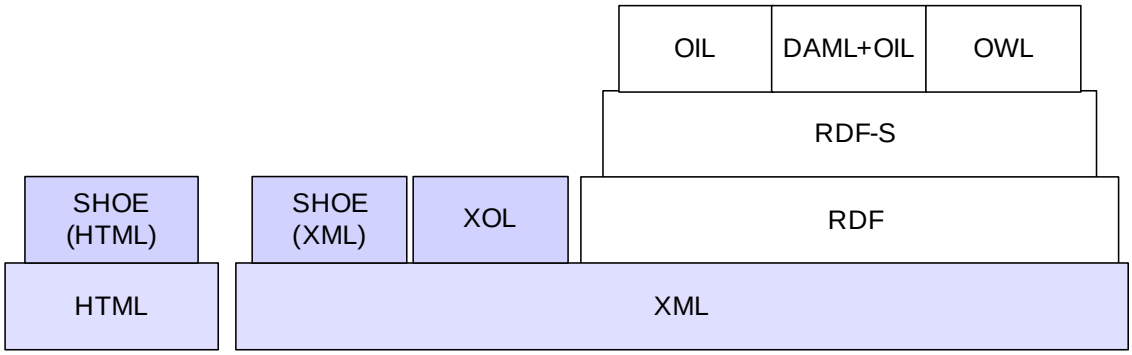


Figura 9: Lenguajes de ontologías basados en lenguajes de marcado

3. 1. 4. 1. SHOE

SHOE [49] es un lenguaje de representación basado en HTML. Es un super conjunto de HTML el cual agrega los tags necesarios para embeber datos semánticos en las páginas Web. Se basa en el modelo de conocimiento de marcos.

Los tags de SHOE están divididos en dos categorías. La primera comprende tags para construir ontologías. La segunda incluye marcas para anotar documentos Web con respecto a una o más ontologías, declarar entidades y hacer inferencias respecto a las mismas bajo las reglas definidas en las ontologías.

3. 1. 4. 2. XML

XML (eXtensible Markup Language) [50] es un lenguaje de marcado para documentos estructurados: documentos con ciertos contenidos (palabras, imágenes, etc.) e indicaciones sobre el rol que cumplen dichos contenidos. Un lenguaje de marcas es un mecanismo para identificar estructuras dentro de un documento. La especificación XML define un estándar sobre cómo agregar marcas a

un documento; por lo que más exactamente debe definirse XML como un metalenguaje que provee una base sintáctica sobre la cual pueden definirse lenguajes de marcas a ser utilizados.

XML representa una primera aproximación a la Web semántica, y aunque no está expresamente pensado para definir ontologías, es un estándar muy extendido. Es un paso hacia una representación explícita de la estructura y los datos de los contenidos de la Web, separada de su presentación en HTML. Permite estructurar datos y documentos en forma de árboles de etiquetas con atributos, aunque no distingue entre objetos y relaciones, ni tiene noción de jerarquía de clases.

3. 1. 4. 3. XMLS

XML Schema (XMLS) [51] fue propuesto originalmente por Microsoft y se transformó en recomendación oficial de W3C en mayo de 2001. Su propósito es describir la estructura de un documento XML, definiendo los bloques que lo pueden constituir.

Un esquema de XML es un documento donde se declara cuál será el contenido válido de un documento más allá de ser correcto sintácticamente. Un documento XML estará "bien-formado" si cumple con la especificación de XML, pero sólo será válido si cumple con su esquema. Con XMLS se puede acordar de antemano las estructuras que se van a utilizar, así como manejar tipos de datos primitivos y derivados.

Un XML Schema define:

- los elementos y los atributos que pueden aparecer en un documento
- qué elementos son hijos, su orden relativo y la cantidad
- si un elemento está vacío o puede incluir texto
- tipos de datos y valores por defecto y fijos para los elementos y atributos

3. 1. 4. 4. RDF

RDF (Resource Description Framework) [52] es un lenguaje para la definición de ontologías y metadatos en la Web. El elemento de construcción básica en RDF es la tripla o sentencia, que consiste en dos nodos (sujeto y objeto) unidos por un arco (predicado), donde los nodos representan recursos, y los arcos propiedades. Por ejemplo una sentencia podría expresar el hecho de que el autor (predicado) del cuadro "Starry Night" (sujeto) fue el pintor Vincent van Gogh (objeto), como se ilustra en la Figura 10.

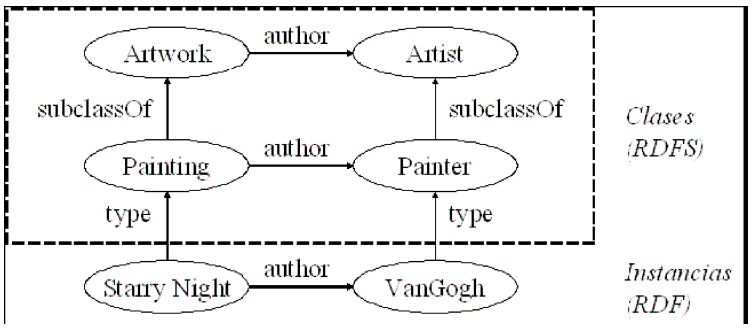


Figura 10: Sujeto – Predicado – Objeto, en RDF

3. 1. 4. 5. RDF Schema

Con RDF Schema (RDFS) [53] se puede definir jerarquías de clases de recursos, especificando las propiedades y relaciones que se admiten entre ellas (ver Figura 10). En RDFS las clases, relaciones, y las propias sentencias son también recursos, y por lo tanto se pueden examinar y recorrer como parte del grafo, o incluso asertar sentencias sobre ellas.

Se ha definido diferentes formas sintácticas para la formulación escrita de RDF, pero quizás la más extendida es la basada en XML. Es por ello que RDFS se presenta a menudo como una extensión de XML. La Figura 11 muestra un ejemplo simplificado de esta sintaxis.


```
<Painter about="vangogh" name="Vincent van Gogh"
         birth="1853" death="1890" nationality="Dutch">
   ...
</Painter>

<Painting about="starrynight" ...>
   <author resource="vangogh">
   ...
</Painting>

<Class about="Painter">
   <subclassOf resource="Artist"/>
</Class>

<Class about="Painting">
   <subclassOf resource="Artwork"/>
</Class>

<Class about="Artist"/>

<Class about="Artwork/>

<Property about="author">
      <domain resource="Artwork"/>
      <range resource="Artist"/>
</Property>
```

Figura 11: RDF-S

3. 1. 4. 6. OIL

Ontology Inference Layer (OIL) [54] sintetiza el trabajo de tres comunidades diferentes para proveer un lenguaje de propósito general basado en etiquetas. Usa sistemas basados en frames, lógica descriptiva y estándares Web (XML y RDF) [17].

Por parte de los lenguajes basados en frames, tiene como primitivas centrales de modelación, clases con propiedades. De la lógica descriptiva, hereda la semántica formal y el soporte eficiente para razonamiento. Además, tiene una sintaxis bien definida en XML y es una extensión de RDF y RDFS.

3. 1. 4. 7. DAML

DAML [55] hereda muchos aspectos de OIL y las capacidades de ambos lenguajes son relativamente similares. Al igual que OIL las características principales son:

- Soportar jerarquía de clases y propiedades
- Permitir la construcción de clases a partir de otras usando combinaciones de intersección, unión, y complemento
- Permitir restricciones de dominio, rango y cardinalidad
- Soportar propiedades transitivas e inversas

3. 1. 4. 8. DAML + OIL

DAML+OIL [56] es un lenguaje que permite expresar los recursos Web con más detalle y claridad que sus predecesores. Extiende los lenguajes RDF y RDFS con primitivas de modelado más ricas.

Se caracteriza por: una semántica bien definida, flexibilidad para formar nuevas expresiones a partir de clases y propiedades y una arquitectura por capas.

3. 1. 4. 9. OWL

OWL (Web Ontology Language) [57] es un lenguaje desarrollado y recomendado por W3C para publicar y compartir ontologías en la Web. Se deriva de DAML+OIL y, al igual que éste, es una extensión de RDFS.

Las declaraciones en OWL definen clases, propiedades e instancias, junto con propiedades ya definidas en RDFS (subclase, subpropiedad, dominio y rango). Además se añaden otras como la relación inversa entre propiedades, y un conjunto de axiomas adicionales para definir restricciones tales como cardinalidad de propiedades y combinaciones de clases (intersecciones, uniones y complemento).

OWL está dividido en tres niveles:

- **OWL Lite**:

Proporciona un subconjunto de OWL sencillo y útil desde el punto de vista de los desarrolladores de herramientas. Se pueden realizar clasificaciones jerárquicas y aplicar restricciones simples.

- **OWL DL (Description Logic)**

Permite dar una mayor expresividad a los elementos de una ontología e incluye todas las construcciones del lenguaje OWL. Limita la expresividad intentando conseguir decidibilidad, es decir, que todas las conclusiones sean realizables en un tiempo finito.

- **OWL Full**

Permite dar una mayor expresividad a las sentencias y da libertad de usar la sintaxis en RDF. Tiene la máxima expresividad pero sin garantía de computabilidad. No se garantiza la eficiencia ni la decidibilidad.

3.1.5. Representaciones del modelo RDF

El modelo RDF (y sus extensiones) tienen dos formatos de representación: la serialización RDF/XML y N-TRIPLES, este último desarrollado por el consorcio W3C. Las principales características y diferencias entre los mismos se mostrarán a través de un ejemplo.

Sea una entidad identificada por la URI *http://choikebilingue/#Africa_del_Este.*

Dicha entidad tiene dos propiedades: término principal en español e inglés, denotados por prolexemeRegionEsp y prolexemeRegionIng y cuyos valores son "Africa del Este" y "East Africa" respectivamente. La Figura 12 muestra dicho modelo RDF en la representación de grafo.

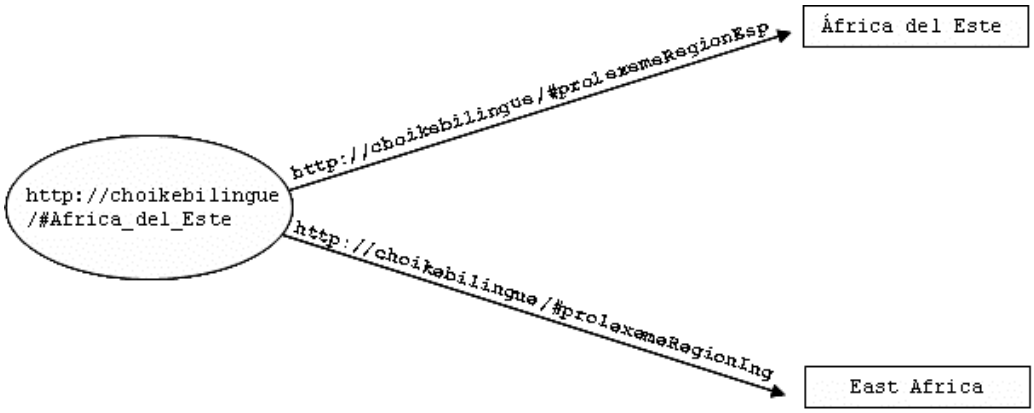


Figura 12: Representación de grafo del modelo RDF

Un grafo RDF puede expresarse mediante la serialización RDF/XML, que es una especificación XML. En la Figura 13 se observa cómo se define la entidad a describir y sus propiedades.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:ck="http:// http://choikebilingue/">
   <rdf:Description rdf:about=”http://choikebilingue/#Africa_del_Este”>
      <ck:prolexemeRegionEsp>África del Este</ck:prolexemeRegionEsp>
      <ck:prolexemeRegionIng>East Africa</ck:prolexemeRegionIng>
   </rdf:Description>
</rdf:RDF>
```

Figura 13: Serialización RDF/XML del grafo de la Figura 12

Una manera equivalente de representar la misma realidad es a través de N-TRIPLES (Figura 14).

```
<http://choikebilingue#Africa_del_Este>
<http://choikebilingue#prolexemeRegionEsp> África del
Este.
<http://choikebilingue#Africa_del_Este>
<http://choikebilingue#prolexemeRegionIng> East Africa.
```

Figura 14: Representación N-TRIPLES del grafo de la Figura 12

Este formato no permite abreviaciones, ni siquiera en las URIs. Cada tripla se escribe en una línea separada y debe finalizar con un punto. Consiste de un sujeto, un predicado y un objeto, que se corresponden con los términos izquierdo, medio y derecho de la tripla respectivamente.

Todos los sujetos y predicados son URIs, pero los objetos pueden ser literales (como en el ejemplo precedente) o URIs.

3.2. Metodologías

Como se expuso en el Capítulo 2, el desarrollo de ontologías es un proceso que requiere de una metodología, entendida como un conjunto de métodos y técnicas que promueven un enfoque para su construcción. A continuación se mencionan algunas de las más conocidas.

3.2.3. OnToKnowledge

OnToKnowledge [27] es un proyecto europeo en el que se construye un ambiente de herramientas basado en ontologías. Los resultados del mismo son un conjunto de herramientas para procesamiento semántico y acceso de los usuarios, OIL y validación vía casos de estudio.

3.2.4. Methontology

Se basa en las actividades identificadas por el estándar IEEE para el desarrollo de software. Estas actividades están registradas en el ciclo de vida de la ontología, que establece los pasos a través de los cuales la ontología se mueve durante el mismo y las actividades a realizar en cada nivel [18].

3.2.5. TOVE (TOronto Virtual Enterprise)

Esta metodología está compuesta por los siguientes pasos [19]:

1. Definir un conjunto de Escenarios de Motivación
2. Definir un conjunto de Preguntas de Competencia Informales que la ontología deberá resolver para dar soporte a los escenarios de motivación.
3. Usando Lógica de Primer Orden, definir la Terminología.
4. Redefinir formalmente las Preguntas de Competencia usando la terminología y lógica de primer orden.
5. Definir la semántica y restricciones sobre la terminología usando lógica de primer orden.

3.2.6. Enterprise Ontology

Esta metodología fue desarrollada como un componente de Enterprise Project [59], un esfuerzo conjunto originado en la Universidad de Edinburgh para desarrollar un marco de modelado de corporaciones. Enterprise Ontology [58] fue construida para servir de base de este marco, el cual incluye además un conjunto de herramientas y métodos [60].

Se compone de las siguientes actividades:

1. Identificar una propuesta de ontología: dominio, alcance, clases, taxonomía, atributos, relaciones, instancias, axiomas.
2. Construir una ontología capturando, codificando e integrando conocimiento apropiado a partir de las ontologías existentes.
3. Evaluar y validar la ontología.
4. Elaborar la documentación.

3.3. Herramientas para desarrollo de Ontologías

A medida que se profundiza en el estudio del dominio de un problema, las ontologías que se pueden construir para su modelado son más voluminosas y complejas. En este punto entran a jugar un papel importante las herramientas, que simplifican el trabajo cuando el tamaño y la complejidad se vuelven difíciles de manejar.

En otros casos, puede ser necesario convertir una ontología de un lenguaje a otro, o entre distintos formatos de un mismo lenguaje. Contar con herramientas que automaticen el proceso disminuye la posibilidad de cometer errores y contribuye a enfocar la energía en las tareas que no pueden ser realizadas automáticamente.

Varias herramientas han aparecido en la última década. A continuación se comentan Ontolingua, WebODE, Protégé, OilEd y Jena.

3.3.3. Ontolingua

Proporciona una sintaxis basada en marcos y traduce toda la información en KIF, que es una forma de codificación de la información en Lógica de Primer Orden [20]. No existe restricción de expresividad. Ontolingua [47] provee un ambiente distribuido de colaboración para buscar, crear, editar, modificar y utilizar ontologías.

3.3.4. WebODE

WebODE se ha desarrollado como un marco escalable para dar soporte a las siguientes actividades:

1. desarrollo, administración y población de ontologías
2. servicios de middleware que permitan el fácil uso e integración de la tecnología de ontologías en los servicios de información
3. suites de desarrollo de aplicaciones basadas en ontologías que faciliten la creación de aplicaciones basadas en ontologías

Ha sido creado para proveer soporte a la metodología de construcción Methontology, aunque puede ser usado en el contexto de otras metodologías o con ningún enfoque metodológico [21].

3.3.5. OilEd

OilEd [61] es un editor de ontologías desarrollado en la Universidad de Manchester que permite al usuario construir las mismas expresándolas en DAML+OIL.

Es un editor simple que no soporta la creación de grandes ontologías ni versionado, y las versiones actuales no proveen un ambiente de desarrollo completo. Es como el "NotePad" de los editores de ontologías, ofrece la mínima funcionalidad como para construirlas.

3.3.6. Protégé

Provee un ambiente integrado para la mayoría de las actividades del ciclo de vida de las ontologías. Es una plataforma gratuita que ofrece un conjunto de herramientas para construir modelos de dominio y ontologías. Puede ser extendida para construir nuevas herramientas y aplicaciones.

Protégé [22] implementa varias estructuras y acciones para modelado del conocimiento que soportan la creación, visualización y manipulación de ontologías en varios formatos de representación.

3.3.7. Jena

Jena [23] es un marco Java para construir aplicaciones de la Web Semántica. Provee un ambiente de programación para RDF, RDF-S, OWL e incluye un motor de inferencia basado en reglas.

Incluye:

- una API RDF
- capacidades de lectura y escritura en lenguaje RDF expresado en los formatos RDF/XML y N-TRIPLES
- una API OWL
- facilidades de consulta y almacenamiento

3.4. Razonadores

Como se expuso en la Sección 3.3, las ontologías pueden tornarse muy complejas, y por tanto es más fácil cometer errores lógicos que la transformen en inconsistente. Los razonadores son herramientas útiles que permiten verificar consistencia, satisfactibilidad de los conceptos y determinar la clase más específica de un individuo, entre otros.

En los siguientes puntos se enumeran y describen brevemente los razonadores más utilizados.

3.4.1. Racer

RACER (Renamed ABox and Concept Expression Reasoner) [62] es un razonador para Lógica Descriptiva que da soporte a OWL Lite y OWL DL. Las características más relevantes son:

- Verificación de relaciones de satisfactibilidad y jerarquía entre conceptos.
- Verificación de consistencia.
- Recuperación de instancias a partir de conceptos especificados o de consultas. Asimismo recupera tuplas de individuos que satisfagan un conjunto de condiciones.
- Determinación del concepto más específico al que pertenece un individuo dado.

3.4.2. Pellet

Pellet [63] es un razonador de código abierto para Lógica Descriptiva, desarrollado sobre la plataforma Java en el Massachussets Institute of Technology (MIT).

Ofrece funcionalidades de análisis y reparación de ontologías. A partir de un conjunto de heurísticas puede detectar y reparar documentos transformándolos de formato OWL Full a OWL DL. Asimismo, es capaz de verificar la consistencia de ontologías y da soporte a consultas formuladas en SPARQL [29] y RDQL [20].

3.4.3. FaCT++

FaCT++ [64] es un razonador de código abierto para Lógica Descriptiva. Fue implementado tomando como base su antecesor FaCT e introduciendo optimizaciones al mismo. Para dotarlo de mayor eficiencia y portabilidad se utilizó C++ para su desarrollo.

Permite verificación de satisfactibilidad, de relaciones de jerarquía entre conceptos y de consistencia de una ontología.

3.4.4. KAON2

KAON2 [28] es un razonador basado en Java disponible gratuitamente para propósitos académicos y de investigación. Mientras que su predecesor KAON utilizaba una extensión propietaria de RDFS, KAON2 está basado en OWL-DL.

Provee facilidades para ingreso de consultas basadas en SPARQL. También implementa un módulo para recuperar instancias de ontologías desde bases de datos relacionales.

3.5. Herramientas para manejo de XML

La población de la ontología es la actividad a través de la cual se incorporan individuos a las clases y relaciones. Dicha carga podría efectuarse manualmente usando el editor o mediante lectura de archivos de entrada, siendo la última la opción más adecuada por trabajar con grandes cantidades de datos.

XML es un lenguaje apropiado para representar mediante diccionarios a las instancias con sus propiedades y relaciones. Esta sección describe las herramientas investigadas para el manejo de XML a través de Java.

3.5.1. DOM

DOM [30] (Modelo de Objetos del Documento) es una interfaz de programación de aplicaciones para documentos válidos HTML y bien construidos XML. Define la estructura lógica de los documentos y el modo en que se accede y manipula. En la especificación DOM, el término "documento" es utilizado en un sentido amplio. Con DOM se pueden construir documentos, navegar por su estructura, y añadir, modificar, o eliminar elementos y contenido.

Como una especificación de W3C, un objetivo importante para DOM es proporcionar una API que pueda ser utilizada en una amplia variedad de entornos y aplicaciones y en cualquier lenguaje de programación.

3.5.2. SAX

SAX [31] (Simple API for XML) es una especificación ampliamente utilizada, que describe cómo los parsers XML pueden pasar información eficientemente de los documentos XML a las aplicaciones. SAX fue implementado originalmente en Java, pero actualmente es soportado por la mayoría de los lenguajes de programación.

SAX es una interfaz de streaming, las aplicaciones reciben información de documentos XML en un flujo continuo, sobre el cual no pueden efectuar ningún tipo de navegación. Este enfoque hace a SAX extremadamente eficiente, manejando documentos XML de casi cualquier extensión en tiempo razonable, pero también demanda mayor esfuerzo de programación. Las interfases con enfoque basado en árboles, como DOM, emplean el enfoque opuesto: menor esfuerzo de desarrollo a costa de mayores recursos computacionales. [24]

3.5.3. JAXP

JAXP [33] (Java API for XML Processing) es una API para procesar datos XML en aplicaciones escritas en Java. Permite efectuar de manera sencilla la validación, parsing, y transformación de documentos XML a través de diferentes funciones.

Soporta los estándares SAX y DOM, y es independiente de las implementaciones de los procesadores XML, permitiendo hacer parsing como un flujo de eventos o construyendo la representación de objetos de ellos.

3.5.4. JDOM

JDOM [32] es una representación Java de un documento XML. Provee una manera de representar los documentos facilitando su lectura, manipulación y escritura. Tiene una API liviana y directa, optimizada para la programación en Java. Es una alternativa para DOM y SAX, aunque se integra bien con ambos.

Se puede construir documentos JDOM a partir de archivos XML, árboles DOM, eventos SAX o cualquier otra fuente. Los documentos JDOM también pueden ser convertidos a los formatos antedichos.

3.5.5. dom4j

dom4j [34] es un framework XML de código abierto para Java. Permite leer, escribir, navegar, crear y modificar documentos XML y está integrado con DOM, SAX y JAXP.

dom4j es una API más simple y liviana que DOM, y hace uso extensivo de APIs estándar de Java tales como las colecciones de Java 2. dom4j soporta completamente el estándar DOM y utiliza JAXP para configurar el parser SAX usar.

Es posible crear documentos dom4j a partir de texto XML, eventos SAX o árboles DOM existentes, y escribir documentos dom4j en los mismos formatos.

dom4j y JDOM son proyectos y APIs diferentes, aunque con objetivos similares. Ambos intentan facilitar la tarea de utilizar XML sobre una plataforma Java. Difieren en su diseño, API e implementación. dom4j está basado en interfases Java y hace uso extensivo del polimorfismo. JDOM y dom4j pueden usar JAXP internamente para hacer el parsing SAX.

3.6. Sistemas de Anotación

La anotación semántica de un documento consiste en relacionar su contenido entero o una parte de él (una palabra o una frase) con un cierto identificador. Tal identificador (generalmente una URI) determina inequívocamente el concepto que es mencionado en el contenido del documento.

Las anotaciones son la base del procesamiento automático de las páginas Web, y por lo tanto la anotación semántica es de fundamental importancia.

La generación de anotaciones semánticas se puede efectuar manual, semiautomática o automáticamente.

3.6.1. Sistemas Manuales

A continuación se enumeran algunos anotadores semánticos manuales disponibles y sus principales características.

3. 6. 1. 1. Annotea

Annotea [65] es un proyecto para mejorar el ambiente colaborativo de la W3C a través del uso de anotaciones compartidas. El usuario recibe el documento y las notas adjuntas de uno o varios servidores y de esta forma compartir ideas.

Annotea es un proyecto abierto, en la medida en que usa y ayuda a mejorar los estándares de la W3C. Con este objetivo en mente, se usa un esquema RDF para describir las anotaciones como metadatos y el protocolo XPointer para indicar las posiciones de las anotaciones en el documento.

Annotea es un proyecto LEAD (*Live Early Adoption and Demonstration*, demostración y adopción práctica temprana), que es una filosofía de la W3C cuyo objetivo es estudiar áreas donde pequeñas piezas deben encajar para hacer funcionar un sistema, pero no se puede tener certeza de qué problemas se encontrarán hasta que se prueba [66].

3. 6. 1. 2. Annozilla

Annozilla [67] está diseñado para ver y crear anotaciones asociadas a una página Web, siguiendo los lineamientos del proyecto Annotea. Este anotador opera almacenando anotaciones en formato RDF en un servidor, utilizando XPointer para identificar la región del documento a anotar.

Annozilla deriva su nombre el navegador Mozilla, del cual utiliza sus características nativas para manipular la metadata, en particular el manejo de RDF para efectuar el parsing.

3. 6. 1. 3. OntoMat-Annotizer

OntoMat-Annotizer [68] es una herramienta de anotación de interactiva amigable con el usuario. Permite efectuar un marcado semántico basado en ontologías y OWL. Incluye un navegador de ontologías para su manipulación y un navegador Web para visualizar las porciones de texto anotadas.

Fue desarrollado para que los usuarios finales pudieran enriquecer sus páginas con metadata; en vez de anotar las páginas Web con un editor de texto común, OntoMat permite resaltar las porciones de texto deseadas a través de interacciones de tipo drag and drop.

3. 6. 1. 4. SemanticWord

SemanticWord [35] es un ambiente basado en Microsoft Word para marcado de contenidos. Extiende la interfaz gráfica del mismo con elementos que permiten asociar marcas OWL con regiones de texto en los documentos. Las marcas están

incorporadas en los documentos y pueden ser extraídas automáticamente a archivos.

Esta herramienta permite la generación de páginas y su marcado simultáneo, provee una biblioteca de plantillas con texto parcialmente anotado y un sistema de extracción de información automático.

3. 6. 1. 5. SMORE

SMORE [36] es un anotador semántico creado por el Grupo MINDSWAP de la Universidad de Maryland. Esta herramienta ofrece a los usuarios la posibilidad de marcar documentos HTML utilizando OWL y ontologías.

Permite la utilización de clases, propiedades e instancias de ontologías preexistentes y provee algunas funcionalidades de edición y creación de ontologías desde cero a partir de términos presentes en documentos Web. Asimismo, es posible crear páginas Web simultáneamente con el marcado de las mismas.

3. 6. 1. 6. Yawas

Yawas [37] (Yet Another Web Annotation System) es una herramienta que permite realizar anotaciones y personalizar los documentos a medida que analiza la información en páginas Web. Es una herramienta de código abierto, muy simple y liviana, que permite resaltar las porciones de texto deseadas.

Codifica las anotaciones en URLs extendidas y utiliza archivos locales para almacenar, recuperar y compartir con otros usuarios las anotaciones. Además proporciona el buscador YawasQuickSearch, que permite buscar en los documentos los términos previamente anotados.

3.6.2. Sistemas Semiautomáticos

Los agentes Web pueden ser diseñados para tratar la información existente en las páginas Web de manera semiautomática. Se trata de convertir la información en conocimiento, referenciando datos dentro de las páginas Web a metadatos con un esquema común consensuado sobre algún dominio.

Estos sistemas explotan técnicas de Procesamiento de Lenguaje Natural (PLN) para extraer las referencias en el texto a ciertos conceptos descritos en ontologías. Estos sistemas requieren generalmente como entrada patrones o corpus de documentos utilizados para entrenar el sistema.

A continuación se comentan algunos de los anotadores semiautomáticos existentes.

3. 6. 2. 1. Melita

Melita [70] es una interfaz de anotación basada en ontologías creada en la Universidad de Sheffield, Reino Unido. Su desarrollo no se orientó a la construcción de una herramienta completa sino únicamente un prototipo de factibilidad.

El proceso de anotación se efectúa mediante la definición de un escenario, un conjunto de tags para anotación organizados en una ontología y un corpus a anotar. Las anotaciones son insertadas seleccionando el tag de la ontología e identificando la porción de texto a anotar en el documento. Mientras el usuario anota el texto, Amilcare (una herramienta de Extracción de Información Adaptativa) se ejecuta en segundo plano aprendiendo cómo reproducir la anotación insertada.

Melita no maneja ontologías OWL, por lo que habría que construir la ontología desde cero utilizando su lenguaje propietario

3. 6. 2. 2. MnM

MnM [38] es una herramienta de anotación que provee soporte automático y semiautomático para anotar páginas Web con contenido semántico.

Integra un navegador Web, un editor de ontologías, medios para efectuar enlaces con servidores de ontologías y vínculos con herramientas de extracción de información.

Permite cargar ontologías preexistentes, crear o cargar bases de conocimiento y poblar las mismas de forma manual o semiautomática mediante entrenamiento con un corpus de prueba. No provee soporte para OWL, solamente para DAML+OIL, WebOnto y RDF.

3. 6. 2. 3. OntoGloss

OntoGloss [71] es una herramienta de anotación basada en ontologías. Permite la anotación semiautomática de nuevos documentos en base a otros ya anotados mediante una interfaz amigable.

Mantiene las anotaciones separadas del documento, en una base de datos, las cuales son cargadas en cada visita al documento. Permite exportar las anotaciones en formato RDF, el cual puede ser manipulado por Sesame, que tiene capacidades de consulta. Soporta servidores de anotación locales y remotos.

3.6.3. Sistemas Automáticos

Como se mencionó en la Sección 2.6, los sistemas de anotación automáticos son herramientas que aplican técnicas de extracción de información de lenguaje natural para generar automáticamente anotaciones en páginas Web.

A los efectos de este trabajo, son muy necesarios, pues contribuyen en gran medida a hacer manejable la anotación masiva de documentos. En la actualidad no es posible encontrar disponible en Internet tanta cantidad de ellos como de anotadores manuales y semiautomáticos, lo que restringe el número de opciones.

En las siguientes secciones se describen los anotadores automáticos encontrados.

3. 6. 3. 1. AeroDAML

AeroDAML [72] aplica técnicas de extracción de información de lenguaje natural para generar automáticamente anotaciones DAML en páginas Web. Soporta anotaciones con una ontología genérica predeterminada y otras que consisten de palabras (entidades) "mapeados" a instancias de clases de la ontología, y relaciones mapeadas a instancia de propiedades.

La versión Web de AeroDAML soporta anotaciones con una ontología genérica predeterminada. El usuario ingresa una URI y AeroDAML retorna la anotación DAML para la página especificada. La versión cliente-servidor de AeroDAML soporta anotaciones con ontologías adaptadas. El usuario ingresa un nombre de archivo y AeroDAML devuelve la anotación DAML para el documento de texto.

AeroDAML consiste de un producto comercial de extracción de información llamado AeroText y componentes para generación DAML.

3. 6. 3. 2. KIM

KIM [39] (Knowledge and Information Management) provee una infraestructura y servicios para anotación semántica automática, indexado y recuperación de contenidos semi o no estructurados.

KIM analiza textos y reconoce referencias a entidades (como personas, organizaciones, ubicaciones, fechas). Intenta vincular la referencia con una entidad conocida, dados una URI única y una descripción en la base de conocimiento. De no existir, genera automáticamente una nueva URI y una nueva descripción. Finalmente, la referencia en el documento se anota con la URI de la entidad. Esta metadata es utilizada posteriormente para indexado semántico, recuperación visualización e hipervinculado automático de documentos.

KIM ofrece un servidor, una interfaz de usuario Web y un plugin para Internet Explorer. Incluye una ontología de alto nivel (KIMO) con aproximadamente 250 clases y 100 propiedades. También cuenta con una base de conocimiento (KIM KB), prepoblada con 200.000 descripciones de entidades.

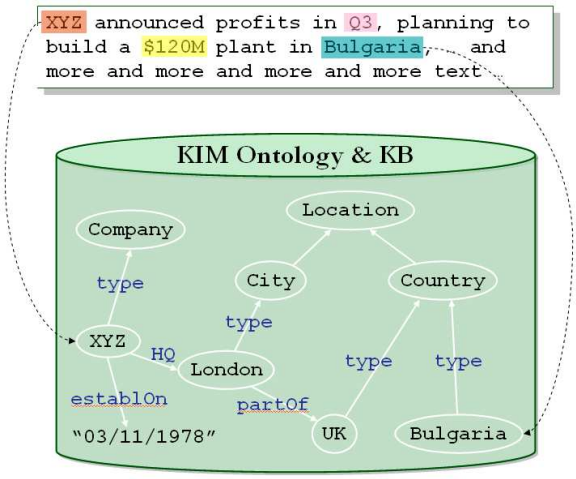


Figura 15: Ontología de KIM y Base de conocimiento

La Figura 15 ilustra la ontología de KIM, su base de conocimiento prepoblada y cómo se relaciona con un documento anotado.

3. 6. 3. 3. Magpie

Herramienta que soporta la interpretación de páginas Web. Magpie ofrece conocimiento complementario, el cual el lector puede invocar para rápidamente obtener acceso a cualquier conocimiento relevante al recurso Web.

Magpie [73] automáticamente asocia una capa semántica basada en ontologías a recursos Web, permitiendo invocar servicios relevantes en un explorador Web estándar.

3.7. Herramientas Elegidas

Las aplicaciones fueron construidas utilizando herramientas seleccionadas en base al Estudio del Estado del Arte, realizado en la etapa inicial del proyecto. En esta sección se describen los motivos que llevan a la elección de cada una de las tecnologías.

Entre las características deseables en las diferentes herramientas, se valoró especialmente que se tratara de componentes de uso libre o de distribución gratuita para fines académicos, así como también que se evitara la instalación de componentes adicionales en el cliente.

3.7.1. Java

Todas las aplicaciones fueron implementadas utilizando el lenguaje Java ya que posee determinadas características que ayudan en el desarrollo:

- Es utilizable en diferentes sistemas operativos
- Es orientada a objetos, lo que facilita la extensibilidad y flexibilidad de la aplicación
- Tiene un gran potencial de expresión desde el punto de vista del programador

3.7.2. NetBeans

NetBeans [42] es el IDE elegido como entorno de desarrollo. Las ventajas principales sobre otras herramientas de desarrollo son su disponibilidad en forma gratuita y que puede ejecutarse sobre varias plataformas: Windows, Linux, Solaris y MacOS. A pesar de que existen otras herramientas que cumplen con estos requisitos, se eligió ésta sobre las demás debido a que ya se tenían conocimientos previos acerca del manejo de la misma.

3.7.3. Tomcat

Se eligió Tomcat 5 [43], que es un servidor de aplicaciones Java cuya utilización es gratuita y con código abierto. Otras características que influyeron sobre su elección son que:

- Ofrece versiones estables
- Es utilizado por grandes industrias y organizaciones, lo que hace difícil encontrar problemas que no hayan sido detectado antes.

3.7.4. OWL

Luego de investigar diferentes lenguajes de ontologías, se seleccionó OWL debido principalmente, a su creciente uso en la actualidad, además de todas las posibilidades y facilidades que este lenguaje brinda. Otros puntos a favor para la elección de OWL son:

- es un lenguaje desarrollado y recomendado por el W3C [44]
- se deriva de varios antecesores, de los cuales hereda buenas características y a las que se le agregan otras más
- permite una mayor expresividad
- al ser muy utilizado actualmente, existe una variedad de herramientas para el desarrollo y manejo de ontologías con el mismo

3.7.5. Protégé

La elección de Protégé [22], está en gran medida ligada a la elección del lenguaje OWL. Adicionalmente, ayudó a su elección la facilidad de uso, la interfaz amigable y las utilidades que brinda. Es también una herramienta gratuita y soporta la creación, visualización y manipulación de ontologías en varios formatos de representación.

3.7.6. Jena

Jena es un marco Java para construir aplicaciones de la Web Semántica que posee APIs para el manejo de RDF, OWL y que permite la lectura y escritura en dichos lenguajes.

Para el manejo de ontologías desde Java es posible la utilización de APIs de Protégé o Jena. Se eligió la segunda opción, ya que ofrece mayor estabilidad e independencia del editor utilizado, además de que la primera, al momento de tomar la decisión, estaba disponible únicamente para una versión Beta del mismo.

3.7.7. Racer

Para verificar la satisfactibilidad y consistencia de la ontología se eligió el razonador Racer debido a que da soporte a OWL Lite y OWL DL. Otra razón trivial, pero no por ello menos importante, es que las integrantes del grupo tenían experiencia previa en la utilización de dicha herramienta.

Cabe acotar que se eligió OWL-DL porque permite modelar la realidad elegida y a su vez asegura la decidibilidad, es decir, que todas las conclusiones sean realizables en un tiempo finito.

3.7.8. dom4j

dom4j [34] es un framework XML para Java y de código abierto. Tiene como ventaja que está integrado con las demás herramientas estudiadas: DOM, SAX y JAXP. Además es una API más simple y liviana que DOM, soporta completamente el estándar DOM y utiliza JAXP para hacer parsing.

dom4j facilita la tarea de utilizar XML en Java.

3.7.9. KIM

Para la elección del anotador se ponderó positivamente los anotadores capaces de manejar un gran número de documentos.

Los anotadores manuales fueron descartados debido a la cantidad de información a anotar, así como también al volumen de páginas existentes.

Los anotadores semiautomáticos son más completos que los anteriores y en algunos casos, tienen la capacidad de reconocer nuevas entidades mediante aprendizaje automático. Tienen la desventaja de que requieren un número considerable de documentos anotados manualmente como parte del entrenamiento del sistema.

Por último, los sistemas automáticos ofrecen múltiples prestaciones y evitan el trabajo de entrenar el sistema. La herramienta de anotación KIM cumple con tres de las características deseables en un anotador: es gratuito con fines académicos, automático y extensible. Además, tiene la ventaja adicional de tener una API que permite realizar consultas semánticas y sintácticas en base al conocimiento adquirido.

4 ChoikeBilingüe

Este capítulo describe el proceso de creación y población de la ontología generada.

Primero se exponen las alternativas de diseño consideradas y se fundamenta la elección realizada. Se hace referencia brevemente a los conceptos, relaciones y propiedades definidas así como también al proceso de carga.

Posteriormente, se detallan los cambios introducidos en el diseño de la ontología original producto de la elección del anotador y las modificaciones que se realizaron en consecuencia.

4.1. Creación de la Ontología Inicial

Uno de los aspectos a trabajar es la creación de una ontología para representar el contenido del dominio de Choike. Choike es una ONG dedicada a la difusión de información de la sociedad civil del hemisferio sur y ofrece un repositorio de documentos en dos idiomas: español e inglés.

Se pretende crear una ontología que incluya los principales conceptos tratados en las páginas del dominio, y que tenga la capacidad de manejar los dos idiomas.

Se consideraron tres alternativas para el diseño de la ontología:

1. La construcción de dos ontologías independientes referentes al mismo dominio, una por idioma, y que estuvieran "mapeadas" por concepto.

   Así, la Tabla 2 presenta las propiedades que se podrían definir para la clase "país" de la ontología en español. Para la ontología en inglés se podría definir una clase "country", con las propiedades mencionadas en la Tabla 3. El "mapeo" se efectuaría entre las clases "país" y "country".

Propiedad	Valor
MainAliasEspañol	República Federativa de Brasil
AliasEspañol	Brasil
AliasEspañol	…

Tabla 2: Propiedades de la clase "país", ontología en español, alternativa 1

Propiedad	Valor
MainAliasIngles	Federative Republic of Brazil
AliasIngles	Brazil
AliasIngles	…

Tabla 3: Propiedades de la clase "country", ontología en inglés, alternativa 1

2. Construir una única ontología que represente a los conceptos independientemente de la lengua en la que son expresados. Estos conceptos están asociados a términos en los dos idiomas.

   Cada concepto corresponde a una clase, la cual tiene asociadas diversas propiedades. Estas propiedades corresponden al término principal y sus alias para cada idioma considerado.

En la Tabla 4 se observan las propiedades a definir, por ejemplo, para el país Brasil.

Propiedad	Valor
MainAliasEspañol	República Federativa de Brasil
MainAliasIngles	Federative Republic of Brazil
AliasEspañol	Brasil
AliasIngles	Brazil
AliasEspañol	…
AliasIngles	…

Tabla 4: Propiedades de alternativa 2

3. Una versión simplificada de la alternativa anterior. Se trata de una ontología que mantiene la idea de una clase por concepto, con propiedades que representan término principal y alias pero con la salvedad que no se discrimina por idioma.

   En la Tabla 5 se presentan las propiedades para el país Brasil, siguiendo la tercera alternativa.

Propiedad	Valor
MainAlias	República Federativa de Brasil
Alias	Federative Republic of Brazil
Alias	Brasil
Alias	Brazil

Tabla 5: Propiedades de alternativa 3

   En la segunda alternativa, cada propiedad recibe un nombre distinto, dependiendo el idioma (MainAliasEspañol, MainAliasIngles, etc.), mientras que en este último caso, se trata de la misma propiedad (Alias).

Los tres modelos propuestos ofrecen una solución a la problemática de la sinonimia, situación en la cual un concepto puede estar asociado a más de un término en un mismo idioma.

La primera alternativa es costosa de extender, ya que requiere la adición de una nueva ontología por cada nuevo idioma. Involucra el agregado de mapeos extra entre los conceptos por lo que contiene mucha información redundante.

Las dos últimas alternativas facilitan por igual la extensibilidad a otros nuevos idiomas mediante modificaciones mínimas.

La segunda alternativa es la que nos parece conceptualmente más adecuada, ya que al igual que la tercera, considera a los conceptos y la relación entre los mismos independientes de la lengua en que se expresen. Además, se lo prefiere sobre la tercera opción, porque hace la distinción entre los diferentes idiomas.

Para la decisión de qué clases serían incluidas en la ontología, se tomó en cuenta el trabajo del Proyecto de Grado OntoChoike [3], también basado en el dominio de Choike. Sólo fueron considerados aquellos conceptos relevantes al dominio.

En el resto de la sección, se describe con más detalle el diseño de la ontología.

4.1.1. Conceptos

En la Tabla 6 se enumeran los conceptos identificados: País, Ciudad, Región, Organización, Evento, Documento, Persona, Cargo y Sigla.

País	Esta clase representa a todos los Países del mundo. Está relacionada con las regiones a las cuales pertenece. Ejemplo: Uruguay, Argentina, Brasil.
Ciudad	Representa todas las ciudades del mundo. Está relacionada con el País al que pertenece y puede ser la capital del mismo. Ejemplo: Montevideo, Bs. Aires, París, San Pablo.
Región	Representa a las regiones, divisiones territoriales según etnia, clima, economía, idioma, y otros más. Ejemplo: África del Este, Melanesia, Oriente Medio.
Organización	Representa a los grupos u Organizaciones internacionales. Ejemplo: Organización de Naciones Unidas, Unión Europea.
Evento	Tiene lugar en un momento determinado. Ejemplo: Asamblea General de la ONU, Comisión Interamericana de DDHH.
Documento	Representa el resultado de un encuentro o acción. Ejemplo: Acuerdo de Libre Comercio, Estatuto de la Corte Penal Internacional.
Persona	Representa a los individuos. Ejemplo: Tabaré Vázquez, Fidel Castro, George Bush.
Cargo	Representa la posición o profesión de un individuo. Ejemplo: director, presidente, secretario.
Sigla	Representa los acrónimos en general. Ejemplo: ONU, UE, FAO, etc.

Tabla 6: Conceptos o clases de la ontología inicial

4.1.2. Relaciones y Propiedades

Los conceptos se relacionan entre sí mediante relaciones. La Figura 16 muestra la relación entre las clases Ciudad, País y Región. Una Ciudad es ciudad de un País, y un País pertenece a una o más Regiones. A su vez una Región puede estar contenida en cero o más Regiones.

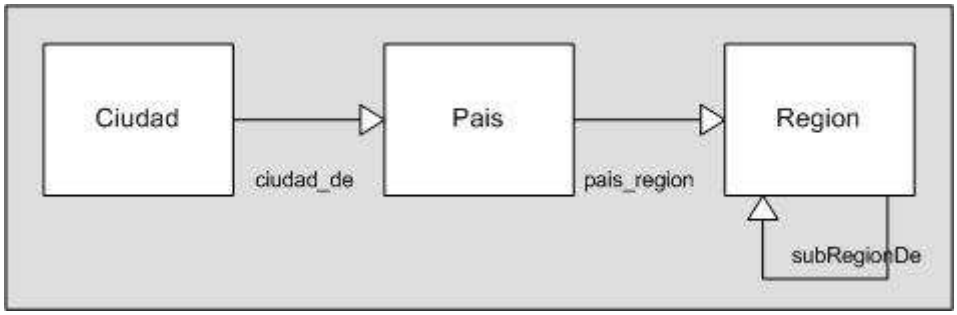


Figura 16: Relaciones entre clases Ciudad, País y Región

La Figura 17 muestra que una Persona tiene uno o más cargos.

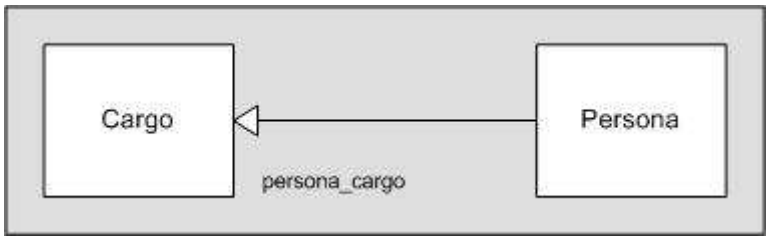


Figura 17: Relación entre clases Persona y Cargo

Por último se encuentra el diagrama de las Organizaciones, Eventos y Documentos. Se indica que todos ellos poseen cero o más siglas que los representan.

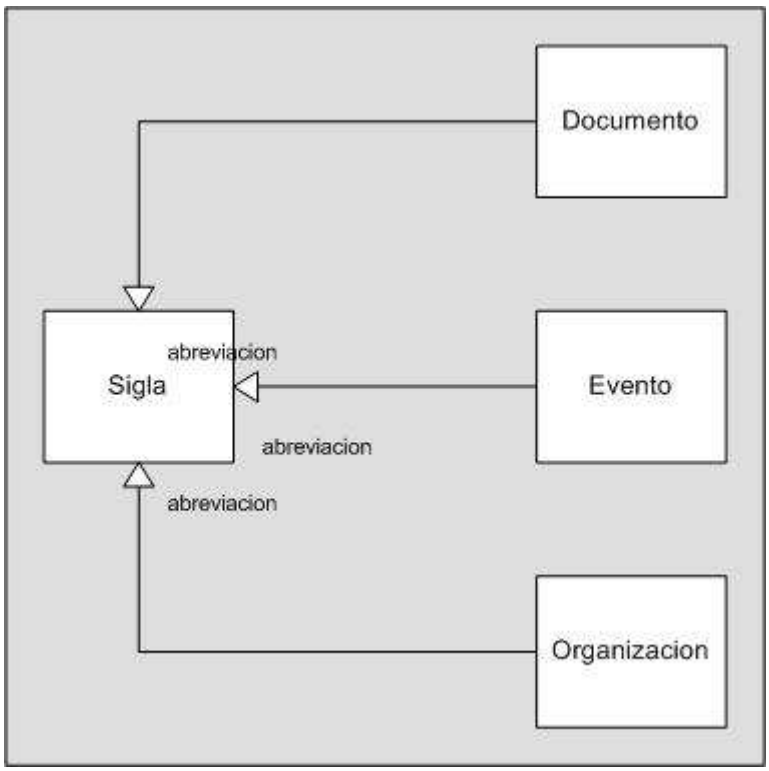


Figura 18: Relaciones de abreviación entre clases Documento, Evento, Organización y Sigla

La herramienta de representación de ontologías elegida (Protégé) distingue dos tipos de propiedades: ObjectProperties y DataTypeProperties.

La primera de ellas corresponde al concepto de Relación explicado en el Capítulo 2, donde el rango o codominio es una clase. Son ejemplos: ciudad_de (va de una Ciudad a un País), subRegionDe (va de una Región a otra Región).

La segunda corresponde a una propiedad simple o atributo; su rango es una cadena de caracteres, una fecha o un valor de verdad, por ejemplo: nombre, edad, fecha del evento.

En la Tabla 7 se detallan las relaciones y propiedades identificadas:

Propiedades comunes a todas las clases: o aliasNombreClaseEsp: Alias o sinónimo en idioma español o aliasNombreClaseIng: Alias o sinónimo en idioma inglés o prolexemeNombreClaseEsp: término principal en idioma español o prolexemeNombreClaseIng: término principal en idioma inglés o estaEnInformes: indica en qué Informes se encuentra dicha instancia de clase
Propiedades de clase Pais: o pais_region: ObjectProperty que indica a qué regiones pertenece un país dado
Propiedades de clase Ciudad: o ciudad_de: ObjectProperty que indica a qué Pais pertenece la Ciudad o es_capital: DatatypeProperty que indica si la ciudad es capital del Pais mencionado en ciudad_de
Propiedades de clase Región: o subRegionDe: ObjectProperty que hace referencia a la región en la cual está contenida dicha Region
Propiedades de clase Organizacion: o abreviacion: ObjectProperty que indica la relación entre la Organización y la sigla que la representa
Propiedades de clase Evento: o abreviacion: ObjectProperty que indica la relación entre el Evento y la sigla que lo representa
Propiedades de clase Documento: o abreviacion: ObjectProperty que indica la relación entre el Documento y la sigla que lo representa
Propiedades de clase Persona: o persona_cargo: ObjectProperty que indica los cargos que ostenta una persona
Propiedades de clase Cargo: o no agrega ninguna propiedad.
Propiedades de clase Sigla: o no agrega ninguna propiedad.

Tabla 7: Relaciones y propiedades de la ontología

4.1.3. Ontología Inicial

En una primera instancia se decidió reflejar en la ontología el dominio completo (Figura 19), incluyendo las secciones en que está dividido el portal de Choike y las páginas en sí. Para ello, se agregaron clases tales como: Ambiente, Comunicación, etc. que representan las secciones del sitio y la clase Informes, que se corresponde con las páginas. El ciclo se cierra a través de una relación entre instancias de las clases base e instancias de informes, los cuales a su vez están relacionados con las secciones del portal. Se logra reunir todo el conocimiento en la ontología: los conceptos, las instancias de los mismos y las anotaciones de los documentos en base a estos conceptos.

Figura 19: Diseño original de la ontología ChoikeBilingüe

4.1.4. Generación y carga de diccionarios

En base al dominio se generaron diccionarios en forma de archivos XML, conteniendo las entidades mencionadas anteriormente. En tal sentido, se crearon los siguientes diccionarios:

- Regiones
- Países
- Ciudades
- Siglas
- Cargo
- Personas
- PersonaCargo

- Eventos
- Organizaciones
- Documentos
- Abreviaciones

Para la construcción de varios diccionarios, se utilizó el archivo XML de salida del proyecto OntoChoike, que contenía las entidades reconocidas en los documentos de Choike. Dado que el presente trabajo utiliza los dos idiomas, español e inglés, y la salida generada por OntoChoike está en español únicamente, fue necesario hacer modificaciones adicionales para generar entidades en ambos idiomas.

Los diccionarios construidos a partir de este archivo XML son: siglas, personas, personaCargo, eventos, organizaciones, documentos y abreviaciones. Por cada clase encontrada en este documento, se agregó una instancia al diccionario correspondiente. Este proceso se realizó mediante la implementación de programas auxiliares utilizando dom4j, que permite el manejo de XML desde Java.

Regiones, países y ciudades fueron construidos a partir de enciclopedias en línea. Éstas están más actualizadas y tienen la ventaja de contener la jerarquía región – país – ciudad existente. Se encontró información en formato de texto, la cual fue almacenada en base de datos para luego ser exportada a formato XML. Las listas mencionadas ya incluían las entidades en español e inglés, y se hizo una verificación para comprobar que no existieran inconsistencias.

El diccionario de cargos se construyó inicialmente en base a OntoChoike, y posteriormente se le agregaron nuevos elementos que se consideraron relevantes, así como sus alias en inglés.

En la Figura 20 se presenta un fragmento del diccionario de Regiones. Cada tag de Región representa una instancia de dicha clase. A su vez, cada elemento contiene atributos que representan las propiedades de dicha clase (ver Tabla 7).

```
<?xml version="1.0" encoding="utf-8"?>
<Regiones>
     <Region   value="África"
               prolexemeRegionEsp="África"
               prolexemeRegionIng="Africa"/>
     <Region   value="África del Este"
               prolexemeRegionEsp="África del Este"
               prolexemeRegionIng="East Africa"
               aliasRegionEsp="África Oriental"
               aliasRegionIng="Eastern Africa"
               subRegionDe="África"/>
</Regiones>
```

Figura 20: Fragmento del diccionario de Regiones

Los diccionarios en español realizados por OntoChoike fueron utilizados como base para construir los nuevos diccionarios.

Se observan varias diferencias con respecto a la ontología creada en dicho trabajo. Además de cambios en la estructura motivados por un énfasis en diferentes

aspectos del dominio, la naturaleza bilingüe de la ontología impone sus propios requerimientos. Esto se refleja en el ejemplo anterior, por la existencia de dos propiedades: prolexemeRegionEsp y prolexemeRegionIng que representan el término principal de cada concepto en español e inglés respectivamente. De forma similar, para los alias (o denominación secundaria) se tienen las propiedades aliasRegionEsp y aliasRegionIng, diferenciándose de los términos principales únicamente en la multiplicidad.

En el ejemplo de la Figura 20, la región "África del Este" tiene como términos principales "East Africa" y "África del Este" en inglés y español respectivamente. Además tiene como alias Eastern Africa y África Oriental en el mismo orden. También se indica que esta región es subregión de África.

Luego de la construcción de los diccionarios, se realizó una carga masiva automática de instancias en la ontología a partir de los mismos. A esto se lo denomina población de la ontología. En la Sección 5.1 se explica el proceso con más detalle.

4.1.5. Anotación semántica

El éxito de la Web Semántica depende de la disponibilidad de ontologías así como también de la existencia de páginas Web anotadas con metadata basada en dichas ontologías. Por esta razón, la capacidad de anotar documentos es crítica.

Las herramientas de anotación semántica permiten estructurar los contenidos en conceptos con significado y añadir información para explicar el contenido de las páginas. Este tipo de información servirá para posibilitar que las búsquedas de información sean más eficientes.

4. 1. 5. 1. Elección del anotador

Se realizó un estudio de las características de los anotadores existentes y ponderando según las necesidades del proyecto.

Se analizó el impacto de relacionar el anotador a la ontología construida y se observó la necesidad de disponer de un anotador que pudiera basarse en la ontología definida para anotar semánticamente, y que a su vez fuera capaz de manejar un gran número de documentos e información.

Las herramientas disponibles en la actualidad condicionan fuertemente la elección del anotador. Los anotadores manuales fueron descartados debido a la cantidad de información a anotar, así como también al volumen de páginas existentes. La anotación manual, página a página, es un trabajo que no agrega valor al proyecto, y requiere intervención manual por cada nueva página que se incorpore al sitio.

Como contrapartida, la mayoría de estos anotadores permite hacer anotaciones sobre los propios documentos HTML, característica que puede ser deseable en algunas situaciones.

Los anotadores semiautomáticos son más completos que los anteriores y en algunos casos, tienen la capacidad de reconocer nuevas entidades mediante aprendizaje automático. Por otra parte, requieren un número de documentos anotados manualmente como parte del entrenamiento del sistema.

Los sistemas automáticos ofrecen múltiples prestaciones y evitan el trabajo de entrenar el sistema. Como se mencionó en el Capítulo 3, la herramienta de anotación KIM cumple con los requisitos: gratuita para fines académicos, automática y extensible. Tiene la ventaja adicional de tener una API que permite realizar consultas semánticas y sintácticas en base al conocimiento adquirido. La arquitectura en capas de KIM permitió integrar la ontología construida en el proyecto con la propia de KIM, agregándole mayor conocimiento a la ontología del proyecto a costa de un proceso de integración no trivial.

Por lo expuesto anteriormente, se decidió realizar cambios a la ontología, modificaciones de tipo estructural, no conceptual. Fue necesario cambiar algunas propiedades así como quitar otras, y realizar variaciones en la configuración predeterminada. Los cambios a la ontología trajeron como consecuencia modificaciones a los diccionarios utilizados para la carga de la misma.

4.2. Cambios en la Ontología

KIM posee un "Conocimiento del Mundo Real (Common World Knowledge)", que consiste de un conjunto de ontologías propias y una base de conocimiento. En todo este "conocimiento" se basa para anotar los documentos.

No sólo anota utilizando sus propias ontologías, sino que permite extender las mismas para luego explotarlas en la fase de la anotación semántica. A su vez, permite eliminar de su motor, por ejemplo, a las ontologías que no se deseen reutilizar, debido a que no se corresponden con el dominio de interés.

Para poder anotar los documentos, era necesario modificar la ontología inicial de ChoikeBilingüe, extendiendo la ontología base de KIM y quitando aquellas que no aportan al proyecto.

Con extender la ontología, se entiende a "colgar" las clases y propiedades de la ontología base de KIM.

4.2.1. Descripción de los cambios

Se efectuaron diversas modificaciones al diseño original, los cuales se detallan a continuación.

4. 2. 1. 1. En las propiedades y clases

- La ontología de KIM posee una clase general para todas las entidades, llamada *Entity*. Esta clase tiene una propiedad *hasAlias* que asocia una entidad a sus "Alias". Esto es utilizado para el caso de los sinónimos.

  Debido a que KIM ya posee una propiedad para representar la sinonimia entre términos, se volvió innecesaria la utilización de las propiedades alias y prolexeme dependientes del idioma definidas inicialmente. Debido a que la nueva ontología es una extensión de la de KIM, todas las clases (subclases de *Entity*) heredan esta propiedad.

  En la Figura 21 se ilustra un ejemplo de cómo se trata la sinonimia.

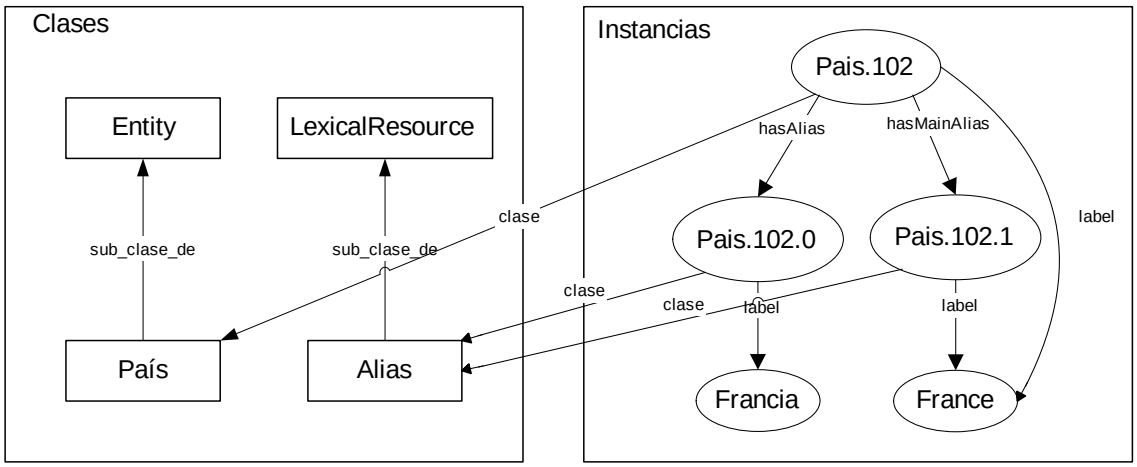


Figura 21: Tratamiento de la sinonimia

El país "Pais.102" correspondiente a Francia tiene dos alias, en este caso uno por idioma, "Pais.102.0" y "Pais.102.1". Éstos son instancias de la clase Alias, y se utilizan para indicar los diferentes sinónimos o alias que tiene la instancia. "Francia" y "France" son simplemente etiquetas (labels) de cada alias, y son utilizados para reconocer la instancia.

- En la ontología de KIM existe una clase llamada *Location* para representar las ubicaciones geográficas. A su vez, esta clase posee una relación (ObjectProperty) llamada subRegionOf, que relaciona una ubicación con la ubicación que lo contiene.
  Las clases Pais, Region y Ciudad de la ontología original pasan a ser subclases de la clase Location, y por lo tanto también heredan esta propiedad. SubRegionOf es el análogo de pais_region, ciudad_de y subRegionDe para el caso de un País, Ciudad y Región respectivamente. Por esta razón, se decidió quitar estas propiedades y reutilizar la propiedad subRegionOf heredada.

- La propiedad Abreviación, en la ontología inicial, era propia de la clase abstracta *Organizaciones* y heredada por las clases *Documento, Evento y Organización*.
  Debido a que fue necesario cambiar las clases para extender la ontología de KIM, ahora estas clases ya no son subclases de *Organizaciones* sino que pasan a ser subclases de clases de la ontología KIM. Es por esto que hubo cambios en esta propiedad. En lugar de tener una única propiedad heredada, ahora cada clase posee su propia ObjectProperty que la relaciona con su sigla correspondiente.

- Las clases *Persona* y *Cargo* en la nueva ontología pasaron a ser subclases de Person y JobPosition, las cuales están relacionadas por la propiedad hasPosition, la cual cumple la misma función que la propiedad persona_cargo de la ontología inicial. Es por ello que persona_cargo fue suprimida.

- La relación entre los documentos y las entidades, antes representada por la propiedad *estaEnInformes*, ahora está enteramente a cargo del motor de KIM, razón por la cual, se suprime también la clase Informes. A partir de los documentos y el conocimiento de la ontología, KIM realiza anotaciones almacenando la relación de qué entidades se encuentran en qué documentos.

- Como desaparece la clase Informes, no tiene sentido la existencia de las clases que modelan las categorías en las que se divide el sitio Web de Choike (Ambiente, Comunicación, etc.). Además, el anotador no tendría forma de relacionar los documentos con las categorías porque los documentos en sí mismos no poseen un identificador de la categoría a la que pertenecen.

En la Tabla 8 se presenta el resumen de los cambios efectuados en las propiedades de la ontología.

Propiedad Anterior	Nueva propiedad	Heredada de KIM?
aliasNombreClaseEsp	hasAlias	Sí
aliasNombreClaseIng	hasAlias	Sí
prolexemeNombreClaseEsp	hasAlias	Sí
prolexemeNombreClaseIng	hasAlias	Sí
pais_region	subRegionOf	Sí
ciudad_de	subRegionOf	Sí
subRegionDe	subRegionOf	Sí
abreviacion (Organización)	abreviacionOrganizacion	No
abreviacion (Documento)	abreviacionDocumento	No
abreviacion (Evento)	abreviacionEvento	No
es_capital	es_capital	No
persona_cargo	hasPosition	Sí
estaEnInformes	No existe propiedad	

Tabla 8: Cambios en las propiedades de la ontología

En la Figura 22 se muestra la ontología inicial (izquierda), en contraposición con la nueva ontología (derecha), producto de los cambios introducidos para poder utilizar el anotador elegido. En la segunda, se observa gráficamente que las clases pertenecientes a ChoikeBilingüe están marcadas con un círculo y las clases restantes son las propias de KIM.

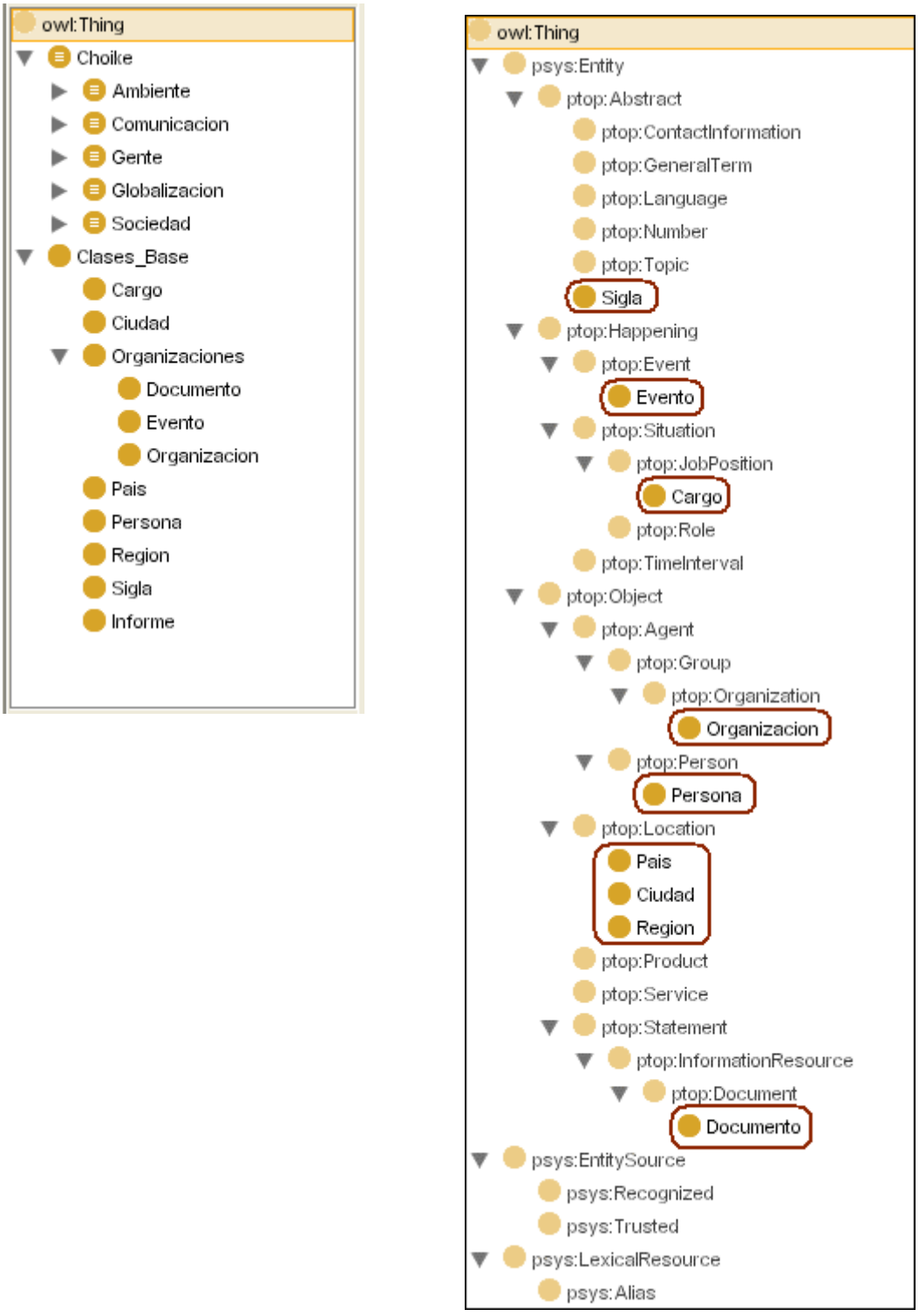


Figura 22: Diseño original versus nuevo diseño de la ontología ChoikeBilingüe

4. 2. 1. 2. En los diccionarios

Los cambios en los atributos y relaciones originaron modificaciones en los diccionarios utilizados en la carga de la ontología: se suprimieron y agregaron atributos en los archivos XML, según el caso.

En la Figura 23 se muestra el mismo ejemplo presentado en la Figura 20, ahora con las nuevas modificaciones.

```
<?xml version="1.0" encoding="utf-8"?>

<Regiones>

        <Region value ="África">

                <alias value ="hasAlias" rango="Alias" hasAlias="África"/>

                <alias value ="hasAlias" rango ="Alias" hasAlias ="Africa"/>

        </Region>

        <Region value ="África del Este">

                <alias value ="hasAlias" rango ="Alias" hasAlias ="África del Este"/>

                <alias value ="hasAlias" rango ="Alias" hasAlias ="East Africa"/>

                <alias value ="hasAlias" rango ="Alias" hasAlias ="África Oriental"/>

                <alias value ="hasAlias" rango ="Alias" hasAlias ="Eastern Africa"/>

                <subRegion value="subRegionOf" rango ="Region" subRegionOf="África"/>

        </Region>

</Regiones>
```

Figura 23: Diccionario de Regiones modificado

4. 2. 1. 3. En la carga y formato de la KB

La KB propia de KIM está en formato N-TRIPLES, por lo que cualquier extensión de la misma debe estar en el mismo formato.

Previo a la utilización de KIM, la implementación del poblado de la ontología daba como resultado un archivo con formato OWL conteniendo la ontología con las instancias cargadas. Luego de conocer la restricción del formato de la KB de KIM, se realizaron cambios para que la salida de la carga de diccionarios devuelva tanto archivos con formato OWL como N-TRIPLES, de extensión .OWL y .NT respectivamente.

5 Solución Construida

Este capítulo ofrece una descripción de los componentes de software construidos. Se comentan las características más relevantes del diseño y la implementación de la aplicación para la población de la ontología – ChoikeBilingüe - y del Buscador Semántico ChoBiSearch.

5.1. Población de la ontología – ChoikeBilingüe

Como se mencionó en el capítulo anterior, es necesario efectuar la carga masiva automática de instancias en la ontología a partir de diccionarios.

La aplicación ChoikeBilingüe toma la ontología construida para el dominio en particular (Choike) y un conjunto de diccionarios en formato XML, que contienen instancias de las clases definidas en la ontología. Es capaz de manejar los diccionarios de entrada y efectuar operaciones para poblar y salvar la ontología.

5.1.1. Diseño

Las funcionalidades requeridas por el sistema, pueden ser incluidas en un único paquete llamado "Ontología" (Figura 24).

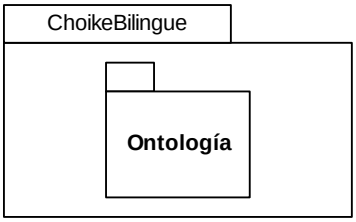


Figura 24: Diagrama de paquetes de ChoikeBilingüe

El mismo se puede dividir en componentes que realizan tareas más específicas, en pos de satisfacer los principios de alta cohesión y bajo acoplamiento (Figura 25).

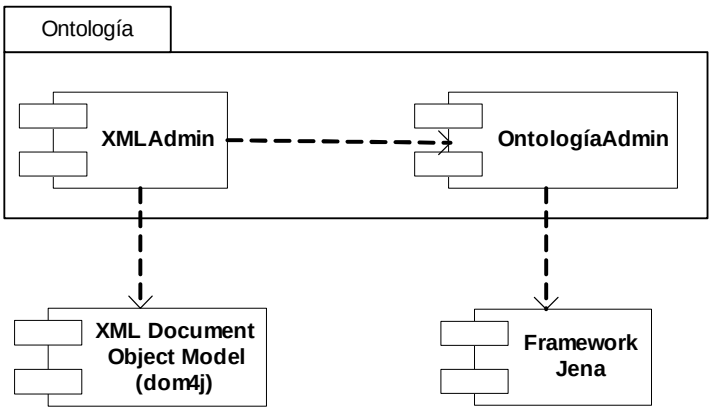


Figura 25: Diagrama de Componentes ChoikeBilingüe

A continuación se describe cada uno de los componentes con más detalle.

XMLAdmin es la clase que permite definir y manipular documentos XML. Esta clase es la encargada de leer los diccionarios para obtener las instancias a cargar en la ontología. Para cargar la misma en base a las instancias, este componente se comunica con OntologiaAdmin, el cual le proporciona servicios necesarios para la manipulación de la ontología.

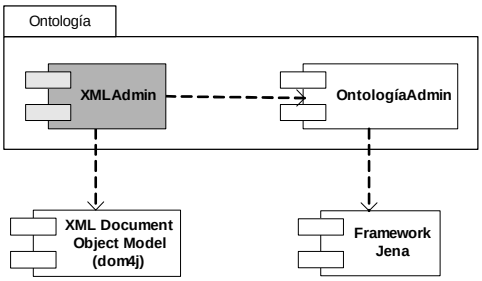


Figura 26: XMLAdmin

OntologiaAdmin es la clase que permite manipular la ontología, esto es: levantar una ontología ya creada (archivo .OWL), agregarle instancias, relaciones, listar atributos y propiedades. Por último, es posible salvarla manteniendo las modificaciones efectuadas, en dos formatos diferentes: OWL y N-TRIPLES.

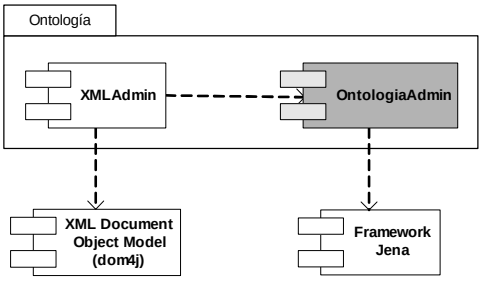


Figura 27: OntologiaAdmin

5.1.2. Implementación

La Figura 28 presenta el diagrama de clases de ChoikeBilingüe.

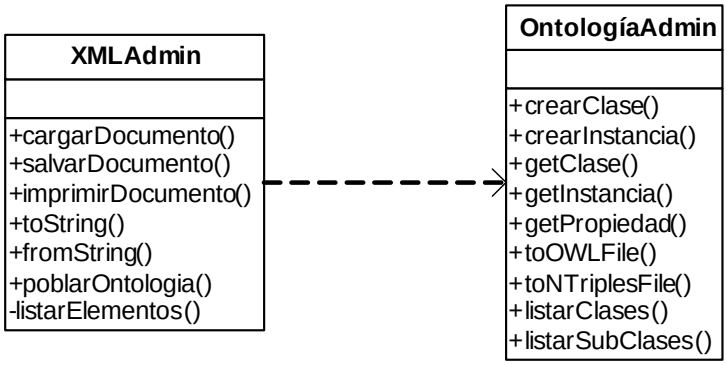


Figura 28: Diagrama de clases de ChoikeBilingüe

Como se mencionó en el Capítulo 3, para el manejo de ontologías desde Java es posible utilizar APIs de Protégé o Jena. La segunda opción ofrece mayor estabilidad e independencia del editor utilizado, además de que la primera, al momento de tomar la decisión, estaba disponible únicamente para una versión Beta del mismo.

Para el manejo de documentos XML (los diccionarios) se utilizó la librería de código abierto dom4j. La misma posee primitivas para recorrer los elementos (tags XML), sus atributos y sus sub-elementos, en forma recursiva.

El proceso de carga se hizo lo suficientemente genérico como para poder cargar cualquier diccionario sin modificación alguna; es independiente de las clases y los atributos que contengan los diccionarios mientras se respete el formato (Figura 23).

La carga de la ontología requiere como entradas, una ontología ya definida (clases, propiedades y axiomas), vacía (sin instancias) y un conjunto de diccionarios.

XMLAdmin es el encargado de levantar los diccionarios y, en base a estos y a los servicios brindados por OntologiaAdmin, realizar la carga de la ontología. El proceso

consiste en recorrer cada elemento de los diccionarios, obteniendo la instancia y la clase a la que pertenece y crear en la ontología, dicha instancia con sus propiedades.

Existen diccionarios que contienen únicamente relaciones entre entidades (persona-cargo y abreviaciones), por razones de claridad. La carga de instancias de relaciones se efectúa de igual forma que la de instancias de clases.

Por último, la ontología poblada se almacena en archivos OWL y N-TRIPLES.

5.2. Buscador semántico – ChoBiSearch

A continuación se presentan las diferentes etapas por las cuales transcurre el proceso de construcción de la aplicación ChoBiSearch.

5.2.1. Diseño

Los requerimientos recogidos conducen claramente a la definición de una arquitectura cliente-servidor. Así, el sistema consta de un servidor central encargado de recibir y procesar las consultas que le son enviadas por el cliente.

La Figura 29 presenta el escenario de distribución esperado para la instalación de la aplicación.

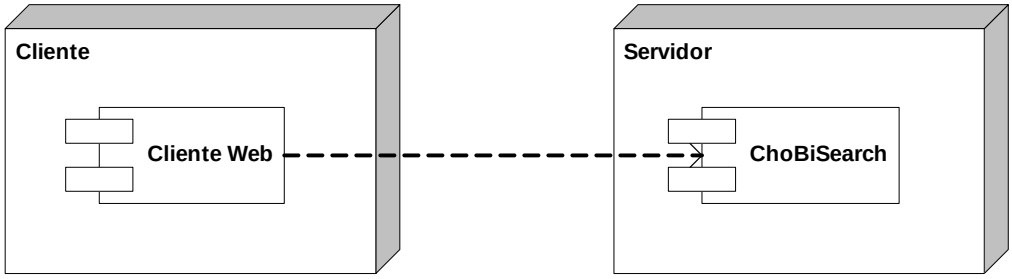


Figura 29: Diagrama de distribución de ChoBiSearch

A continuación se describen los aspectos principales del los nodos Cliente y Servidor.

Cliente: representa el equipo donde se ejecuta el Cliente Web. El mismo, es cualquier navegador de Internet, como ser Mozilla Firefox e Internet Explorer. El usuario realiza las consultas, las cuales son enviadas al Servidor para ser procesadas y los resultados recibidos son desplegados por el navegador.

No existen componentes lógicos de la aplicación ejecutándose en este nodo.

Servidor: representa el equipo donde se ejecuta la aplicación ChoBiSearch. Es el encargado de obtener, procesar y devolver las consultas enviadas desde el Cliente Web.

El diseño general de la arquitectura se puede observar en la Figura 30, mediante un diagrama de componentes lógicos. Se trata de un diseño basado en componentes de propósito claro, concreto y con alto grado de cohesión.

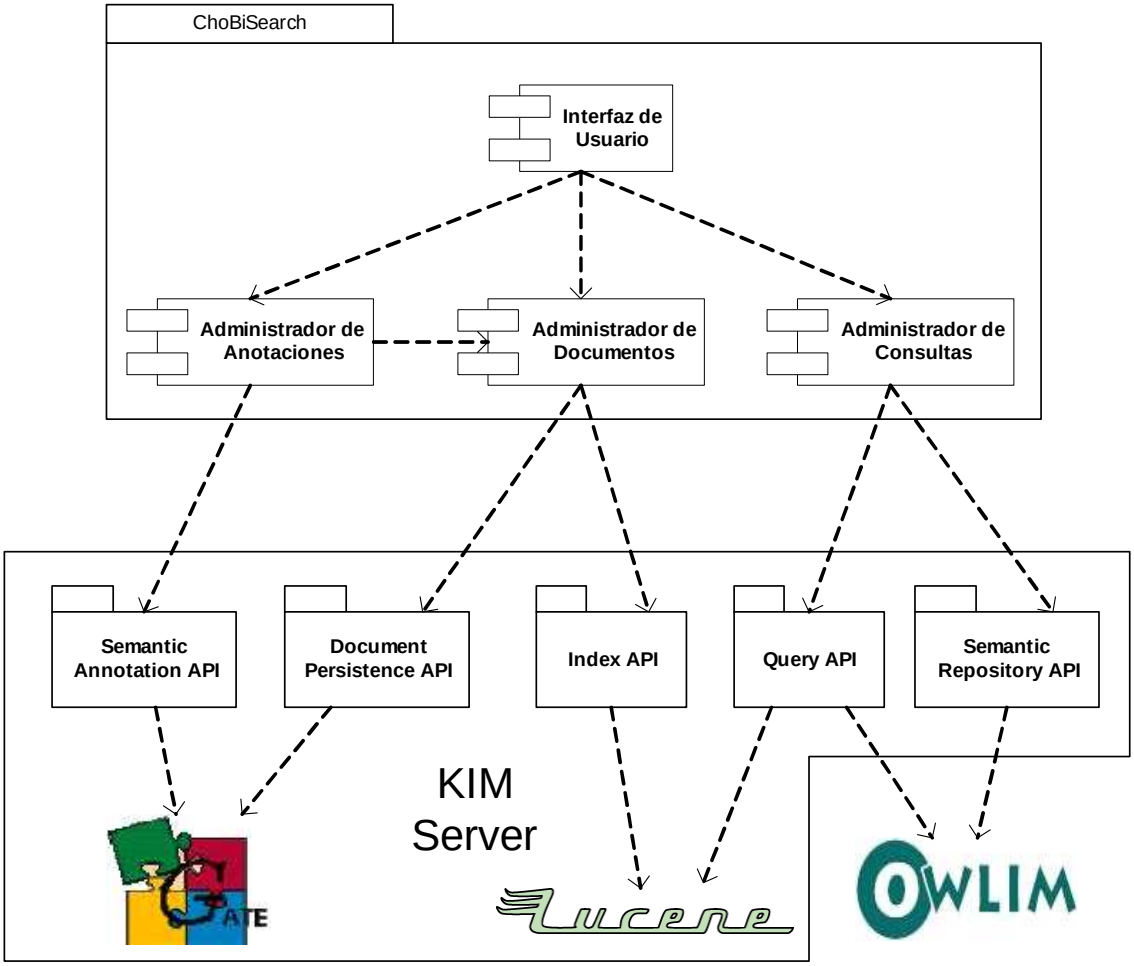


Figura 30: Vista lógica de ChoBiSearch

La división hecha en la figura anterior tiene el cometido de mostrar la separación entre el sistema construido (ChoBiSearch) y el sistema de KIM con el cual se comunica el primero. La arquitectura del servidor KIM se obtuvo de la documentación de Ontotext [83].

Luego de presentada una vista general del sistema, se detallan las principales funcionalidades de cada uno de los componentes.

El Administrador de Anotaciones es el encargado de manejar las anotaciones de los documentos. Es capaz de crear, manipular y explotar las mismas.

Utiliza servicios del Administrador de Documentos para obtener aquellos que serán anotados o cuyas anotaciones serán recorridas.

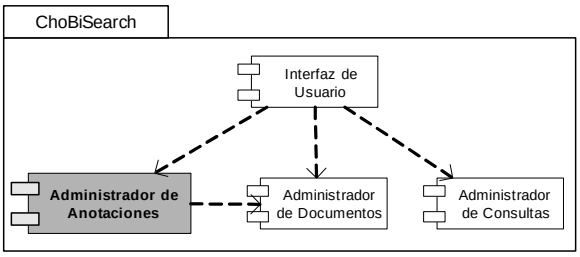


Figura 31: Componente Administrador de Anotaciones

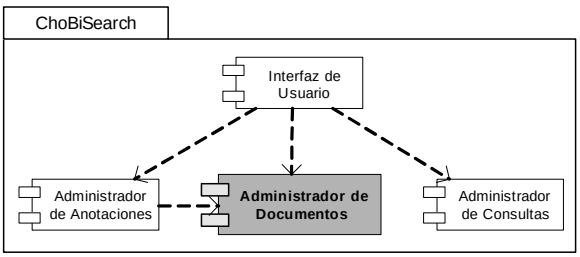


Figura 32: Componente Administrador de Documentos

El Administrador de Documentos es el componente responsable de gestionar los documentos anotados en una persistencia administrada por KIM.

Permite crear, eliminar y almacenar documentos; manipular su contenido y sus propiedades (título, fecha de modificación, URL).

El Administrador de Consultas es el conjunto de clases cuya principal funcionalidad es la de armar consultas en base a los datos ingresados por un usuario.

Genera una consulta comprensible por la QueryAPI de KIM, la ejecuta, recibe como resultado un conjunto de documentos y los transforma en un formato apto para ser desplegado en la interfaz gráfica.

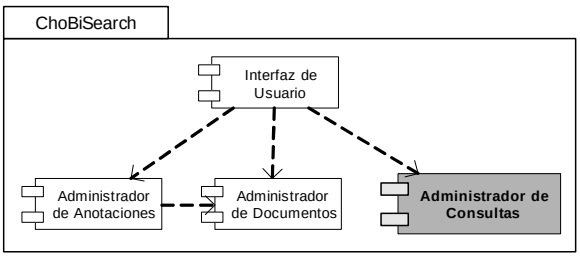


Figura 33: Componente Administrador de Consultas

Una vez mencionados los componentes del sistema construido, se pasan a detallar las diferentes APIs que brinda KIM y que son consumidas por los mismos.

KIM provee funcionalidad e infraestructura para anotación semántica, indexado y recuperación, así como manejo de documentos y consultas sobre la base de conocimiento. Las APIs que brindan dicha funcionalidad son:

Corpora API: Permite la creación de documentos a partir de una URL o contenido textual. Además, permite el manejo de conjuntos de anotaciones de estos documentos, y sus propiedades.

Index API:
Brinda las siguientes funcionalidades:

- Crear, eliminar y optimizar el índice
- Indexar documentos respecto a su contenido textual e identificadores de las entidades que contiene
- Búsqueda de documentos dada una consulta y criterios de ordenamiento

Persist API (o Document Persistence API): Permite cargar, almacenar, eliminar y sincronizar documentos, así como obtener documentos a partir de determinadas restricciones.

Index And Persist API: Combina la funcionalidad de las APIs de Persistencia e Indexamiento y por lo tanto provee ambos: indexamiento y almacenamiento de documentos sobre el mismo almacenamiento físico. Su implementación más reciente está basada en Lucene IR Engine (de Apache).

Semantic Annotation API: Brinda una cantidad de métodos que toman como entrada un texto o documento y provee anotaciones semánticas como resultado. El resultado del proceso de anotación es un conjunto de anotaciones que son integradas en el documento retornado.

Semantic Repository API: Provee una API sobre repositorio RDF(S). Actualmente está implementado sobre la API Sesame.

Query API: Brinda funcionalidad para realizar consultas sobre el repositorio semántico y el almacenamiento de documentos. Ofrece infraestructura para crear consultas semánticas que especifican restricciones sobre entidades, incluyendo sus atributos y propiedades (como relaciones a otras entidades). Estas consultas semánticas podrían ser usadas para recibir las entidades relevantes del repositorio semántico o los documentos que refieren a estas entidades. Además, ofrece la posibilidad de crear y ejecutar consultas combinadas (semántica + terminológica) que permite realizar búsquedas más complejas.

A su vez, la plataforma KIM está basada en otras plataformas, robustas y de código abierto, especializadas en tres dominios diferentes: repositorios OWL, extracción de información y recuperación de información.

Los recursos de conocimiento son guardados en el repositorio *OWLIM*, basado en Sesame, el cual provee infraestructura de almacenamiento y consulta.

OWLIM [79] es una denominación para OWLMemSchemaRepository SAIL (Storage and Inference Layer) para Sesame, que soporta razonamiento sobre OWL. Es una implementación en-memoria, el contenido completo del repositorio es cargado y mantenido en memoria principal, para hacer más eficientes la recuperación y las consultas. Asimismo, OWLIM incluye una persistencia confiable, basada en archivos en formato N-TRIPLES [78].

Sesame [80] es un framework Java de código abierto para almacenamiento, consulta y razonamiento con RDF y RDF Schema. Puede ser usado como una base de datos para los mismos, o como una biblioteca Java para aplicaciones que necesiten trabajar internamente con RDF. Sesame ofrece herramientas para tokenizar, interpretar, consultar y almacenar información proveniente de archivos RDF de gran tamaño en la propia aplicación en construcción, en una base de datos separada o en un servidor remoto.

La plataforma GATE [81] se ha utilizado como base para el proceso de extracción de información y para la administración de contenidos y anotaciones. Provee técnicas de análisis de texto, sobre las cuales se han construido extensiones potenciadas semánticamente, específicas para la extracción de información en KIM.

El motor de recuperación de información Lucene [82] es el encargado de realizar indexado y recuperación con respecto a entidades nombradas y evaluación de relevancia de acuerdo a las mismas [77].

5.2.2. Implementación

A continuación se presenta el diagrama de clases y se profundiza en las características más importantes.

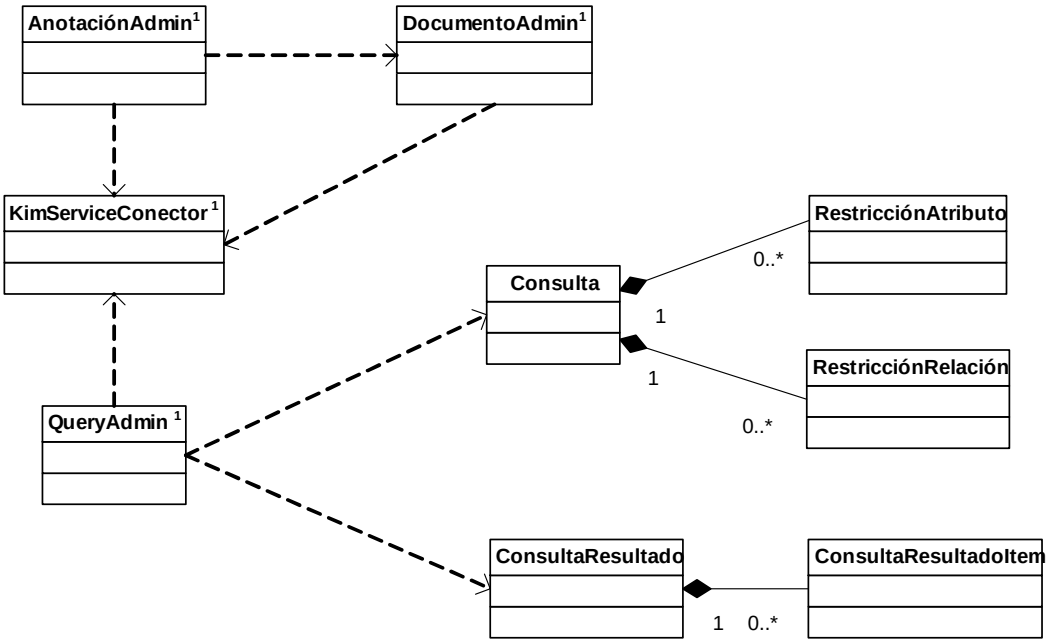


Figura 34: Diagrama de clases de implementación de ChoBiSearch

KimServiceConector es la clase que se conecta al servidor de KIM y que por lo tanto, permite utilizar todos los servicios provistos por el mismo a través de sus APIs.

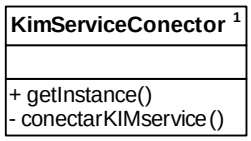


Figura 35: Clase KimServiceConector

DocumentoAdmin y AnotacionAdmin se corresponden con los componentes Administrador de Documentos y Administrador de Anotaciones respectivamente. Estas clases fueron descriptas en la sección anterior, razón por la cual no se profundiza en esta sección.

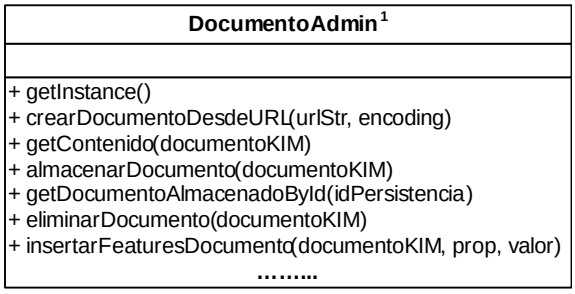


Figura 36: Clase DocumentoAdmin

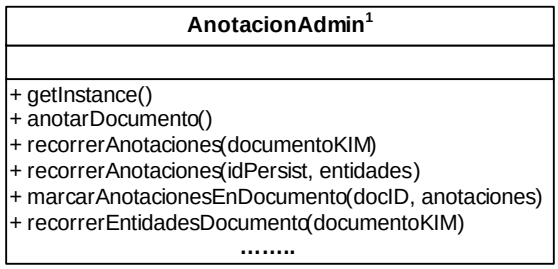


Figura 37: Clase AnotacionAdmin

Todos los controladores fueron creados utilizando el patrón Singleton, que permite tener una única instancia de la clase en el sistema: KimServiceConector, DocumentoAdmin, QueryAdmin y AnotacionAdmin.

El componente Administrador de Consultas está conformado por un conjunto de clases que permiten administrar las diferentes consultas. Por un lado, se tiene la

clase controladora llamada QueryAdmin, que se encarga de recibir las solicitudes y actuar según corresponda. Las principales funciones se enumeran en la Figura 38.

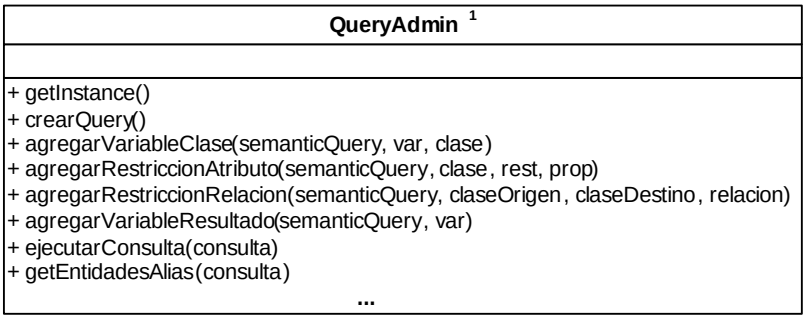


Figura 38: QueryAdmin

Básicamente, las tareas principales de esta clase abarca la creación de consultas vacías, a las cuales permite añadir restricciones tanto de atributos como de relaciones. Una restricción de atributo podría ser que el nombre de un país, considerando todos sus alias posibles, sea exactamente igual a Francia. Como restricción de relación podría ser una ciudad ubicada en el país Francia.

X: Pais
Y: Ciudad
X Ξ ”Francia”
Y subRegionOf X

Figura 39: Ejemplo parcial de consulta

Otra posibilidad que brinda esta clase es la de indicar las variables de la consulta (X o Y en este caso) que deberán estar contenidas en el conjunto de documentos a devolver. Este tipo de variable se denomina “variable resultado”.

En el ejemplo anterior podrían recuperarse documentos que contengan {X}, {Y} o {X, Y}; Si se quisiera obtener los documentos que incluyen ciudades de Francia bastaría con construir la consulta:

X: Pais
Y: Ciudad
X Ξ ”Francia”
Y subRegionOf X
Variables Resultado: {Y}

Figura 40: Ejemplo completo de consulta de Fig.38

Cabe aclarar que este formato de consulta está dado por la clase Consulta, la cual se detalla en profundidad más adelante.

Además de generación de consultas, ofrece la funcionalidad de ejecución de las mismas. Para ello, recibe una instancia de la clase Consulta, armada a partir de los datos ingresados por un usuario desde la interfaz gráfica. Luego, convierte dicha consulta en una consulta semántica comprensible por la Query API de KIM, la ejecuta y obtiene una colección de documentos. A partir de esta colección, construye una instancia de la clase ConsultaResultado, que será utilizada para desplegar en la interfaz gráfica, datos de los documentos recuperados.

Por último, otra de las funciones más importantes es la de obtener, a partir de una instancia de Consulta, los alias de las entidades que satisfacen la misma. Esto se utiliza para resaltar las ocurrencias de dichas entidades en los documentos resultado, al desplegarlos al usuario.

Como se mencionó anteriormente, QueryAdmin utiliza servicios brindados por las clases Consulta y ConsultaResultado, que se detallan a continuación.

La clase Consulta está conformada por tres componentes: una colección de variables involucradas en la consulta (denominadas variables definición), con su respectiva clase, una colección de restricciones y una colección de variables resultado (las cuales fueron mencionadas en la Figura 40).

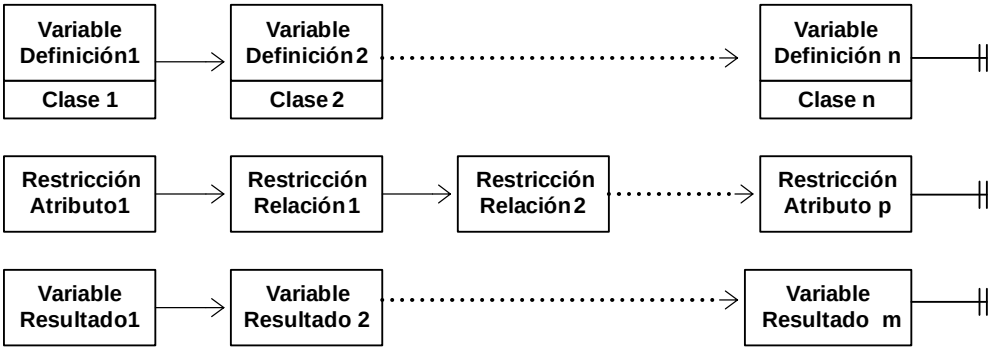


Figura 41: Componentes de la clase Consulta

La colección de restricciones consta de un conjunto de restricciones de atributo (clase RestriccionAtributo) y restricciones de relación (clase RestriccionRelacion). Es indiferente el orden en el que son definidas las restricciones en la colección, el resultado no cambia. En los párrafos siguientes se detallará un poco más la estructura de las restricciones.

A continuación se muestra el diagrama de clases para Consulta, RestriccionAtributo y RestriccionRelacion.

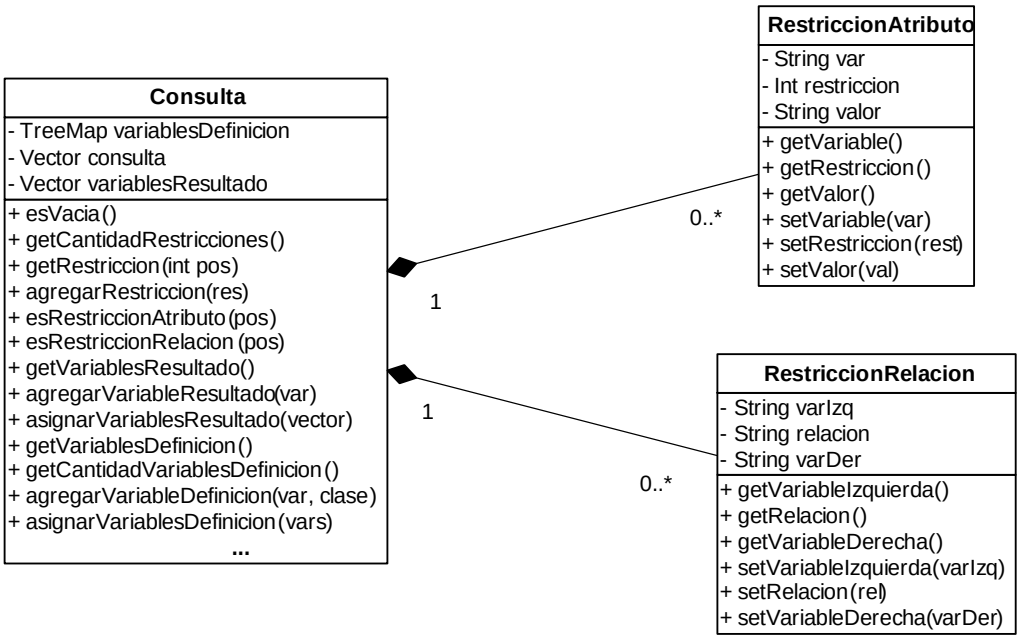


Figura 42: Diagrama de clases para Consulta

La clase Consulta contiene los datos de una consulta semántica. Como se mencionó anteriormente, es manejada por QueryAdmin para poder ejecutar una consulta. Brinda funciones para agregar, modificar y eliminar variables definición, variables resultado y restricciones tanto de atributo como de relación.

RestriccionAtributo se compone de una variable, una restricción y un valor literal. Los valores posibles de restricción son EXACTAMENTE_IGUAL, CONTIENE, COMIENZA_CON, entre otros. En términos de las definiciones introducidas en la Sección 4.1.2, una RestriccionAtributo se corresponde con una DatatypeProperty.

En el ejemplo de la Figura 40, *X ∃ "Francia"* (X EXACTAMENTE_IGUAL Francia), X es la variable, EXACTAMENTE_IGUAL es la restricción y "Francia" el valor literal.

Las variables utilizadas en una RestriccionAtributo pertenecen al conjunto de las variables definición de esa consulta. Es decir, las variables involucradas en una restricción deben ser definidas previamente.

RestriccionRelacion se compone de dos variables y una relación que las vincula. Ambas variables deben pertenecer al conjunto de las variables definición. La relación puede ser una de las siguientes: subRegionOf, es_capital, abreviacionOrganizacion, y las demás mencionadas en la Tabla 7, dependiendo de las variables involucradas. RestriccionRelacion se corresponde con una ObjectProperty (ver Sección 4.1.2).

En el ejemplo de la Figura 40, *Y subRegionOf X*, Y es la variable izquierda, X es la variable derecha y subRegionOf es la relación, que indica que Y es subregión de X.

Como se muestra en el diagrama de clases siguiente, la clase ConsultaResultado está compuesta de un conjunto de documentos (ConsultaResultadoItem).

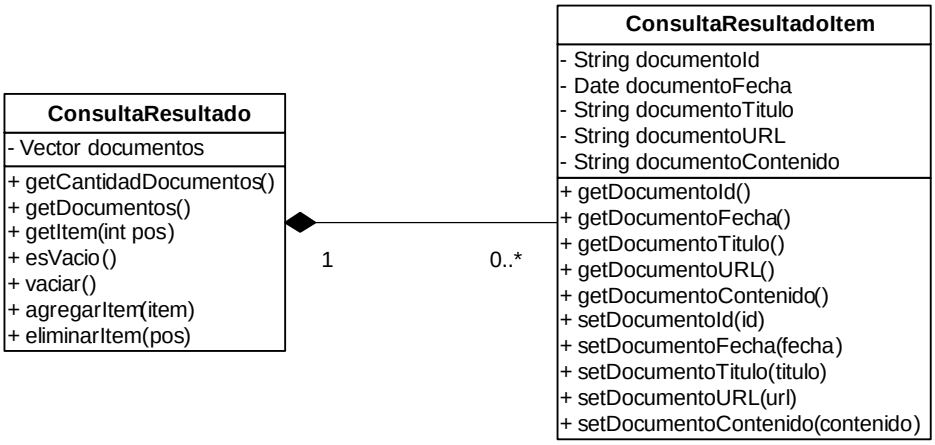


Figura 43: ConsultaResultado

ConsultaResultado permite agregar, eliminar y obtener ítems, entendiendo por ítem a un documento de la colección.
Cada ítem consta de: un identificador (obtenido luego del proceso de anotación), última fecha de modificación, título, URL y el contenido textual. El identificador es utilizado internamente por los controladores para el manejo de cada documento. Los demás atributos son desplegados vía interfaz gráfica cuando se devuelven los resultados al usuario.

5.3. Anotación de Documentos

Luego de la población de la ontología, y como requisito para la ejecución de las consultas semánticas, es necesario realizar la anotación del corpus de documentos.

Para realizar la anotación de documentos, existe una herramienta brindada por el anotador KIM, pero que no permite anotar entidades con caracteres especiales (tildes, diéresis, etc.). Como el idioma español contiene dichos caracteres, y las entidades y documentos también, esta opción de anotación no fue aceptable.

Es por esto que se creó una pequeña aplicación que permitiera anotar todo tipo de caracteres. Esta herramienta utiliza la API de anotación semántica de KIM.

Entonces, posteriormente a la carga de la ontología, se genera una base de conocimiento (en formato N-TRIPLES) que luego es utilizada por el proceso de anotación de los documentos.

6 Pruebas y Resultados

6.1. Introducción

Esta sección pretende ilustrar los resultados obtenidos con la utilización de las diferentes herramientas: el poblador de ontología (ChoikeBilingüe), el anotador de documentos y el buscador semántico (ChoBiSearch). Estos resultados se analizaron por separado.

En las pruebas realizadas, se utilizó la salida de una etapa anterior como entrada para las pruebas de la etapa siguiente. Es decir, la salida de la población de la ontología (ontología poblada) se utilizó para la anotación de los documentos, y a su vez, la anotación de los documentos generó la metadata necesaria para poder realizar consultas luego con ChoBiSearch.

Para la población de la ontología lo importante es la cantidad de instancias cargadas en la ontología resultado, así como el tiempo de ejecución de la misma. Por lo tanto en las pruebas se observaron tanto el volumen de datos como la performance de la aplicación.

Las pruebas realizadas del anotador semántico, se basaron principalmente en observar el tiempo que insumía para una carga de documentos grande. Resulta interesante saber el tiempo transcurrido en la anotación de un gran volumen de documentos, como es el que se maneja en un portal Web, y más específicamente en Choike.

Para el buscador semántico (ChoBiSearch) es interesante determinar la fiabilidad de los resultados obtenidos al realizar determinadas consultas que se consideraron relevantes. Es por esto que se obtuvieron los resultados, se compararon con una búsqueda terminológica (por palabra clave) y para determinar la calidad de los resultados se utilizaron los indicadores mencionados en la Sección 2.7: *Recall* y *Precision*.

6.2. Población de la ontología

Para la población de la ontología, se comenzó con una ontología inicial vacía, sin instancias, y se corrió el proceso de carga a partir de los diccionarios.

Como se mencionó previamente, los diccionarios a cargar fueron: regiones, países, ciudades, siglas, cargos, personas, personaCargo, eventos, organizaciones, documentos y abreviaciones. Estos diccionarios constan de una gran cantidad de datos, lo que se traduce en un número de instancias considerable en la ontología.

Los resultados obtenidos al correr la aplicación se detallan en la Tabla 9. Esta información se presenta discriminada por clase:

Clase	Nº instancias
Sigla	1284
Evento	670
Cargo	48
Organizacion	1331
Persona	1020

Pais	233
Region	39
Ciudad	1450
Documento	301
Total	**6376**

Tabla 9: Instancias cargadas en la ontología

El proceso de carga de las 6376 instancias insumió solamente 33 segundos en una computadora con las características de hardware que se mencionarán en la Sección 6.5.

La ontología generada consta de una cantidad de datos importante y se considera que el tiempo insumido para la carga de la misma es despreciable. Por lo tanto, la ontología resultado colma las expectativas tanto en cantidad de instancias como en tiempo insumido en la población.

6.3. Anotación de Documentos

La prueba de anotación se hizo con un muestreo de 1000 documentos, 500 en español y 500 en inglés, los cuales tenían un tamaño promedio de 20KBytes. Dichas anotaciones, se basaron en la KB generada en la población de la ontología y en la base de conocimiento propia de KIM.

El tiempo necesario para llevar a cabo la anotación de los 1000 documentos, fue de 14 minutos 9 segundos, lo que da un promedio de 0.849 segundos por documento.

No se encontró material disponible sobre tiempo insumido en la anotación con otras herramientas automáticas, como para poder comparar estos resultados. Igualmente, se considera que es un buen tiempo tomando en cuenta el volumen de documentos a anotar (1000), y el tamaño de la KB (6370 instancias, más las aproximadamente 187000 instancias propias de KIM).

El tiempo insumido en la anotación de los documentos no fue considerado de suma relevancia debido a que se trata de un proceso que no se realizará asiduamente. Por el contrario, la idea es que se anoten todos los documentos una única vez y solamente se vuelva a utilizar el anotador en caso de haber cambios en documentos o se agreguen otros.

6.4. Búsquedas semánticas

Para entender mejor el resultado de las pruebas, se comienza haciendo una pequeña introducción en la forma de utilizar el buscador, en donde se explica mediante ejemplos las consultas que se pueden realizar y los resultados esperados.

6.4.1. Utilización del buscador

6. 4. 1. 1. Criterios Generales

En todos los casos, las búsquedas se realizan por conceptos, los cuales se especifican mediante las listas desplegables *Clase X, Clase Y* y *Clase Z*. Las categorías sobre las que se permite buscar son:

- País
- Ciudad
- Región
- Organización
- Evento

- Documento
- Persona
- Cargo
- Sigla

Es posible especificar restricciones de diferente grado sobre los nombres de los conceptos. Las mismas son:

Es desconocido	No se especifica restricción alguna.
Es exactamente igual a	Al menos uno de los nombres del concepto debe ser exactamente igual al término ingresado.
Comienza con	Al menos uno de los nombres del concepto debe comenzar con término ingresado.
Termina con	Al menos uno de los nombres del concepto debe terminar con término ingresado.
Contiene	Al menos uno de los nombres del concepto incluye el término ingresado.

Tabla 10: Restricciones sobre los nombres de los conceptos

Los conceptos están vinculados entre sí mediante relaciones. La Tabla 11 enumera las relaciones que pueden ser especificadas en las consultas.

personaCargo	Permite especificar los cargos que puede tener una persona.
subRegionDe	Relaciona dos ubicaciones entre sí. Permite hacer consultas tales como que una ciudad es subRegionDe un país, un país es subRegionDe una región, etc.
abreviacionOrganizacion	Relaciona una organización con una sigla.
abreviacionEvento	Relaciona un evento con una sigla.
abreviacionDocumento	Relaciona un documento con una sigla.

Tabla 11: Relaciones que pueden tener lugar entre los conceptos

6. 4. 1. 2. Búsqueda Básica

Se denomina Búsqueda Básica a la que incluye un concepto y una restricción sobre su nombre. Las restricciones que se pueden aplicar son las mencionadas en la Tabla 10.

A continuación se presentan ejemplos de búsquedas utilizando las restricciones "Exactamente igual", "Contiene" y "Termina con" y se explican los resultados obtenidos.

- Restricción Exactamente igual

La búsqueda a efectuar es de los documentos que contengan ubicaciones cuyo nombre sea exactamente igual a "Brasil". Esto se expresa seleccionando los valores de las listas desplegables e introduciendo el texto como se muestra en la Figura 44.

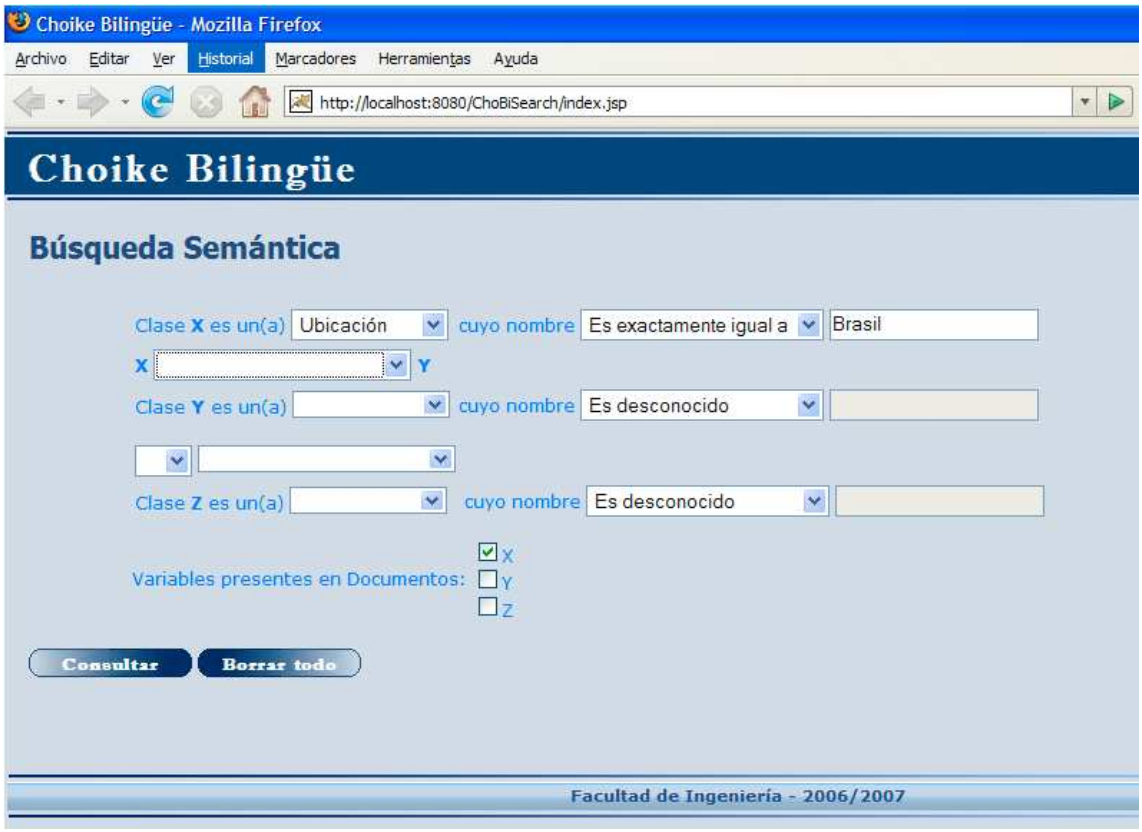


Figura 44: Búsqueda de ubicaciones de nombre "Brasil"

Presionando el botón Consultar, el servidor procesa los valores ingresados y devuelve una pantalla de resultados. Haciendo "click" sobre uno de los documentos obtenidos, se accede al contenido del mismo.

ChoBiSearch devuelve documentos que contienen el nombre de la entidad exactamente como fue ingresado, así como aquellos que contienen algún alias de la misma. Así, se devuelven documentos como el de la Figura 45. Se observa que el texto está en idioma inglés y se ha encontrado la palabra "Brazil", que es uno de los términos en inglés asignados al país Brasil.

Figura 45: Texto del documento de nombre "444", conteniendo el término "Brazil"

Observar que se hizo una búsqueda en idioma español, y se obtuvo como resultado documentos que contienen términos en español e inglés referidos al concepto solicitado.

- Restricción Contiene

La búsqueda a efectuar es de los documentos que contengan ubicaciones cuyo nombre contenga el término "Estados". Esta búsqueda es similar a la anterior, estableciendo "Contiene" como restricción de nombre y "Estados" como texto a buscar.

La consulta devuelve varios documentos, entre ellos el de la Figura 46.

En este documento se han reconocido las ubicaciones "México" y "Brasil". Esto se debe a que una de las posibles denominaciones de México es "Estados Unidos Mexicanos". Análogamente, Brasil también tiene entre sus posibles alias "Estados Unidos do Brasil", por esta razón el buscador lo devuelve.

También se han recuperado documentos en inglés donde el buscador reconoció el país "Estados Unidos" por su alias "US".

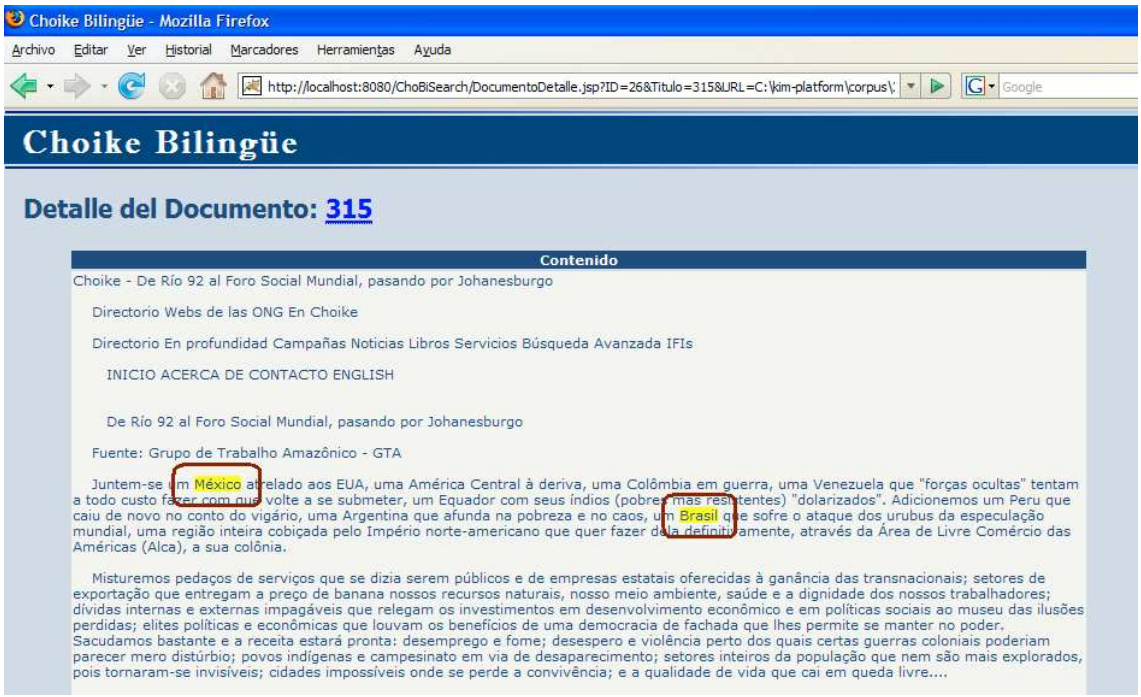


Figura 46: Texto del documento de nombre "315", acerca de ubicaciones cuyos nombres contienen el término "Estados"

- Restricción Termina con

La búsqueda a efectuar es de los documentos que contengan ubicaciones cuyo nombre termine con el término "Sur". Esta búsqueda es similar a las anteriores, estableciendo "Termina con" como restricción de nombre y "Sur" como texto a buscar.

En alguno de los documentos recuperados aparece "República de Corea", la cual satisface la búsqueda porque tiene entre sus alias "Corea del Sur".

6.4.2. Pruebas

Luego de anotados los documentos, se procedió a la explotación de los procesos anteriores: las consultas semánticas.

Las pruebas realizadas en las búsquedas semánticas tienen como resultado la cantidad de documentos devueltos para una determinada consulta. También se considera la relevancia de cada uno de estos documentos según la consulta ingresada, lo que lleva a determinar la calidad de los resultados.

En esta sección se presentan dos tipos de pruebas: el primer tipo pretende ilustrar las ventajas que tiene un buscador semántico sobre uno por palabra clave y el segundo pretende demostrar una deficiencia en la anotación de documentos, que lleva a que en la etapa de las búsquedas se obtengan resultados de mala calidad para determinadas consultas. Estas pruebas se detallan en las secciones 6.4.1 y 6.4.2 respectivamente.

6. 4. 2. 1. Prueba Tipo 1

Se decidió realizar la prueba con un muestreo de 1000 documentos seleccionados aleatoriamente de Choike, salvo por la restricción de que 500 documentos se tomaron en idioma español y 500 en inglés. Estos documentos, son los mismos que se anotaron en la prueba de la Sección 6.4.

La consulta "Pais = Brasil" retornó 190 documentos que contenían dicha entidad. Para poder obtener una estimación de la calidad de la información devuelta por el buscador, se debe tener la información de cuáles son los documentos relevantes para la consulta realizada.

En la Tabla 12 se detallan la cantidad de documentos recuperados, discriminado entre los documentos relevantes reconocidos y los que no, así como también los documentos relevantes y no reconocidos. Los resultados obtenidos por ChoBiSearch son comparados con los resultados de una búsqueda terminológica, como la que brinda el explorador de Windows.

Documentos relevantes: Total = 198			
Pais = Brasil		**ChoBiSearch**	**Búsqueda Terminológica**
Recuperados	Total	190	111
	Relevantes	190	101
	No relevantes	0	10
No recuperados		8	97

Tabla 12: Documentos devueltos

A partir de este resultado se realizó un análisis de la fiabilidad del mismo en base a los medidores *Recall* y *Precision*. Como se mencionó en la Sección 2.7, los indicadores quedan determinados por las siguientes fórmulas:

Recall = relevantes recuperados / relevantes en la colección
Precision = relevantes recuperados / total de documentos recuperados

Análisis de los resultados:

Indicador	ChoBiSearch	Búsqueda Terminológica
Recall	190/198 = **0.96**	101/198 = **0.51**
Precision	190/190 = **1**	101/111 = **0.91**

Tabla 13: Indicadores *Recall* y *Precision* para la consulta

1. La búsqueda semántica reconoció el 96% de los documentos relevantes, mientras que la búsqueda terminológica reconoció el 51%
2. Todos los documentos recuperados por la búsqueda semántica eran relevantes, mientras que el 9% de los documentos devueltos por la búsqueda terminológica no lo eran.

De todo lo anterior se puede observar que la búsqueda semántica resultó mucho más efectiva que la terminológica. Además de que devuelve la mayor cantidad de documentos, no tiene "falsos positivos" (no existe ruido).

El principal problema de la búsqueda terminológica es que no resuelve los problemas de sinonimia y multilingüismo mencionados en la Sección 2.7.2.1. Es decir, no devuelve los documentos que contienen sinónimos del término introducido en la consulta, o el mismo término pero en otro idioma. En lugar del término en sí (Brasil), los documentos contienen por ejemplo el término en inglés (Brazil). Es por esto que en la prueba realizada no se devolvieron 97 de los 198 documentos.

Si bien mediante una búsqueda sintáctica, buscando por cada uno de los alias de la entidad Brasil, se encontrarían los mismos documentos que en la búsqueda semántica, se estaría obligando a que el usuario conociera todos los alias existentes para dicho país. Se está forzando al usuario a tener un mayor conocimiento del dominio del necesario, además de ser una tarea tediosa el ingreso de todos estos datos. La búsqueda semántica devuelve todos los documentos al costo de escribir solamente un nombre de la entidad a buscar.

Otra de las ventajas del buscador semántico es que permite hacer consultas del estilo "Ciudades de Uruguay", es decir, consultar los documentos que mencionan ciudades ubicadas en el Uruguay, sin incluir explícitamente los nombres de dichas

ciudades. En este caso se está explotando la capacidad de inferencia de la ontología definida, donde la clase Ciudad está relacionada con la clase Pais a través de la relación "subRegionOf".

6. 4. 2. 2. Prueba Tipo 2

La siguiente prueba tiene el cometido de analizar el comportamiento del anotador, así como de las consultas semánticas frente a términos iguales, pero que tienen un significado diferente, dependiendo del documento y el contexto en el que se encuentran. Este problema se llama polisemia, y también fue mencionado en la Sección 2.7.2.1.

La idea de la prueba consiste en agregar en la base de conocimiento, tanto una Persona como una Ciudad, que se llamen igual. Para este ejemplo se usó la Ciudad Washington, y la Persona con el mismo nombre.

En los documentos donde se realizaron las consultas, se nombraban en algunos casos a la persona y en otros a la ciudad. De los 1000 documentos, en 99 de ellos se mencionaba el término "Washington".

La Tabla 14 presenta los resultados obtenidos para la consulta Ciudad = Washington, discriminando documentos recuperados y no recuperados. La Tabla 15 muestra los valores de *Recall* y *Precision* obtenidos en la consulta.

Documentos relevantes: Total = 38			
Ciudad = Washington		**ChoBiSearch**	**Búsqueda Terminológica**
Recuperados	Total	65	99
	Relevantes	38	38
	No relevantes	27	61
No recuperados		0	0

Tabla 14: Documentos devueltos (Consulta Ciudad = Washington)

Análisis de los resultados:

Indicador	ChoBiSearch	Búsqueda Terminológica
Recall	38/38 = **1**	38/38 = **1**
Precision	38/65 = **0.58**	38/99 = **0.38**

Tabla 15: Indicadores *Recall* y *Precision* para documentos en la consulta Ciudad = Washington

1. Tanto en la búsqueda semántica como en la terminológica se reconocieron todos los documentos relevantes de la colección
2. El 58% de los documentos recuperados por la búsqueda semántica eran relevantes, mientras que el 62% de los documentos devueltos por la búsqueda terminológica no lo eran

La Tabla 16 presenta los resultados de la consulta Persona = Washington, discriminando por documentos recuperados y no recuperados. La Tabla 17 muestra los valores de *Recall* y *Precision* obtenidos en la consulta.

Documentos relevantes: Total = 29			
Persona = Washington		**ChoBiSearch**	**Búsqueda Terminológica**
Recuperados	Total	1	99
	Relevantes	1	29
	No relevantes	0	70
No recuperados		28	0

Tabla 16: Documentos devueltos (Consulta Persona = Washington)

Análisis de los resultados:

Indicador	ChoBiSearch	Búsqueda Terminológica

Recall	1/29 = **0.03**	29/29 = **1**
Precision	1/1 = **1**	29/99 = **0.29**

Tabla 17: Indicadores *Recall* y *Precision* para documentos en la consulta Persona = Washington

1. La búsqueda semántica reconoció solamente el 3% de los documentos relevantes, mientras que la búsqueda terminológica reconoció todos
2. El 100% de los documentos recuperados por la búsqueda semántica eran relevantes, mientras que el 71% de los documentos devueltos por la búsqueda terminológica no lo eran

De todo lo anterior, se puede decir que ambos buscadores son muy malos a la hora de resolver el problema de la polisemia. El buscador sintáctico, a pesar de que resultó un poco mejor que el semántico, siempre devuelve todos los documentos y entidades donde se encuentra el término buscado, a pesar de que no se correspondan con la clase buscada (baja *Precision*).

Analizando el problema de la polisemia, e investigando cómo trata KIM este problema, tanto en las anotaciones como en las consultas, se dedujo que en realidad el problema de los malos resultados en la búsqueda, no se debían al buscador semántico sino a la anotación de los documentos.

Las consultas semánticas no reconocen nada, sino que devuelven los documentos que contienen anotaciones de entidades con determinado nombre (Washington en este caso). Es en la fase de anotación cuando se toma la decisión de si la ocurrencia de un nombre "Washington" hace referencia a una entidad de la clase Persona o a una de Ciudad.

Al momento de anotar un documento, un módulo gazetteer busca la primera entidad que contenga el mismo nombre que el texto del documento (Washington). Si la primera entidad que encuentra es la Persona Washington, entonces el documento queda anotado con esta referencia. En caso contrario se anota con una referencia a la Ciudad Washington. Cuál será la entidad reconocida en el documento, depende de cómo Sesame (la plataforma que administra el repositorio semántico) carga las instancias y cuál de ellas será ingresada primero en las listas gazetteer.

En conclusión, no existe una regla para saber con qué entidad será anotado un término en un documento, en el caso de existir varias entidades con el mismo nombre. Eso sí, si las entidades fueran anotadas correctamente, el buscador semántico se comportaría del mismo modo que en la Prueba Tipo 1. El problema de la polisemia entonces, debería atacarse en la etapa de anotación, quizás permitiendo de alguna forma cambiar las anotaciones hechas en los documentos, para evitar estos problemas de ambigüedad. En el Capítulo 7, como una mejora al producto, se describe la forma de solucionar este problema.

6.5. Plataforma

Esta sección describe la plataforma en que corrió el servidor donde se ejecutaron las pruebas.

Durante el proyecto se realizaron pruebas sobre dos equipos diferentes con las características enumeradas en las Tabla 18.

CPU	Intel Celeron M 1.6 GHz Genuine Intel T1350 1.86 GHz
Memoria RAM	1 GB
Sistema operativo	Windows XP Home
JVM	Versión 1.5.0_08

Servidor de aplicaciones	Apache Tomcat 5.5.17

Tabla 18: Computadora utilizada

Respecto a las herramientas utilizadas, se enumeran las versiones instaladas de cada una de ellas:

KIM Platform	Versión 1.6.8.14
dom4j	Versión 1.6.1
Jena	Versión 2.4

Tabla 19: Herramientas utilizadas

Por último, los navegadores en los cuales se probó la aplicación de búsquedas semánticas fueron Firefox e Internet Explorer, aunque no se descarta que funcione correctamente para otros navegadores como Netscape y Opera.

En la Tabla 20 se describen las versiones de navegadores en las cuales se ejecutó la aplicación.

Mozilla Firefox	Versiones 1.5.0.11 y 2.0.0.3
Internet Explorer	Versiones 6.0 y 7.0

Tabla 20: Navegadores utilizados

7 Conclusiones

El objetivo del presente proyecto se puede expresar sintéticamente como efectuar recuperación de información bilingüe en la Web Semántica sobre un dominio en particular.

En la etapa inicial se estudiaron y analizaron los conceptos y herramientas relacionados a la Web Semántica. Entre otros temas, se investigaron las herramientas y lenguajes para la definición y manipulación de ontologías. Se incorporaron posteriormente al estudio, los anotadores semánticos, que pueden ser considerados como críticos por su escasez y poca madurez. La elección de esta herramienta fue fundamental, debido a los requisitos particulares que debía cumplir para ser de utilidad.

Este proyecto tuvo una fuerte componente de investigación de temáticas que están en pleno desarrollo. Se analizaron varias herramientas, muchas de las cuales no estaban totalmente maduras o contaban con escasa o nula documentación. Resultó muy interesante introducirse en un tema de constante crecimiento, a pesar de que lo incipiente de su desarrollo fue a veces un obstáculo.

Posteriormente, en la etapa de desarrollo, se creó una ontología bilingüe para estructurar los conceptos, relaciones y propiedades del dominio de Choike. Luego, se trabajó en la creación automática de la base de conocimiento asociada a esta ontología, la cual es cargada a partir de diccionarios.

La anotación semántica de documentos, actividad fundamental para la concreción de la Web Semántica, se basó en los productos obtenidos en los pasos anteriores: la ontología bilingüe y la base de conocimiento. En esta etapa se anotaron los documentos proporcionados por el sitio Web elegido para el estudio. Finalmente, se procedió a la explotación de la metadata generada en la etapa anterior, mediante la implementación de un buscador semántico, que permite recuperar información relevante de las páginas del sitio.

Como principal resultado del trabajo, se obtuvo un producto capaz de efectuar búsquedas semánticas sobre el dominio elegido. Para medir la calidad de las mismas, se compararon los resultados obtenidos con los de un buscador por palabra clave. A tales efectos, se analizaron los indicadores de Recall y Precision para los resultados de ambos buscadores, y de acuerdo a los valores obtenidos se puede afirmar que el buscador semántico es superior en ambos aspectos. En particular, debido a que los documentos están dotados de significado, por contener anotaciones que hacen referencia a entidades de la ontología, el buscador semántico introduce escaso o ningún "ruido" a los resultados.

Otra de las ventajas del buscador semántico sobre el terminológico, es que resuelve en forma satisfactoria los problemas de multilingüismo y sinonimia. Es decir, devuelve documentos que contienen las entidades buscadas, referidas por su nombre en inglés o por denominaciones secundarias en español.

El buscador semántico permite hacer consultas que explotan la capacidad de inferencia de la ontología definida. Es el caso de la consulta "Ciudades de Uruguay", que sin incluir explícitamente nombres de ciudades en la misma, devuelve el conjunto de documentos relevantes asociado.

En base al objetivo planteado al comienzo del proyecto y a un análisis general del trabajo realizado, se puede concluir que el mismo ha sido cumplido satisfactoriamente.

7.1. Mejoras y Trabajo a Futuro

A lo largo del proyecto se identificaron algunos puntos que podrían ser mejorados, así como errores detectados en su construcción. Asimismo se presentan trabajos futuros sugeridos y pueden servir de guía para la evolución de la aplicación desarrollada.

A continuación se mencionan los puntos considerados más relevantes.

7.1.1. Consideraciones generales

La ontología construida pasó por dos etapas: la creación inicial, la cual se basó en la ontología del proyecto OntoChoike y su evolución a partir de la elección del anotador. Este proceso se describe con mayor detalle en el Capítulo 4.

La evolución de la ontología se trató, básicamente, de su adaptación para extender la ontología de KIM. Esta herramienta ya incluye una base de conocimiento que consta de una gran cantidad de instancias. A su vez, se contaba con la KB construida para la ontología ChoikeBilingüe, la cual podía tener instancias definidas en común con la KB de KIM. En resumen, existían instancias definidas en ambos repositorios de información.

Esto provocaba que a la hora de anotar un documento no estuviera determinado exactamente con qué entidad sería anotado, un término en un documento podía ser vinculado a cualquiera de las dos entidades que representan al mismo concepto.

Por ejemplo, en KIM estaba definida la entidad Location Egypt, que representa al país Egipto. A su vez, ChoikeBilingüe tiene definido el País Egipto, con alias Egypt. Entonces, los documentos que contienen el término Egypt pueden ser anotados indistintamente con cualquiera de las dos entidades. Además de no ser totalmente correcto conceptualmente, esta característica presenta problemas en el momento de ejecutar las consultas. Si se consulta por la clase País=Egypt, sólo devuelve los documentos anotados con la instancia de País, no con la de Location.

Se evaluaron diferentes soluciones para este problema:

- Quitar las instancias propias de KIM y solamente utilizar las de ChoikeBilingüe. Esta opción tiene como desventaja que se desaprovecha una KB muy rica en información; se constató en la práctica que la pérdida era significativa.
- Quitar las instancias propias de ChoikeBilingüe y utilizar únicamente la KB de KIM. Esta opción no es viable por no contar con los alias en español, ya que la KB de KIM sólo contiene alias en inglés.
- Editar la KB de KIM para eliminar las instancias ya definidas en ChoikeBilingüe. Este trabajo debía ser realizado en forma manual, lo cual lo hacía inviable debido a la cantidad de información contenida en el repositorio de KIM. No se encontró la manera de automatizar esta tarea.
- Por último, utilizar ambas bases de conocimiento. Como se mencionó anteriormente, esta solución no está exenta de inconvenientes, ya que conlleva problemas para las consultas. Se encontró el workaround de hacer referencia a la clase padre (perteneciente a KIM) en las consultas, observando que era una solución satisfactoria al problema. Es decir, una consulta que hace referencia al País Egypt se traduce como la consulta que refiere a la Location Egypt. De esta forma, se devuelven los documentos que contienen alguna de las dos entidades, Location o País (subclase de Location).

Finalmente, se decidió por la última opción por tener costo mínimo, no tener grandes desventajas y seguir aprovechando ambas bases de conocimiento. Como trabajo a futuro, se podría considerar el análisis de una solución conceptualmente

más adecuada, que no fue posible realizar en esta oportunidad por restricciones de tiempo.

7.1.2. Herramienta de anotación de documentos

Anotación recursiva en directorios

La herramienta solicita la ruta del directorio en el que se encuentran los documentos y anota únicamente aquellos ubicados bajo el mismo, sin considerar los documentos que se puedan encontrar en sus subdirectorios. Esto puede ser un problema si las páginas a anotar están distribuidas en una estructura de directorios. Se podría dar la opción de anotar el contenido de la carpeta especificada y de todas sus subcarpetas.

Se considera que esta optimización no es muy costosa y podría llevar a lo sumo un par de días. No se realizó dicha implementación debido a que no era un requisito, se priorizaron otras mejoras y la verificación de la aplicación en general.

Tratamiento de la polisemia

El diseño original de KIM fue pensado para realizar anotaciones automáticamente, dejando en manos del anotador la decisión de qué clase es cada término. Por esta razón no existe ninguna funcionalidad brindada por KIM que permita editar la clase a la cual pertenece una entidad anotada en un documento.

Según el equipo de soporte de KIM, existe la posibilidad de hacerlo manualmente, utilizando las APIs provistas. La solución propuesta es dado un documento que contiene una entidad anotada erróneamente, recorrer mediante código todas las anotaciones del mismo hasta encontrar la anotación en cuestión, eliminarla y agregar una nueva anotación con la clase correcta. Por último, se debe sincronizar el documento modificado para actualizarlo en la metadata.

Más específicamente, dirigido a los desarrolladores que puedan extender esta aplicación, se debe obtener el documento utilizando la función getGateDocument() de la clase KIMDocumentImpl. Una vez obtenido el mismo, se deben recuperar las anotaciones utilizando la función getAnnotations(). Para editar alguna de sus anotaciones asociadas, no alcanza con modificarla sino que es necesario eliminarla, crear una nueva y agregarla al conjunto. Una vez completado este proceso, se debe sincronizar el documento a través de la función brindada por la API de persistencia.

7.1.3. Buscador semántico ChoBiSearch

Lectura dinámica de las entidades

La versión actual del buscador semántico permite el armado de consultas mediante la selección de clases y restricciones utilizando combos. Los mismos son estáticos y contienen los nombres de las clases definidas en la ontología. Se podría dotar de mayor flexibilidad a la herramienta haciendo que los mismos se carguen dinámicamente desde un archivo, o mejor aún, desde la ontología, utilizando Jena o alguna API provista por KIM.

El costo asociado a la implementación de esta mejora no es significativo para el caso de la carga desde archivo. Para el segundo caso, se tiene el costo de la lectura de la ontología, que es mayor que el primero y depende en gran medida de las funcionalidades ofrecidas por las APIs.

Visualización de la ontología

A los efectos de ofrecer una aplicación más amigable al usuario, se podría diseñar una interfaz que muestre la taxonomía de la ontología tal como lo hace Protégé. Esa vista de la ontología permitiría el armado de las consultas mediante

interacciones drag and drop, seleccionando las clases y propiedades que intervienen en las mismas.

Esta mejora tiene un costo considerable, debido a que se necesita leer la ontología utilizando funcionalidades disponibles, seguida de la implementación de una interfaz gráfica compleja.

Consultas predefinidas

Sería de utilidad contar con un conjunto de consultas ya definidas, de uso común, para facilitarle el trabajo de armado de las mismas al usuario. Por ejemplo: ciudades ubicadas en un país, personas con determinado cargo, entre otros.

Una vez seleccionada la consulta predefinida, se presenta al usuario un cuadro de diálogo donde debe ingresar los filtros correspondientes a la misma. Si eligió la consulta ciudades ubicadas en un país, debe ingresar un nombre de país.

Esta utilidad se podría implementar sin demasiado costo, ya que las consultas predefinidas utilizarían los mismos servicios que se usan para las consultas dinámicas ya implementadas.

Restricciones

Las restricciones se utilizan para realizar filtros sobre los nombres de entidades. Las restricciones brindadas por la aplicación son: es desconocido, es exactamente igual, comienza con, termina con y contiene. En la Figura 47 se puede observar una consulta donde se elige el filtro a aplicar.

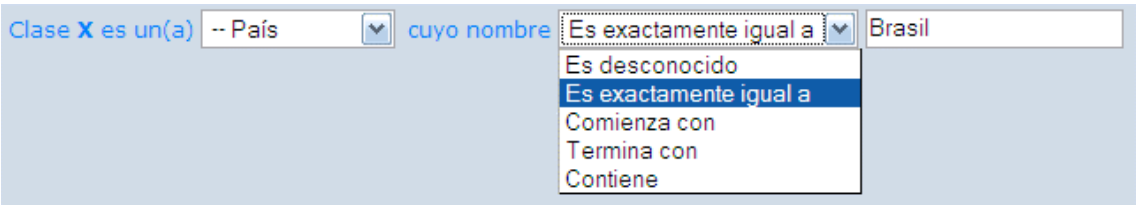


Figura 47: Filtros de las consultas

- La restricción "es desconocido" es utilizada para permitir cualquier valor en el nombre de una entidad. Es decir, para el caso de la Figura 47 se puede utilizar este filtro si se desea obtener todos los países. La consulta definida sería "País" cuyo nombre "es desconocido".
  Esta restricción no se comporta correctamente, ya que devuelve menos documentos de los que debería. Estudiando la traza del error, se detectó que es una falta que proviene de la API de KIM utilizada por el buscador semántico para ejecutar las consultas, y no del buscador en sí.
- La API de KIM también provee restricciones de "mayor que" y "menor que", para el filtrado de los nombres. Las mismas no pudieron ser incluidas en la solución, debido a que presentaban un comportamiento erróneo. Este bug fue reportado a la organización Ontotext [76], por lo cual es posible que surja una solución al problema en futuras versiones de KIM.

8 Glosario

Anotación: Proceso de asociar metadatos a un recurso, que lo describen en todo o en parte.

Anotación semántica: Anotación en la que los metadatos están definidos formalmente y son procesables por máquina.

API: Application Programming Interface. Conjunto de especificaciones de comunicación entre componentes de software. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general.

Bug: Es un error de software que pude provocar una salida incorrecta o un resultado inesperado.

Cliente: Cualquier programa que utilice los servicios de otro. En la Web, un cliente Web es un programa, tal como un navegador, editor o robot de búsqueda, que lea o escriba información en la Web.

DatatypeProperty: Son propiedades que vinculan un individuo con un literal.

Gate: Plataforma de procesamiento de lenguaje natural y extracción de información, desarrollada en la Universidad de Sheffield en colaboración con Ontotext.

Hiperónimo: Palabra cuyo significado engloba el de otras más específicas. Ejemplo: "mueble" es un hiperónimo de "silla".

IDE: Entorno integrado de desarrollo (Integrated development environment).

Índice Inter-Lingua: Este índice permite interconectar los lenguajes de forma tal que es posible ir de una palabra en un lenguaje a palabras similares en otros lenguajes.

Inferencia: En el contexto de lenguajes de razonamiento automático y ciencia de la computación en general, se llama inferencia a la derivación de nuevos datos a partir de información existente. En general, se puede definir como el proceso de derivar conclusiones lógicas a partir de premisas que se sabe o se asumen verdaderas.

Información: Significado que el ser humano le asigna a los datos utilizando las convenciones conocidas y generalmente aceptadas.

KIM (Knowledge and Information Management) provee una infraestructura y servicios para anotación semántica automática, indexado y recuperación de contenidos semi o no estructurados.

Knowledge Base: Una Base de Conocimiento o KB, es un tipo especial de base de datos para la administración del conocimiento. Provee medios para recolección, organización y recuperación del conocimiento. En el contexto de este trabajo, se puede ver como la ontología poblada.

Lenguaje de Razonamiento: Es un lenguaje capaz de expresar razonamiento y por tanto realizar inferencia. También se lo refiere como lógico, deductivo o basado en reglas.

Lista Gazetteer: Lista de entidades que se utiliza muy frecuentemente para reconocimiento de las instancias nombradas. Dicha lista puede contener referencias geográficas, tales como nombres de ciudades, países, etc., así como también nombres propios de personas y organizaciones.

Lucene: Motor de Recuperación de Información de código abierto, desarrollado por Apache.

Metadato: Dato que describe otro dato. En el campo de la Web Semántica los metadatos se utilizan para describir recursos.

Módulo Gazetteer de KIM: Es un componente utilizado en el proceso de anotación semántica para buscar alias de entidades en un documento. Dichos alias fueron previamente catalogados como entidades "Trusted" en la KB. Su rol principal es proveer una vía rápida para la anotación de entidades con sus respectivos alias.

N-TRIPLES: Formato desarrollado por W3C, que permite representar modelos RDF.

ObjectProperty: Son propiedades que vinculan un individuo con otro.

Ontología: Vocabulario formal de los conceptos relevantes en un dominio, las propiedades que los relacionan y, quizás, las reglas que rigen el funcionamiento de dicho dominio.

Ontología multilingüe: está definida por una ontología y un conjunto de diccionarios (un diccionario por cada lenguaje). La misma contiene un conjunto de conceptos identificados C, relacionados por un conjunto de relaciones R (Guyot).

Open Source: Software cuyo código es distribuido y modificado libremente.

OWL: Lenguaje de Ontologías Web (Ontology Web Language). Lenguaje de definición de ontologías para la Web.

OWLIM: Denominación para OWLMemSchemaRepository SAIL (Storage and Inference Layer) para Sesame. Provee infraestructura de almacenamiento y consulta.

PLN: Procesamiento de Lenguaje Natural. Es una subdisciplina de la Inteligencia Artificial, que estudia métodos y técnicas eficientes desde un punto de vista computacional para la comprensión y generación de lenguaje natural.

***Precision*:** Cantidad de documentos relevantes recuperados sobre el total de documentos obtenidos.

Razonamiento: Método para derivar conocimiento (potencialmente nuevo) a partir de hechos (positivos o negativos).

RDF: Marco para Descripción de Recursos (Resource Description Framework). Es un lenguaje para la definición de ontologías y metadatos en la Web.

RDFS: Esquema RDF (Resource Description Framework Schema). Permite definir jerarquías de clases de recursos, especificando las propiedades y relaciones que se admiten entre ellas.

***Recall*:** Cantidad de documentos relevantes recuperados sobre el total de documentos relevantes.

Recuperación de Información: Dada una necesidad de información a través de una consulta, para un perfil de usuario, y un conjunto de documentos, ordenar los documentos de más a menos relevantes para esa necesidad, y presentar un subconjunto de los más relevantes. (Baeza-Yates)

Recurso: Cualquier cosa que pueda ser interesante describir en una aplicación de Web Semántica. Una página Web, un correo electrónico, un fichero, son recursos, pero también lo son una persona, un coche, incluso una idea. Los recursos se identifican por medio de URIs.

Red Semántica: Es un mecanismo para representar conceptos mediante grafos etiquetados dirigidos, compuestos de un conjunto de nodos y un conjunto de enlaces unidireccionales.

Servidor: Programa que ofrece un servicio (típicamente información) a otro programa denominado cliente. Un servidor Web contiene páginas Web y permite a los clientes leer o escribir en ellas.

Sesame: Framework Java de código abierto para almacenamiento, consulta y razonamiento con RDF y RDF Schema. Puede ser usado como una base de datos para los mismos, o como una biblioteca Java para aplicaciones que necesiten trabajar internamente con RDF. Fue desarrollado por Aduna y Ontotext.

SGML: Lenguaje de Marcación Generalizado (Standard Generalized Markup Language). Estándar para lenguajes marcado de, base de HTML y precursor de XML.

Singleton: Es un patrón de diseño de software, el cual establece una restricción en la creación de objetos pertenecientes a una clase o el valor de un tipo, a un único objeto.

Sistemas de Anotación Manual: Sistemas que, típicamente, proporcionan una interfaz de usuario que permite a los anotadores humanos ver y navegar simultáneamente las ontologías y los recursos Web, usando el conocimiento modelado en las ontologías para agregar anotaciones a los mismos.

Sistemas de Anotación Automáticos: Son herramientas de anotación que aplican técnicas de extracción de información de lenguaje natural para generar automáticamente anotaciones en páginas Web.

Taxonomía: Ciencia que trata de los principios, métodos y fines de la clasificación. Se aplica en particular, dentro de la biología, para la ordenación jerarquizada y sistemática, con sus nombres, de los grupos de animales y de vegetales.

URI: Identificador de Recursos Uniforme (Uniform Resource Identifier). Es el formato utilizado en la Web Semántica para asociar identificadores a recursos.

Usuario: Persona que hace uso del sistema.

W3C: World Wide Web Consortium. Organismo encargado de desarrollar estándares para la Web.

Web Semántica: Es una extensión de la Web actual en la cual la información recibe un significado bien definido, permitiendo a las computadoras y las personas trabajar en cooperación de mejor forma. (Tim Berners-Lee y cols.).

XML: Lenguaje de Marcas Extensible (eXtensible Markup Language). Es una especificación que define una forma estándar de agregar marcas a un documento. Es un lenguaje para documentos que contienen información estructurada.

9 Referencias Bibliográficas

[1] Joost Breuker, Abdullatif Elhag, Emil Petkov y Radboud Winkels. "IT Support for the Judiciary: Use of Ontologies in the e-Court Project"
University of Amsterdam, Department of Computer Science and Law (LRI), Amsterdam, The Netherlands.

[2] Paul Buitelaar, Diana Steffen, Martin Volk, Dominic Widdows, Bogdan Sacaleanu, Špela Vintar, Stanley Peters, Hans Uszkoreit
"Evaluation Resources for Concept-based Cross-Lingual Information Retrieval in the Medical Domain".

[3] Natalia Chiaro, Pablo Damonte. Año 2004-2005. "Generación de descripciones de Web Semántica en una red de ONG". Facultad de Ingeniería, Universidad de la República Oriental del Uruguay.

[4] Guía Breve de Web Semántica
http://www.w3c.es/Divulgacion/Guiasbreves/WebSemantica
Último Acceso: 11/07/2006

[5] Tutorial: Semantic Web
http://www.w3schools.com/semweb/default.asp
Último Acceso: 12/07/2006

[6] Revista InfoVis.Net: La Web Semántica, hoy
http://www.infovis.net/printMag.php?num=131&lang=1
Último Acceso: 12/07/2006

[7] Estudios de Lingüística Española (M. Chantal Pérez, Universidad de Málaga)
http://elies.rediris.es/elies18
Último Acceso: 18/07/2006

[8] Thomas Gruber 1993. "Toward Principles for the Design of Ontologies Used for Knowledge Sharing".
http://www.cise.ufl.edu/~jhammer/classes/6930/XML-FA02/papers/gruber93ontology.pdf
Último Acceso: 19/07/2006

[9] Gruber, T. R. (1993). A Translation Approach to Portable Ontology Specifications.
Último Acceso: 25/01/2007

[10] Ontologías
http://www.hipertexto.info/documentos/ontologias.htm
Último Acceso: 26/07/2006

[11] Artículo: Evolución de los lenguajes utilizados en la construcción de la Web Semántica. (Néstor Darío Duque Méndez, Julio César Chavarro Porras, Ricardo Moreno Laverde).
Último Acceso: 22/04/2007

[12] Jacques Guyot, Saïd Radhouani, Gilles Falquet. "Ontology-Based Multilingual Information Retrieval"

[13] Atanas Kiryakov, Borislav Popov, Ivan Terziev, Dimitar Manov, Damyan Ognyanoff. "Semantic Annotation, Indexing, and Retrieval".

[14] CMSA: A Method for Construction and Maintenance of Semantic Annotations
http://cs.nju.edu.cn/people/gchen/paper/papers/ISPA2005-han.pdf
Último Acceso: 24/01/2007

[15] Ricardo Baeza-Yates. "Recuperación de la Información: Modelos, Estructuras de Datos, Algoritmos y Búsqueda en la Web."

[16] Marcos Campal, Eduardo Fraschini. Año 2004. "MegúSearch (Meta Motor de Búsqueda)". Facultad de Ingeniería, Universidad de la República Oriental del Uruguay.

[17] Artículo: Manejo de Ontologías en Sistemas Multiagentes por medio de un Agente de Ontologías aplicado a JITIK
http://homepages.mty.itesm.mx/ceballos/docs/MIT_Ceballos.pdf
Último Acceso: 30/07/2006

[18] Methontology
http://rhizomik.net/~roberto/thesis/html/Methodology.html
Último Acceso: 06/08/2006

[19] *TOronto Virtual Enterprise* (TOVE)
http://www.eil.utoronto.ca/enterprise-modelling/entmethod/index.html
Último Acceso: 31/07/2006

[20] Problemática y Tendencias en la Arquitectura de Metadatos Web (Pedro Manuel Díaz Ortuño) Facultad de Ciencias de la Documentación. Universidad de Murcia.
http://www.um.es/fccd/anales/ad06/ad0603.pdf
Último Acceso: 19/07/2006

[21] Artículo: WebODE: an integrated workbench for ontology representation, reasoning and Exchange. (Óscar Corcho, Mariano Fernández-López, Asunción Gómez-Pérez, Óscar Vicente).
http://www.cs.man.ac.uk/~ocorcho/documents/EKAW2002_CorchoEtAl.pdf
Último Acceso: 31/07/2006

[22] Protégé
http://protege.stanford.edu/
Último Acceso: 31/07/2006

[23] Jena – A Semantic Web Framework for Java
http://jena.sourceforge.net/
Último Acceso: 11/03/2007

[24] Simple API for XML
http://www.megginson.com/downloads/SAX/
Último Acceso: 15/04/2007

[25] Semantic Matching: Formal Ontological Distinctions for Information Organization, Extraction, and Integration (Nicola Guarino)
http://old.ulstu.ru/people/SOSNIN/umk/Basis_of_Artificial_Intelligence/mirrors/www.ladseb.pd.cnr.it/infor/ontology/Papers/SCIE97.pdf
Último Acceso: 19/07/2006

[26] Pellet: A Practical OWL-DL Reasoner. (Evren Sirin, Bijan Parsia , Bernardo Cuenca Grau, Aditya Kalyanpur, Yarden Katz).

[27] OntoKnowledge
http://www.ontoknowledge.org/
Último Acceso: 24/04/2007

[28] KAON2
http://kaon2.semanticweb.org/
Último Acceso: 24/04/2007

[29] SPARQL
http://www.w3.org/TR/rdf-sparql-query/
Último Acceso: 24/04/2007

[30] DOM
http://www.w3.org/DOM/
Último Acceso: 24/04/2007

[31] SAX
http://www.saxproject.org/
Último Acceso: 24/04/2007

[32] JDOM
http://www.jdom.org/
Último Acceso: 24/04/2007

[33] JAXP
http://java.sun.com/webservices/jaxp/
Último Acceso: 24/04/2007

[34] dom4j
http://www.dom4j.org/
Último Acceso: 24/04/2007

[35] SemanticWord
http://mr.teknowledge.com/daml/SemanticWord/SemanticWord.htm
Último Acceso: 24/04/2007

[36] SMORE
http://www.mindswap.org/2005/SMORE/
Último Acceso: 24/04/2007

[37] YAWAS
http://www.fxpal.com/people/denoue/yawas/
Último Acceso: 24/04/2007

[38] MnM
http://www.fxpal.com/people/denoue/yawas/
Último Acceso: 24/04/2007

[39] KIM
http://www.ontotext.com/kim/index.html
Último Acceso: 24/04/2007

[40] CYC
http://www.cyc.com/
Último Acceso: 26/04/2007

[41] RDQL
http://www.w3.org/Submission/RDQL/
Último Acceso: 24/04/2007

[42] NetBeans
http://www.netbeans.org/
Último Acceso: 12/05/2007

[43] Apache Software Foundation
http://tomcat.apache.org/
Último Acceso: 12/05/2007

[44] World Wide Web Consortium (W3C)
http://www.w3.org/
Último Acceso: 13/05/2007

[45] A Comparative Study of Ontology Languages and Tools (Xiaomeng Su, Lars Ilebrekke)
http://www.vf.utwente.nl/~xsu/paper/caise02WorkshopCRC.pdf
Último Acceso: 15/05/2007

[46] Evaluating Knowledge Representation and Reasoning Capabilities of Ontology Specification Languages (Oscar Corcho, Asunción Gómez-Pérez)
http://www.cs.man.ac.uk/~ocorcho/documents/ECAI00WS_CorchoGomezPerez.pdf
Último Acceso: 15/05/2007

[47] Ontolingua
http://www.ksl.stanford.edu/software/ontolingua/
Último Acceso: 15/05/2007

[48] OCML
http://kmi.open.ac.uk/projects/ocml/
Último Acceso: 15/05/2007

[49] SHOE
http://www.cs.umd.edu/projects/plus/SHOE/
Último Acceso: 15/05/2007

[50] XML
http://www.w3.org/XML/
Último Acceso: 15/05/2007

[51] XML Schema
http://www.w3.org/XML/Schema
Último Acceso: 15/05/2007

[52] RDF
http://www.w3.org/RDF/
Último Acceso: 15/05/2007

[53] RDF Schema
http://www.w3.org/TR/rdf-schema/
Último Acceso: 15/05/2007

[54] OIL
http://www.ontoknowledge.org/oil/
Último Acceso: 15/05/2007

[55] DAML
http://www.daml.org/
Último Acceso: 15/05/2007

[56] DAML+OIL
http://www.daml.org/2001/03/daml+oil-index
Último Acceso: 15/05/2007

[57] OWL
http://www.w3.org/TR/owl-features/
Último Acceso: 15/05/2007

[58] Enterprise Ontology
http://www.aiai.ed.ac.uk/project/enterprise/enterprise/ontology.html
Último Acceso: 15/05/2007

[59] Enterprise Project
http://www.aiai.ed.ac.uk/project/enterprise/
Último Acceso: 15/05/2007

[60] Artículo: The Enterprise Ontology (Mike Uschold, Martin King, Stuart Moralee and Yannis Zorgios)
http://www.eee-con.de/german/information/98-ker-ent-ontology.pdf
Último Acceso: 15/05/2007

[61] OilEd
http://oiled.man.ac.uk/
Último Acceso: 15/05/2007

[62] Racer Systems
www.racer-systems.com
Último Acceso: 15/05/2007

[63] Pellet
http://pellet.owldl.com/
Último Acceso: 15/05/2007

[64] FaCT++
http://owl.man.ac.uk/factplusplus/
Último Acceso: 15/05/2007

[65] Annotea
http://www.annotea.org/
Último Acceso: 15/05/2007

[66] Annotea
http://www.tejedoresdelweb.com/307/article-5817.html
Último Acceso: 15/05/2007

[67] Annozilla (Annotea on Mozilla)
http://annozilla.mozdev.org/
Último Acceso: 15/05/2007

[68] OntoMat Annotizer
http://annotation.semanticweb.org/ontomat/index.html
Último Acceso: 15/05/2007

[69] Búsqueda de información multilingüe: estado del arte (Fernando López-Ostenero, Julio Gonzalo, Felisa Verdejo).

[70] Melita
http://www.aktors.org/technologies/melita
Último Acceso: 19/05/2007

[71] Ontogloss: An Ontology-Based Annotation Tool
http://linguistlist.org/emeld/workshop/2005/papers/mostowfi-paper.html
Último Acceso: 19/05/2007

[72] AeroDAML
http://updates.zdnet.com/tags/AeroDAML.html
Último Acceso: 19/05/2007

[73] Magpie
http://kmi.open.ac.uk/projects/magpie
Último Acceso: 18/05/2007

[74] Evaluación de la recuperación de documentos (Luis Jiménez Cuadrado)
http://evaluacion-ri.uc3m.es/medidas.html
Último Acceso: 16/05/2007

[75] La evaluación en recuperación de la información (Raquel Gómez Díaz)
http://www.hipertext.net/web/pag238.htm
Último Acceso: 16/05/2007

[76] Ontotext
http://www.ontotext.com/index.html
Último Acceso: 21/05/2007

[77] D2.6.1 Massive Automatic Annotation
http://www.sekt-project.org/rd/deliverables/wp02/sekt-d-2-6-1-Massive%20Automatic%20Annotation%20V1.pdf
Último Acceso: 04/06/2007

[78] OWLIM – a Pragmatic Semantic Repository for OWL
http://www.ontotext.com/publications/ssws_owlim.pdf
Último Acceso: 04/06/2007

[79] OWLIM Semantic Repository
http://www.ontotext.com/owlim/index.html
Último Acceso: 05/06/2007

[80] Sesame
http://www.openrdf.org/
Último Acceso: 05/06/2007

[81] GATE – General Architecture for Text Engineering
http://gate.ac.uk/
Último Acceso: 05/06/2007

[82] Lucene
http://lucene.apache.org/java/docs/index.html
Último Acceso: 05/06/2007

[83] Arquitectura de KIM

http://www.ontotext.com/kim/KIMPlatform.pdf
Último Acceso: 05/06/2007