TESIS DOCTORAL

PROCEDIMIENTO SEMI-AUTOMÁTICO PARA TRANSFORMAR LA WEB EN WEB SEMÁNTICA

Luis Criado Fernández

Ingeniero de Telecomunicación

por la Universidad Alfonso X el Sabio

Departamento de Inteligencia Artificial

Escuela Técnica Superior de Ingeniería Informática

Universidad Nacional de Educación a Distancia (UNED)

Madrid 2009

Departamento de Inteligencia Artificial
Escuela Técnica Superior de Ingeniería Informática
Universidad Nacional de Educación a Distancia (UNED)

PROCEDIMIENTO PARA TRANSFORMAR LA WEB EN WEB SEMÁNTICA

Por: Luis Criado Fernández
Ingeniero de Telecomunicación
por la Universidad Alfonso X el Sabio

Director de la Tesis: Dr. D. Rafael Martínez Tomás

Dedicado a mi hermana Terica y a mi tío Rafa.

Agradecimientos

Esta tesis es el resultado de cuatro años de duro trabajo, donde la dedicación se ha compartido con una vida laboral ajena por completo. Una aventura llena de dificultades que he superado con perseverancia, paciencia, imaginación, entusiasmo y sobre todo, con voluntad.

En primer lugar quiero dar las gracias al Departamento de Inteligencia Artificial de la ETS de Informática de la UNED, en especial a mi director de tesis Dr. Rafael Martínez Tomás, por su dedicación incondicional, por sus consejos para orientar el trabajo y sobre todo por mantener siempre una actitud positiva ante mis propuestas. Considero que nuestras reuniones han sido siempre muy enriquecedoras y me han ayudado enormemente a clarificar conceptos e ideas. Quiero manifestar también mi gratitud al Dr. José Ramón Álvarez Sánchez por su ayuda en la instalación del servidor Linux dedicado a ejecutar nuestro Buscador Semántico.

La voluntad es para mí el principal atributo para concluir con una labor de largo recorrido, pero el desarrollo y fortalecimiento de esta cualidad no solo depende de uno mismo, sino de quienes te rodean. Es por esto que quiero agradecer a mi novia, principal tónico de mi voluntad, su apoyo constante, sus ánimos, y su compresión. Quiero agradecer mucho a Amor –que así se llama ella- el sacrificio de soportarme cuando me pongo tan monotemático y sobre todo, el esfuerzo de adaptar nuestros planes personales a mi horario particular.

A mi familia en general, por su apoyo y sobre todo, por su comprensión al descuidar tantas reuniones familiares. Mis padres y hermanos es otro de los tónicos que me han servido para hacer de la voluntad un aliado y no un enemigo. Agradezco especialmente a mis hermanos, a Alfonso, Alberto y Belén. A mi madre, siempre atenta a los avatares y preocupaciones de sus hijos; pero también empatizando con nuestros sueños, y esto en mi caso ha sido una bendición. A mi padre, mi mejor confidente, quien ha leído tantos borradores, quien ha estado al tanto de cada pequeño progreso, de cada cambio en la exposición. Con quien he compartido mis pensamientos mientras entrenábamos juntos en el gimnasio.

Siempre han sido muy importantes para nosotros, mis tias, pendientes y preocupadas por todo lo que nos ocurre, muchas gracias a todas; Mari, Elvi, Jovi, Pili, Amparito y Cuqui. Pero ahora que redacto estas líneas me viene a la memoria, que desde niño, Elvi siempre me dijo que con esfuerzo todo se logra, y si esta tesis ve ya el fín, desde luego, es un ejemplo de esto que siempre me ha dicho. Tambíen quiero hacer llegar mi agradecimiento a mis tios;

Manolo, Luis y Carmelo quien aún desde la distancia, en aquel momento estaba fuera de España, ha seguido mi trabajo, gracias por tus consejos y por leer tantos borradores. Mi especial gratitud para D. José María Sobrino, que aunque no le conozco en persona, ha tenido siempre la amabilidad de leer y opinar constructivamente sobre los borradores que le he enviado.

En el entorno de mi vida profesional, quiero trasmitir mi agradecimiento al equipo del Área de Innovación Tecnológica del CRTM (Antonio, Andrés, Ricardo, Ioannis y Pablo) por su apoyo e interés por la evolución de mi trabajo.

Cuando se trabaja en una tesis, el doctorando, al menos en mi experiencia, tiene la impresión de que el trabajo está casi terminado en varias ocasiones. Sin embargo, al repasar el texto, probablemente por exceso de perfeccionismo, uno tiende a encontrar deficiencias. De manera, que hay una labor final de pequeños ajustes y correcciones que resultan muy tediosos. Por eso, las últimas lecturas me han parecido tan importantes. Quiero dar las gracias a dos personas que se han sumado a estas últimas lecturas, como son María Teresa Pascual Ogueta y la Dra. María Jesús Taboada Iglesias de la Universidad de Santiago de Compostela. Por último, doy las gracias al resto de amigos que se han interesado por la evolución de mi trabajo.

INDICE

INTRODUCCIÓN Y RESUMEN II

RESUMEN Y APORTACIONES III
ORGANIZACIÓN DE LA TESIS V

CAPÍTULO 1: EVOLUCIÓN DE LA WEB Y SU PROBLEMA ACTUAL 1

1.1. INICIO DE LA WEB 3
1.2. EL PROBLEMA QUE TIENE LA WEB SIN SEMÁNTICA 5
1.3. INICIOS DE LA WEB SEMÁNTICA 6
1.4. LA WEB SEMÁNTICA Y EL PROCESADO DE LENGUAJE NATURAL. 8
1.5. PLATAFORMA TECNOLÓGICA DE LA WEB SEMÁNTICA 9
1.5.1. Estandares, editores y frameworks 9
1.5.1.1. XML 10
1.5.1.2. RDF y RDFS 14
1.5.1.3. Semántica sobre RDF y RDFS 22
1.5.1.4. SPARQL 23
1.5.1.5. OWL 23
1.5.1.6. OWL DL base de la Web Semántica 25
1.5.1.7. Editor ontológico: PROTÉGÉ 26
1.5.1.8. Razonadores DL 36
1.5.1.9. Interfaces de programación 37
1.5.2. Microformatos y vocabularios. 47
1.5.3. Proyectos europeos sobre la WS: NeOn y ASK-IT 48
1.6. ARQUITECTURA DE LA WEB SEMÁNTICA 50
1.7. EL PROBLEMA DE IMPLANTAR LA WEB SEMÁNTICA 52
1.8. ANOTACIÓN SEMÁNTICA. 55
1.8.1 Anotación Manual. 61
1.8.2 Anotación semi-automatizada. 63
1.8.3 Conclusión sobre herramientas de población de ontologías. 65
1.9. CONSECUENCIAS Y OBJETIVOS 66

CAPÍTULO 2: PROPUESTA: POBLACIÓN DE ONTOLOGÍAS EN BASE A VISTAS SEMÁNTICAS 67

2.1. ETAPAS DEL PROCESO DE TRANSFORMACIÓN 68
2.1.1. Identificación de ontologías 70
2.1.2. Extracción de lenguaje natural y análisis morfosintáctico. 71
2.1.3. Interpretación del Lenguaje Natural conducido por ontologías 73
Vistas semánticas 75
Página semántica 76
2.2. ¿CÓMO VINCULAR UNA VISTA SEMÁNTICA CON SU PÁGINA WEB? 78
2.3. MIGRACIÓN DE UN SITIO WEB A UN SITIO WEB SEMÁNTICO 80
2.4. ¿QUIÉN MANTIENE LAS VISTAS SEMÁNTICAS? 83
2.5. ALGUNAS CONCLUSIONES. 84

CAPÍTULO 3: IMPLEMENTACIÓN DE LA HERRAMIENTA DE TRANSFORMACIÓN "SW2SWS" 85

3.1. IDENTIFICACIÓN ONTOLÓGICA. 86
3.1.1. Ontologías utilizadas. *87*
3.1.2. Repositorio de identificadores de ontologías *88*
3.1.2.1. Extracción de identificadores de clases y propiedades de una ontología 88
3.1.2.2. Traducción de identificadores de la ontología al idioma del contenido. 93
3.1.3. El proceso de identificación. Métodos. *96*
3.1.3.1. Identificación exacta 97
3.1.3.2. Identificación basada en la raíz de términos 100
3.1.3.3. Identificación basada en WordNetRDF 101
3.1.4. Comentarios sobre la metodología de identificación *102*
3.1.5. Herramienta automática para la identificación ontológica de un sitio web *106*
3.2. EXTRACCIÓN Y ANÁLISIS 108
3.2.1. Preprocesado *109*
3.3.2. Procesado de Lenguaje Natural (PLN) *109*
3.2.2.1. PLN en Esperanto 110
3.2.2.2. PLN en Castellano 115
3.2.3. Representación formal en XML *124*
3.3. INTERPRETACIÓN: GENERACIÓN DE VISTAS SEMÁNTICAS 126
3.3.1. Alcance *127*
3.3.2. Formalización de una Vista Semántica de primer orden. *127*
3.3.3. Generación de un sitio web semántico mediante la herramienta sw2sws. *129*
3.3.4. Mantenimiento de un sitio web semántico. *130*

CAPÍTULO 4: EL BUSCADOR SEMÁNTICO BASADO EN VISTAS SEMÁNTICAS 131

4.1. CRITERIOS DE FILTRADO DE INFORMACIÓN EN LOS BUSCADORES NO SEMÁNTICOS. 132
4.2. ¿CÓMO SE IMAGINA DESDE EL CONSORCIO WORLD WIDE WEB LOS BUSCADORES SEMÁNTICOS? 134
4.3. SITUACIÓN DE LOS BUSCADORES SEMÁNTICOS A PRINCIPIOS DEL 2009 135
4.3.1 Buscadores semánticos no orientados a usuarios *136*
4.3.2 Buscadores semánticos orientados a usuarios. *137*
4.3.3 Conclusión. *139*
4.4. CONTEXTO DE LA BUSQUEDA. 140
4.5. CRITERIOS DE FILTRADO DE INFORMACIÓN EN LOS BUSCADORES SEMÁNTICOS 141
4.6. BREVE COMENTARIO SOBRE EL MODELO DE DATOS DEL BUSCADOR SEMÁNTICO 142
4.7. ARQUITECTURA TEÓRICA DEL BUSCADOR SEMÁNTICO BASADO EN VISTAS SEMÁNTICAS 143
4.8. PROTOTIPO DE BUSCADOR: VISSEM 145
4.8.1. Inicializando la BBDD del buscador VISSEM. *145*
4.8.2. Mantenimiento de vistas semánticas en el buscador semántico. *147*
4.8.3. ¿Cómo se utiliza el buscador VISSEM?. *147*
4.8.4. Requisitos funcionales en el diseño de buscadores semánticos. *152*
4.9. ANOTACIÓN SEMÁNTICA DE LOS BUSCADORES SEMANTICOS. 154
4.10. VISSEM ENRIQUECIDO CON SU PROPIA ANOTACIÓN SEMÁNTICA. 156

CAPÍTULO 5: VALORACIÓN, CONCLUSIONES Y TRABAJOS FUTUROS. 159

5.1. APORTACIONES 160
5.2. VALORACIÓN DE LA PROPUESTA 161
5.3. CONCLUSIONES 165
5.4.1. Líneas de Investigación 166
5.4.1.1. Sincronización de Páginas Semánticas. 166
5.4.1.2. Optimización de los procesos de transformación basados en PLN. 167
5.4.1.3. Contenido Semántico de naturaleza Dinámica 167
5.4.1.4. Algoritmos para la extracción de identificadores y traducción al idioma del contenido 168
5.4.1.5. Estrategias para el tratamiento masivo de ontologías en el proceso de identificación 168
5.4.1.6. Selección automática de contextos para aplicaciones semánticas 169
5.4.1.7. Motor de BBDD basado en lógica 169
5.4.2. Trabajos de Innovación Tecnológica 169
5.4.2.1. Optimización del Buscador Semántico: VISSEM 169
5.4.2.2. Web Service para RO y RIO 170
5.4.3. Oportunidades Empresariales 170
5.4.3.1. Herramienta profesional para generación de páginas semánticas basadas en ontologías concretas 170
5.4.3.2. Diseño específico de Ontologías 171

BIBLIOGRAFÍA 172

ANEXO A: INSTALACIÓN DE LAS HERRAMIENTAS IMPLEMENTADAS. 193

A.1. HERRAMIENTAS IMPLEMENTADAS. 193
A.2. DESCARGA DE HERRAMIENTAS 194
A.3. CÓMO AÑADIR UNA NUEVA ONTOLOGÍA EN SW2SWS. 198
A.3.1. Extracción de todas las clases y propiedades 198
Configurar el fichero “ontologia.properties” 199
Ejecutar el proceso de extraer palabras de la ontologia 200
A.3.2. Traducción de identificadores al lenguaje del contenido 201
A.4. PROCESO DE IDENTIFICACIÓN DE UN SITIO WEB. 204
A.5. EXTRACCIÓN DE LENGUAJE NATURAL Y SU REPRESENTACIÓN FORMAL EN XML. 205
A.6. GENERACIÓN DE VISTAS SEMÁNTICAS. 207
A.7. HERRAMIENTA PARA TRANSFORMAR UN SITIO WEB EN UN SITIO WEB SEMÁNTICO (SW2SWS). 208
A.8. CARGA DE ONTOLOGÍAS EN LA BBDD DEL BUSCADOR SEMÁNTICO. 209
A.9. CARGA DE VISTAS SEMÁNTICAS EN LA BBDD DEL BUSCADOR SEMÁNTICO. 211
A.10. MIGRACIÓN DEL MODELO DE JENA A VISSEM. 213

ANEXO B: DIAGRAMAS DE LAS APLICACIONES 215

TERICA.JAR 215
HERRAMIENTA DE TRANSFORMACIÓN SW2SWS 225
BUSCADOR VISSEM 236

Glosario y acrónimos.

Anotación: Se entiende por anotación semántica, el proceso de representar estructuradamente información, que en origen carece de estructura, desde un punto de vista semántico. Con independencia del lenguaje de anotación y de que sea embebido o no.

Clase: Representa un concepto, es una idea genérica. Por ejemplo "Persona" puede ser una clase, mientras que "Luis" es un individuo o instancia de dicha clase.

Embebido: Este término, muy utilizdo en informática, nos permite expresar la idea de que un programa, o incluso una parte en lenguaje etiquetado, esta contenido o encerrado dentro de otro. Por ejemplo, se puede hablar de codigo JavaScript embebido en HTML. Es decir, dentro de una página web escrita en HTML, puede haber un programa en JavaScript.

Instancia: Representan objetos particulares de una clase. Es un individuo concreto.

Ontología: Es una representación de un dominio de conocimiento expresado jerárquicamente mediante conceptos que se nombran en lenguaje natural (organizado en clases), de los cuales se especifican sus características o atributos (slots o propiedades) y las relaciones entre ellos (restricciones).

OWL: Es un lenguaje de ontologías, orientado a la Web, basado en lógica descriptiva. El lenguaje OWL ha tenido varias versiones, la vesion 1.0 (2004) y version 1.1 (2006)) tiene tres variantes: OWL Lite, OWL DL y OWL Full. El 2 de diciembre del 2009 el Consorcio de la W3C publicó el borrador de las especificaciones "OWL 2", en el cual se definen otros tres sublenguajes llamados OWL 2 EL, OWL 2 QL y OWL 2 RL.

PLN: Procesado de lenguaje natural.

Propiedad: Es una caracteristica o atributo de un objeto.

RO: Repositorio de ontologías.

RIO: Repositorio de identificadores de ontologías.

Semántico: En el contexto de esta tesis, el término "semántico" se refiere a significado respecto a los sistemas informáticos.

Sitio web semántico: es un sitio web que además incorpora también anotaciones semánticas sobre su contenido, en un lenguaje soportado por una lógica descriptiva, como es el caso de OWL.

Usuarios activos: Denominamos así al conjunto formado por los webmaster y los usuarios de la Web 2.0 que participan de forma activa en los contenidos.

Vista semántica: es una estructura de información formado por una o varias interpretaciones que proporcionan diferentes visiones de acuerdo a diferentes ontologías de un mismo contenido

Tablas

Tabla 1. 1: Ranking de buscadores diciembre 2007 2
Tabla 1. 2: Solución 1, parte 1 12
Tabla 1. 3: Solución 1, parte 2 12
Tabla 1. 4: Solución 2, parte 1 13
Tabla 1. 5: Solución 2, parte 2 13
Tabla 1. 6: Ejemplo de tripleta básica expresada en RDF 15
Tabla 1. 7: Ejemplo 2 de tripleta en RDF 17
Tabla 1. 8: Otra forma de plantear el ejemplo 2 de tripleta en RDF 17
Tabla 1. 9: Clase mamifero hereda de vertebrado 27
Tabla 1. 10: Clases disjuntas 28
Tabla 1. 11: Propiedades de objetos 28
Tabla 1. 12: Propiedad transitiva 28
Tabla 1. 13: Propiedad transitiva en OWL 29
Tabla 1. 14: Propiedades de vertebrados 29
Tabla 1. 15: Clases como restricciones 31
Tabla 1. 16: Restricciones propiedades tipo dato en OWL (necesarias) 32
Tabla 1. 17: Restricciones propiedades tipo dato en OWL (necesarias y suficientes) 32
Tabla 1. 18: Clase perro en OWL 34
Tabla 1. 19: Creación de un modelo 38
Tabla 1. 20: Escribir un modelo básico con Jena 39
Tabla 1. 21: Resultado RDF obtenido con Jena 40
Tabla 1. 22: Declaración de un modelo 41
Tabla 1. 23: Definición e inicialización 41
Tabla 1. 24: Escribir modelo 41
Tabla 1. 25: Representación RDF 42
Tabla 1. 26: Lista de todos los enunciados 43
Tabla 1. 27: Ejecución de extración de enunciados 43
Tabla 1. 28: Unión con JENA de dos modelos 45
Tabla 1. 29: Ejemplo vcard con RDF 46

Tabla 2. 1: Varias ontologías referenciadas desde una misma página web 78
Tabla 2. 2: Ejemplo de vinculación entre Interpretación y página web 79

Tabla 3. 1: Ontologías soportadas en el prototipo 87
Tabla 3. 2: Descripción XML de la lista de términos de la ontología a traducir 92
Tabla 3. 3: Ejemplo de formato de diccionario inglés-esperanto 95
Tabla 3. 4: Mapeo de términos ontológicos para el ejemplo 100
Tabla 3. 5: Relación de webs (proceso identificación) 106
Tabla 3. 6: Representación de letras 111
Tabla 3. 7: Codificación UTF-8 esperanto 112
Tabla 3. 8: Información obtenida de SVMTools 121
Tabla 3. 9: Etiquetas herramienta SVMTools (2ª parte) 123

Tabla 5. 1: Resultados de la transformación 162

Figuras

Figura 1. 1: Ranking de buscadores, agosto 2007 2
Figura 1. 2: Evolución de sitios web 5
Figura 1. 3 11
Figura 1. 4: Esquema gráfico de una tripleta 16
Figura 1. 5: Ejemplo 2 de tripleta gráfica 16
Figura 1. 6: Ontología vertebrados. 27
Figura 1. 7: Definicion de propiedades en PROTÉGÉ 30
Figura 1. 8: Restricciones de propiedades tipo dato 31
Figura 1. 9: Jerarquía de clases 33
Figura 1. 10: Instancia de la clase perro en PROTÉGÉ 35
Figura 1. 11: Instancia de la clase perro. 35
Figura 1. 12: Esquema más sencillo: un recurso y una propiedad 39
Figura 1. 13: Propiedades con valores 40
Figura 1. 14: Dos modelos que pertenecen a la misma ontología 44
Figura 1. 15: Union de dos modelos 45
Figura 1. 16: Arquitectura de la Web Semántica en 2000 50
Figura 1. 17: Arquitectura de la Web Semántica en 2006 50
Figura 1. 18: DAFO Web sin semántica (Web 2.0) 53
Figura 1. 19: DAFO Web Semántica (Web 3) 54
Figura 1. 20: Clasificación Reeve-Han 56
Figura 1. 21: Clasificación de Rolando Beltrán 57
Figura 1. 22: Propuesta de clasificación de herramientas de anotación 59
Figura 1. 23: Anotación Amaya 11.2 62
Figura 1. 24: Cuadro resumen de herramientas de anotación 65

Figura 2. 1: Objetivo GRDDL 68
Figura 2. 2: Etapas del proceso de transformación 69
Figura 2. 3: Ejemplo de captura de texto 71
Figura 2. 4: Extracción 73
Figura 2. 5: Ejemplo de anotación. 73
Figura 2. 6: Página semántica basada en vista semántica 77

Figura 3. 1: Representación general del proceso de tranformación. 86
Figura 3. 2: Proceso de generación de los nombres a mapear 90
Figura 3. 3: Organigrama para extraer identificadores 91

Figura 3. 4: Esquema traducción de identificadores. 95
Figura 3. 5: Organigrama de identificación. 97
Figura 3. 6: Captura pantalla identificación exacta 98
Figura 3. 7: Identificación ontológica basada en raices 101
Figura 3. 8: Respuesta on line WordNetRdf 102
Figura 3. 9: Ejemplo identificación exacta español-inglés 103
Figura 3. 10: Ejemplo 2 identificación exacta español-inglés 104
Figura 3. 11: Ejemplo identificación por raiz con esperanto 105
Figura 3. 12: Segundo ejemplo con esperanto 105
Figura 3. 13: Características del proyecto 107
Figura 3. 14: Progreso de identificación 107
Figura 3. 15: Informe resumen de identificación 108
Figura 3. 16: Etapas de extracción. 109
Figura 3. 17: Esquema de formación de palabras en Esperanto 113
Figura 3. 18: Métodos de la clase gramaticaEsperanto 114
Figura 3. 19: Ejemplo de analisis sintáctico 119
Figura 3. 20: Respuesta de SVMT 121
Figura 3. 21: Ejemplo de análisis 122
Figura 3. 22: Ejemplo de extracción 125
Figura 3. 23: Ejemplo de extracción 2 126
Figura 3. 24: Anotación automática 128
Figura 3. 25: Instancia que invoca a la ontología madre, visor SWOOP 129
Figura 3. 26: Herramienta html2ws - Vistas Semánticas 130

Figura 4. 1: Buscador Actual 134
Figura 4. 2: Buscador semántico 135
Figura 4. 3: Arquitectura de Naveganza 138
Figura 4. 4: Arquitectura del buscador semántico 144
Figura 4. 5: Funcionamiento de las aplicaciones en la Web Semántica 144
Figura 4. 6: Inicialización de ONTOLOGIAS en la BBDD del buscador semántico 146
Figura 4. 7: Tablas en la BBDD que representan las ontologías 146
Figura 4. 8: Comparación con Google, ejemplo 1 148
Figura 4. 9: Comparación con Yahoo, ejemplo 1 149
Figura 4. 10: Comparación con Google, ejemplo 2 149
Figura 4. 11: Comparación con Yahoo, ejemplo 2 150
Figura 4. 12: Comparación Hakia y Powerset (ejemplo 2) 151
Figura 4. 13: Ejemplo 3, buscador UNED 151

Figura 4. 14: (ver instancia) de Pitbull..... 152
Figura 4. 15: Clase genérica caniche..... 155
Figura 4. 16: Clase genérica doberman. 155
Figura 4. 17: ¿ Cúal es el perro más peligroso ? 157
Figura 4. 18: ¿ Cúal es el animal más rápido ? 158
Figura 4. 19: ¿Qué es más peligroso un caniche o un doberman ? 158
Figura 5. 1: DAFO de la Web Semántica basada en vistas semánticas 164

Introducción y resumen

La Web Semántica tiene como objetivo fundamental que las páginas webs no sólo las entiendan las personas sino que también puedan ser usadas como fuente de conocimiento por sistemas informáticos. La ingente cantidad de información que contiene la Web así lo exige. Esta tesis se orienta de forma genérica hacia este objetivo.

La predicción de Nova Spivack, Radar Networks [56] (Figura i) sobre el futuro de la Web establece 5 hitos. En dicha figura se representa como la Web ha evolucionado, pero también se representan las expectativas de evolución, como es el caso de la Web Semántica o Web 3.0

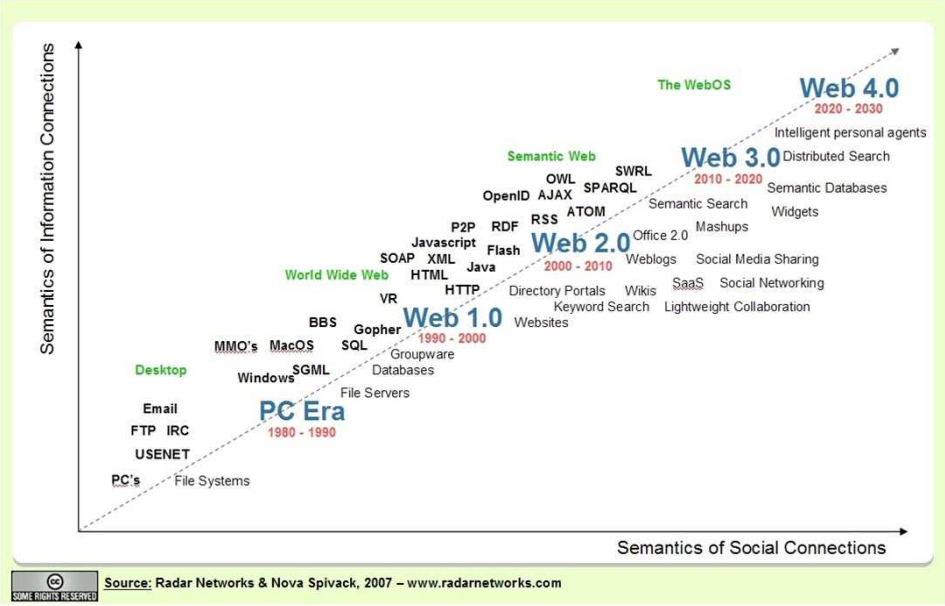


Figura i: Evolución y predicción de la Web

La Web 1.0, es una web, donde la participación de los usuarios en su construcción es inexistente, es decir, todo el contenido es mantenido por los webmasters. Pero esta situación ha cambiado, ahora estamos en la Web 2.0, que es una web donde además de webmasters aparecen usuarios finales activos, usuarios que participan en los contenidos de Internet mediante wikis, blogs, foros, etc. La Web 3.0 es el siguiente paso evolutivo, y añade, a las capacidades anteriores contenido semántico para los sistemas informáticos. Por lo tanto, el concepto de la Web 3.0 contiene el concepto de Web Semántica. En cierta medida la Web Semántica es invisible para las personas pero sus efectos serán espectaculares para los usuarios, especialmente en lo referente al acceso eficiente de la información.

A lo largo de esta tesis, nos referiremos reiteradas veces al webmaster y a los usuarios de la Web 2.0 que participan de forma activa en los contenidos, de forma conjunta como **"**usuarios activos**"**.

Si está claro el objetivo de la Web Semántica, también está claro que hasta el día de hoy no se ha conseguido que se convierta en una realidad. Se ha trabajado mucho, se han definido especificaciones que garantizan la interoperabilidad, y sobre esto, se han desarrollado herramientas, interfaces de programación de aplicaciones (API) y entornos de trabajo que describiremos en el capítulo 1, pero los logros obtenidos no se han extendido porque globalmente no son suficientemente eficientes y tampoco son herramientas dirigidas a usuarios finales.

Resumen y aportaciones

El concepto de Web Semántica exige una representación formal de la información de acuerdo a ontologías de referencia que doten a la Web de semántica para los sistemas informáticos. Hay un acuerdo generalizado de que esto se haga a través de lenguajes estándar de etiquetado. Pero también exige que haya suficientes anotaciones semánticas de este tipo, es necesaria una cierta "masa crítica" para que tenga sentido global en la Web. Y esto no se ha conseguido fundamentalmente por la complejidad que plantea realizar la anotación de forma manual. Sólo cuando se tenga la facilidad de generar suficientes anotaciones semánticas, ya sea de forma automática o semi-automática, se podrá extender la semántica en los contenidos de la Web. A partir de esta situación ya se podrán desarrollar aplicaciones que aprovechen o saquen partido de esa semántica, las aplicaciones semánticas. Y hacia este problema se orienta, ya más particularmente, nuestra investigación.

Así, la principal aportación específica de esta tesis es la propuesta de un procedimiento para contribuir en la extensión de la población de ontologías, que facilita a un usuario activo el etiquetado semántico de la información que gestiona, y que ya ha descrito en texto en su página HTML, de acuerdo a la ontología u ontologías que el sistema ha identificado como más afines a sus contenidos. En nuestro trabajo se tiene muy en cuenta esta última posibilidad, el contenido a etiquetar puede hacer referencia a diferentes temas o puede interpretarse desde diferentes puntos de vista, es decir, el proceso puede "poblar" diferentes ontologías desde el mismo contenido, lo que en este trabajo denominaremos generar diferentes "*vistas semánticas*".

Pero además un sitio web semántico debe ser compatible con la Web actual, es decir, el proceso de anotación no debe afectar al funcionamiento actual de cualquier buscador. En consecuencia, al transformar un sitio web en un sitio web semántico se obtendrá funcionalidades semánticas que podrán ser explotadas por un buscador semántico, pero cuando sea tratado por un buscador ordinario existirá compatibilidad total y el buscador ordinario lo tratará como si fuera un sitio web más. También en esta tesis se ha tenido en cuenta esta exigencia, las *vistas semánticas* se mantienen diferenciadas de la página HTML, accesibles pero sin afectar a los buscadores habituales.

Hemos definido unas etapas de transformación que deben realizarse de forma secuencial. La primera que denominamos *identificación* permite asociar la ontología u ontologías que están más cercanas al contenido de la página web. Esta selección de ontologías es fundamental para que en la etapa siguiente, que denominamos *extracción*, se procese el texto a nivel morfológico y sintáctico. Finalmente, la última etapa que hemos denominado *interpretación* se encarga de la anotación semántica. La anotación se hace en nuestro estudio en OWL DL por ser el lenguaje estándar para la descripción de semántica en la Web [Smith *et al*; 2004] y permitir las inferencias propias de la lógica descriptiva SROID(D) en el que se sustenta.

En el desarrollo, la metodología empleada se ha basado en simplificar la problemática sin perder la categoría conceptual para poder abarcar todo el ámbito de la propuesta, compuesta por una secuencia de procesos que se desarrollan a lo largo de la tesis. Es decir, se ha planteado un escenario simplificado que recrea los elementos fundamentales de la Web actual para proponer una estrategia de migración o transformación hacia la Web Semántica. Las conclusiones alcanzadas son el resultado de un proceso de autocorrección experimental. Hemos implementado por completo la propuesta de esta tesis que puede ser verificada por cualquier investigador siguiendo las indicaciones del anexo A.

Para realizar esta transformación o migración, se ha implementado una herramienta prototipo (*sw2sws*) que automatiza las tres etapas que hemos presentado. Se ha probado sobre sitios webs reales. Nuestra herramienta prototipo automatiza el proceso de anotación con las ontologías usadas en la tesis, pero es fácilmente adaptable para soportar otras. Además nuestro enfoque acepta la posibilidad de intervención del usuario (proceso semiautomático) que complete o mejore cualquiera de las fases del proceso global.

La calidad de la anotación obtenida depende de varios factores; como son la propia calidad de la ontología con respecto a la que anota (afinidad, precisión, estandarización, completitud, etc), la claridad del contenido y la capacidad de extracción y análisis, condicionada, en gran medida, al procesado de lenguaje natural (PLN). Esta tesis no pretende resolver el problema del PLN para la anotación; no obstante, para probar el proceso, hemos realizado un pequeño módulo de PLN que permite mostrar la viabilidad para usuarios activos, usuarios que participan en los contenidos y que son inexpertos en las técnicas de la Web Semántica.

Alcanzado el objetivo principal, para mostrar cómo explotar esta información que ya tiene semántica y cerrar todo la secuencia del proceso, nos hemos visto en la necesidad de diseñar e implementar un prototipo propio de buscador semántico, al que hemos denominado *Vissem*, capaz de interpretar preguntas en lenguaje natural y efectuar las búsquedas correspondientes sobre las instancias de los sitio web semánticos que hemos generado.

Organización de la tesis

El capítulo 1, se dedica a repasar la evolución de la Web. Procuraremos contestar las preguntas que nos hacemos todos cuando se aborda el tema, como ¿qué problema tiene la Web sin semántica? ¿por qué es importante el procesado de lenguaje natural para la Web Semántica? y ¿cuál es el estado del arte de la Web Semántica? (en el momento de redactar esta tesis). Finalmente, analizaremos el problema de implantar la Web Semántica y definimos más ampliamente los objetivos de la tesis.

En el capítulo 2 se propone un procedimiento para la automatización de la anotación semántica, fundamental para alcanzar el objetivo de transformar la Web actual en una Web Semántica. Es de destacar, que en este capítulo se presenta un nuevo concepto como es el de la *vista semántica*, que a su vez permite definir otros conceptos, que aunque puedan parecer más familiares, nunca han sido concretados como son los conceptos de *páginas semánticas* y *sitios web semánticos*. El capítulo 3, pone en práctica todos los conceptos e ideas presentadas en el capítulo anterior, por medio de una herramienta prototipo de transformación que hemos

desarrollado. Con ambos capítulos se cubre el propósito principal de la investigación, que tal y como indica el título es un procedimiento para transformar la Web en Web Semántica.

Pero ¿cómo comprobar que un sitio Web ahora es también un sitio Web Semántico?, ¿cómo saber si las anotaciones semánticas corresponden con el contenido realmente?. Con este fin hemos construido una aplicación web semántica que utilice la información de un sitio web semántico que previamente hemos generado (capítulos anteriores), de manera, que el capítulo 4 se dedica al buscador semántico, basado en nuestro concepto de *sitio web semántico.* Este capítulo sirve para ver la utilidad del procedimiento de transformación que proponemos.

El capítulo 5 recoge nuestra valoración, nuestra conclusión y presenta varias propuestas de trabajos futuros.

Finalmente hay que mencionar que hemos incluido dos anexos; el primero describe cómo cualquier investigador puede comprobar nuestros resultados usando las herramientas que hemos implementado. Se detalla a nivel de usuario avanzado, explicando cómo compilar y utilizar cada funcionalidad por separado y de forma integrada, pero no se entra a nivel de código. El otro anexo es una descripción de la librería “terica,jar” que contiene todas las funcionalidades necesarias para que funcione nuestra herramienta de transformación “Sw2sws”.

Capítulo 1: Evolución de la Web y su problema actual

Todos sabemos que Internet ha tenido un crecimiento espectacular y continúa imparable. Internet ha proporcionado nuevas oportunidades económicas, sociales y tecnológicas. Todo tipo de organizaciones, empresas y universidades han visto en la Web una forma de promocionarse, establecer relaciones económicas y publicar e intercambiar información de todo tipo.

La Web es también el mayor repositorio de información y en él, se puede buscar información sobre cualquier tema imaginable. Las herramientas para localizar información en la world wide web son los *buscadores*. La importancia de estas herramientas es tal, que desde hace años, suelen ser el punto de partida para navegar por Internet y es por este motivo, que la mayoría de los "portales", incluyen alguna utilidad para buscar información contenida tanto en sus propios servidores como en Internet. La proliferación de estas herramientas ha sido enorme, se han implantado miles de buscadores, y ha habido mucha competencia, pero hace pocos años, por la calidad de sus búsquedas, se estabilizó un ranking donde los buscadores como Yahoo o Google permanecen inmutables ocupando el segundo y primer puesto respectivamente. Según el análisis de ComScore (ver figura 1.1), en agosto del 2007 se realizaron más de 61.000 millones de búsquedas; el 60,7% fueron a través de Google, el 14% de Yahoo y el resto se reparte prácticamente entre otros 8 buscadores.

Top 10 Search Properties Worldwide* August 2007 Total World Age 15+, Home and Work Locations** Source: comScore qSearch 2.0	
Search Property	**Searches (MM)**
Worldwide	*61,036*
Google Sites	37,094
Yahoo! Sites	8,549
Baidu.com Inc.	3,253
Microsoft Sites	2,166
NHN Corporation	2,044
eBay	1,319
Time Warner Network	1,212
Ask Network	743
Fox Interactive Media	683
Lycos, Inc.	441

Figura 1. 1: Ranking de buscadores, agosto 2007

Según otra fuente, como la "The Nielsen Company" [62] (ver tabla 1.1), en diciembre del 2007 el 56,3% de las busquedas fueron a través de Google, el 17,7% de Yahoo y el resto se reparte entre otros 8 buscadores.

Provider	Searches (000)	Share of Searches	Searches per Searcher
1. Google Search	4,062,536	56.3%	37.9
2. Yahoo! Search	1,273,688	17.7%	22.4
3. MSN/Windows Live Search	995,899	13.8%	31.7
4. AOL Search	339,761	4.7%	15.2
5. Ask.com Search	159,529	2.2%	10.0
6. My Web Search	70,630	1.0%	10.4
7. Comcast Search	34,715	0.5%	10.1
8. NexTag Search	29,019	0.4%	2.9
9. AT&T Worldnet Search	25,159	0.3%	9.1
10. BizRate Search	17,205	0.2%	2.8

Source: Nielsen Online, MegaView Search

Tabla 1. 1: Ranking de buscadores diciembre 2007

Pero aunque los buscadores utilizan cada vez más y mejores algoritmos y servidores, la ingente cantidad de datos está planteando serios problemas en la calidad de información que recibe el usuario. Sin embargo, este problema no es nuevo y ya fue previsto antes de que Internet existiera [38].

1.1. Inicio de la Web

En 1945, el Director de la Oficina de Desarrollo e Investigación Científica (EE.UU.), el Doctor Vannevar Bush, escribió el artículo "As We May Think" para "The Atlantic Online" [37], en el que expresaba su preocupación por la ingente cantidad de información que existía y estaba siendo generada, y el poco tiempo y los ineficientes sistemas que había para encontrarla. Así, y basándose en la tecnología existente en aquel momento, describió un dispositivo personal, al que llamó "memex". Este dispositivo no llegó a construirse pero es el primer antecedente de solución al problema de la gestión a gran escala de la información. Vannevar Bush fue un pionero, un visionario que ya previó diversas formas (mecánicas) para estructurar, organizar y acceder a los textos, es decir, de nuevos procesos para su producción y uso [38]. Es así como nace la idea de hipertexto, aunque el término "hipertexto" fue acuñado por Ted Nelson en 1965, en su artículo "A File Structure for the Complex, the Changing, and the Indeterminate" [59], que leyó durante la vigésima conferencia anual de la Association of Computer Machinery (ACM). Ted Nelson ideó un modelo (Xanadu) para la interconexión de documentos electrónicos.

El antecedente directo de Internet es ARPANET (Advanced Research Projects Agency Network) que fue creada por encargo del Departamento de Defensa de los Estados Unidos. El primer enlace de ARPANET se estableció el 21 de noviembre de 1969 entre UCLA (University of California, Los Angeles) y Stanford. El 5 de diciembre del mismo año, toda la red inicial estaba lista. En 1972, Ray Tomlinson inventó el correo electrónico. Ese mismo año, Vinton G. Cerf y Robert E. Kahn presentaron el protocolo TCP/IP, considerados padres de Internet, recibieron en el año 2004 el premio Turing. En 1973, el protocolo FTP ya estaba definido e implementado.

Entre 1982 y 1986, Cerf diseñó el MCI MAIL, primer servicio comercial de correo electrónico que se conectaría a Internet. La World Wide Web fue inventada en 1989 por un informático del CERN (Organización Europea de Investigación Nuclear) llamado Tim Berners-Lee. Era un sistema de hipertexto para compartir información, basado en Internet, concebido originalmente para servir como herramienta de comunicación entre los científicos nucleares del CERN. Tim Berners-Lee había estado experimentando con hipertexto desde 1980, año en que programó Enquire, un programa para almacenar piezas de información y enlazarlas entre ellas. Enquire se ejecutaba en un entorno multiusuario y permitía acceder a varias personas a los mismos datos. Tim Berners-Lee [38] entregó su propuesta al CERN en 1989; en septiembre de 1990 recibió el visto bueno y, junto con Robert Cailliau, comenzó a escribir el nuevo sistema de hipertexto. A finales de 1990, el primer browser de la historia, World Wide Web, ya tenía forma. En aquella época casi todo el mundo utilizaba TeX y PostScript, pero éstos eran demasiado complicados teniendo en cuenta que debían ser leídos por todo tipo de ordenadores. Así, tanto el lenguaje de intercambio (HTML), como el protocolo de red (HTTP) se diseñaron para ser realmente muy simples.

A principios de 1993 había alrededor de 50 servidores [38]. Ya había un navegador gráfico, pero solo funcionaba en una plataforma. Por otro lado los navegadores en modo línea se habían extendido a todas las plataformas, pero su uso era tedioso. En Febrero del mismo año, se lanzó la primera versión alfa del navegador "Mosaic for X", desarrollado en el NCSA (National Center for Supercomputing Applications). En Abril el CERN declaraba la WWW como tecnología de acceso gratuito. En septiembre, ya había versiones de Mosaic para PC y Macintosh, y había más de 500 servidores. Es el comienzo del crecimiento explosivo de la Web. A finales del 94 había más de 10.000 servidores y 10 millones de usuarios. En 1997, más de 650.000 servidores. En los años siguientes se tiende a la unificación y compatibilidad [39].

1.2. El problema que tiene la web sin semántica

En el 2003, la Web es algo cotidiano para una gran parte de los millones de usuarios de Internet que hay en todo el mundo. Sus utilidades son diversas, su impacto en la economía mundial es apreciable. No sólo hay documentos de texto: hay imágenes, vídeos, música, se pueden comprar cosas, se pueden hacer reservas de viajes, se ha creado un mundo virtual que ofrece acceso a información, servicios y ocio.

El volumen es enorme y su crecimiento es exponencial. Según Netcraft [56] (ver figura 1.2), en mayo del 2009 el número de sitios webs es de 236 millones.

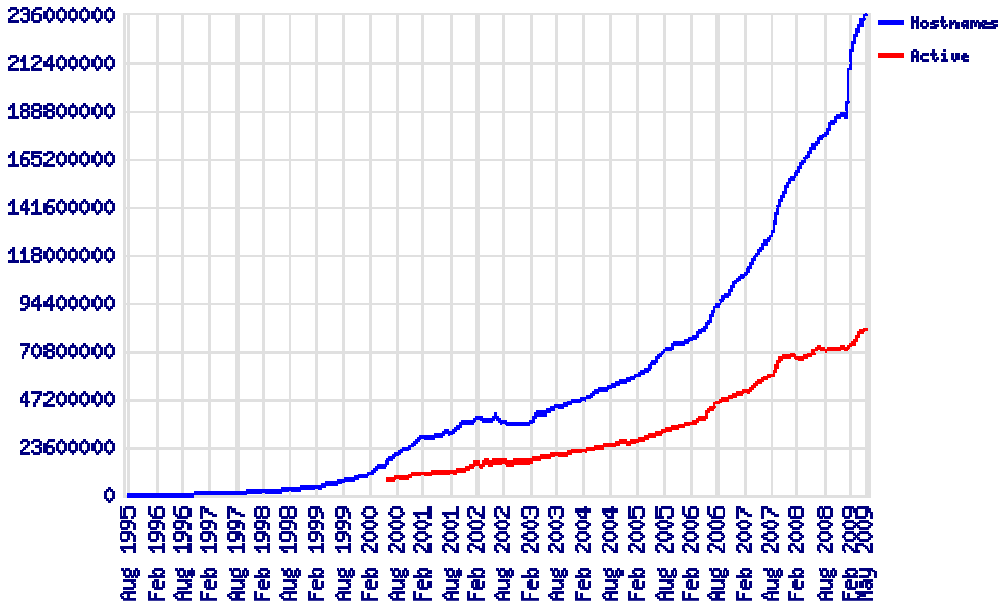


Figura 1. 2: Evolución de sitios web

Es muy difícil estimar el tamaño de la Web, ya que un sitio web consta de varias páginas web, pero en el 2001, Bergman realizó un estudio, en el cual se calculaba que el volumen de páginas web estáticas era del orden de $4*10^9$ (equivalente a entre 14 y 28 millones de libros) [Bergman; 2001]. Teniendo en cuenta, la gráfica de Netcraft y el trabajo de Bergman, se deduce que aproximadamente $35*10^6$ sitios webs corresponden con $4*10^9$ páginas web estáticas, de manera que considerando la misma relación, en mayo del 2009 se podría estimar en $26,9*10^9$ páginas web estáticas, equivalente a más de 94 millones de libros en el caso más

pesimista. Este volumen de información supera a cualquier biblioteca, como dato, una de las bibliotecas consideradas más grandes del mundo es la biblioteca de Washington conocida como biblioteca del congreso (fundada en 1800) que cuenta con 29 millones de libros.

Pero todo esto no es más que la denominada web superficial. Se ha calculado que la web profunda, aquella que se genera dinámicamente mediante el acceso al contenido a base de datos (BBDD), puede contener un tamaño de información varios cientos de veces superior a la web estática y crece a un ritmo aún mayor [O'Neill et al; 2003].

Este gran desarrollo ha convertido el proceso de acceso a la información en un problema cada vez más crítico. Los buscadores funcionaron muy bien durante estos años; sin embargo, el extraordinario crecimiento que ha experimentado la Web empieza a mostrar sus primeros síntomas frustrantes. Cada día el usuario necesita dedicar más tiempo para filtrar las páginas web que devuelve una búsqueda, es decir, la calidad de la información es cada vez peor. Como estrategia para resolver la situación de futuro, el World Wide Web Consortium (W3C) ha propuesto el uso de ontologías y, en el 2004, logró desarrollar las especificaciones OWL DL, que permitirán la evolución hacia la Web Semántica. Es decir, no es una alternativa el implantar la Web Semántica, lo que ocurre es que si no se implanta llegará un momento que la tecnología actual no pueda soportar índices aceptables de funcionamiento. De manera, que implantar la Web Semántica es una necesidad para resolver el problema que ya se está produciendo.

1.3. Inicios de la Web Semántica

La Web Semántica es una propuesta realizada por el propio inventor del HTML y fundador de la W3C (Consorcio de la World Wide Web), Tim Berners-Lee [6], que en septiembre de 1998 publicó dos artículos, “Semantic Web Road Map” y “What the Semantic Web can represent” [Berners-Lee 1; 1998], en los que se presentaba la idea de Web Semántica como una extensión de la Web actual dotada de significado, esto es, un espacio donde la información tendría un significado bien definido, de manera que pudiera ser interpretada tanto por agentes humanos como por agentes computerizados.

Dotar a la Web actual de significado tiene como efecto inmediato que la búsqueda de información sea más eficiente computacionalmente y sobre todo más adecuadas a las necesidades del usuario. Por supuesto, se mantienen los principios fundamentales que han hecho un éxito de la Web actual, como son los principios de descentralización y compartición [Castells; 2002].

En 1998, Tim Berners-Lee aseguraba que *"A Semantic Web is not Artificial Intelligence*" [Berners-Lee; 1998], es más, estaba convencido que se trataba de un problema de estructura, de organizar la información de forma adecuada como se puede comprobar en este extracto: "*The concept of machine-understandable documents does not imply some magical artificial intelligence which allows machines to comprehend human mumblings. It only indicates a machine's ability to solve a well-defined problem by performing well-defined operations on existing well-defined data. Instead of asking machines to understand people's language, it involves asking people to make the extra effort*". De hecho, trabajando en esta línea de convencimiento logró, desde el Consorcio de la World Wide Web, impulsar el desarrollo de los estandares semánticos. En 1999 aparece RDF, en diciembre del 2000 aparece la primera especificación del lenguaje DAML+OIL [60], fruto de la cooperación entre los grupos de trabajo de OIL y DARPA (US Defense Advanced Research Projects Agency), quienes anteriormente habían desarrollado el lenguaje DAML (DARPA's Agent Markup Language) con la finalidad de extender el nivel de expresividad de RDFS y en el 2004 OWL fue especificado, de tal manera que en este año, ya sabíamos como organizar la información de forma adecuada y como tratar ontologías con lógica descriptiva. Sin duda, la aceptación de OWL por la Comunidad Científica es un gran pacto para favorecer la implantación de la Web Semántica. Muy pronto prosperaron APIS para distintos lenguajes de programación que permiten tratar la información con arreglo a estos estándares.

En el 2004 la mayor parte de investigadores siguen pensando que "*The Semantic Web is Not Artificial Intelligence on the Web*", estas palabras de Ivan Herman pronunciadas en Bilbao en "La Gira Estándares W3C visita Deusto (2004)" corroboran con contundencia la principal postura de los investigadores y que son muy similares a las que decía Berners-Lee, en 1998, "…documentos comprensibles por máquinas no implica Inteligencia Artificial…" [Berners-Lee; 1998]. Incluso se ha llegado a pensar que no solo la Web Semántica no es inteligencia artificial sino que será una herramienta de ayuda para el desarrollo de la misma, como asegura Berners-Lee, en el 2005, [Carvin; 2005].

1.4. La Web Semántica y el procesado de lenguaje natural.

Desde el 2004 hasta el 2007 podemos presumir de pocos avances significativos, al menos, en la realidad del usuario final, pero en el 2008 la tendencia ha cambiado, ya que han aparecido prototipos de buscadores que fundamentalmente trabajan en RDF. Sabemos que el problema no se puede resolver sólo con especificaciones y formatos, por esta razón, después de casi diez años la Web Semántica no está aún implantada. Para que la Web Semántica sea abordable hace falta algo más que es complementario a OWL pero necesario, esto es aplicar técnicas para procesar lenguaje natural. En base al trabajo realizado en esta tesis pensamos que el éxito en la implantación de la Web Semántica depende en gran medida de la mejora en las técnicas de procesado de lenguaje natural (PLN) que proporcionan la posibilidad de construir herramientas semi-automáticas e incluso automáticas que permitan transformar un sitio web en un sitio web semántico.

Las ontologías son la base de la Web Semántica, pero las ontologías por si mismas no son suficientes. Hacen falta instancias respecto a los conceptos y/o atributos que se especifican en las ontologías. Dicho de manera sencilla, hace falta mecanismos de anotación semántica, que permiten realizar una correspondencia entre las información de un sitio web y los conceptos de las ontologías. Pero RDF (como veremos en el epígrafe 1.5.1.3) no es suficiente, es necesario incorporar OWL DL, ya que la potencia de la Web Semántica reside en la capacidad de explotar mecanismos de inferencia. De manera, que la Web Semántica no existe aún, tal y como indicábamos en la introducción de la tesis, debido a que no hay anotaciones semánticas OWL DL suficientes (representan el conocimiento) y esto es consecuencia directa de la complejidad que plantea realizarlo manualmente. Por consiguiente, es necesario desarrollar herramientas que permitan automatizar procesos de anotación semántica basadas en OWL-DL. Por esto es tan importante el PLN en la implantación de la Web Semántica, ya que un PLN permitirá plantear procedimientos para realizar anotaciones semánticas, que es sin duda la clave del éxito de este nuevo paradigma y como aún no está resuelto es el principal argumento de los investigadores que piensan que la Web Semántica no es posible, como defendió Nor Naaman (Yahoo Research Berkeley) [61] con una ponencia titulada "The Semantic Web is Dead" en el “16th Internacional World Wide Web Conference” [63] celebrado 8-12 de mayo del 2007 en Alberta (Canada).

Cuando se tenga la capacidad de generar anotaciones semánticas, ya sea de forma automática o semi-automática, todos los actores que participan en los contenidos; como son los webmasters, los usuarios de wikis, usuarios de blogs, ect… podrán dotar de semántica aquellos

contenidos que dependen de ellos y en este momento tendremos contenido semántico. Llegados a este punto se podrán desarrollar aplicaciones semánticas, como por ejemplo, buscadores semánticos de propósito general.

1.5. Plataforma tecnológica de la Web Semántica

En este epígrafe examinamos la plataforma tecnológica con la que cuenta la Web Semántica en el momento de la redacción de esta tesis. Presentando los estándares, editores y entornos de trabajo actuales y también los proyectos europeos más importantes en el ámbito de este nuevo paradigma.

1.5.1. Estandares, editores y frameworks

La Web Semántica se basa en una serie de estándares que permitirán gestionar computacionalmente la información de un dominio de conocimiento. Se parte de la idea de definir una ontología en términos computables, con el objetivo de que las páginas web que se puedan asociar a esa ontología permitan hacer búsquedas inteligentes. Entendiendo por búsquedas inteligentes, aquellas que puedan hacer inferencias con lógica descriptiva.

Para llegar a definir una ontología es necesario conocer los estándares que permiten realizar esto. De forma, que en este epígrafe, se presentan todos los estándares que se deben conocer antes de desarrollar una ontología para el entorno de la Web Semántica. Se ha procurado dar una visión general de las características que aporta cada estándar para converger en OWL DL. Estos estándares sobre los que se construye la Web Semántica son RDF (representado en XML) y OWL (Lenguaje Web de Ontologías). En realidad, se trata de una serie de estándares embebidos unos en otros. No se pretende explicar el detalle de dichos estándares pero es necesario entender que sólo OWL aporta la semántica demandada, que RDF permite la localización de cualquier recurso y que XML es la forma de expresarlo.

Enfatizamos el hecho que para entender el proceso de transformación de un sitio web a un sitio web semántico, que se propone, se define y se implementa en esta tesis, se basa en anotar semánticamente parte del contenido en OWL DL-RDF-XML de forma automática. Es por esta razón importante, repasar los fundamentos de cada uno de los estándares, ya que, el concepto de *vistas semánticas* que detallaremos en el siguiente capítulo requiere de este conocimiento.

1.5.1.1. XML

XML es un lenguaje de etiquetas y sólo eso. Sin embargo, este sencillo lenguaje ofrece un formato de documento portable y flexible; portable porque se puede utilizar en cualquier plataforma; flexible porque puede representar cualquier tipo de dato que se pueda codificar como texto [Harold et al; 2005].

Estas etiquetas de las que se compone XML se denominan elementos, cada elemento se compone de una etiqueta de apertura "<nombre_tag>" y de una etiqueta de cierre "</nombre_tag>" y el contenido entre ambas es el valor del elemento.

Un elemento puede tener atributos (propiedades) que nos ofrecen información sobre el mismo. Por ejemplo, si tenemos una etiqueta del estilo <imagen path="c:\pruebas\" fichero="prueba.jpg"> significará que el elemento imagen tiene dos atributos; el atributo path indica el directorio donde está la imagen, y el atributo fichero nos proporciona el nombre de la imagen.

Los documentos XML pueden empezar con un prólogo, en el que se define la versión de XML y la codificación ***<?xml version="1.0" encoding="UTF-8"?>***, así como la reglas sintácticas que definen el propio documento (DTD) o en un fichero externo DTD o XML Schema.

Todos los documentos XML, sin excepción, deben estar bien estructurados. Esto significa que deben cumplirse varias reglas:

1. Debe haber un único elemento raíz.
2. Toda etiqueta de inicio debe tener una etiqueta de cierre coincidente.
3. Los elementos se pueden anidar, pero no superponer.

Los valores de los atributos deben estar entrecomillados

El Modelo de Objeto de Documento (DOM), que es el que hemos utilizado, es una interfaz neutral a la plataforma y al lenguaje de programación para acceder y actualizar ficheros XML. Este modelo DOM permite acceder a un archivo XML a través de una estructura arborescente, compuesta por nodos elemento y nodos de texto.

El árbol será construido en memoria y, por lo tanto, se produce una penalización de consumo de memoria RAM al usar dicho modelo. Sin embargo, la ventaja de DOM es que es más simple de programar y mantener que otros modelos. Sirva como ejemplo que un documento XML se puede convertir en un árbol con tres líneas de código.

```
// Paso 1: Crea un objeto DocumentBuilderFactory
  DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

// Paso 2: Crea un objeto DocumentBuilder
  DocumentBuilder db = dbf.newDocumentBuilder();

// Paso 3: Parsea el documento
  Document doc = db.parse(fichero);
```

Si el valor de **doc** es null significa que el fichero XML esta mal construido, en caso contrario, quiere decir, que esta bien construido

Semántica sobre XML

La flexibilidad de XML permite diferentes representaciones de la misma información, sin ambigüedad y bien representada. En consecuencia, es posible que la misma información tenga diferentes formatos XML lo que implica que XML no es adecuado para incluir semántica [Abián; 2005] por si mismo, hay que construir por encima de él (como se hizo con RDF y OWL).

Pongamos un ejemplo muy sencillo para que quede claro que XML por si mismo no es suficiente para incluir semántica, debido a que su naturaleza flexible permite diferentes representaciones para describir la misma información, permitiendo una diversidad tan rica que resulta muy difícil, sino imposible, tratar con objetivo semántico.

EJEMPLO DE AMBIGÜEDAD SEMÁNTICA: El alumno LUIS estudia la asignatura código 2003.

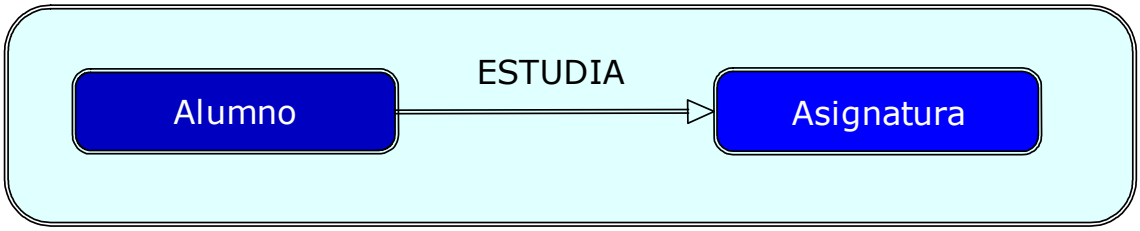


Figura 1. 3

Una posible solución puede ser la que hemos denominado “solucion1”. Se trata de un fichero XML bien estructurado y con unas reglas sintácticas adecuadas definidas a través de un esquema en el fichero XSD (ver tabla 1.2 y 1.3)

solucion1.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<alumno nombre="Luis"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=".\solucion1.xsd">
     <estudia>
          <asignatura codigo="2003"/>
     </estudia>
</alumno>
```

Tabla 1. 2: Solución 1, parte 1

solucion1.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
     <xs:element name="alumno">
          <xs:complexType>
               <xs:sequence>
                    <xs:element ref="estudia"/>
               </xs:sequence>
                    <xs:attribute name="nombre"
                    type="xs:string" use="required"/>
          </xs:complexType>
     </xs:element>
     <xs:element name="asignatura">
          <xs:complexType>
               <xs:attribute name="codigo" type="xs:short"
               use="required"/>
          </xs:complexType>
     </xs:element>
     <xs:element name="estudia">
          <xs:complexType>
               <xs:sequence>
                    <xs:element ref="asignatura"/>
               </xs:sequence>
          </xs:complexType>
     </xs:element>
</xs:schema>
```

Tabla 1. 3: Solución 1, parte 2

Otra solución (ver tabla 1.4 y 1.5) pudiera ser:

Solucion2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<asignatura codigo="2003"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=".\solucion2.xsd">
     <alumno nombre="Luis" estudia="Si"/>
</asignatura>
```

Tabla 1. 4: Solución 2, parte 1

Y su fichero sintáctico asociado:

Solucion2.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
     <xs:element name="alumno">
          <xs:complexType>
               <xs:attribute name="nombre" type="xs:string"
               use="required"/>
               <xs:attribute name="estudia"
               type="xs:boolean" use="required"/>
          </xs:complexType>
     </xs:element>
     <xs:element name="asignatura">
          <xs:complexType>
               <xs:sequence>
                    <xs:element ref="alumno"/>
               </xs:sequence>
                    <xs:attribute name="codigo"
                    type="xs:short" use="required"/>
          </xs:complexType>
     </xs:element>
</xs:schema>
```

Tabla 1. 5: Solución 2, parte 2

Nótese que en este caso incluso se ha definido un tipo de dato booleano. Pero aún hay más, lo representado con esquemas, podría también haber sido especificado mediante DTDs [29].

En conclusión, el lenguaje de etiquetas XML presenta dos importantes características:

- Puede representar cualquier tipo de dato que se pueda codificar como texto sin ambigüedad sintáctica y bien estructurado.

- Es muy flexible, tanto que permite diferentes formatos para representar la misma información. Es decir, XML no tiene una forma única de representar los mismos datos, consecuentemente no es adecuado para incluir semántica.

Por lo tanto, si se añade una capa superior se puede restringir esta flexibilidad, que es un inconveniente para la incorporación de semántica, pero que, por otro lado, permite realizar el desarrollo deseado. De manera, que XML será el soporte para expresar funcionalidades añadidas que convergen a un estándar semántico.

1.5.1.2. RDF y RDFS

Según la W3C, el lenguaje de etiquetas RDF (Resource Description Framework o Infraestructura para la Descripción de Recursos) [Lassila et al; 2001] tiene como objetivo general definir un mecanismo para describir recursos. El mecanismo que utiliza RDF para la definición de recursos es la tripleta, este modelo proporciona un ámbito semántico muy simple, que se basa en tres términos; sujeto o recurso, propiedad o predicado y objeto o literal.

Todas las cosas descritas por expresiones RDF se denominan sujeto o recurso. Un recurso puede ser una página web completa o una parte de ella. Pero un recurso puede ser también una colección completa de páginas (un sitio web completo). Por si esto fuera poco, un recurso puede ser también un objeto físico, por ejemplo un libro impreso. Los recursos se designan siempre por URIs (**U**niform **R**esource **I**dentifier, identificador uniforme de recurso, definido en RFC 2396). Cualquier cosa puede tener un URI; la extensibilidad de URIs permite la introducción de identificadores para cualquier entidad imaginable.

Una propiedad es un aspecto específico, característica, atributo, o relación utilizado para describir un recurso. Para cada propiedad hay que definir sus valores permitidos, los tipos de recursos que puede describir, y sus relaciones con otras propiedades.

Un recurso específico junto con una propiedad y el valor de esta última determinan una sentencia RDF [RDF statement]. El objeto de una sentencia (es decir, el valor de la propiedad) puede ser otro recurso o pude ser un literal; es decir, un recurso (especificado por un URI) o una cadena simple de caracteres [string] u otros tipos de datos primitivos definidos por XML.

Por lo tanto, el sujeto o recurso es la parte a que se refiere la sentencia, la parte que identifica la característica del sujeto es la propiedad o predicado, por último, la parte que identifica el valor de la propiedad es el objeto o literal.

Supongamos que queremos expresar algo sencillo, del estilo, *"la persona X es el autor del sitio web Z", particularizando más "Luis Criado es el autor del sitio web* http://www.luis.criado.org" (ver Tabla 1.6 y Figura 1.4)

En sintaxis RDF

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:s="http://www.ia.criado.org/">
        <rdf:Description about="http://www.luis.criado.org">
                <s:Creator>Luis Criado</s:Creator>
        </rdf:Description>
</rdf:RDF>
```

Tabla 1. 6: Ejemplo de tripleta básica expresada en RDF

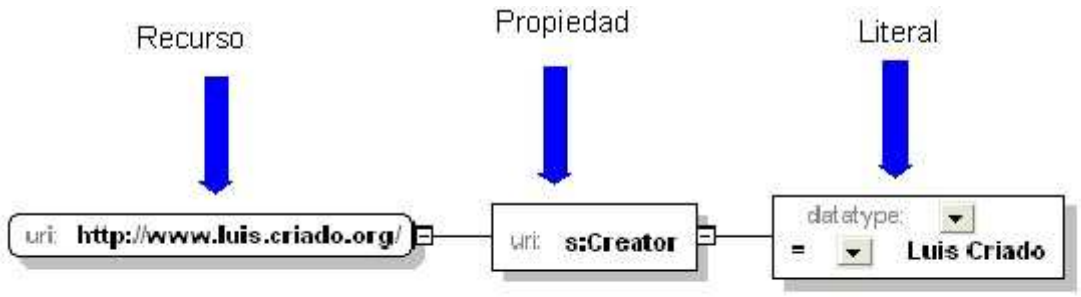


Figura 1. 4: Esquema gráfico de una tripleta

Obsérvese que en este caso, según RDF, "http://www.luis.criado.org" es el sujeto. Sin embargo, en castellano sería "Luis Criado", ya que, responde a la pregunta "¿quien es el *autor del sitio web* http://www.luis.criado.org?". Precisamente, para evitar malas interpretaciones utilizaremos los conceptos de recurso, propiedad y literal.

Ahora se añaden dos propiedades y la frase queda como "Luis Criado tiene como dirección de correo luis@criado.org y es el autor del sitio web http://www.luis.criado.org" (ver Figura 1.5, Tabla 1.7 y Tabla 1.8)

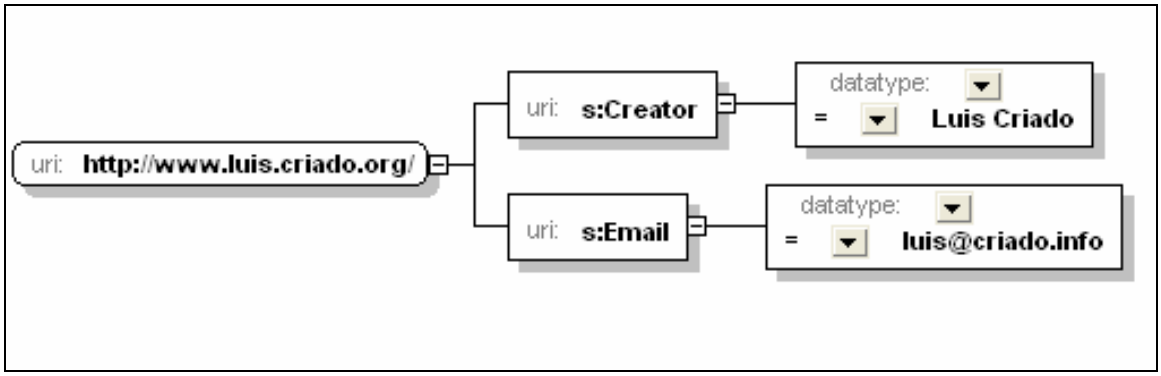


Figura 1. 5: Ejemplo 2 de tripleta gráfica

En sintaxis RDF

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:s="http://www.ia.criado.org/">
        <rdf:Description about="http://www.luis.criado.org">
                <s:Creator>Luis Criado</s:Creator>
                <s:Email>luis@criado.info</s:Email>
        </rdf:Description>
</rdf:RDF>
```

Tabla 1. 7: Ejemplo 2 de tripleta en RDF

El siguiente ejemplo es el mismo pero hemos incorporado una definición de tipos de datos.

En sintaxis RDF

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:s="http://www.ia.criado.org/">
        <rdf:Description about="http://www.luis.criado.org">
       <s:Creator rdf:datatype="http://www.w3.org/2001/XMLSchema#Name">Luis Criado</s:Creator>
                <s:Email rdf:datatype="http://www.w3.org/2001/XMLSchema#string">luis@criado.info</s:Email>
        </rdf:Description>
</rdf:RDF>
```

Tabla 1. 8: Otra forma de plantear el ejemplo 2 de tripleta en RDF

Las especificaciones de RDF [Lassila et al; 2001] definen dos sintaxis básicas: la sintaxis serializada y la sintaxis serializada abreviada que tiene a su vez tres formas de abreviación. RDF requiere de esquemas RDFS [Brickley et al; 2001] que sirven para definir las propiedades aplicables a las clases de objetos y describir las relaciones entre ellas. El núcleo del vocabulario del esquema se define en un namespace denominado ‘rdfs’, y se identifica por el URI de referencia http://www.w3.org/2000/01/rdf-schema#. Esta especificación utiliza también el prefijo 'rdf' para referirse al namespace principal de RDF en el URI referenciado por http://www.w3.org/1999/02/22-rdf-syntax-ns#.

SERIALIZADA

La sintaxis serializada RDF básica en notación EBNF es la siguiente:

```
 ((1)) RDF           ::= (('<rdf:RDF>')) description* (('</rdf:RDF>'))
 ((2)) description   ::= '<rdf:Description' idAboutAttr? '>' propertyElt*
                         '</rdf:Description>'
 ((3)) idAboutAttr   ::= idAttr | aboutAttr
 ((4)) aboutAttr     ::= 'about="' URI-reference '"'
 ((5)) idAttr        ::= 'ID="' IDsymbol '"'
 ((6)) propertyElt   ::= '<' propName '>' value '</' propName '>'
                         | '<' propName resourceAttr '/>'
 ((7)) propName      ::= Qname
 ((8)) value         ::= description | string
 ((9)) resourceAttr  ::= 'resource="' URI-reference '"'
((10)) Qname         ::= (( NSprefix ':' )) name
((11)) URI-reference ::= string, interpreted per ((URI))
((12)) IDsymbol      ::= (any legal XML name symbol)
((13)) name          ::= (any legal XML name symbol)
((14)) NSprefix      ::= (any legal XML namespace prefix)
((15)) string        ::= (any XML text, with "<", ">", and "&" escaped)
```

El elemento **RDF** marca los límites en un documento XML entre los que el contenido está dispuesto a ser mapeado a una instancia de modelo de datos RDF. El elemento **RDF** es opcional si el contenido puede entenderse como RDF desde el contexto de la aplicación.

El elemento **Description** proporciona una forma de dar el nombre adecuado a varias sentencias. Cuando se especifica el atributo **about** con **Description**, la sentencia se refiere al recurso cuyo identificador determina el elemento **about**. Un elemento **Description** sin un atributo **about** representa un nuevo recurso.

El elemento **Description** puede contener más de un elemento ***propertyElt*** con el mismo nombre de propiedad. Dentro del elemento ***propertyElt***, el atributo **resource** específica que otros recursos son el valor de esta propiedad.

Los ***Strings*** deben ser XML bien formados. Los nombres de propiedad pueden asociarse con un esquema. Esto puede hacerse mediante *namespace*.

ABREVIADA

Las especificaciones de RDF definen también una forma abreviada que es una forma XML más compacta.

Se definen tres formas de abreviación para la sintaxis básica serializada.

Primera forma abreviada

Se puede utilizar para propiedades no repetidas dentro de un elemento **Description** donde los valores de dichas propiedades son literales. En este caso, las propiedades se pueden expresar como atributos XML del elemento **Description**. Por ejemplo:

```
<rdf:RDF>
  <rdf:Description about="http://www.w3.org/Home/Lassila">
    <s:Creator>Ora Lassila</s:Creator>
  </rdf:Description>
</rdf:RDF>
```

Se puede expresar como:

```
<rdf:RDF>
  <rdf:Description about="http://www.w3.org/Home/Lassila"
                  s:Creator="Ora Lassila" />
</rdf:RDF>
```

Segunda forma abreviada

La segunda forma abreviada trabaja sobre elementos **Description**. Esta forma abreviada puede emplearse para sentencias específicas donde el objeto de la declaración es otro recurso y el valor de cualquier propiedad dada son **strings**. En este caso se usa una transformación similar de elementos XML que se nombran como atributos XML: las propiedades del recurso en el elemento **Description** anidado puede escribirse como atributos XML del elemento ***propertyElt*** en el que se comprende **Description**.

La siguiente frase de ejemplo:

El individuo al que se refiere el identificador de empleado id 85740 se llama Ora Lassila y tiene la dirección de correo lassila@w3.org. Ese individuo creó el recurso http://www.w3.org/Home/Lassila

se escribe en RDF utilizando la forma serializada explícita como:

```
<rdf:RDF>
 <rdf:Description about="http://www.w3.org/Home/Lassila">
  <s:Creator rdf:resource="http://www.w3.org/staffId/85740"/>
 </rdf:Description>

 <rdf:Description about="http://www.w3.org/staffId/85740">
  <v:Name>Ora Lassila</v:Name>
  <v:Email>lassila@w3.org</v:Email>
 </rdf:Description>
</rdf:RDF>
```

De esta forma se evidencia que se están describiendo dos recursos separados, pero lo que está menos claro es que el segundo recurso se use en la primera descripción. Esta misma expresión podría escribirse de la forma que se señala a continuación para mostrar esta relación más obvia. Nótese que para la máquina, no habría diferencia:

```
<rdf:RDF>
 <rdf:Description about="http://www.w3.org/Home/Lassila">
  <s:Creator>
   <rdf:Description about="http://www.w3.org/staffId/85740">
    <v:Name>Ora Lassila</v:Name>
    <v:Email>lassila@w3.org</v:Email>
   </rdf:Description>
  </s:Creator>
 </rdf:Description>
</rdf:RDF>
```

Utilizando esta segunda sintaxis abreviada, el elemento interior **Description** y las expresiones de las propiedades que contiene pueden escribirse como atributos del elemento **Creator**:

```
<rdf:RDF>
 <rdf:Description about="http://www.w3.org/Home/Lassila">
  <s:Creator rdf:resource="http://www.w3.org/staffId/85740"
   v:Name="Ora Lassila"
   v:Email="lassila@w3.org" />
 </rdf:Description>
</rdf:RDF>
```

Cuando usamos esta forma abreviada, el atributo **about** del elemento anidado **Description** se convierte en un atributo de **resource** en el elemento **propertyElt,** de igual forma que el recurso designado por el URI es en ambos casos el valor de la propiedad **Creator**.

Tercera forma abreviada

La tercera forma de sintaxis abreviada básica se aplica al caso común de un elemento **Description** que contenga una propiedad **type**. En este caso, el tipo de recurso definido en el esquema que corresponde al valor de la propiedad **type** puede utilizarse directamente como un nombre de elemento.

Supongamos, por ejemplo, el fragmento RDF anterior al que hemos añadido el recurso **http://www.w3.org/staffId/85740** que representa una instancia (objeto específico de la categoría) de una persona, podríamos escribirlo en una sintaxis serializada completa, de manera, que tendríamos:

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:s="http://description.org/schema/">
 <rdf:Description about="http://www.w3.org/Home/Lassila">
  <s:Creator>
   <rdf:Description about="http://www.w3.org/staffId/85740">
    <rdf:type resource="http://description.org/schema/Person"/>
    <v:Name>Ora Lassila</v:Name>
    <v:Email>lassila@w3.org</v:Email>
   </rdf:Description>
  </s:Creator>
 </rdf:Description>
</rdf:RDF>
```

y utilizando la tercera forma abreviada, quedaría de la siguiente forma:

```
<rdf:RDF>
 <rdf:Description about="http://www.w3.org/Home/Lassila">
  <s:Creator>
   <s:Person about="http://www.w3.org/staffId/85740">
    <v:Name>Ora Lassila</v:Name>
    <v:Email>lassila@w3.org</v:Email>
   </s:Person>
  </s:Creator>
 </rdf:Description>
</rdf:RDF>
```

En notación EBNF la sintaxis abreviada sustituye las producciones ((2)) y ((6)) de la gramática para la sintaxis serializada básica, de la siguiente forma:

((2a)) description ::= '<*rdf*:Description' idAboutAttr? propAttr* '/>'
| '<*rdf*:Description' idAboutAttr? propAttr* '>'
propertyElt* '</*rdf*:Description>'
| typedNode
((6a)) propertyElt ::= '<' propName '>' value '</' propName '>'
| '<' propName resourceAttr? propAttr* '/>'
((16)) propAttr ::= propName '="' string '"'
(with embedded quotes escaped)
((17)) typedNode ::= '<' typeName idAboutAttr? propAttr* '/>'
| '<' typeName idAboutAttr? propAttr* '>'
property* '</' typeName '>'

1.5.1.3. Semántica sobre RDF y RDFS

Hasta este momento hemos visto que XML y XML Schema aportan la sintaxis para crear documentos bien estructurados, pero sin semántica. El lenguaje RDF y RDF Schema proporciona un modelo de datos para los recursos y las relaciones que se puedan establecer entre ellos, se incorpora la descripción de propiedades y clases de los recursos, además se aporta una semántica básica para establecer jerarquías de generalización entre dichas propiedades y clases. Pero esto no es suficiente para la interoperabilidad semántica.

RDF tiene aún muchas carencias [Abián; 2005]:

- No se pueden declarar restricciones de rango (rdfs:range) sólo para una clase, debido a que rdfs:range define el rango de una propiedad para todas las clases.
- No se pueden representar algunas características de las propiedades. En concreto no se puede declarar que una propiedad es transitiva (si una clase), y tampoco se pueden declarar propiedades del estilo; menor que, simétrica, inversa y única.
- No se puede reflejar que determinadas clases son disjuntas. Por ejemplo, supongamos que en RDF declaramos la clase persona, y las subclases mujer y hombre, entonces no podríamos especificar que una persona no puede ser a la vez mujer y hombre.

- No permite expresar restricciones de cardinalidad. Lo que significa que una propiedad no está restringida en cuanto al número de valores que puede tomar.

1.5.1.4. SPARQL

Una vez establecidos los lenguajes de etiquetas RDF y RDFS que proporcionan la posibilidad de introducir semántica en Web es inmediato plantearse el uso de otro lenguaje que permita consultas sobre el primero. Actualmente se está utilizando el lenguaje SPARQL (basado en RDQL) que es un SQL sobre RDF [Prud'hommeaux et al; 2006], es decir, el lenguaje de consultas para RDF. SPARQL es un lenguaje cuyas especificaciones son del 20 de febrero del 2006; de manera que es un lenguaje muy nuevo, pero ya existen implementaciones, como el caso de Jena [44], que veremos más adelante. Este lenguaje de consultas, al igual que RDF, se basa en la tripleta.

1.5.1.5. OWL

El Lenguaje de Ontologías Web (OWL) en palabras del profesor Ian Horrocks [35], es un "*lenguaje de representación del conocimiento descriptivo y basado en lógica*". Este lenguaje permite la definición de ontologías estructuradas pensado para la Web. Los lenguajes anteriores, como DAML-OIL, se utilizaron para desarrollar herramientas y ontologías para comunidades de usuarios concretos (particularmente en las ciencias y en aplicaciones de comercio electrónico de compañías específicas), pero no fueron definidos para ser compatibles con la arquitectura de la World Wide Web, y en consecuencia con la Web Semántica .

OWL se ha construido sobre RDF y RDF Esquema, permitiendo representar ontologías a partir de un vocabulario más amplio y una sintaxis más fuerte que RDF. El lenguaje OWL proporciona la funcionalidad de aplicar lógica descriptiva. Es por esto que resulta muy interesante el hecho de que OWL puede embeberse en RDF y en consecuencia puede tratarse con SPARQL.

Sintaxis

OWL tiene tres variantes [54]:

- OWL Lite: Esta variante se recomienda para cubrir necesidades de clasificación jerárquica y restricciones simples. Por ejemplo, soporta restricciones cardinales, pero solamente permite valores cardinales de 0 ó 1. *OWL Lite* ofrece una rápida ruta de migración para tesauros y otras taxonomías.

- OWL DL: Las especificaciones del 2004 se basaban en la lógica descriptiva SHOIN(D). Pero OWL continua evolucionando, de manera que en diciembre del 2006 se especifica OWL 1.1 que establece como lógica descriptiva a SROID(D). En definitiva, esta variante de OWL ofrece máxima expresividad conservando la eficacia computacional, además se garantiza la decidibilidad.

- OWL Full: Proporciona soporte a usuarios que requieren el máximo de expresividad y la libertad sintáctica de RDF sin garantías computacionales y por lo tanto sin garantía de decidibilidad.

Cada uno de estos sub-lenguajes es una extensión del anterior. El conjunto siguiente de relaciones es correcto, sin embargo, no sus inversas.

- Cada ontología legal *OWL Lite* es una ontología legal *OWL DL*.
- Cada ontología legal *OWL DL* es una ontología legal *OWL Full*.
- Cada conclusión válida *OWL Lite* es una conclusión válida *OWL DL*.
- Cada conclusión válida *OWL DL* es una conclusión válida *OWL Full*.

En OWL, las clases o conceptos se organizan jerárquicamente mediante "*subClassOf*". Cada clase solo puede tener un padre. La clases pueden contener diversas propiedades especificas del concepto que definen y generales, procedentes de la herencia de otras clases, aunque su valor puede ser particularizado.

Una opción realmente interesante es definir un dominio de una propiedad, de manera, que se limita los individuos a los que se aplica la propiedad. Si una propiedad relaciona un individuo a otro individuo, y la propiedad tiene una clase como uno de sus dominios, entonces el individuo debe pertenecer a la clase. Por ejemplo, la propiedad "hasChild" debe ser establecida para tener el dominio de *Mamífero*. Desde aquí un razonador (ver apartado "1.5.1.8. Razonadores DL") puede deducir que si Pedro hasChild Ana, entonces Pedro debe ser un Mamífero. Los individuos son instancias de clases, y las propiedades deben usarse para

relacionar un individuo con otro.

OWL Lite utiliza únicamente algunas de las características del lenguaje OWL y está más limitado en el uso de características que OWL DL y OWL Full. Por ejemplo, en OWL Lite, las clases sólo pueden ser declaradas en términos de superclases definidas (las superclases no pueden ser expresiones arbitrarias), y sólo pueden ser usados ciertos tipos de restricciones de clase. Además, únicamente se permite la equivalencia entre clases y relaciones de subclases entre clases cuando se trata de clases definidas, no en el caso de expresiones de clases arbitrarias. Igualmente, las restricciones en OWL Lite usan sólo clases definidas. OWL Lite tiene además una noción limitada de cardinalidad - las únicas cardinalidades que se pueden definir explicitamente son 0 ó 1.

OWL DL y OWL Full utilizan el mismo vocabulario aunque OWL DL está sujeto a algunas restricciones. De forma general, OWL DL requiere separación de tipos (una clase no puede ser un individuo o una propiedad, una propiedad no puede ser tampoco un individuo o una clase). Esto implica que no se pueden aplicar restricciones a elementos del lenguaje de OWL (algo que se permite en OWL Full). Además, OWL DL requiere que las propiedades sean del tipo ObjectProperties o del tipo DatatypeProperties: DatatypeProperties son relaciones entre las instancias de clases y literales de RDF y tipos de datos de esquema XML, mientras que ObjectProperties son relaciones entre instancias pertenecientes a dos clases.

1.5.1.6. OWL DL base de la Web Semántica

Como hemos visto en el apartado “1.5.1.3” XML/ XMLS aportan la sintaxis para crear documentos bien estructurados, pero sin semántica y RDF/RDFS proporcionan un modelo de datos para cualquier recurso y las relaciones que se puedan establecer entre ellos, incorporando definición de clases y propiedades, con algunos mecanismos para establecer jerarquías de generalización entre dichas propiedades y clases. Pero esto no es suficiente para la interoperabilidad semántica. OWL DL se define sobre las especificaciones anteriores para corregir las deficiencias semánticas que hemos identificado hasta este momento, de manera que al incorporar OWL DL obtenemos las siguientes ventajas:

- Se pueden declarar restricciones de rango, por propiedad perteneciente a cada clase.

- Se pueden tratar las propiedades transitivas, de relación de orden (menor que, mayor que, etc…), de simetría, inversa y relación única.

- Se puede indicar que determinadas clases son disjuntas. Por ejemplo, supongamos que en OWL declaramos la clase persona, y las subclases mujer y hombre, entonces se puede especificar que una persona no puede ser a la vez mujer y hombre.

- Se permite expresar restricciones de cardinalidad.

1.5.1.7. Editor ontológico: PROTÉGÉ

Actualmente hay una gran variedad de editores de ontologías. Una buena herramienta, muy fácil de usar pero no gratuita, es la propuesta por ALTOVA que ha sido bautizada como "SemanticWorks". Esta herramienta de diseño visual para RDF-OWL [3] es una buena opción para aquellas personas que conocen bien XML y han usado el editor de XML de ALTOVA llamado "XMLSpy". Sin embargo, hoy por hoy, la herramienta más popular para la edición de ontologías es PROTÉGÉ [43], que es fácil de usar, incorpora un panel para realizar consultas SPARQL y está bien documentado, además es gratuita.

En este epígrafe vamos a explicar como se puede escribir una ontología con ayuda de PROTÉGÉ. No tratamos de hacer un manual, solo de explicar cómo podemos crear una ontología de ejemplo en OWL de manera sencilla y rápida. Vamos a crear una ONTOLOGÍA para hacer una clasificación de VERTEBRADOS muy básica. Esta ontología será el ejemplo principal, aunque usaremos referencias a otros ejemplos que no pertenecen a dicha ontología de vertebrados para mostrar algunas otras funcionalidades de OWL con mayor claridad.

Aunque en la actualidad se dispone de varios métodos para la construcción de ontologías, como por ejemplo; DILIGENT, HCOME, METHONTOLOGY, OTK methodology, y Ontology Development 101. Nos hemos decidido por seguir este último, ya que los ejemplos estan basados en la herramienta PROTÉGÉ [Noy et al; 2005]:

- Definir las clases en la ontología de forma jerárquica
- Definir las características (propiedades o slots) de cada clase
- Definir los valores permitidos que tiene cada característica.

Una vez que se tenga una ontología se podrá realizar instancias, es decir, atribuir individuos de una clase determinada.

En PROTÉGÉ-OWL todas las clases nacen de una clase genérica llamada “owl:Thing”, tal y como se comento en el apartado dedicado a OWL. De forma que cuando añadimos una clase raíz debe concebirse como una subclase de “owl:Thing”.

Figura 1. 6: Ontología vertebrados.

Para escribir la jerarquía de clases debemos saber que los vertebrados (ver Figura 1.6) se clasifican en peces, anfibios, reptiles, aves y mamíferos. Estos últimos pueden ser de tres tipos: placentarios, monotremas (ovíparos) y marsupiales (poseen bolsas para proteger a sus crías, como es el caso del canguros). Los reptiles pueden ser de cuatro tipos: cocodrilos y caimanes, lagartos y serpientes, tortugas y, finalmente, tuátaras. En cuanto a los anfibios, se distinguen tres grandes grupos: cecilidos, salamandras, y sapos y ranas. Si bien existen peces con esqueleto cartilaginosos, también hay una gran variedad de peces con esqueleto óseo; en este caso, se clasifican por su hábitat como de peces de agua dulce, de rio-mar y de mar.

Siguiendo con nuestro principal ejemplo, la clase mamífero, que hereda de la clase vertebrados, se expresa en OWL de la siguiente forma:

```
<owl:Class rdf:about="#mamiferos">
  <rdfs:subClassOf rdf:resource="#vertebrados"/>
</owl:Class>
```

Tabla 1. 9: Clase mamifero hereda de vertebrado

Llegados a este punto, comenzamos a establecer la relaciones entre clases de forma que, si un animal es un mamífero, entonces no puede ser al mismo tiempo un pez, un reptil, un anfibio o un ave, por lo que se trata de clases disjuntas.

```
<owl:Class rdf:about="#mamiferos">
  <owl:disjointWith>
    <owl:Class rdf:about="#aves"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#reptiles"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#anfibios"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#peces"/>
  <rdfs:subClassOf rdf:resource="#vertebrados"/>
</owl:Class>
```

Tabla 1. 10: Clases disjuntas

Hay dos tipos de propiedades:

Propiedades de objetos: Son las que relacionan un objeto con otro. Por ejemplo:

```
<owl:ObjectProperty rdf:ID="daClaseDe">
  <owl:domain rdf:resource="#Profesor"/>
  <owl:range rdf:resource= "#Asignatura"/>
    <rdfs:subPropertyOf rdf:resource="#estáRelacionadoCon"/>
</owl:ObjectProperty>
```

Tabla 1. 11: Propiedades de objetos

Se pueden usar varias propiedades interesantes, como la inversa, simétrica, transitiva, etc. La propiedad transitiva se formula de la siguiente forma:

$$P \text{ inverseOf } Q \Rightarrow P(x,y) \Leftrightarrow Q(y,x)$$

Tabla 1. 12: Propiedad transitiva

Esta propiedad en OWL se representa de la siguiente manera:

```
<owl:ObjectProperty rdf:ID="isOfferedAt">
   <rdfs:range rdf:resource="#Destination"/>
   <rdfs:domain rdf:resource="#Activity"/>
   <owl:inverseOf>
     <owl:ObjectProperty rdf:about="#hasActivity"/>
   </owl:inverseOf>
 </owl:ObjectProperty>
```

Tabla 1. 13: Propiedad transitiva en OWL

Propiedades de tipo de dato: Son las que relacionan objetos con valores de tipo dato, como son; *enteros, literales, etc*. Normalmente se utilizan los tipos de datos de XML Schema. Hemos identificado características comunes para todos los vertebrados que toman valores diferentes en cada clase, de manera que asignamos estas propiedades a la clase "vertebrados" para que hereden estas propiedades todas las subclases. Las propiedades identificadas junto con los valores asignados para cada clase se muestran en la siguiente tabla:

	PECES	*ANFIBIOS*	*REPTILES*	*AVES*	*MAMIFEROS*
RESPIRACION	branquias	Branquias Pulmonar Cutánea	Pulmonar	Pulmonar	Pulmonar
NUM. PATAS	Sin patas	Cuatro o ninguna	Cuatro o ninguna	dos	cuatro
REPRODUCCIÓN	huevos	huevos	huevos	huevos	Huevos o interna
SANGRE	fría	fría	fría	caliente	caliente
PIEL	escamas	Desnuda, Húmeda y escurridiza	Gruesa con escamas	plumas	Piel (o no)

Tabla 1. 14: Propiedades de vertebrados

En PROTÉGÉ se define en la pestaña “properties”, tal y como se aprecia en la siguiente captura de pantalla:

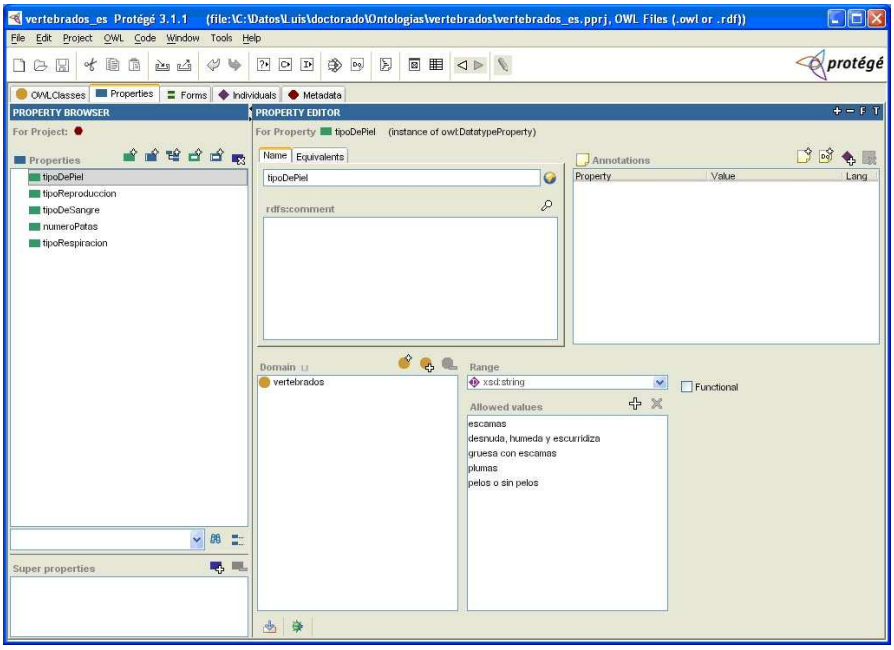


Figura 1. 7: Definicion de propiedades en PROTÉGÉ

En cuanto a las restricciones se distinguen tres mecanismos en OWL para realizar restricciones:

1. Clases como restricciones
2. Restricciones de propiedades tipo dato
3. Restricciones de propiedades de objeto

1.-) En este caso, se puede declarar que la clase C satisface ciertas condiciones o restricciones (todos sus individuos deben satisfacerlas). La manera de anotar en OWL esta forma de restricción puede verse en la (tabla 1.15).

```
<owl:Class rdf:about="#personalDocente">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#daClaseDe"/>
   <owl:someValuesFrom rdf:resource="#Asignatura"/>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

Tabla 1. 15: Clases como restricciones

Esto tiene el mismo efecto que decir que C es una subclase de C', donde C' incluye todos los objetos que satisfacen dichas condiciones. Por lo tanto, se puede construir clases que contienen restricciones y que, cuando otras heredan de ellas, automáticamente incorporan las restricciones de la clase padre.

2.-) Como se ha visto anteriormente, hemos establecido restricciones en cada clase. Estas pueden ser condiciones necesarias o necesarias y suficientes. Estas restricciones de clase se basan en formular reglas en base a las propiedades. En nuestro ejemplo, sabemos que las aves tienen plumas y dos patas. No existe ningún otro vertebrado que tenga estas características, de manera que son necesarias y suficientes.

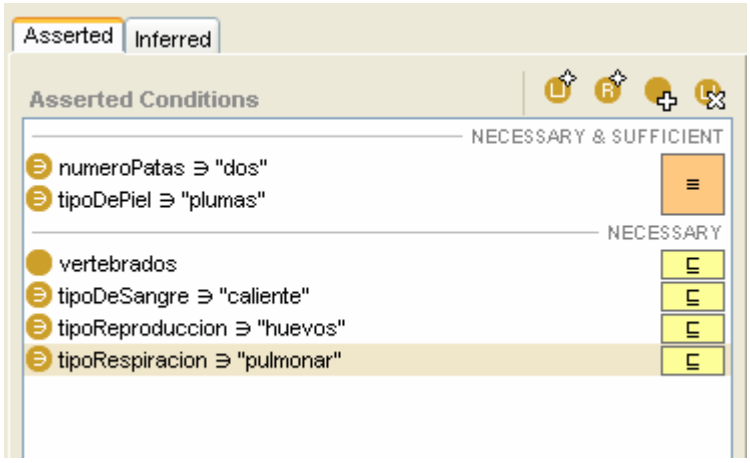


Figura 1. 8: Restricciones de propiedades tipo dato

Las reglas necesarias en OWL son:

```
</owl:equivalentClass>
 <rdfs:subClassOf>
   <owl:Restriction>
     <owl:onProperty>
       <owl:DatatypeProperty rdf:ID="tipoReproduccion"/>
     </owl:onProperty>
     <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string">huevos
          </owl:hasValue>
   </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
   <owl:Restriction>
     <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string">caliente
          </owl:hasValue>
     <owl:onProperty>
       <owl:DatatypeProperty rdf:ID="tipoDeSangre"/>
     </owl:onProperty>
   </owl:Restriction>
 </rdfs:subClassOf>
```

Tabla 1. 16: Restricciones propiedades tipo dato en OWL (necesarias)

Las reglas necesarias y suficientes en OWL son:

```
<owl:equivalentClass>
   <owl:Class>
     <owl:intersectionOf rdf:parseType="Collection">
       <owl:Restriction>
         <owl:onProperty>
           <owl:DatatypeProperty rdf:ID="numeroPatas"/>
         </owl:onProperty>
         <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string">dos
              </owl:hasValue>
       </owl:Restriction>
       <owl:Restriction>
         <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string">plumas
              </owl:hasValue>
         <owl:onProperty>
           <owl:DatatypeProperty rdf:ID="tipoDePiel"/>
         </owl:onProperty>
       </owl:Restriction>
     </owl:intersectionOf>
   </owl:Class>
```

Tabla 1. 17: Restricciones propiedades tipo dato en OWL (necesarias y suficientes)

En resumen, se pueden utilizar las siguientes restricciones:

allValuesFrom: Indica que todos los valores deben ser de un tipo. Los que no tiene ningún valor, también cumplen la condición.

someValuesFrom: Al menos la propiedad debe tener un valor, por ejemplo, un estudiante es una persona que cursa al menos una asignatura. Pero puede tener más.

hasValue: La propiedad tiene un único valor.

minCardinality, maxCardinality: Restringen el número máximo/mínimo de valores

3.-) Las restricciones de propiedades de objeto van a permitir trabajar con lógica de primer orden fácilmente, ya que, aportan propiedades especiales, como son:

TransitiveProperty: Si P(x,y) y P(y,z) entonces P(x,z)
SimmetricProperty: Si P(x,y) entonces P(y,x)
FunctionalProperty. Si una propiedad es una FunctionalProperty, entonces no tiene más que un valor para cada individuo. Esta característica se denomina como una propiedad única.

InverseFunctionalProperty, inverseOf: Si P(x,y) y P(z,y) entonces x = z..

Una propiedad puede ser establecida para ser la inversa de otra propiedad. Si se establece la propiedad P1 como inversa de la propiedad P2, entonces si X se relaciona con Y por la propiedad P2 implica que Y estará relacionada con X por la propiedad P1. Por ejemplo, si **hasChild** es la inversa de **hasParent** y *Juana* **hasParent** *Luisa*, entonces, un razonador deduce que *Luisa* **hasChild** *Juana*.

Una instancia es un individuo de una clase. Para realizar una instancia es necesario antes definir la jerarquía de clases completa. En nuestro caso hemos partido de la clase "vertebrados" hasta llegar a la subclase "perros" (ver Figura 1.9).

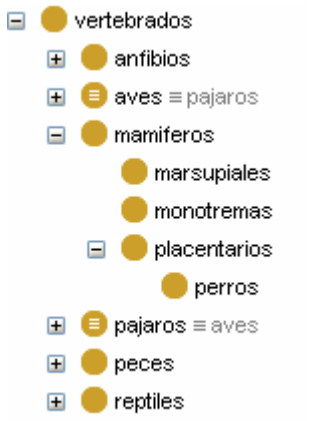


Figura 1. 9: Jerarquía de clases

Definimos la clase "perros" desde PROTÉGÉ y obtendremos automáticamente su representación en OWL (Tabla 1.18).

La clase perro en OWL es:

```
<owl:Class rdf:about="#placentarios">
   <rdfs:subClassOf rdf:resource="#mamiferos"/>
 </owl:Class>
 <owl:Class rdf:ID="perros">
   <rdfs:subClassOf>
    <owl:Restriction>
      <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >ladrar</owl:hasValue>
      <owl:onProperty>
        <owl:DatatypeProperty rdf:ID="sonido"/>
      </owl:onProperty>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Class rdf:ID="placentarios"/>
   </rdfs:subClassOf>
 </owl:Class>
```

Tabla 1. 18: Clase perro en OWL

Una vez definido la clase “perros” podemos realizar una instancia de una raza de perros en particular, como el “doberman”. En la Figura 1.10 se muestra una captura de pantalla de PROTÉGÉ en la cual se ha definido “doberman”, la Tabla 1.18 es su representación en OWL.

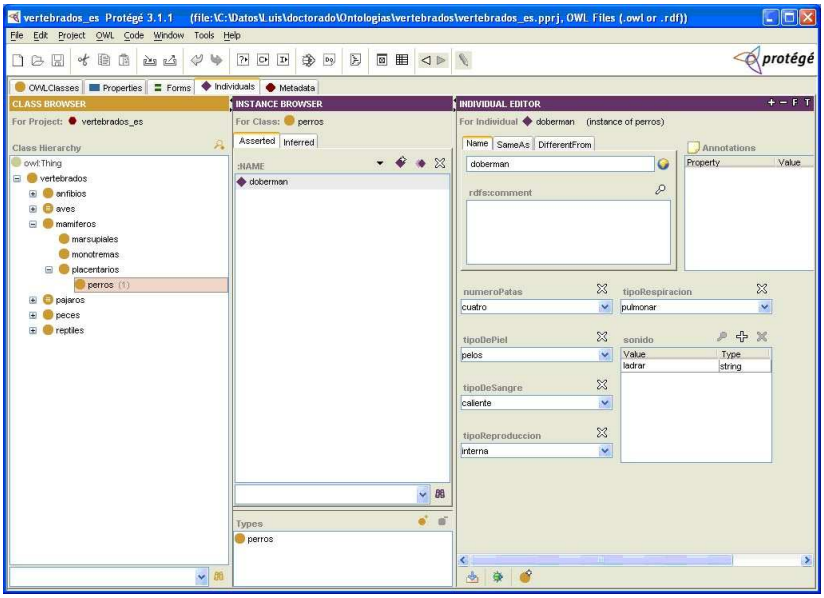


Figura 1. 10: Instancia de la clase perro en PROTÉGÉ

```
<perros rdf:ID="doberman">
   <numeroPatas rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >cuatro</numeroPatas>
   <tipoDeSangre rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >caliente</tipoDeSangre>
   <tipoRespiracion rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >pulmonar</tipoRespiracion>
   <tipoReproduccion rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >interna</tipoReproduccion>
   <tipoDePiel rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >pelos</tipoDePiel>
 </perros>
```

Figura 1. 11: Instancia de la clase perro.

1.5.1.8. Razonadores DL

La lógica descriptiva se basa en representar el conocimiento utilizando una terminología o vocabulario del dominio (TBOX) y un conjunto de afirmaciones (ABOX) [66].

Por ejemplo, si tenemos como conceptos atómicos "Persona" y "Mujer" y como relaciones "tieneHijo", "tieneMarido" y "tieneEsposa", podríamos tener como TBOX algo como lo siguiente:

$$Hombre \equiv Persona \cap -Mujer$$
$$Madre \equiv Mujer \cap \exists tieneHijo.Persona$$
$$Padre \equiv Hombre \cap \exists tieneHijo.Persona$$
$$Padres \equiv Madre \cap Padre$$
$$Progenitor \equiv Madre \cup Padre$$
$$Esposa \equiv Mujer \cap \exists tieneMarido.Hombre$$
$$Marido \equiv Hombre \cap \exists tieneEsposa.Mujer$$

La información recogida en la ABOX podría ser algo como:

$$tieneHijo(Luis, Alfonso)$$
$$tieneHijo(Teresa, Alfonso)$$
$$Padre(Luis)$$
$$Madre(Teresa)$$
$$Padres(Luis, Teresa)$$
$$Progenitor(Luis, Teresa)$$
$$Esposa(Teresa, Luis)$$
$$Marido(Luis, Teresa)$$

Las herramientas que permiten razonar sobre las TBOX y ABOX se denominan "razonadores DL" y permiten comprobar la coherencia de la ontología y operar con ella aplicando mecanismos de lógica descriptiva, es de destacar las siguientes funcionalidades [67]:

- Comprobación de coherencia de una ontología (se comprueba que la ontología no contiene hechos contradictorios). Si no se cumple, la ontología es inconsistente.

- Satisfacibilidad: El razonador determina si es posible que una clase tenga instancias. En el caso de que un concepto sea insastisfacible, la ontología será inconsistente.

- Clasificación de la ontología. A partir de los axiomas declarados en la *T-Box* de la ontología las relaciones de subclase entre todos los conceptos declarados explícitamente (tienen una URI) para construir la jerarquía de clases. Esta jerarquía de clases se puede utilizar para formular *queries* como: todas las subclases de un concepto, inferir nuevas subclases de un concepto, las superclases directas, etc.

- Instanciación: Un razonador DL puede inferir cuáles son las clases a las que directamente pertenece un objeto o instancia. Si además se utiliza la jerarquía inferida, es posible obtener todas la clases a las que indirectamente pertenece una instancia dentro de la ontología.

Hay muchos razonadores para lógica descriptiva, a continuación se enumeran un par de razonadores implementados en Java:

- KAON2 [68] es un razonador semántico desarrollado por la Universidad de Manchester, la Universidad de Karlsruhe y el "Research Center for Information Technologies" (FZI), que permite la manipulación y ejecución de inferencias con ontologías representadas en OWL DL, SWRL y F-Logic.

- Pellet [69] es un razonador desarrollado por la Universidad de Maryland que permite realizar inferencias con ontologías OWL DL.

1.5.1.9. Interfaces de programación

Hay varios interfaces de programación de aplicaciones (API) o frameworks (entornos de trabajo) que permiten la creación y tratamiento de ontologías en sintaxis OWL. En esta tesis, nos hemos centrado en los APIs para Java. Entre estos APIs hay que mencionar; OWL API [9], Sesame+OWLIM [10] [11] y Jena [44].

El OWL API está enfocado a OWL lite y OWL DL, y, por supuesto, implementa un motor de inferencia, pero es un API que carece de una mínima documentación. Ni siquiera en el momento de plantear esta tesis, expone en su web un solo ejemplo. Afortunadamente hay más

APIs para Java. Una alternativa es Sesame (API Java para RDF), siempre y cuando lo completemos con OWLIM, que proporciona una capa de inferencia de alto rendimiento para el tratamiento de OWL basado en SESAME. OWLIM puede manejar millones de declaraciones explícitas en un ordenador de sobremesa. La limitación principal de OWLIM es la lentitud en la operación de borrado. Sin embargo, la descarga y la evaluación de las consultas son muy rápidas. Otra opción es usar Jena. Este API integra RDF y OWL, y, al igual que SESAME-OWLIM, dispone de poca documentación pero nos permite afrontar desarrollos para el tratamiento de archivos en OWL. La razón de decidirnos por Jena, que es el único entorno de programación que comentaremos, es por haber encontrado mayor documentación y más ejemplos en el momento de afrontar esta investigación, pero es sólo una herramienta, por lo que no condiciona de ninguna manera las ideas y planteamientos que este trabajo propone.

Jena

Jena es un API para Java desarrollado por los laboratorios de investigación semántica de HP [44]. Este API aporta, por un lado, el tratamiento de cualquier archivo RDF y/o OWL, entendiendo por tratamiento cualquier tarea de lectura o escritura, como puede ser parsear, navegar entre los nodos, añadir información, etc. Por otro lado, provee de un "Reasoner", esto es, un motor de razonamiento que implementa la funcionalidad de inferir o deducir nuevo conocimiento. En base a lo expuesto vamos a dedicar este apartado a presentar solamente las funcionalidades que proporciona el API Jena que hemos utilizado en esta tesis, ya que con ello construiremos la herramienta de ayuda a usuarios activos de la web y el prototipo de Buscador Semántico de propósito general.

Tanto si trabajamos con RDF como con OWL se debe declarar un objeto "Model" que representa un modelo en memoria de RDF-OWL [Powers; 2003]. Para declarar y crear un modelo (vacío) es suficiente la línea de código de la tabla siguiente:

```
// Creamos un modelo vacio
    Model model = ModelFactory.createDefaultModel();
```

Tabla 1. 19: Creación de un modelo

Una vez creado un modelo deberemos declarar al menos un recurso y alguna propiedad asociada; vamos a usar algunos ejemplos de la documentación oficial de Jena. Para ello nos situamos en el caso más sencillo que se representa en el siguiente esquema:

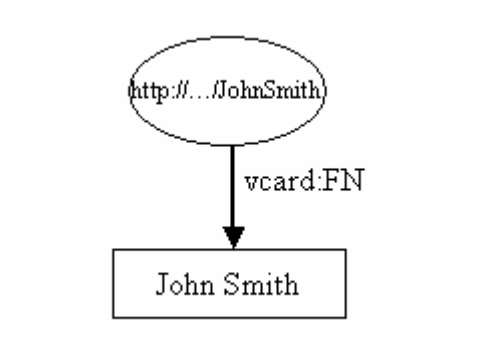


Figura 1. 12: Esquema más sencillo: un recurso y una propiedad

En este caso solo tenemos un recurso y una propiedad asociada, de manera, que una vez que hemos creado el modelo vacío, habrá que crear el recurso. Para crear el recurso utilizaremos el método "createResource()" y para añadir una propiedad "addProperty()" (ver la tabla siguiente)

```
String personURI    = "http://www.john.smith.org";
String fullName     = "John Smith";

// Creamos el recurso
Resource johnSmith = model.createResource(personURI);

// Añadimos una propiedad
johnSmith.addProperty(VCARD.FN, fullName);

// escribimos el modelo
model.write(System.out);
```

Tabla 1. 20: Escribir un modelo básico con Jena

Antes de continuar, es interesante comentar que VCARD es un vocabulario embebido en RDF, al igual que OWL puede considerarse otro vocabulario embebido en RDF. Sin embargo, VCARD es algo muy simple que se refiere a "tarjeta de presentación". La ejecución de estas líneas de código (Tabla 1.20) da lugar al siguiente fichero RDF (Tabla 1.21):

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#" >
  <rdf:Description rdf:about="http://www.john.smith.org">
   <vcard:FN>John Smith</vcard:FN>
  </rdf:Description>
</rdf:RDF>
```

Tabla 1. 21: Resultado RDF obtenido con Jena

Para profundizar un poco más vamos a agregar al ejemplo anterior más detalles. En particular vamos a agregar otra propiedad que a su vez toma otro recurso como valor. El siguiente modelo (Figura 1.13) proporciona un ejemplo de una propiedad de este tipo.

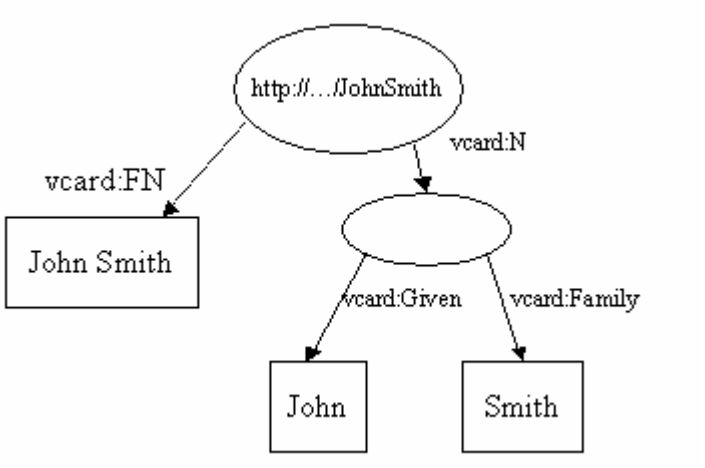


Figura 1. 13: Propiedades con valores

De la misma forma que en el ejemplo anterior hay que declarar el modelo (Tabla 1.22):

```
// Creamos un modelo vacio
    Model model = ModelFactory.createDefaultModel();
```

Tabla 1. 22: Declaración de un modelo

También definimos algunas variables tipo String (Tabla 1.23):

```
// Definimos varibles
    String personURI    = "http://www.john.smith.org";
    String givenName    = "John";
    String familyName   = "Smith";
    String fullName     = givenName + " " + familyName;
```

Tabla 1. 23: Definición e inicialización

Se debe añadir una nueva propiedad (N) que representa la estructura del nombre de John Smith. Dicha propiedad toma un recurso como valor en vez de un literal. Es importante aclarar que el recurso que representa al nombre compuesto no tiene URI asociado, es por esto, que se denomina "nodo en blanco". El código para construir este ejemplo puede verse en la tabla siguiente:

```
 Resource johnSmith
     = model.createResource(personURI)
      .addProperty(RDF.value, fullName)
      .addProperty(VCARD.N,
                model.createResource()
                    .addProperty(VCARD.Given, givenName)
                    .addProperty(VCARD.Family, familyName));

// escribimos el modelo
model.write(System.out);
```

Tabla 1. 24: Escribir modelo

Tras ejecutar el código de la "Tabla 1.24" se obtiene el resultado siguiente en RDF (Tabla 1.25).

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#" >
  <rdf:Description rdf:about="http://www.john.smith.org">
   <rdf:value>John Smith</rdf:value>
   <vcard:N rdf:nodeID="A0"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="A0">
   <vcard:Given>John</vcard:Given>
   <vcard:Family>Smith</vcard:Family>
  </rdf:Description>
</rdf:RDF>
```

Tabla 1. 25: Representación RDF

Cada arco en un modelo RDF (Figura 1.13) se denomina enunciado. Cada enunciado expresa un hecho sobre un recurso. Un enunciado o sentencia, como decíamos en el apartado dedicado a RDF, tiene tres partes: el sujeto (el recurso sobre el cual se expresa el enunciado), el predicado (la propiedad que da nombre al arco) y el objeto (el recurso o literal que toma la propiedad como valor). Por lo tanto, un modelo RDF es un conjunto de enunciados. La clase "Model" proporciona el método "listStatements()" que retorna un iterador sobre todos los enunciados del modelo. El interfase "Statement" proporciona métodos para acceder al recurso, a la propiedad y al objeto de un enunciado.

El siguiente código Java-Jena (Tabla 1.26) extiende el ejemplo anterior para listar todos los enunciados del modelo. La Tabla 1.27 muestra los resultados de la ejecución.

```
StmtIterator iter = model.listStatements();
// imprime el predicado, sujeto y recurso        while (iter.hasNext()) {
   Statement stmt      = iter.nextStatement();         // get next statement
   Resource  subject   = stmt.getSubject();   // get the subject
   Property  predicate = stmt.getPredicate(); // get the predicate
   RDFNode   object    = stmt.getObject();    // get the object
   System.out.print("sujeto ["+subject.toString()+"]");
   System.out.print(" predicado[" + predicate.toString() + "] ");
   if (object instanceof Resource) {
      System.out.print(" recurso ["+object.toString()+" ]");
   } else {         // object is a literal
      System.out.print(" recurso_literal[" + object.toString() + "]");
   }
   System.out.println(" .");
}
```

Tabla 1. 26: Lista de todos los enunciados

El resultado es del estilo siguiente:

```
Sujeto[http://www.john.smith.org]
predicado[http://www.w3.org/1999/02/22-rdf-syntax-ns#value]
recurso_literal[John Smith].
……..
sujeto[6952c226:10a37d51c9d:-8000]
predicado[http://www.w3.org/2001/vcard-rdf/3.0#Family]
recurso_literal[Smith].
```

Tabla 1. 27: Ejecución de extración de enunciados

Las tres operaciones que proporciona Jena para manipular modelos son:

- UNIÓN: todos los enunciados de los dos modelos se suman
- INTERSECCIÓN: Solo los enunciados comunes
- DIFERENCIA: Los enunciados no comunes

Estas operaciones otorgan una potencia extraordinaria, ya que, integran la información procedente de diferentes fuentes. Es decir, cada modelo puede estar en una máquina distinta y a partir de ellos se realiza una de las operaciones descritas anteriormente. Supongamos que tenemos dos modelos, que representan instancias de la misma ontología (Figura 1.14)

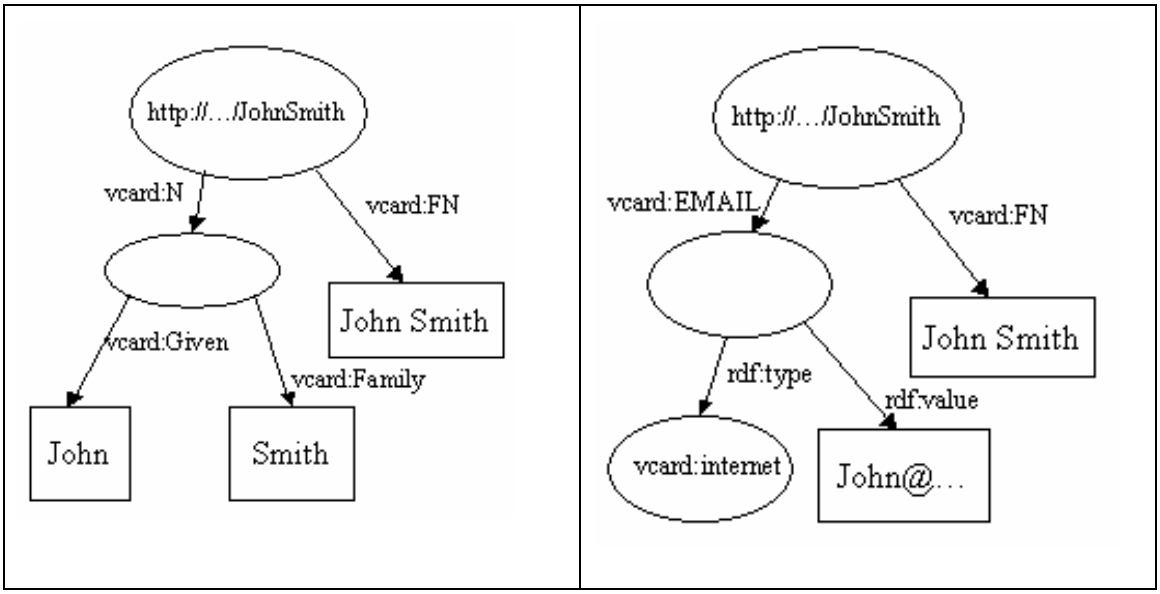


Figura 1. 14: Dos modelos que pertenecen a la misma ontología

Entonces si realizamos la operación de UNIÓN como se especifica en la Tabla 1.28, obtendremos un modelo unificador (Figura 1.15)

```
// read the RDF/XML files
model1.read(new InputStreamReader(in1), "");
model2.read(new InputStreamReader(in2), "");

// merge the Models
Model model = model1.union(model2);

// print the Model as RDF/XML
model.write(system.out, "RDF/XML-ABBREV");
```

Tabla 1. 28: Unión con JENA de dos modelos

Obtendremos como resultado el siguiente modelo

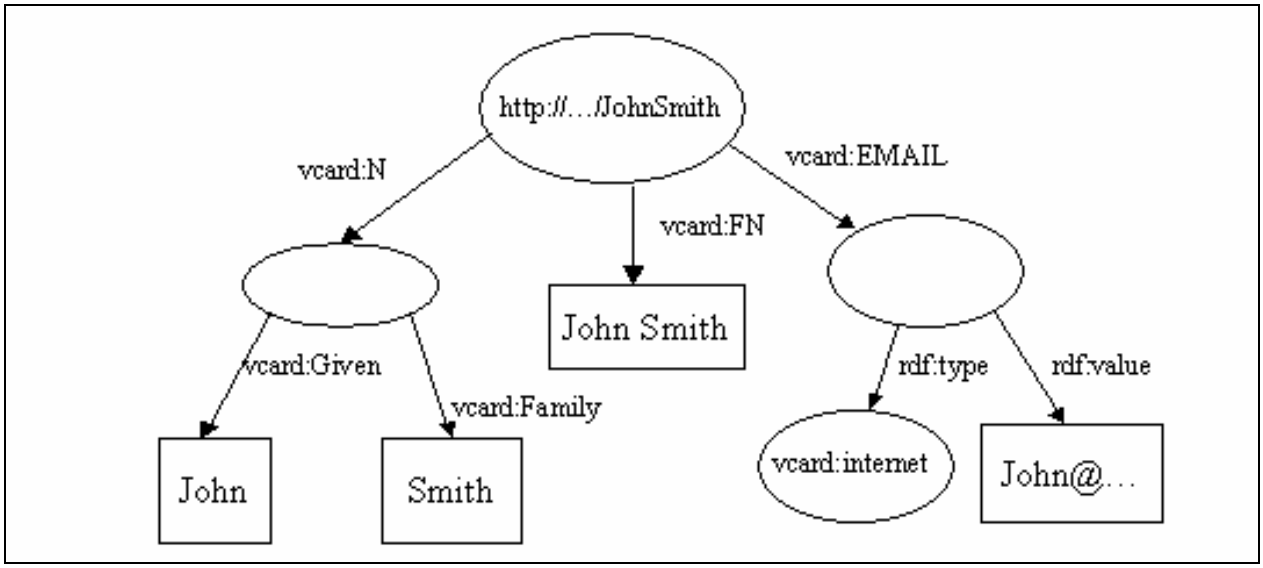


Figura 1. 15: Union de dos modelos

Expresado en RDF:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#">
 <rdf:Description rdf:about="http://somewhere/JohnSmith/">
   <vcard:EMAIL>
     <vcard:internet>
       <rdf:value>John@somewhere.com</rdf:value>
     </vcard:internet>
   </vcard:EMAIL>
   <vcard:N rdf:parseType="Resource">
     <vcard:Given>John</vcard:Given>
     <vcard:Family>Smith</vcard:Family>
   </vcard:N>
   <vcard:FN>John Smith</vcard:FN>
 </rdf:Description>
</rdf:RDF>
```

Tabla 1. 29: Ejemplo vcard con RDF

Como se puede comprobar, esta versión de Jena carga en memoria RAM la unión de todos los modelos, sean ontologías o instancias y procedentes de cualquier fuente (tripletas persistentes o ficheros owl). Pero el razonador de Jena es inestable cuando opera con volumenes de información grandes procedentes de varias fuentes. En nuestros experimentos nos ha fallado por problemas de memoria en varias ocasiones, incluso ampliando la memoria de la máquina virtual java, de forma que finalmente, solo utilizamos la funcionalidad de volcar ontologías e instancias al modelo persistente. Finalmente, comentar que, basándose en el API de Jena, se ha desarrollado otro API llamado Jastor [12] que aporta algunas funcionalidades interesantes al desarrollador. En pocas palabras, permite generar una jerarquía de clases y métodos automáticamente en lenguaje Java para tratar un fichero particular en OWL; es decir, genera código fuente. Sin embargo, para nuestro desarrollo no servirá, ya que nuestro problema es enfrentarnos a OWL desconocidos y tratarlos en tiempo real, por lo que no se puede plantear una etapa de compilación posterior a la adquisición de un determinado fichero OWL.

1.5.2. Microformatos y vocabularios.

Una propuesta para generar semántica son los microformatos [19], etiquetas que intentan enriquecer el HTML actual dotándolo de semántica; en concreto es interesante comentar los Metadatos Dublin Core. Este microformato trata de introducir contenido semántico aprovechando las características de los atributos "cid" ó "class" usada por algunas etiqueta de HTML. Sin embargo, HTML fue concebido para la presentación de información y no para estructurarla, por lo que parece difícil pensar que con pequeños cambios obtendremos un HTML semántico.

Dos de los ejemplos más conocidos de la Web Semántica son RSS y FOAF. RSS [16] es un vocabulario RDF basado en XML que permite la catalogación de información (noticias y eventos) de tal manera que sea posible encontrar información precisa adaptada a las preferencias de los usuarios, y FOAF es un proyecto de Web Semántica que permite crear páginas web para describir personas [17], vínculos entre ellas y cosas que hacen y crean. Se trata de un vocabulario RDF, que permite tener disponible información personal de forma sencilla y simplificada para que pueda ser procesada, compartida y reutilizada. Sin embargo, ambas aplicaciones incorporan una semántica elemental que es la que proporciona la tecnología en la que se sustentan, es decir, RDF. Pero, como se discutió anteriormente, RDF tiene muchas carencias [Abián; 2005] que fueron detalladas en el apartado dedicado al mismo y que podemos resumir diciendo que tiene limitado las restricciones de rango y cardinalidad, limita en exceso la representación de propiedades y no permite clases disjuntas. Hoy parece que con OWL se ha logrado superar las restricciones de RDF y que es la manera de representar conocimiento.

De forma, que para plantear aplicaciones semánticas de futuro tenemos que consolidar una tecnología que permita tal desarrollo. La “Red Temática de Web Semántica” española, es un ejemplo del esfuerzo de investigación que sin duda ha contribuido en esta consolidación. En esta red participan un total de 200 investigadores con nacionalidad o residencia española distribuidos entre 27 grupos de investigación españoles y algunos centros de investigación extranjeros. En los últimos 5 años, los grupos de investigación integrantes de la red han participado en aproximadamente 150 Proyectos (MKBEEM, OntoWeb, PIKON, Esperonto, KW, OntoGrid, SEEMP, etc) financiados con recursos públicos y privados.

1.5.3. Proyectos europeos sobre la WS: NeOn y ASK-IT

Aunque hay muchísimos proyectos relacionados con la Web Semántica, vamos a comentar dos de los más importantes, en base a su presupuesto, como son NeOn y ASK-IT.

El proyecto NeOn [13], dirigido por el profesor Enrico Motta [14] del Knowledge Media Institute (KMi) perteneciente a la Open University de Reino Unido [15], es un proyecto europeo que cuenta con un presupuesto de 14,7 millones de euros. NeOn está enfocado hacia dos sectores tan diferentes como el farmacéutico y el de la agricultura y pesca, que manejan un enorme volumen de datos distribuidos difíciles de integrar y tratar con la tecnología actual. Con NeOn será posible ordenar esta información de manera sensata y sencilla, asociando etiquetas semánticas a los datos que, a su vez, se pueden combinar para permitir a los usuarios localizar datos conectados por su significado o por las palabras que se usan en su descripción. Las instituciones de investigación que participan en el proyecto NeOn son equipos líderes a escala mundial en varios campos relacionados con la Web Semántica.

El proyecto ASK-IT [70] es otro gran proyecto europeo con un presupuesto de alrededor de 15 millones de euros, es un proyecto donde se intenta aplicar la tecnología de la Web Semántica con el objetivo de apoyar y promocionar soluciones a la movilidad de las personas con limitaciones para desplazarse. Este proyecto se ha dividido en cinco subproyectos, el primero de ellos comenzó en octubre del 2004 y el último terminó en diciembre del 2008:

- SP1: Especificación de contenidos adecuados (*Content for All*)
- SP2: Diseño de aplicaciones accesibles (*Tools for All*)
- SP3: Implantación de la plataforma (*Ambient Intelligence Framework*)
- SP4: Desarrollo de los pilotos (*Accesible Europe*)
- SP5: Actividades horizontales (*Horizontal Activities*)

Se ha creado el Consorcio Ask-It, que cuenta con más de 50 socios de 15 países europeos y un país americano. En Ask-It participan empresas, universidades, administraciones y organizaciones que representan colectivos con dificultades de movilidad; como pueden ser las organizaciones de ancianos y personas con discapacidad.

Los casos de uso que se han considerado se pueden resumir en nueve categorías:

- Categoria 0: Registro del usuario.
- Categoria 1: Planificación del viaje y guiado.
- Categoria 2: Almacenamiento de información.
- Categoria 3: Servicios de transporte.
- Categoria 4: Asistencia personal.
- Categoria 5: Control ambiental.
- Categoria 6: E-Services.
- Categoria 7: Servicios de viajes y ocio.
- Categoria 8: Contactos sociales.
- Categoria 9: Emergencias.

En consecuencia, en Ask-It se han desarrollado varias ontologías disponibles [71]. La primera es una ontología general para Ask-It y posteriormente se han diseñado las siguientes ontologías:

- Servicios de apoyo personal.
- Transporte.
- Turístico y ocio.
- El trabajo, los negocios y el contenido de apoyo a la educación.
- Las relaciones sociales y comunitarias de fomento de los servicios.

Sobre estas ontologías se ha diseñado un sistema que permite que un usuario se conecte desde cualquier terminal; un teléfono móvil, un portátil, una PDA, etc.. y solicite una información al servidor Ask-It que determinará en que ciudad está de entre las ciudades pilotos que han participado (Madrid, Genoa, Athens/Thessaloniki, Bucharest, Nuremberg, Helsinki, Newcastle y Den Haag). Una vez localizado, imaginemos que está en Madrid y se solicita un servicio para conocer un destino, el sistema mediante el uso de ontologías localizará la ruta más adecuada en transporte público, conectándose a un servicio web del Consorcio de Transportes de Madrid (CRTM), después conectará con el Ayuntamiento de Madrid para buscar los puntos de interés en un radio de 300 metros del destino y con esta información el sistema Ask-It contestará al usuario. Pero la respuesta será diferente en función de las dificultades de movilidad del usuario.

1.6. Arquitectura de la Web Semántica

La capa de aplicación del protocolo TCP/IP ha sido subdividida en varias capas para definir el conjunto de protocolos de la Web Semántica. El Consorcio de la World Wide Web establece la siguiente arquitectura para la Web Semántica (Figura 1.16) [1]:

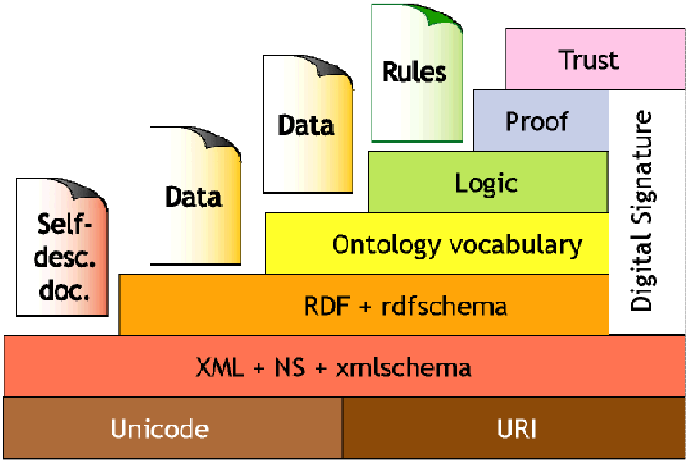


Figura 1. 16: Arquitectura de la Web Semántica en 2000

Como puede apreciarse, la arquitectura de la Web Semántica ha sido estructurada por niveles. Pero esta arquitectura continua evolucionando, de hecho durante el 2005 y el 2006 sufrió algunas modificaciones (Figura 1.17). Aunque los elementos principales se conservan, se observa como se van incorporando nuevas especificaciones, como OWL o SPARQL .

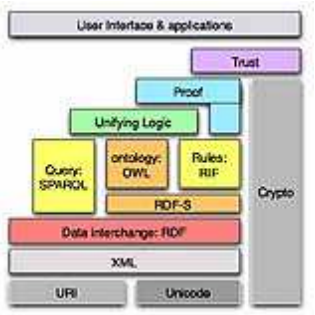


Figura 1. 17: Arquitectura de la Web Semántica en 2006

Ambas representaciones coinciden en los niveles sobre los que se sustenta la Web Semántica, aunque en la arquitectura del 2006 se añade una última capa con la que interactua el usuario.

El nivel URI se definen los recursos distribuidos por la red. Los URIs identifican de forma inequívoca un recurso en la red. Este identificador además cumple con la función de identificador de objetos en el mundo real.

El nivel XML es el nivel sintáctico. Permite establecer la forma de expresar los protocolos superiores. Los entusiastas de XML opinan que algún día todos los navegadores procesarán XML en vez de HTML, a través de una migración progresiva mediante XHTML, SVG (Scalable Vector Graphic), Xlink, etc. Pero nosotros pensamos que este enfoque no es operativo para avanzar hacia la Web Semántica, ya que, en el caso de producirse, estimamos que su implantación sería muy costosa y, desde luego, a muy largo plazo.

El nivel de recurso lo forman los datos RDF, el nivel de ontologías e instancias viene formado por diferentes especificaciones como son RDFS, SPARQL y OWL. Por encima, ya se puede plantear un nivel de lógica descriptiva, que pretende dar flexibilidad a la arquitectura para realizar consultas e inferir conocimiento a partir de las capas anteriores. El nivel de seguridad permite asignar grados de confianza y seguridad a los distintos recursos distribuidos en la web, a través de firmas digitales, redes de confianza u otras técnicas de autentificación por red. Finalmente y sobre todo esto, estarán las aplicaciones semánticas.

1.7. El problema de implantar la Web Semántica

Para estudiar la problemática de la tecnología de la Web actual y compararla con la Web Semántica vamos a utilizar una metodología (muy popular en el campo de la economía) denominada DAFO. Esta metodología consiste en el estudio de la situación competitiva de una empresa dentro de su mercado, así como de las características internas de la misma para determinar sus debilidades, amenazas, fortalezas y oportunidades. Realizaremos, por lo tanto, un análisis DAFO adaptado a la tecnología Web. Esta adaptación la definimos de la siguiente manera:

- Debilidades: Son aspectos que limitan o reducen la capacidad de desarrollo efectivo de la tecnología, constituyen una amenaza para la consolidación de su implantación y deben, por tanto, ser controladas y superadas.
- Fortalezas: Son ventajas competitivas que deben y pueden servir para competir con otras tecnologías alternativas.
- Amenazas: Son las circunstancias de futuro que pueden impedir o reducir el uso de una tecnología.
- Oportunidades: Son las circunstancias de futuro que pueden favorecer el uso de la tecnología

La gran ventaja que tiene la Web hoy en día es lo fácil que resulta diseñar un sitio web, pues ser webmaster no requiere grandes conocimientos. Sin duda esto es su gran *fortaleza*. Además, parece que en el futuro la Web será cada vez más utilizado porque tal vez se ha convertido en el medio más popular para el acceso a la información. El problema es que esta popularidad que hoy ya existe está afectando negativamente a la eficacia de las búsquedas: cada vez los buscadores proporcionan más información de peor calidad, es decir, devuelven más páginas de información que no está buscando el usuario, de manera que éste pierde cada vez más tiempo en encontrar lo que verdaderamente necesita. En definitiva, se ha alcanzado un punto de inflexión donde las ventajas se convierten en desventajas. El siguiente DAFO (Figura_1.18) recoge esta situación.

Figura 1. 18: DAFO Web sin semántica (Web 2.0)

Afortunadamente se han desarrollado una series de estándares que permitirán que las búsquedas de información sean más eficaces. En realidad, cuando estos estándares en los que se basa la Web Semántica (como son RDF y OWL) se incorporen a la web, las búsquedas serán eficaces independientemente del volumen de información. El problema es que RDF y OWL no son sencillos y requieren personas especializadas. Nosotros identificamos dos grupos de problemas en la implantación de la Web Semántica. Un primer problema, en esta visión de la Web, es la definición de ontologías. Toda la Web Semántica se basa en la existencia de ontologías, y estas han de ser miles (seguramente expresados en varios idiomas). Además, deben desarrollarse ontologías nuevas a medida que se necesiten, por lo que constantemente será necesario añadir ontologías nuevas en la Web Semántica. Como es inviable que los usuarios anoten directamente en código OWL DL, se requieren herramientas que faciliten esta tarea, esto sin tener en cuenta la dificultad que supone diseñar una ontología. Aunque, como ya dijimos, actualmente ya se dispone de editores ontológicos, como por ejemplo, el editor PROTÉGÉ de la Universidad de Stanford [43]. Un segundo problema, probablemente más importante, ya que condiciona el éxito de la Web Semántica, es el problema de la anotación semántica, es decir, cómo los webmasters y resto de usuarios que participan en contenidos pueden expresar su información en OWL.

Figura 1. 19: DAFO Web Semántica (Web 3)

La oportunidad principal para implantar la Web Semántica, es en realidad, la debilidad de la Web actual (ver Figura 1.19), pero esto no será posible si no se logra suavizar la complejidad de la tecnología para que los webmasters y usuarios activos puedan adaptarse a ella. Hay que tener muy en cuenta que realizar un proceso de implantación implica una transformación de las páginas actuales a páginas semánticas, es decir, se debe realizar un proceso de migración y de mantenimiento.

Por lo tanto, el verdadero problema que hoy por hoy tiene la Web Semántica es que nadie se preocupa de sus actores fundamentales, es decir, de los webmasters y de los usuarios activos de la Web 2.0. Estos actores son los responsables de mantener las páginas webs y por lo tanto, de los contenidos, de manera que plantear que dichos actores deben asumir transformar sus sitios web en sitios web semánticos sin facilitarle herramientas, es inviable. Sencillamente porque el grado de complejidad que la Web Semántica exige es tal, que se requiere herramientas semi-automáticas o automáticas que faciliten este proceso de transformación. Si no se dispone de estas herramientas, es posible que la Web Semántica no pueda ser implantada. Por si esto fuera poco, no basta con transformar un sitio web una vez, sino que hay que mantener esta información actualizada mediante una representación basada en ontologías constantemente. Debemos esforzarnos en proporcionar herramientas suficientes para que dichos actores puedan adaptarse al nuevo paradigma. Estos actores deben ocuparse, a nuestro juicio, de tres acciones de gran trascendencia para el éxito de la Web Semántica:

- Identificación del dominio de conocimiento al que pertenece la página o sitio web.
- Anotación formal de contenidos.
- Actualización y mantenimiento (coherencia de contenido con anotación semántica)

1.8. Anotación semántica.

La intensidad con que la comunidad científica está abordando el reto de la Web Semántica y la diversidad de proyectos y trabajos, hacen difícil conocer rigurosamente el estado del arte. Pero podemos obtener una visión general si enfocamos el estado del arte en base a la causa de los problemas planteados en el epígrafe "1.7. El problema de implantar la Web Semántica" que es el volumen y el crecimiento de la Web actual combinado con la complejidad de la tecnología de la Web Semántica (epígrafe "1.5. Plataforma tecnológica de la Web Semántica"). En consecuencia la Web Semántica requiere superar dos grandes bloques de dificultades:

1º.-, Aprendizaje de ontologías (ontology learning).

El objetivo es disponer de ontologías que representen el conocimiento de toda la Web, de manera que se investiga en métodos de construcción semi-automáticos o automáticos de ontologías a partir de fuentes de textos en lenguaje natural que no sólo se centra en la obtención de jerarquías de conceptos, sino que tiene en cuenta también un amplio conjunto de relaciones semánticas entre ellos. Existen varias propuestas en el ámbito del aprendizaje de ontologías, pero nosotros no las vamos a desarrollar, ya que esta tesis no se encuadra en dicho ámbito.

2º.- Población de ontologías (ontology population).

En este ámbito de investigación, se parte de que las ontologías ya existen, de manera que se trata de rellenar éstas con instancias. El objetivo entonces, es realizar anotaciones semánticas de todos los sitios webs de acuerdo a esas ontologías. Al igual que en "ontology learning", las investigaciones actuales se orientan hacia métodos de construcción semi-automáticos o automáticos. Pero actualmente, estamos lejos para abordar la transformación de la Web actual en una Web donde partiendo de cada página web, que forma un sitio web, se describa el contenido formalmente anotado respecto a una o varias ontologías, es decir, herramientas de transformación desde la Web hacia Web Semántica.

Por esta razón nos interesa el estado del arte concretamente en población de ontologías.

Estas dos grandes líneas de trabajo son imprescindibles para la implantación de la Web Semántica. Ambos problemas, son complicados de superar, por lo que se han convertido en argumentos claros para los detractores de la Web Semántica. Sin embargo, son también las principales áreas de trabajo para los investigadores. Por supuesto, hay más ámbitos de estudio, como puede ser el problema de buscar la correspondencia entre conceptos, ya que, los mismos conceptos pueden formar parte de distintas ontologías (conocido como ontology mapping). La necesidad de una metodología aceptada por toda la comunidad investigadora de construcción de ontologías (Ontology Engineering Methodology). Incluso la definición de métodos de evaluación "benchmarking" aplicado a la tecnología de Web Semántica [García Castro; 2008]. Pero, como hemos dicho, nos centraremos en "población de ontologías" restringido exclusivamente en anotación de texto. Es decir, no entraremos en anotación de imágenes, audio, multimedia, etc.

En los últimos años han aparecido varias herramientas de anotación semántica o de población de ontologías y aunque todas ellas son primitivas respecto a la necesidad de transformar la Web en Web Semántica. La diversidad de las mismas nos obliga a clasificarlas. Hemos identificado varios criterios para esta tarea. Seguidamente exponemos los que hemos considerado más relevantes:

1. Atendiendo a las estrategias de extracción:

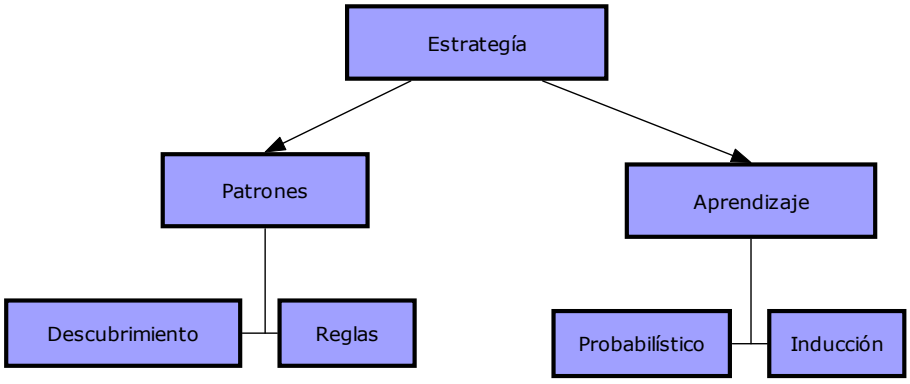


Figura 1. 20: Clasificación Reeve-Han

La figura anterior representa la clasificación de Reeve y Han [*Reeve et al*; 2005]. Según la cual, los métodos de anotación atendiendo a la estrategia de extracción, serian de dos tipos fundamentales; basados en patrones, o bien, basados en aprendizaje. Los primeros identifican

cada entidad a partir de un conjunto inicial de etiquetas, estas pueden ser reglas fijas o de descubrimiento, que consiste en un entrenamiento previo del sistema en base a textos ya anotados adecuadamente, En cuanto a las estrategias basadas en aprendizaje, se puede distinguir de tipo probabilístico o de inducción. Los métodos probabilísticos predicen el uso de una palabra (o una frase) en relación a una ontología, mientras que los métodos de inducción emplean la estructura lingüística de los textos para obtener patrones que reconozcan entidades [Danger Mercaderes; 2007].

2. Atendiendo al grado de automatismo de las herramientas.

   Nos ha parecido interesante el punto de vista de Rolando Beltrán [*Rolando Beltrán*; 2008] que realiza una clasificación basada en el nivel de apoyo a la automatización de las herramientas de anotación y el método con el cual lo hacen (ver Figura 1.21)

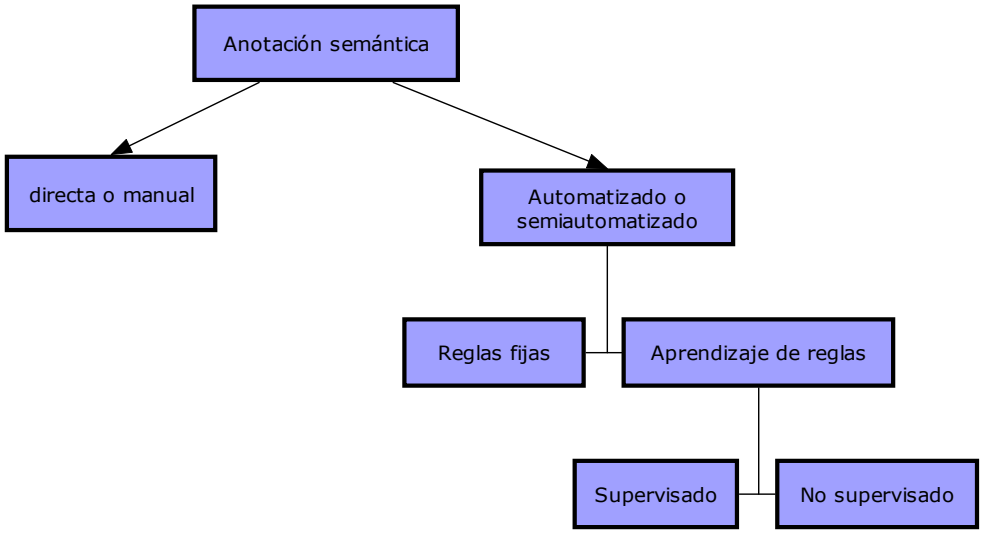


Figura 1. 21: Clasificación de Rolando Beltrán

Como anotaciones directas o manuales se incluyen las herramientas que permiten al usuario hacer anotaciones semánticas sobre un contenido dado, el nivel de automatización es nulo, es el usuario quien se encarga de analizar los documentos, tratando de identificar las entidades semánticas, y las relaciones entre estas. En la otra categoría, anotación automatizada o semi-automatizada, se distingue una pequeña clasificación en base a algunas estrategias de extracción.

3. Atendiendo al lugar donde se realiza la anotación.

   Las herramientas de anotación se englobaron en dos grupos:

   - Herramientas de anotación externa. Estas herramientas permiten asociar meta información a páginas web que ya están presentes en la Web. La meta información no se almacena dentro de la misma página, sino que se almacena de forma externa.
   - Herramientas de anotación de autor. Estas herramientas permiten incluir la información estructurada dentro de las propias páginas.

   Es decir, herramientas donde la anotación se realiza de forma embebida o herramientas donde la anotación no es embebida. Esta fue la clasificación del proyecto RODA (Red de conocimiento Descentralizado a través de Anotaciones) [72]. Sin embargo, parece que las clasificaciones de los últimos años de las herramientas de anotación se han olvidado por completo de este detalle. El proyecto RODA, cofinanciado por el PROFIT 2003, en el cual participaron Robotiker, la Fundación IBIT (Islas Baleares para la Innovación Tecnológica) y el IAT (Instituto Andaluz de Tecnología), fue pionero al realizar una clasificación de herramientas de anotación, pues se realizo entre enero del 2003 y marzo del 2004. Es decir, el Consorcio de la World Wide Web acababa de definir OWL. De forma, que las herramientas de anotaciones evaluadas estaban en las primeras fases y como mucho realizaban anotaciones en RDF

En base a todos estos puntos de vista, hemos intentado realizar una nueva clasificación combinando los tres enfoques (ver Figura 1.22) y procurando adaptarnos mejor a las herramientas analizadas.

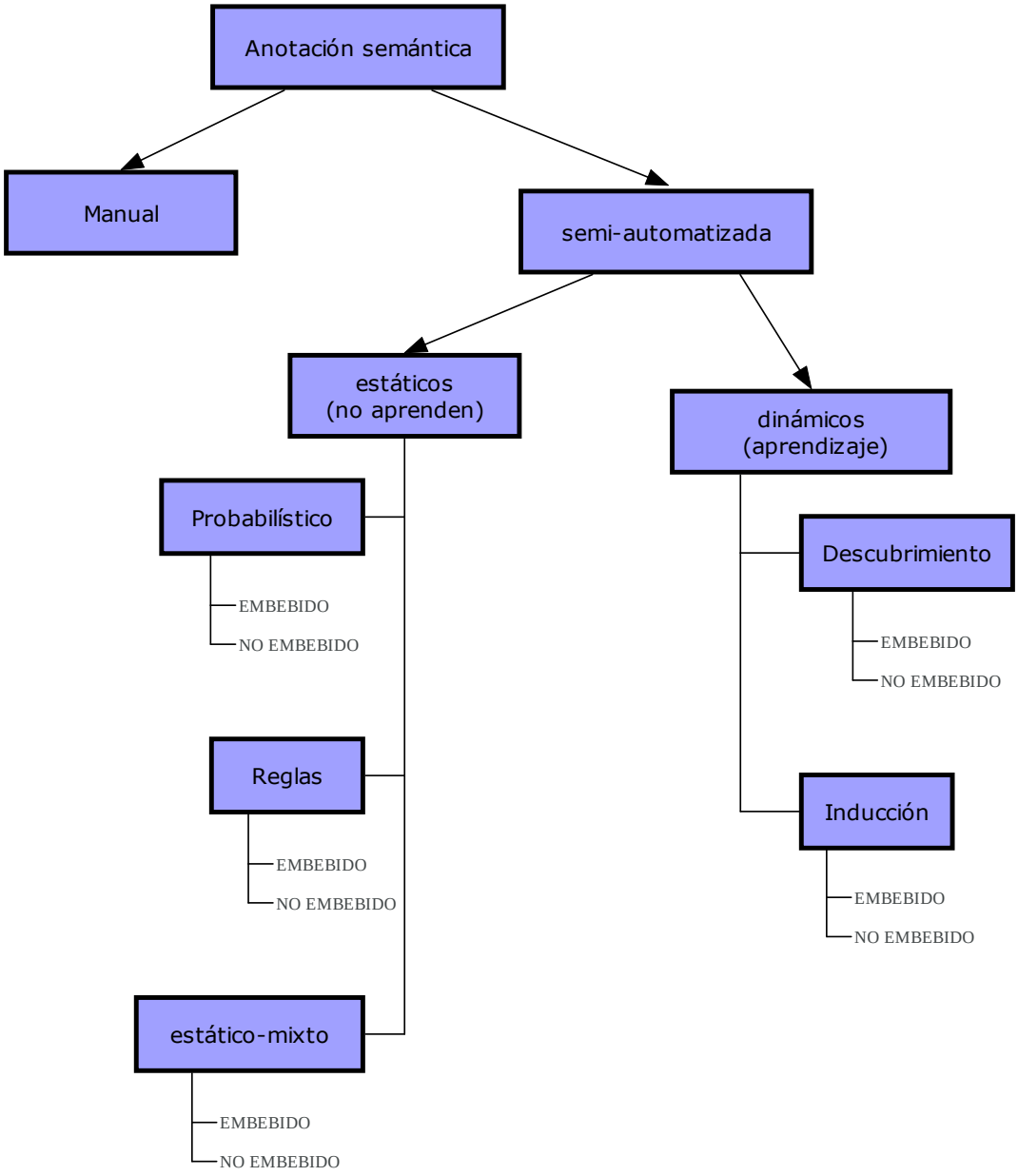


Figura 1. 22: Propuesta de clasificación de herramientas de anotación

En primer lugar hay que distinguir entre sistemas de anotación manual y semi-automática. No consideramos sistemas de anotación automáticos, sencillamente porque estamos muy lejos. Actualmente no existe ninguna herramienta automática, aunque algunos autores se empeñen en declarar que si lo son. Las herramientas y los métodos orientados a procesos semi-automáticos, podemos clasificarlos a su vez. en aquellas que utilizan algún sistema de aprendizaje y las que no. En este sentido, no nos ha parecido acertada la clasificaciones anteriores, ya que, en la clasificación Rolando Beltrán solo se tiene en cuenta métodos o herramientas basadas en reglas, y en la clasificación de Reeve y Han, se discierne entre aprendizaje y patrones, sin considerar bucles en la clasificación, ya que los métodos de

inducción, permiten, a partir de determinadas estructuras lingüísticas, extraer patrones, para posteriormente reconocer entidades. Luego en este caso, el patrón es resultado de un aprendizaje.

En la clasificación propuesta, hemos huido de conceptos como "patrón", ya que en realidad un patrón siempre se utiliza y no es útil para distinguir estrategias. Hemos agrupado las herramientas y métodos semi-automáticos en estáticos y dinámicos, siendo estos últimos los que evolucionan en el tiempo, pues incorporan mecanismos de aprendizaje. Los estáticos pueden ser de varios tipos; basados en probabilidad (usualmente se utiliza la frecuencia de aparición de términos), basados en reglas (que asocian ciertas características del texto con una entidad) o incluso combinando ambas estrategias, En cuanto a los dinámicos, distinguimos dos categorías; por descubrimiento, y por inducción, que como hemos comentado anteriormente los métodos de inducción emplean la estructura lingüística de los textos para obtener patrones que reconozcan entidades. Finalmente, en cualquier categoría de la clasificación se distingue entre que la anotación sea embebida o no. Esto es lo que en el proyecto RODA se denomino anotación externa (para no embebido) y herramientas de autor para referirse a la anotación embebida. En las conclusiones de RODA se considero que la "anotación externa es útil para realizar comentarios personales sobre páginas web, para mantener discusiones en grupo sobre estas páginas, para compartir información, como herramientas de ayuda para el mantenimiento de bookmarks compartidos, etc.". Sin embargo, la importancia de no realizar la anotación embebida es enorme, pues realizarlo de forma externa permite dos caracteristicas fundamentales para la implantación de la Web Semántica y que veremos a lo largo del desarrollo de esta tesis, y son:

1.- Compatibilidad entre la Web actual y la futura Web Semántica.

2.- La anotación externa puede ser una estrategia que facilite mucho más el desarrollo de buscadores semánticos que la anotación embebida.

Seguidamente comentamos algunas herramientas que hemos analizado:

1.8.1 Anotación Manual.

Aunque no conviene extendernos demasiado en la anotación manual, ya que, el volumen de la Web (como se ha explicado con anterioridad) pensamos que obliga a utilizar mecanismos automáticos si queremos implantar la Web Semántica. Sin embargo, aún hoy se mantienen este tipo de herramientas y representan el primer intento de población de ontologías. De manera, que describimos las herramientas más conocidas de este tipo:

Annotea [81]: Es una herramienta que pretende mejorar la anotación mediante la colaboración de los usuarios y fue iniciativa de la W3C. En Annotea [81] una anotación puede ser un comentario, una nota, una explicación o cualquier texto que se pueda adjuntar a un documento web externamente, es decir, sin necesidad de modificar el documento original. En este proyecto se ha desarrollado un navegador propio llamado Amaya para poder ver y crear las anotaciones. De forma, que para su empleo, el usuario final sólo necesita tener instalado el navegador Maya o la extensión "Annotea Ubimarks" para Mozilla y Firefox. Con estos navegadores, el usuario puede realizar anotaciones y ver las realizadas por el resto de usuarios que se reciben de uno o varios servidores adjunto a una página web, de esta manera, el usuario puede ver qué piensan otros usuarios sobre el mismo documento.

La creación y el mantenimiento de anotaciones se realizan en RDF, dichas anotaciones son almacenadas de forma externa, en un servidor de base de datos. Como crítica hay que señalar que cuando hemos probado la herramienta, no es posible realizar anotaciones en páginas que no cumplan estrictamente las especificaciones de la W3C, lo cual parece razonable en principio, pero la realidad es que son una minoría de páginas web las que cumplen dichas especificaciones y de hecho los navegadores actuales son robustos a ello. De forma que interesan herramientas de anotación robustas frente a HTML mal formado, ya que en caso contrario no podrá ser usado. Para comprobar esto es suficiente con abrir un blog en una de las dos plataformas más populares como son blogspot y wordpress, añadir algún gadged como un contador de visitas y comprobar que la herramienta Annotea falla. Otra cuestión que nos parece muy importante es que el objetivo de anotar semánticamente el contenido web es que posteriormente pueda ser explotado por los buscadores semánticos. ¿Cómo realizar esto con anotaciones tan abiertas como utiliza Amaya 11.2 del proyecto Annotea? (ver Figura)

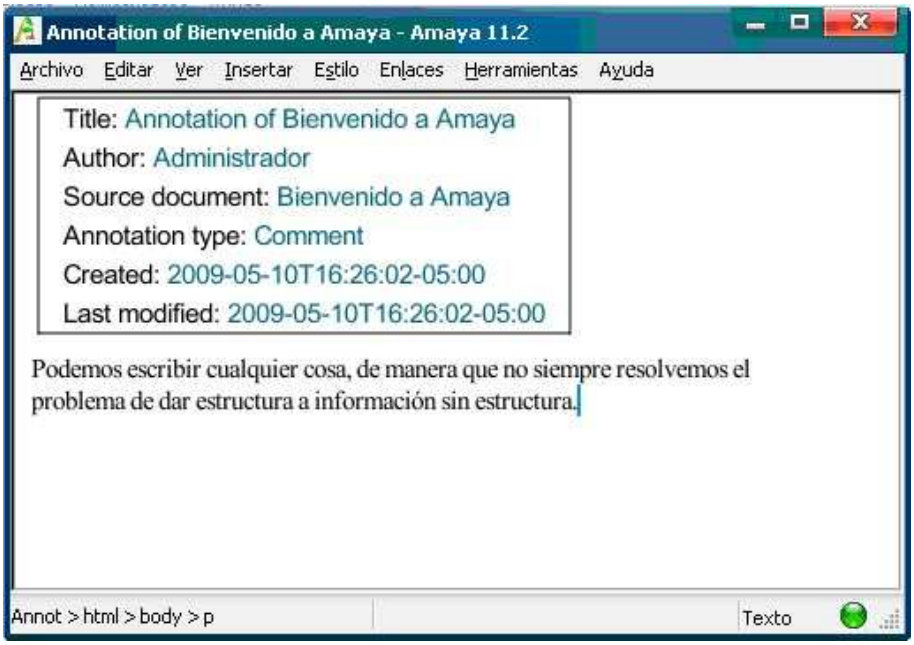


Figura 1. 23: Anotación Amaya 11.2

Yawas (Yet Another Web Annotation System): YAWAS [72] es una herramienta que permite a los usuarios realizar anotaciones para expresar sus opiniones y personalizar los documentos a medida que analizan la información en páginas web. Se trata de un editor de anotaciones creado por expertos que consideran que las anotaciones almacenadas en servidores, limitan la extensión de la tecnología de anotaciones en la Web. La propuesta es codificar las anotaciones en URLs extendidas, que pueden estar disponibles fácilmente y ser insertadas en documentos. Se trata de que el usuario disponga de una herramienta de trabajo similar a un libro de favoritos pero personalizado y adaptado a su necesidad. Para su empleo, el usuario final sólo necesita tener instalado el navegador Web Internet Explorer o el de Netscape, junto con cierto software de Yawas para estos navegadores. Con el navegador, el usuario puede realizar anotaciones y ver sólo las notaciones realizadas por él.

Trellis Web [85]: Entorno interactiva que permite a los usuarios añadir sus observaciones, puntos de vista y conclusiones como la información que analizan realizando anotaciones semánticas a los documentos y a otros recursos on-line. En realidad es una forma de adquirir conocimiento donde el usuario puede añadir nuevo conocimiento al sistema basándose en su propia experiencia a medida que analiza la información. Para su empleo, el usuario final sólo necesita tener instalado un navegador Web capaz de soportar "frames". Con el navegador, el usuario puede realizar anotaciones y ver las notaciones realizadas por el resto de usuarios.

1.8.2 Anotación semi-automatizada.

Melita [78]: Es una herramienta de anotación de texto semi-automática basada en ontologías que tiene integrada una máquina de extracción de información basada en el sistema Amilcare [92]. Los usuarios gestionan todo el proceso de anotación. Sin embargo, algunos de los pasos se pueden automatizar y gestionar fácilmente. Las principales capacidades de Melita se pueden resumir en: la tarea de gestión, la extracción, el aprendizaje, y el etiquetado de la Información de forma autónoma.

AKTive Media [94]: Se trata de un sistema de anotación para texto e imágenes. El objetivo es de automatizar el proceso de anotación, sugiriendo el conocimiento al usuario de un modo interactivo mientras el usuario anota y de ahí reduce al mínimo el esfuerzo de usuario. El sistema trabaja activamente en el fondo, actuando recíprocamente con servicios de web y consultando a una BBDD de anotación central para buscar el contexto el conocimiento específico. La herramienta puede trabajar con varios lenguajes de ontologías como RDFS, OWL, DAML. También soporta anotación de imágenes para diferentes formatos (JPG, GIF, BMP y PNG)

MnM: Según el sitio web oficial, MnM [73] es una herramienta de anotación semántica tanto automática como semi-automático orientado a páginas web. La herramienta (implementada en java) integra un navegador web con un editor de ontologías y proporciona un API para la integración de servidores y herramientas de extracción de información. Para la extracción de información se emplea un método de inducción. En base a nuestras pruebas, antes de poder usarse es necesario que el usuario realice una larga lista de pasos. La anotación es en RDF o DAML+OIL

Pankow [*CIMIANO et al*; 2004]: Es un método para encontrar todos los nombres propios en una página web y entonces emplea un conjunto de patrones sintácticos (calculados manualmente) que conectan conceptos de una ontología a identificadores para generar un conjunto de patrones para cada nombre propio. Por último, encuestan a Google para darle ranking a cada patrón candidato y seleccionar el mejor (más frecuente) patrón para cada instancia (nombre propio). De esta forma, cada nombre propio de la página web es anotado con el concepto que define el patrón seleccionado [Danger Mercaderes; 2007].

COHSE (The Conceptual Open Hypermedia Project) : Proyecto de investigación sobre métodos que mejoren significativamente la calidad, consistencia y la amplitud de documentos web enlazados mientras se recuperan (cuando los lectores navegan sobre los documentos) y al mismo tiempo que se crean (cuando los autores crean los documentos). COHSE [80] utiliza tres tecnologías: un servicio de razonamiento de ontologías que se utiliza para representar un sofisticado modelo conceptual de términos documentales y sus relaciones, un servicio abierto de enlaces hipermedia basado en Web, que sea escalable, y la integración de estos dos para poder enlazar documentos vía metadatos describiendo sus contenidos. Para su empleo, el usuario final sólo necesita tener instalado el navegador Web Mozilla, junto con cierto software de COHSE para este navegador. Con el navegador, el usuario puede realizar anotaciones y ver las notaciones realizadas por el resto de usuarios.

SemTag [*DILL et al*; 2003]: Describe un método para la anotación semántica de instancias de una ontología. Para ello emplean la taxonomía de la ontología, es decir, la clasificación de conceptos. Dichos conceptos son nombrados con un identificador al que se asocia sinónimos o palabras cercanas.

SHOE Knowledge Annotator [76]: Herramienta que permite realizar anotaciones en documentos web sin tener que preocuparse de los códigos HTML pues añade las etiquetas necesarias automáticamente. Las etiquetas están divididas en dos categorías: para la construcción de ontologías y para anotar documentos web para suscribirlos a una o más ontologías, declarar entidades de datos y realizar afirmaciones sobre las entidades según las normas establecidas por las ontologías.

Armadillo [93]: Basado en las reglas y entidades extraídas por Amilcare. Esta herramienta induce los patrones de páginas web de sitios con una estructura altamente regular. Los patrones encontrados (atributos de una ontología particular) son entonces verificados y analizando las estadísticas resultantes de consultas a servicios web. Los patrones mas representativos pueden ser finalmente empleados para extraer información de páginas web.

AeroDAML [*Kogut et al*; 2001]: Es una herramienta para crear anotaciones semánticas de forma automática, que se basa en técnicas de análisis provenientes del campo del procesamiento de lenguaje natural, para asignar una clase o propiedad a una palabra en el contenido tratado. El funcionamiento de AeroDAML consiste en que el usuario entra la URL del documento (web) y la aplicación genera la anotación semántica para ese documento. Esta solución esta soporta DAML y RDF.

SMORE [77]: Herramienta que permite a los usuarios etiquetar sus documentos en RDF utilizando ontologías existentes en Internet asociadas a los elementos y términos específicos.

1.8.3 Conclusión sobre herramientas de población de ontologías.

Las herramientas que hemos evaluado, aunque probablemente existan muchas más de características similares, no resuelven el problema de poblar cualquier ontología y tampoco ofrecen solidez a la industria.

Herramienta	Manual	Semi-automáticas									Embebido			Lenguaje					
		Estáticos (no aprenden)					Dinámicos (aprenden)												
	anotación manual	Probabilistico	Reglas	Estático-mixto	otros		Descubrimiento	Inducción	otros		Si	No	No se sabe	XML	OWL	RDF	OWL, RDF y otros	otros o no se sabe	Obsoleta
Annotea / Annozilla	X											X							
Yawas	X												X						
Trellis Web	X											X							
Melita												X							X
AKTive Media					X								X				X		
MnM (v2.1)								X			X								
Pankow		X											X					X	
COHSE												X							X
SemTag					X								X					X	
SHOE Knowledge Annotator											X							X	
Armadillo			X										X					X	
AeroDAML					X								X			X			
SMORE											X					X			

Figura 1. 24: Cuadro resumen de herramientas de anotación.

En el cuadro resumen (ver Figura 1.24) se puede observar, que son excepcionales las herramientas que dan soporte a la anotación en OWL, probablemente por la complejidad del mismo y que las herramientas disponibles, aún siendo generosos y considerándolas semi-automáticas, no están orientadas a transformar un sitio web por completo, es decir, no son capaces de realizar la anotación de todas las páginas que componen el sitio web. En cuanto a la calidad de los resultados de anotación que ofrecen estas herramientas, todas afirman tener unos índices altísimos, superiores al 90%. Sin embargo, la mayor parte de los estudios se basan en

evaluar la calidad del resultado obtenido en base a la formulación que establece Baeza y Ribeiro en su libro "Modern Information Retrieval" [Beza *et al*; 1999]:

$$precisión = \frac{Total\_entidades\_recuperados\_correctos}{Total\_entidades\_recuperados}$$

$$recuperación = \frac{Total\_entidades\_recuperados\_correctos}{Total\_entiadades\_a\_recuperar}$$

$$F1 = \frac{2 * precisión * recuperación}{precisión + recuperación}$$

Hay que decir, que estas expresiones pueden ser útiles con un volumen de información reducido. Pero cuando nos enfrentamos a cualquier tipo de texto y a un volumen considerable de información, como el que puede contener un sitio web completo, parece difícil que estos parámetros se puedan calcular con rigor. Por ejemplo, cómo podemos saber todas las posibles entidades que se podrían recuperar, ya que dichas entidades estarán en función de la ontología u ontologías que se utilicen para su extracción.

1.9. Consecuencias y objetivos

Como hemos visto, las herramientas de anotación semántica actuales son insuficientes para plantear un proceso de población de ontologías completo. Esta situación de incapacidad de expresar la información de las páginas web en un lenguaje de etiquetado como puede ser OWL frena la posible implantación del nuevo paradigma Web. De forma que nuestra investigación se centra en ésta problemática concreta.

Por otro lado hemos observado, en las herramientas examinadas en el epígrafe anterior, que no se ocupan de la relación entre anotación y explotación de la misma por parte de un buscador semántico. No se debe olvidar el objetivo fundamental de la Web Semántica, que és, una solución al acceso de información en el extraordinario volumen de información de la Web que ya tiene y que previsiblemente continuará creciendo cada vez más rápido. De forma, que en nuestra tesis procuraremos cerrar el ciclo, es decir, nos ocuparemos de la población de ontologías, pero también de la explotación de las anotaciones semánticas que se automaticen, por parte de los buscadores semánticos.

Capítulo 2: Propuesta: Población de ontologías en base a vistas semánticas

En este capítulo se describen los diferentes pasos que planteabamos en el resumen, desde un punto de vista teórico. Además de nuevos conceptos sobre los cuales descansan estas etapas.

Como hemos visto en el capítulo anterior, inicialmente la Web Semántica sirve para solucionar el problema de búsqueda en la enorme cantidad de información que tiene la Web, en consecuencia, una aplicación necesaria de la Web Semántica son los buscadores semánticos. A finales del 2007 y principios del 2008, han aparecido algunos buscadores semánticos de propósito general, como son Hakia [51], Powerset [49] y True Knowledge [50], pero aún no obtienen mejores resultados que el famoso buscador Google. De hecho, ninguno de ellos, que serán analizados en el capítulo dedicado a los buscadores semánticos, utilizan anotaciones OWL, aunque por lo menos, utilizan interfaces de procesado de lenguaje natural (PLN). Por otro lado, la razón por la que esta tecnología no arranca no depende solo de los llamados buscadores semánticos. El problema es más complejo. La Web Semántica no puede construirse por un "buscador" debe desarrollarse desde la cooperación de la anotación semántica.

Lo que proponemos en esta tesis es un procedimiento semi-automático o automático de anotación en OWL basado en la cooperación de los usuarios. Las aplicaciones semánticas, como son los buscadores semánticos, podrán diseñarse e implementarse realmente cuando exista información semántica con la que se puedan aportar funcionalidades distintivas respecto a la Web actual. Este es el objetivo principal de esta tesis: proporcionar un procedimiento para generar automáticamente información semántica (páginas semánticas) a partir del contenido que tienen las páginas web.

2.1. Etapas del proceso de transformación

El Consorcio de la World Wide Web ha publicado (11 septiembre 2007) su recomendación de GRDDL (Gleaning Resource Descriptions from Dialects of Languages) [55], que son especificaciones que permiten transformar datos expresados en formato XML (como XHTML) en datos RDF. Pero el objetivo de esta tesis (ver Figura 2.1) no es ir desde páginas web bien formadas en XHTML, que poca gente utiliza, hacia estructuras de datos representadas en RDF, ya que como hemos explicado en el capítulo anterior no aportan verdadera semántica. Lo que pretendemos hacer es transformar páginas HTML mal formadas, las que existen de forma masiva, en páginas semánticas, concepto que definiremos en este capítulo.

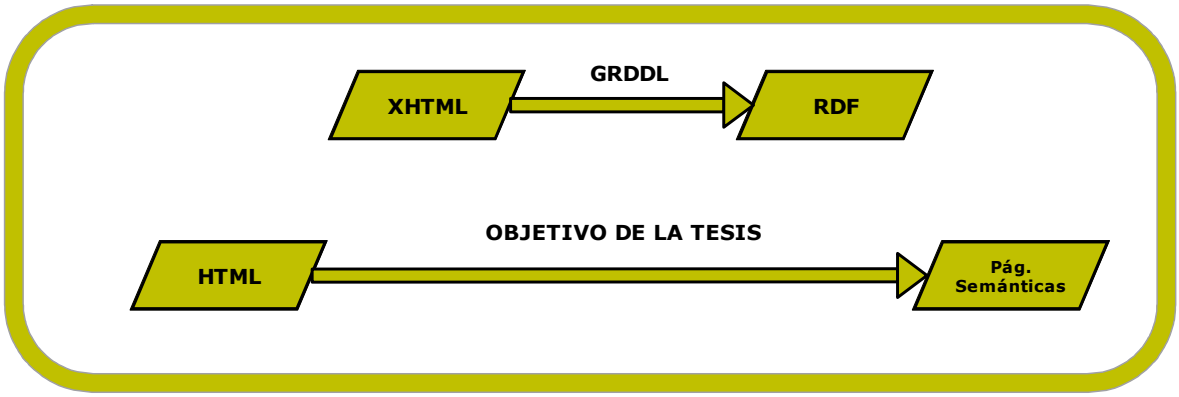


Figura 2. 1: Objetivo GRDDL

Esta transformación debe ser compatible con la Web actual, es decir, por un lado se debe lograr convertir una página web en una página semántica (estructura semántica que entienden personas y sistemas informáticos); y por otro lado, se debe realizar esta transformación de tal forma que pueda ser utilizada tanto por los buscadores actuales como por los buscadores del futuro (semánticos). Es más, las páginas semánticas, que explicaremos en este capítulo, deben diseñarse para permitir compatibilidad completa, es decir, que puedan ser usadas por cualquier aplicación de la Web actual y por cualquier aplicación de la Web Semántica del futuro.

Para poder hacer esto, es necesario dividir el problema en etapas. En esta búsqueda de simplificación pero sin perder la categoría conceptual, hemos distinguido tres etapas fundamentales que permiten la transformación de un sitio web ordinario a un sitio web semántico. En este epígrafe se presentan las etapas de forma muy general. Pero se amplían en el

capítulo siguiente, en el cual profundizaremos en la herramienta “sw2sws”.

El proceso de transformación (ver Figura 2.2) se logra mediante una aproximación de etapas sucesivas, cada etapa del esquema está asociada con un ejemplo en la parte inferior que representa el objetivo de cada una de ellas, así por ejemplo, la primera etapa consiste en averiguar el contexto seleccionando una o varias ontologías, esta etapa la denominamos a lo largo de esta tesis “identificación”. En la etapa siguiente, no solo se extrae contenido, sino que se realiza un análisis morfosintáctico, representando este contenido en categorías sintácticas como son sujeto, verbo, complemento directo e indirecto. Una vez que se extrae lenguaje natural, se puede anotar semánticamente en lenguaje OWL DL, esta última etapa de anotación la denominamos “interpretación”. En los apartados siguientes explicaremos, en detalle y a nivel teórico, cada una de estas etapas que hemos presentado.

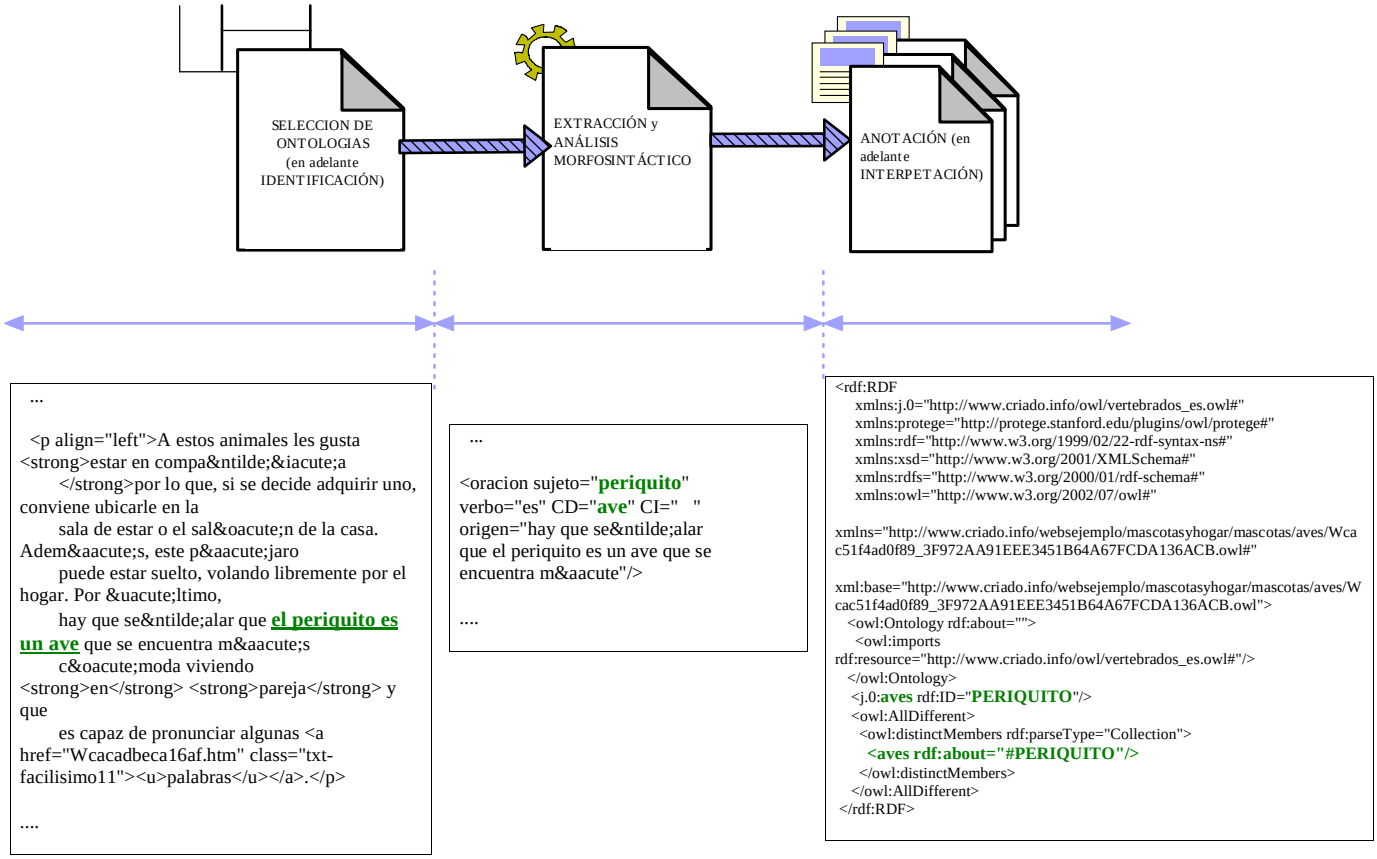


Figura 2. 2: Etapas del proceso de transformación

2.1.1. Identificación de ontologías

En primer lugar, y antes de poder realizar un proceso de transformación, es necesario conocer la temática de la página web que se desea transformar, es decir, determinar de qué trata la página web. Una forma de conocer la temática de una página web es mediante el uso de ontologías. Siendo una temática una o varias ontologías, por ejemplo, podemos decidir que una página web trata de gastronomía, pero tal vez no se dispone de una ontología de gastronomía sino de un conjunto de ontologías de gastronomía especializadas (una ontología de vinos, otra de cocina italiana, cocina madrileña, etc…). De manera, que para los procesos automáticos, el proceso de identificación ontológico consistirá en determinar qué ontologías pueden estar relacionadas con el contenido de una página web. Por supuesto, todo esto que se ha explicado para una página web hay que generalizarlo para el conjunto de todas las páginas web que constituyen el sitio web.

El proceso de Identificación puede ser relativamente rápido cuando trabajamos con un número pequeño de ontologías, ya que, en realidad no requiere un procesado de lenguaje natural (PLN), puede actuar por frecuencia cruzando conceptos con las ontologías como se verá detalladamente en el capítulo 3. Sin embargo, una herramienta comercial deberá tratar de identificar el contenido de un sitio web con el mayor número de ontologías posibles y esta es la principal dificultad, ya que, contrastar el contenido del sitio web con un mayor número de ontologías supone elevar el coste computacional y por consiguiente al tiempo de procesamiento. El resultado de esta etapa es proporcionar información por cada página web indicando, en el caso en que sea posible, si el contenido se asocia a una o varias ontologías.

Para realizar esta tarea de asociar el contenido de una página web con una o varias ontologías, se lee cada fichero HTML y cada término de texto embebido en la página se compara con los identificadores de cada una de las ontologías, estos identificadores deben estar en el mismo idioma, por lo que previamente hay que tener en cuenta un proceso de extracción de identificadores y traducción al idioma del contenido como se explicará en el siguiente capítulo. En la Figura 2.3, se ha subrayado, la frase que tiene sentido para una de las ontologías de ejemplo que hemos utilizado, ya que el término “ave” es una clase de “vertebrado”. Sin embargo, en esta etapa únicamente se selecciona la ontología “http://www.criado.info/owl/vertebrados_es.owl”, dejando para la etapa posterior, las tareas de extracción y análisis.

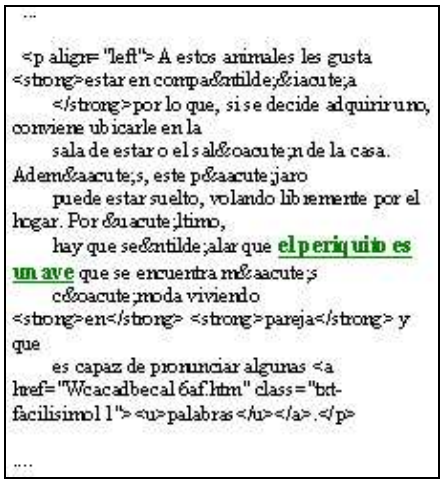


Figura 2. 3: Ejemplo de captura de texto

2.1.2. Extracción de lenguaje natural y análisis morfosintáctico.

El propósito de esta etapa es la extracción de lenguaje natural escrito. Para ello es necesario realizar un análisis morfosintáctico. Este es un campo donde se ha trabajado durante años y que no es el objetivo de esta investigación. Sin embargo, es preciso el utilizar una etapa de extracción y análisis para poder transformar un sitio web. En el capítulo 3, daremos detalles de algunas herramientas que hemos utilizado en nuestras pruebas.

Conviene recordar que la idea de Web Semántica surge desde el convencimiento de que se trata de un problema de estructuración de la información [Berners-Lee; 1998]. Hoy en dia, algunos investigadores piensan, que el problema no se puede resolver sólo con especificaciones y formatos; es necesario encontrar la forma de automatizar la anotación semántica y su actualización, es la razón de que, después de casi diez años, la Web Semántica no esté implantada. La solución se basa en el uso del lenguaje natural, esto es aplicar técnicas del campo de la Inteligencia Artificial y/o del campo de la Lingüística computacional. Berners-Lee, en 1998, decía que "…documentos comprensibles por máquinas no implica Inteligencia Artificial…". Sin embargo, en base a tendencias actuales y a los experimentos y resultados de esta tesis pensamos justamente lo contrario, que el éxito en la implantación de la Web Semántica depende de técnicas de Inteligencia Artificial, en particular, de las técnicas de procesado de lenguaje natural conducidas por ontologías, que proporcionan la posibilidad de construir herramientas semi-automáticas e incluso automáticas. Estas herramientas de anotación semántica permitirán transformar un sitio web en un sitio web semántico.

Pero, ¿por qué es necesario el procesado de lenguaje natural para implantar la Web Semántica?. La razón es la enorme cantidad de páginas web que existen. Por ejemplo, en mayo del 2009, solo en páginas web estáticas se estima en $26,9*10^9$, lo que equivale, en el caso más pesimista, a más de 94 millones de libros. Téngase en cuenta que una de las bibliotecas consideradas más grandes del mundo, la biblioteca de Washington, conocida como biblioteca del congreso (fundada en 1800) cuenta con 29 millones de libros En consecuencia, es inviable realizar una transformación manual.

El trabajo de transformar, mantener o adecuar más de 236 millones de sitios web para que incorpore algo de semántica, es una obra de proporciones titánicas que parece inabordable. La deducción es que este trabajo de transformación solo se podrá abordar si se dispone de herramientas que resuelvan el problema de la anotación de forma automática, mediante procesado de lenguaje natural conducido por ontologías.

La pregunta siguiente es: ¿quién realizará este trabajo?. Solo hay dos alternativas:

1.- Las propias aplicaciones semánticas.

2.- Idear una forma de cooperación, en la que los usuarios activos dispongan de un mecanismo semi-automático, o incluso automático para realizar anotaciones semánticas en sus contenidos.

Superada la fase anterior, la fase de identificación ontológica, el ámbito de trabajo será igual o más reducido, pero nunca mayor. Debido a que muchas páginas web habrán sido perfectamente identificadas respecto a una o varias ontologías, y es posible que otras no se lograsen relacionar con ninguna de ellas. Sin embargo, el proceso es mucho más complejo que la fase anterior, ya que esta tarea esta vinculada al PLN. Pero el procesado de lenguaje natural se aplica solo a las páginas en que el proceso de identificación ontológico tuvo éxito.

Después de filtrar los tags HTML de cada página web, se analiza cada fichero TXT resultante (Figura 2.4). El fichero IDE es un fichero índice que especifica las páginas identificadas y la ubicación de los ficheros filtrados. El análisis de cada fichero filtrado se realiza procesando cada frase a nivel morfológico y posteriormente, mediante un modelo de oración simple se asocian categorías sintácticas. El detalle de esta etapa se explica en el capítulo siguiente.

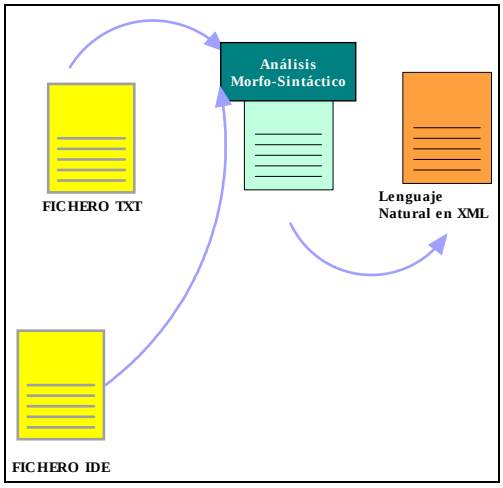


Figura 2. 4: Extracción

2.1.3. Interpretación del Lenguaje Natural conducido por ontologías

La interpretación es un proceso de anotación y es la etapa final en el proceso de transformación. La anotación se puede considerar como una información sobre las entidades o conceptos de una ontología (Figura 2.5), que aparecen en un texto y su situación en el mismo, o también las referencias que hay en un texto sobre un repositorio semántico en el que hay más conocimiento [Atanas *et al*; 2003]

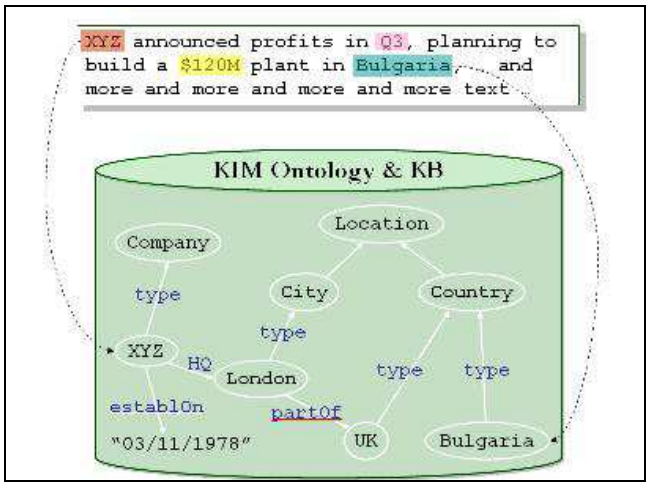


Figura 2. 5: Ejemplo de anotación.

Una vez clasificadas las herramientas de anotación (ver epígrafe 1.8), conviene considerar también quien realizará la tarea de anotación. Entendemos que hay varias alternativas, estas son:

1. La anotación lo realiza personal experto en ontologías

2. Mediante etiquetado colaborativo, tendencia consistente en la anotación manual por parte de los usuarios. El Dr. Mor Naaman, científico investigador de Yahooo Research Berkeley, declaró el 16 de mayo del 2007 en la Conferencia Internacional de World Wide Web de Alberta, en Canadá, que la Web Semántica "Está muerta" [59], ya que considera que no hay esperanza de resolver el problema de la anotación semántica a nivel global. Pero aclaro, que al decir que está muerta se refiere solo a la gran visión de una Web Semántica, que piensa no se logrará, debido a que no se puede esperar que los usuarios realizen anotaciones semánticas complejas. Desde luego, como hemos dicho en esta tesis, solo con herramientas automáticas es posible que los usuarios activos realicen anotaciones semánticas.

3. Anotación automática centralizada: Se trata de idear métodos para que una aplicación semántica sea capaz de dotar de semántica a la información que utiliza. Por ejemplo, si se trata de un buscador semántico, éste sería capaz de rastrear sus referencias para extraer información y sobre ésta realizar la anotación semántica del contenido de los sitios web. Esta información semántica obtenida será utilizada por el propio motor de búsqueda en las respuestas al usuario. Sin embargo, esta opción parece inviable debido al volumen de la Web.

4. Anotación automática colaborativa: La idea se fundamenta en la anotación por parte de los usuarios activos de contenido semántico, pero mediante el uso de herramientas automáticas. Por ejemplo, un webmaster podría tener una herramienta como la presentada en esta tesis que automatizara el proceso de transformación de su sitio web a un sitio web semántico, que después se daría de alta en un buscador semántico, de tal manera que el buscador indexaría directamente esta información. El objetivo es el mismo que la anotación automática centralizada, pero con la diferencia fundamental que el buscador solo incorpora las anotaciones y no las genera. Este enfoque aporta solución a

cualquier tipo de aplicación semántica. Los entornos colaborativos no son una utopía, ya que dichos entornos aplicados a Internet han sido y están siendo un éxito transformando la Web hacia la Web 2.0

De manera, que el procedimiento que proponemos, puede considerarse un procedimiento orientado hacia la idea de "anotación automática colaborativa", siendo la etapa de interpretación, una herramienta de anotación externa basada en ontologías y que consiste en generar una estructura de *vistas semánticas* para obtener una *página semántica*. Estos nuevos conceptos se definen a continuación.

Vistas semánticas

Entendemos por *vista semántica* (*VS*) la estructura que proporciona diferentes visiones de acuerdo a diferentes ontologías, es decir, proporciona diferentes interpretaciones de una misma página web. En la expresión (2.1) una vista semántica (VS) de orden "n" es un conjunto de "n" interpretaciones cada una de acuerdo a una ontología diferente. Donde I es una interpretación de una página web, que es una descripción del contenido mediante instancias de acuerdo a una determinada ontología. Cada interpretación tiene un número indefinido de instancias, es decir, de anotaciones semánticas. Pero todas las instancias de una interpretación se corresponden con una determinada ontología.

$$VS^n = \underset{i=1}{\overset{n}{\{}} I_i \}$$

(2. 1)

En definitiva, una *vista semántica* es la visión que alguien puede tener de una página web conociendo un conjunto de ontologías y que permite cubrir parcialmente o por completo, el contenido de una página web.

Página semántica

Una página semántica puede definirse de varias formas. Estas son las siguientes:

- OWL embebido en HTML: Con esta propuesta no se garantiza la convivencia del actual sistema con el nuevo. Incorporando dentro de código HTML el código OWL se podría declarar la ontología a la que pertenece, así como las instancias y/o declaraciones de hechos en OWL-RDF. Sin embargo, plantea un problema bloqueante en la actualidad, ya que, un fichero HTML no se puede parsear correctamente en la mayoría de los casos, debido a que el código HTML (aunque puede ser definido con un esquema XML) habitualmente no estará bien formado. Según una investigación realizada por Opera Software (3,5 millones de sitios analizados), los sitios web que usan los estándares de W3C constituyen solo el 4,13% [87]. Pero los navegadores son robustos a HTML mal formado, como por ejemplo, que falte el cierre de determinadas etiquetas. Además de esto, HTML tiene una serie de "tags" incompatibles con la concepción de XML, como es el caso de la etiqueta de salto de línea "
". Esta dificultad hace poco viable el embeber semántica en un fichero HTML. Aunque se puede solucionar imponiendo XHTML, que es HTML bien formado, acorde a XML. Actualmente ya hay trabajos del Consorcio de la World Wide Web [Adida et al; 2006] en los que se especifica que XHTML puede embeber RDF, un ejemplo claro es el borrador de documento "RDF/A Primer 1.0" que se ha hecho público el 10 de marzo del 2006. Pero, aplicar esta solución en este momento, implica que la Web Semántica no será realizable a corto y medio plazo.

- Mediante el concepto de *vistas semánticas*: Este enfoque es una evolución sobre el caso anterior, dando la flexibilidad suficiente para que sea viable. Se trata de asociar a cada página HTML una *vista semántica*. Esta es la definición de concepto que aplicamos en esta tesis.

  Por lo tanto, una página semántica (PS) es un par de elementos tal y como se detalla en la siguiente expresión:

$$PS^n = \{PW, VS^n\}$$

(2. 2)

Obsérvese que PS y VS concuerdan en orden "n". La página semántica (ver Figura 2.6) sigue la arquitectura de la W3C y está constituida por dos elementos; uno simple, la página web (PW), y el otro, compuesto, ya que es una *vista semántica* (*VS*) de un determinado orden. Las diferentes interpretaciones (I) de una PW de acuerdo a cada una de las ontologías, y que a su vez son un conjunto de instancias sobre la misma, forman una vista semántica de orden "n". Así, una página semántica de grado 1, es aquella que se compone de una página web y una vista semántica de grado 1. Lo que significa que solo tiene una interpretación de acuerdo a una única ontología. Una página semántica de grado 2 está formada por una página HTML y una vista semántica de grado 2, lo que significa dos interpretaciones asociadas a dos ontologías, es decir, en la misma página HTML se ha encontrado contenido relacionado con dos ontologías diferentes. La idea de esta estructura de página semántica es que el elemento PW es lo que utiliza y entienden las personas a través del navegador y la *VS* es lo que entenderán los sistemas informáticos, como un buscador semántico.

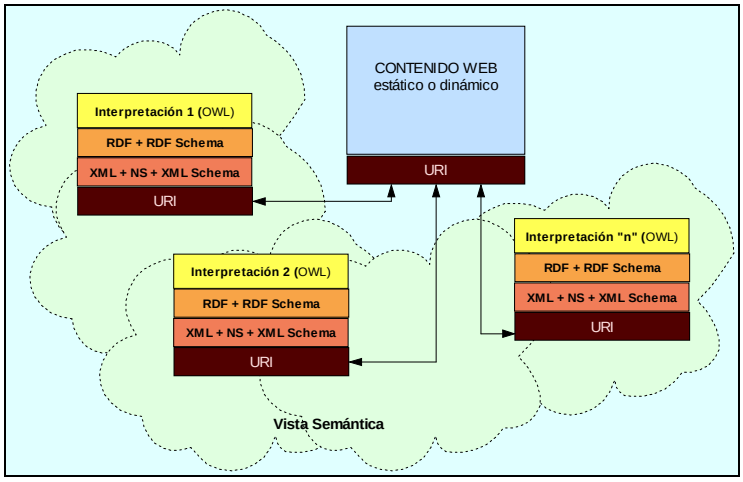


Figura 2. 6: Página semántica basada en vista semántica

2.2. ¿Cómo vincular una vista semántica con su página web?

Para poder generar una *página semántica* previamente se tiene que tener una *vista semántica* (del orden que corresponda). El problema es cómo la *vista semántica* se vincula con la página web de la cual se originó.

La *vista semántica* son agrupaciones OWL que contienen instancias de ontologías y representan parte del contenido extraído de la página HTML. Se puede vincular una página HTML con varias ontologías simplemente añadiendo un "MetaTag" por ontología, se trata, pues, de una relación 1:n, tal y como se aprecia en el siguiente ejemplo:

```
<meta uri-owl-asociado="www.sitio.web.com/mi_pagina1">
<meta uri-owl-asociado="www.sitio.web.com/mi_pagina2">
<meta uri-owl-asociado="www.sitio.web.com/mi_pagina3">
```

Tabla 2. 1: Varias ontologías referenciadas desde una misma página web

Sin embargo, esta solución no es aplicable al contenido dinámico, ya que, hasta aquí el concepto de *vista semántica* sirve para representar información de naturaleza estática. Generalizando el concepto para contenido de texto de todo tipo, sea ésta información estática o dinámica, se debe entender por *vistas semánticas* lo siguiente: Una vista semántica es una estructura de información formado por una o varias interpretaciones que proporcionan diferentes visiones de acuerdo a diferentes ontologías de un mismo contenido. Cada interpretación especifica la ontología concernida y declara las instancias extraídas del contenedor web, siendo éste, de naturaleza estática o dinámica. Estas interpretaciones pueden ser ficheros OWL, en el caso de contenido estático, o tripletas almacenadas en una base de datos, si es información de cualquier naturaleza.

Por lo tanto, hay que tener en cuenta que el contenido dinámico no dispone de estos metadatos, por lo que parece más oportuno relacionar cada página estática y/o dinámica mediante un URI que es un localizador universal de recursos. De forma que cada contenido (estático o dinámico) puede relacionarse con su *vista semántica* constituyendo una página semántica compatible con un buscador actual, pero con las funcionalidades necesarias para proporcionar al buscador semántico capacidades inferenciales.

La forma de referenciar ontologías e instancias asociadas al contenido de una página web (estática o dinámica) expresado en OWL DL, permite no solo recoger el contenido de una página web mediante una sola ontología, sino que nos otorga la posibilidad de relacionar una página web o un contenido web con varias ontologías. Por lo tanto, surge aquí la pregunta de cómo implementar esto con URIs. La solución es que cada URL de un contenido web es también un URI, de manera que basta especificar este valor en el atributo "xml:base" dentro del tag "rdf:RDF". El ejemplo OWL (Tabla 2.2) muestra una instancia obtenida de una página web que trata de perros. Obsérvese como el atributo del tag mencionado apunta a la URL (se ha definido como URI la URL exacta donde reside la fuente del contenido) donde está la página, obsérvese cómo también se indica la ontología sobre la que se instancia mediante "owl:imports rdf:resource".

```
<rdf:RDF
     xmlns:j.0="http://www.criado.info/owl/vertebrados_es.owl#"
     xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns="http://213.139.18.202/owl/Perrilandia/Perrilandia/
                         aguai/index_3F972AA91EEE3451B64A67FCDA136ACB.owl#"
    xml:base="http://213.139.18.202/owl/Perrilandia/Perrilandia/
                         aguai/index_3F972AA91EEE3451B64A67FCDA136ACB.owl">
   <owl:Ontology rdf:about="">
     <owl:imports rdf:resource="http://www.criado.info/owl/vertebrados_es.owl#"/>
   </owl:Ontology>
    <j.0:perros rdf:ID="Spaniel Agua Irlandes"/>
    <owl:AllDifferent>
      <owl:distinctMembers rdf:parseType="Collection">
        <perros rdf:about="#Spaniel Agua Irlandes"/>
      </owl:distinctMembers>
    </owl:AllDifferent>
 </rdf:RDF>
```

Tabla 2. 2: Ejemplo de vinculación entre Interpretación y página web

Este mecanismo va a permitir que el software "entienda" parte del contenido de las páginas web a través de las *vistas semánticas* y que una vez procesada la información de estas *vistas semánticas*, a través, por ejemplo, de un buscador semántico de propósito general, proponga al usuario no la *vista semántica* que se procesa para "entender" el contenido, sino la página web asociada a ésta.

La siguiente línea de la Tabla 2.2 establece el URI de la instancia:

```
xml:base="http://213.139.18.202/owl/Perrilandia/Perrilandia/aguai/index_3F972AA91EEE3451B64A67FCDA136ACB.owl
```

Es de gran interés comentar que el nombre del fichero al que se referencia "index_3F972AA91EEE3451B64A67FCDA136ACB.owl", está formado por el nombre de la página web original al que se le ha concatenado una cadena hexadecimal de 32 caracteres, que es el resultado de aplicar una función hash al contenido del fichero HTML original. Una función hash sirve para generar un identificador resumen, utilizando una longitud fija de bytes y de forma unívoca, de un archivo de cualquier tamaño. En concreto la función que hemos utilizado es la función MD5. Generando el MD5 en el URI de la instancia se obtienen dos ventajas:

- Se genera un auténtico identificador de recursos URI, que identifica inequívocamente la instancia
- La función hash (MD5) permite comprobar si la página a la que se refiere ha sido modificada, esto es fundamental para mantener la coherencia entre instancias OWL y contenido HTML.

2.3. Migración de un sitio web a un sitio web semántico

Vamos a precisar particularmente el concepto de *sitio web semántico*. Entendemos por "semántico" que tenga significado respecto a los sistemas informáticos. En principio, se podría decir que un sitio que incorpore etiquetado que permita clasificar la información y gestionarla, podría considerarse un *sitio web semántico*. Bajo esta definición se podría plantear considerar sitios de este tipo aquellos basados en microformatos o en vocabularios RDF (ver epígrafe 1.5.2). Sin embargo, ambas incorporan una semántica elemental que ya utilizan los buscadores actuales y que parece estar aún muy lejos del concepto de Web Semántica que propuso Tim Berner-Lee en 1998. De manera, que para que la Web Semántica arranque como tal, hacen falta

mayores prestaciones.

Por esta razón hay que completar la definición anterior con el concepto de lógica. Podríamos decir que un *sitio web semántico* es aquel que además de incorporar etiquetado que permita clasificar la información, ésta se representa en un lenguaje sobre el cual se pueda aplicar mecanismos de lógica descriptiva, como es el caso de OWL. Incorporando OWL DL en el etiquetado, se abre la posibilidad del uso de razonadores DL, que diferencian el TBOX del ABOX. El *TBox* incluye toda la terminología, es decir, el vocabulario de un dominio de aplicación en función de:

- Conceptos: denotan clases o conjunto de individuos.
- Roles: denotan relaciones binarias entre los individuos.
- Un conjunto de descripciones complejas sobre este vocabulario (restringidos, por su puesto, por el lenguaje de descripción)

Mientras que el *ABox* contiene afirmaciones acerca de individuos nombrados en términos de vocabulario. Una base de conocimiento basado en *TBox* y *ABox* posibilita inferencias que permiten derivar "conocimiento" implícito a partir del "explícito" de la base de conocimiento.

Si consideramos aceptada la definición anterior sobre el concepto de *sitio web semántico*, debemos ocuparnos ahora del problema de cómo lograr transformar un sitio web actual (HTML) en un *sitio web semántico* (HTML conviviendo con OWL-DL).

El objetivo es que a partir de la Web actual, cada usuario activo coopere en la construcción de la Web Semántica transformando su sitio web (SW) definido en la expresión (2.3) en un sitio web semántico (SWS) (ver expresión 2.4) mediante una herramienta automática. Tengase en cuenta que un SW es un conjunto de páginas web (PW) y que el parámetro "m" será menor o igual a "p".

$$SW = \bigcup_{K=1}^{P} \{ PW \}$$

(2. 3)

$$SWS = \bigcup_{j=1}^{m} \{ PS_j^{n(j)} \}$$

(2. 4)

Como hemos dicho, una página semántica (PS), que puede tener ordenes distintos para cada página web, es una estructura que debe representar la información de tal forma que tenga sentido tanto para personas como para sistemas informáticos. Obsérvese en (2.4) que el número de páginas semánticas será “m” y que cada una de ellas tendrá un orden “n” que pueden no coincidir, es por ello que lo indicamos como “n(j)”. La página semántica es compatible con los buscadores ordinarios (actual), ya que en (2.2) se ha definido como el conjunto formado por dos elementos; la página web y la vista semántica de un orden determinado. De manera, que los buscadores tradicionales usarán paginas web, mientras que los buscadores semánticos usarán también las (2.1) *vistas semánticas*.

En definitiva, para transformar un sitio web en un sitio web semántico (SWS) hay que realizar tres etapas:

1. Asignar a un sitio web las ontologías concernidas: proceso de Identificación Ontológica.
2. Extraer el contenido de cada página del sitio web.
3. Obtener las instancias de acuerdo a las ontologías de la etapa anterior, es decir, generar las *vistas semánticas* de todo el sitio web. De esta forma se conseguirá migrar un sitio web a un sitio web semántico, ya que las páginas webs se habrán transformado en páginas semánticas. Un sitio web semántico se puede entender como un conjunto de páginas semánticas como indica (2.4).

2.4. ¿Quién mantiene las Vistas Semánticas?

Como es conocido, en la Web la responsabilidad del mantenimiento se reparte entre varios actores: los webmaster construyen sus páginas webs y se preocupan por actualizar su contenido; por otro lado, los buscadores y agentes indexan el contenido de estas páginas para que el usuario final de Internet pueda acceder a ellas. Con el nuevo paradigma el papel del webmaster tiene mayor trascendencia, ya que deberá relacionar sus páginas web normales con una o varias ontologías y expresar el contenido de su web en uno o varios ficheros adicionales en OWL. Por lo tanto, la Web Semántica la siguen manteniendo los mismos actores que la Web actual, pero el funcionamiento del nuevo paradigma introduce una dependencia enorme con el webmaster, hasta el punto que, si éste no realiza bien su trabajo, la Web Semántica no funcionará. Es por esto que hay que esforzarse en simplificar la labor del webmaster con herramientas automáticas para asegurar el éxito de la Web Semántica.

Para garantizar el éxito de la Web Semántica se debe cuidar también al resto de usuarios activos, es decir, aquellos que intervienen en el contenido, como son los usuarios de blogs, los usuarios que actualizan contenidos en wikis, en foros, etc….

Como hemos dicho, el contenido puede ser de naturaleza estática, esto es, que la información reside en el fichero HTML, o dinámica, en cuyo caso la información se guarda en una base de datos (BBDD) y sólo se genera una página HTML cuando se solicita una información determinada. Aunque esta tesis aborda el tratamiento para las páginas estáticas, pensamos que todas las ideas y conceptos presentados y desarrollados son aplicables al caso del contenido dinámico siempre que dicho contenido pueda ser localizado mediante un URI, tal y como se ha indicado en el apartado anterior (ver apartado de líneas futuras).

2.5. Algunas conclusiones.

Repasando todo lo expuesto, puede observarse que aunque en la evolución de nuestro razonamiento se ha pensado en la posibilidad de embeber semántica en contenido web, finalmente se ha resuelto mediante URIs en las propias interpretaciones que constituyen cada *vista semántica*. En todo el proceso de transformación, en ninguna etapa, se modifica página web alguna. El proceso de migración hacia un sitio web semántico solo añaden nuevas estructuras que se vinculan automáticamente con ontologías y páginas web ya existentes. Por lo tanto, todo lo que se ha añadido no afecta al funcionamiento actual de la Web, es como si no existiera para los buscadores actuales, de manera que existe compatibilidad plena. Sin embargo, se posibilita el funcionamiento de aplicaciones semánticas que habrá que implementar e implantar en un futuro cercano. En definitiva, este enfoque permite una etapa de migración sin ocasionar ninguna perdida de servicio para los actuales buscadores, ya que las páginas webs actuales no experimentarán ningún cambio y seguirán existiendo exactamente igual a como son (la Web Semántica no obliga a cambiar las páginas web). La estrategia para garantizar la compatibilidad con los buscadores actuales consiste en crear *vistas semánticas* asociadas a las páginas web, siendo cada interpretación de la vista quien establece la referencia a la ontología concernida y a la página web de la que extrae su contenido.

El objetivo por lo tanto, es que al implantar aplicaciones semánticas, como por ejemplo, un buscador semántico de propósito general, los sitios web semánticos no perturben el funcionamiento actual de Internet, aunque proporcionen la estructura de información necesaria para que los buscadores semánticos puedan explotar su principal funcionalidad como es la capacidad de procesar contenido, razonar con éste, combinarlo y realizar nuevas deducciones lógicas para, finalmente, proporcionar la mejor información posible al usuario.

Capítulo 3: Implementación de la herramienta de transformación "sw2sws"

En este capítulo se especifica la propuesta mediante un prototipo que hemos implementado denominado "sw2sws" y que permite transformar un sitio web en un sitio web semántico desde la cooperación de los usuarios activos, tal y como indicabamos con la expresión (2.4).

Explicaremos en detalle como realizar esto. Pero antes vamos a enumerar las condiciones de contorno en el que hemos probado nuestra herramienta:

1. contenido: estático
2. idioma: castellano
3. ontologías: las tratadas en esta tesis como ejemplos
4. identificación: método exacto
5. extracción: PLN de oración simple

En el siguiente esquema (Figura 3.1) se puede ver una representación general del proceso. El esquema se ha dividido en tres partes limitadas por puntos discontinuos que iremos comentando a lo largo de este capítulo.

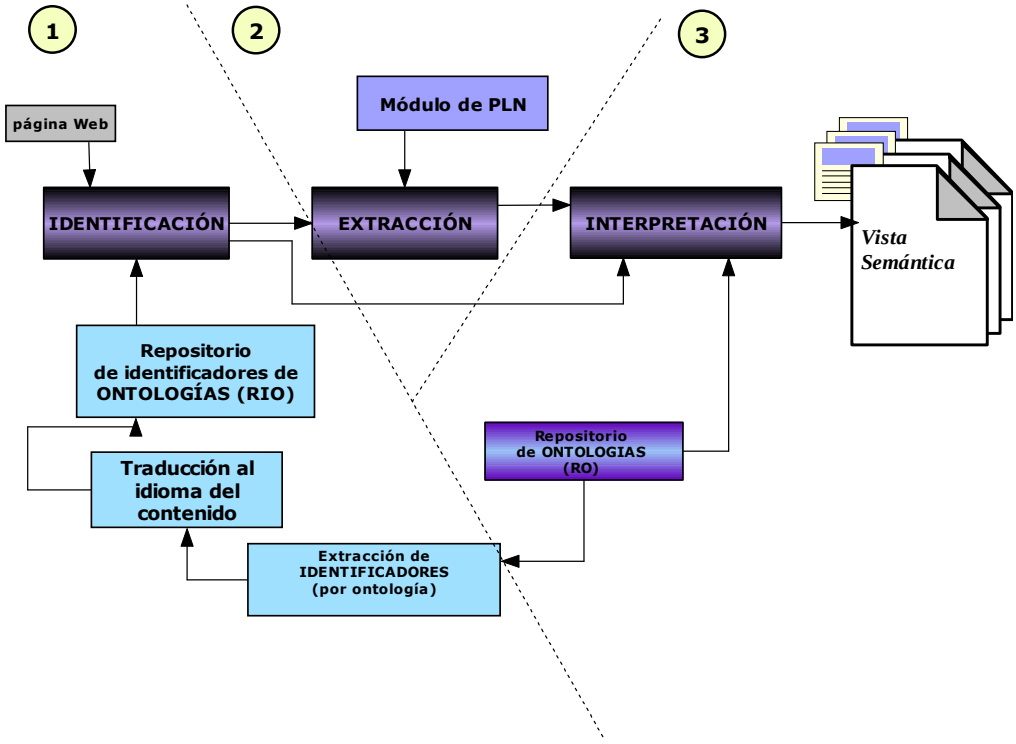


Figura 3. 1: Representación general del proceso de tranformación.

3.1. Identificación ontológica.

Antes de poder representar adecuadamente la información contenida en una página web es necesario conocer el contexto o los contextos ontológicos sobre los cuales tratar el contenido web. Como nosotros no hemos encontrado experiencias automáticas de este tipo lo hemos denominado “identificación ontológica”.

Pensamos que este proceso de identificación debe ser muy rápido, ya que permite asignar páginas web a una o varias ontologías. De manera, que es importante comparar cada página web con una amplia biblioteca de ontologías. Es por esta razón por la que hemos optado por no usar técnicas de PLN en esta etapa (que en etapas posteriores sí utilizaremos) y solo ocuparnos de la coincidencia entre los identificadores de cada ontología y el contenido de la página a identificar en el caso más sencillo. Aunque el objetivo final es más ambicioso y no debe limitarse a comparaciones rígidas, sino asociar las palabras con su concepto, sinónimos e incluso, reflejar la cercanía semántica de una palabra con respecto a otra [Barrón Cedeñoa; 2005].

En este trabajo hemos considerado que a mayor frecuencia de coincidencia mayor probabilidad de que el contenido tenga una mayor relación semántica con la ontología.

Hemos realizado las siguientes suposiciones con el objetivo de simplificar el problema:

1. Las ontologías están definidas y disponibles mediante un URI en formato OWL-DL.
2. Las ontologías son representaciones de conocimiento aceptadas por la comunidad. Lo que proponemos es un procedimiento para automatizar la anotación semántica por lo que dejamos al margen la problemática de la cercanía ontológica o cualquier otra valoración sobre las ontologías empleadas. Lo importante para nuestras pruebas es disponer de ontologías y anotar cualquier texto con respecto a ellas.
3. Cada ontología puede estar etiquetada en cualquier idioma.

3.1.1. Ontologías utilizadas.

El prototipo funciona solo en base a un universo reducido de ontologías (ver Tabla 3.1)

ONTOLOGÍA	URI
gedcom.owl	http://www.daml.org/2001/01/gedcom/gedcom#
food.owl	http://www.w3.org/2001/sw/WebOnt/guide-src/food#
wine.owl	http://www.w3.org/2001/sw/WebOnt/guide-src/wine#
pizza.owl	http://www.co-ode.org/ontologies/pizza/2005/10/18/pizza.owl#
vertebrados_es.owl	http://www.criado.info/owl/vertebrados_es.owl#

Tabla 3. 1: Ontologías soportadas en el prototipo

La razón de elegir estas cinco ontologías se debe a un criterio de búsqueda de ejemplos sencillos que pudieramos abordar. Conocida es la ontología de la “pizza” de la herramienta Protégé a la que hemos sumado un par de ontologías muy relacionadas como son las ontologías “food” y “wine” del consorcio de la W3C. Un ejemplo que nos parecía también que podíamos usar es el del parentesco, por lo que nos decidimos por “gedcom”. Pero todas estas ontologías están en inglés. De manera, que hemos desarrollado nuestra propia ontología en castellano utilizando un caso muy sencillo como es una clasificación de vertebrados.

Sin embargo, todas estas ontologías no se pueden usar directamente en el proceso de identificación, ya que deseamos minimizar el tiempo de procesado. En consecuencia es necesario realizar una etapa previa cada vez que se añade una nueva ontología a la herramienta sw2sws.

3.1.2. Repositorio de identificadores de ontologías

El repositorio de identificadores de ontologías (RO) es la columna vertebral del proceso de identificación, ya que en base a este repositorio se realizan todas las comprobaciones definidas en los métodos de identificación establecidos en este capítulo.

En la Figura 3.1, en la parte numerada como 1, se puede ver que a partir del RO, que es un conjunto de ontologías expresado el OWL DL, se obtienen los identificadores de cada ontología. Entendemos por identificadores de una ontología el conjunto de términos con las que se han nombrado las clases y propiedades que forman la misma, y que se corresponden con palabra del idioma en el que se desarrollo la ontología. Pero puede ocurrir, que los identificadores extraídos, se correspondan con palabras de un idioma diferente al del contenido del sitio web del que se pretende realizar un proceso de *identificación*. Por lo tanto, hay que tener previsto un proceso de traducción de identificadores al idioma del contenido del sitio web (ver Figura 3.1). A este resultado final lo hemos denominado repositorio de identificadores de ontologías (RIO). De manera que por cada ontología nueva que se quiera incorporar a la herramienta, es necesario realizar estos dos procesos para añadirlo al RIO.

3.1.2.1. Extracción de identificadores de clases y propiedades de una ontología

En este apartado explicamos como obtener los identificadores de una ontología de forma automática, aunque la calidad de este automatismo depende mucho de las decisiones que se tomaron en el diseño de la misma.

El proceso de extracción de los nombres (identificadores) de las clases y propiedades precisa obtener un modelo en memoria de la ontología; una vez cargado el modelo ontológico se extrae el identificador con el que se nombra las clases y las propiedades. Pero hemos comprobado, que es muy habitual nombrar las clases y propiedades con varias palabras concatenadas. De manera que hemos previsto esta casuística en el proceso de extracción de identificadores.

Hemos usado las librerías Jena para modelar la ontología en OWL mediante la clase “OntModel”:

```
OntModel owlModel =
ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM_RULE_INF);
```

La definición ontológica se establece de la siguiente manera:

```
owlModel.getDocumentManager().addAltEntry( namespace, source );
```

Donde la variable “namespace” identifica el URI y la variable “source” se refiere a la dirección de Internet donde podemos acceder a la descripción de la ontología en formato OWL, es decir, el URL.

Como se aprecia en la Figura 3.2. se cargará el fichero de la ontología a través de Internet en memoria local aplicando el método “read” al objeto “owlModel” (owlModel.read(source);). En este momento tenemos un modelo de ontología en memoria que nos permitirá obtener la lista de clases y propiedades, de manera que para listar todas las clases habrá que aplicar el método “listNamedClasses()” al objeto “owlModel” que hemos definido anteriormente.

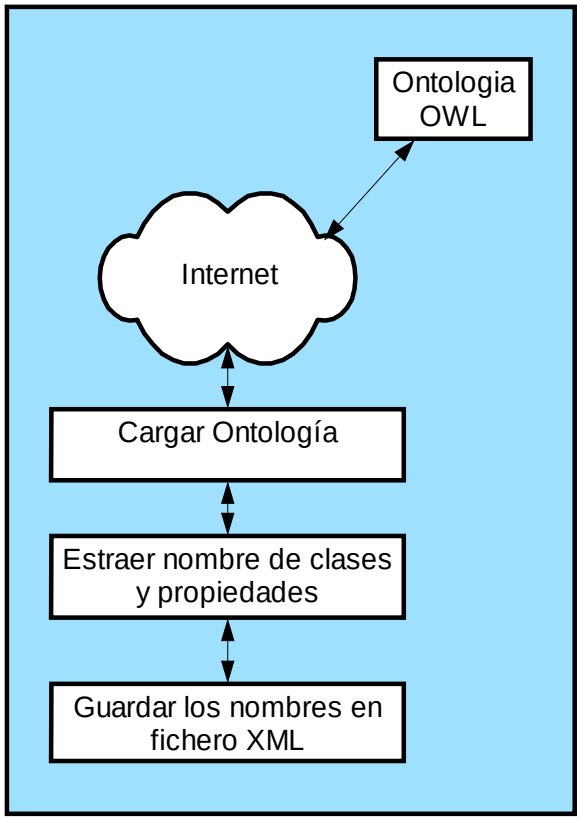


Figura 3. 2: Proceso de generación de los nombres a mapear

Este método nos permite iterar un bucle definido de la siguiente forma:

for (Iterator it = owlModel.listNamedClasses(); it.hasNext();)

Para capturar el nombre de cada clase será necesario antes crear un objeto de tipo “OntClass” al cual asignamos el valor de “it” mediante un patrón de tipo “OntClass idClase = (OntClass) it.next();”, finalmente capturaremos el nombre de la clase aplicando el método “getLocalName()” a la variable “idClase”

El nombre de todas las propiedades se obtiene de forma similar, aplicando el método “listOntProperties()” al objeto “owlModel”, de forma que el bucle se define como:

for (Iterator it2 = owlModel.listOntProperties(); it2.hasNext();)

Para la captura de los nombres de las propiedades será necesario crear un objeto de tipo “OntProperty” al cual asignamos el valor de “it2” mediante un patrón de tipo “OntProperty idPropiedad = (OntProperty) it.next();”, de manera que el nombre de la propiedad puede obtenerse aplicando el método “getLocalName()” a la variable “idPropiedad”.

Tanto para recuperar los nombres de las clases como las propiedades hemos usado Jena, la ventaja es que independientemente de los anidamientos y relaciones que pueda existir en un fichero OWL, la librería Jena es capaz de obtener esta información, de forma, que no hay que preocuparse en implementar rutinas de navegación por un fichero OWL sobre XML que puede presentar diferencias significativas de una ontología a otra.

Una vez cargado una ontología en memoria mediante Jena, por cada nombre de clase o propiedad se ha comprobado si contiene varias palabras, en cuyo caso se han extraído cada palabra por separado, despreciando las palabras de menos de tres caracteres (ver Figura 3.3), ya que, suelen ser elementos de enlace (sin significado en la ontología). Posteriormente se ha generado un archivo de salida en formato XML que contiene todas las palabras extraídas de la ontología, estas son las palabras clave que deberán traducirse al idioma del contenido que se desea identificar. Se ha comprobado que es muy frecuente la convención de usar letras minúsculas para la letra inicial de los nombres de propiedades, y mayúsculas para las clases, tal y como se especifica en las recomendaciones de W3C [Brickley et al; 2001]

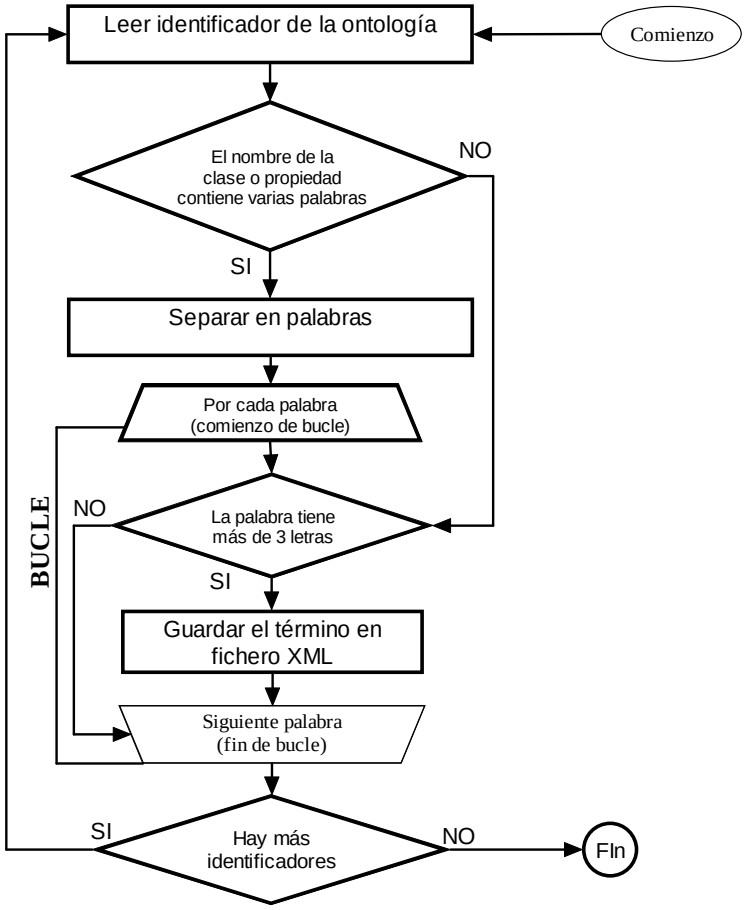


Figura 3. 3: Organigrama para extraer identificadores

El fichero XML que se obtiene posee una estructura muy simple, solo tiene tres etiquetas; la etiqueta <ontologia> incorpora el atruibuto “url” que es la dirección donde reside la descripción de la ontología, la etiqueta <idioma> se refiere al idioma de origen en que se escribió la ontología, este “tag” puede anidar a su vez todos los términos que se especifican en dicha ontología mediante la etiqueta <termino>.

Solo la etiqueta <termino> puede repetirse indefinidamente y contiene la palabra que posteriormente tendrá que ser traducida al idioma del contenido. El fichero que contiene las palabras “clave” extraidos de los identificadores de las clases y propiedades presenta el siguiente aspecto (Tabla 3.2):

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<ontologia url="http://protege.cim3.net/file/pub/ontologies/travel/travel.owl" uri="http://www.owl-ontologies.com/travel.owl#">
 <idioma origen="ingles">
      <termino origen="city"
      <termino origen="sunbathing"
      <termino origen="sports"
      <termino origen="capital"
      <termino origen="budget"
      <termino origen="hotel"
      ……
      <termino origen="rating"
      <termino origen="museums"
      <termino origen="town"
      <termino origen="family"
      <termino origen="contact"
      <termino origen="backpackers"
      <termino origen="hiking"
      <termino origen="part"
      <termino origen="offered"
      <termino origen="street"
      <termino origen="mail"
      <termino origen="code"
</idioma>
</ontologia>
```

Tabla 3. 2: Descripción XML de la lista de términos de la ontología a traducir

3.1.2.2. Traducción de identificadores de la ontología al idioma del contenido.

Toda la Web Semántica se basa en la existencia de ontologías, y éstas van a ser miles, expresadas en varios idiomas. Además, deberán desarrollarse constantemente ontologías nuevas a medida que se vayan necesitando. Por esta razón son tan importantes las investigaciones sobre técnicas de “ontology learning”, las cuales se ocupan de proporcionar metodologías para automatizar al máximo el proceso de generación de ontologías con el objetivo final de adquirir conocimiento desde un texto en lenguaje natural y representarlo automáticamente en lenguaje ontológico (OWL) [Valencia García; 2005] [Valencia García et al; 2004]. De esta forma se obtiene la infraestructura necesaria para abordar la creación de la Web Semántica (en el momento de redactar esta tesis, Swoogle [41] cuenta con enlaces a más de 10.000 ontologías).

Como ejemplo de la necesidad de traducciones de ontologías es interesante comentar que el 29 de marzo de 2007, en la jornada técnica “Web Semántica. II edición”, celebrada en la sede de ROBOTIKER (Bilbao) fue comentada esta circunstancia por varios ponentes, en concreto se ha explicado que para el proyecto EuroWorksafe [86] se desarrolló una ontología sobre el cáncer partiendo de la taxonomía sobre salud de la Agencia Europea para la Seguridad del trabajo. Esta ontología ha sido traducida a todos los idiomas de la Unión Europea.

Como las ontologías se ocupan de representar un dominio de conocimiento definiendo conceptos, a través de sus propiedades o características y estableciendo relaciones, tanto con las propiedades como con otros conceptos, significa que la ontología tiene una dependencia directa con el lenguaje particularmente en el etiquetado. De forma que, suponiendo que se tiene una ontología aceptada como válida y expresada en un idioma cualquiera ¿cuál debe ser el paso siguiente?, ¿se debe generar versiones de cada ontología para los diferentes idiomas?, y si se opta por este camino ¿cuánto tiempo se puede tardar en este proceso?, ¿quedaría algún idioma sin su traducción?.

Nuestro planteamiento, para nuestra herramienta sw2sws ha sido la de traducir sólo los identificadores de la ontología y no la ontología completa, es decir el proceso de anotación se basa en la ontología original, sin embargo el proceso de *identificación* se realiza sobre el RIO. La justificación de esta estrategia se fundamenta en las siguientes razones:

1. Como norma general no se dispone de versiones traducidas de una misma ontología a varios idiomas. Las que estan disponibles son excepciones.

2. Aún en el caso de disponer de versiones traducidas de ontologías, podrian evolucionar con independencia, lo cual parece contrario a la idea de unificar el conocimiento.

Seria deseable que exista un sitio en Internet donde las ontologías sean aceptadas y compartidas por cualquier herramienta de transformación de sitio web a sitio web semántico, deberemos pensar también en generar repositorios de identificadores ontológicos, que facilitan la traducción “on line” de los identificadores a cualquier idioma. Éste parece ser uno de los requisitos funcionales de la herramienta de transformación para los usuarios activos. Ya que, lo primero que debe hacer la herramienta es identificar la o las ontologías a las que se refieren el contenido de sus páginas. Por ejemplo, si la página está escrita en “italiano” y la ontología más adecuada está escrita en “inglés” se debe realizar un proceso que sea capaz de saltar la barrera del idioma en que se definió originalmente la ontología y que accediendo a un RIO obtenga la traducción “on line” del identificador.

En nuestra herramienta el RIO, esta constituido por un conjunto de ficheros XML que incorporan, por un lado, las palabras “claves” que identifican propiedades y clases de la ontología en el lenguaje natural con el que se definió y, por otro, las palabras equivalentes en el idioma del contenido. Para obtener estos ficheros de traducción de identificadores, hay que disponer de dos entradas (ver Figura 3.4):

1. Acceso a un diccionario entre el idioma original de la ontología y el idioma que usa en su contenido el usuario activo (ver Tabla 3.3).
2. Extraer previamente los nombres de clases y propiedades (ver Tabla 3.2).

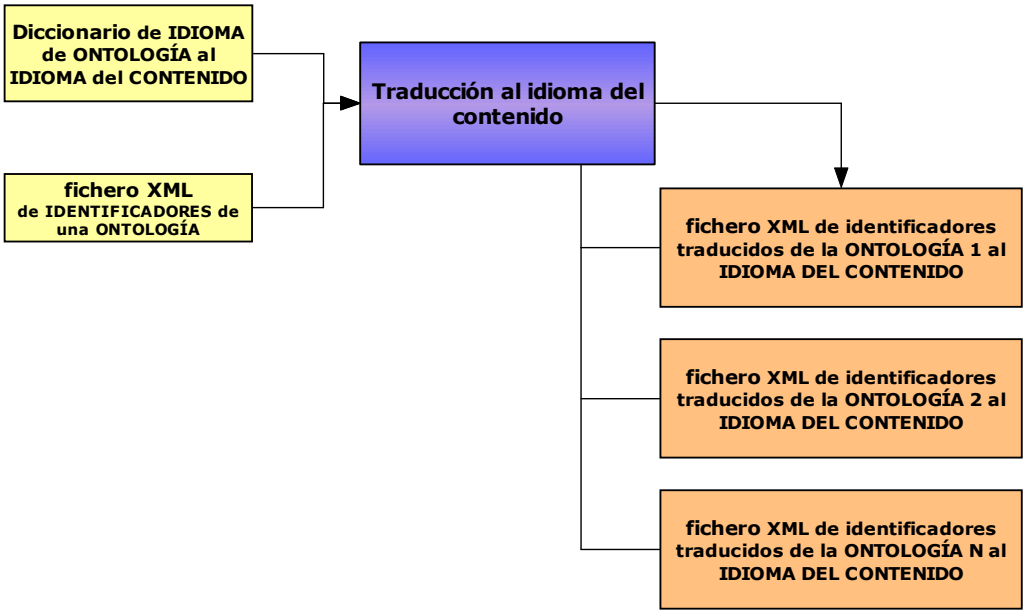


Figura 3. 4: Esquema traducción de identificadores.

Nuestras pruebas a lo largo de esta investigación utilizan castellano, esperanto e inglés. En Internet se pueden encontrar varios diccionarios de inglés-esperanto gratuitos, de forma que hemos usado la base de datos de uno de ellos. La base de datos contiene 54.499 términos y esta información la hemos volcado en un fichero XML con el formato de la Tabla 3.3.

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<idioma origen="INGLES" destino="ESPERANTO">
...
        <termino origen="Andes" destino="Andoj"/>
        <termino origen="andiron" destino="morelo"/>
        <termino origen="Andorra" destino="Andoro"/>
        <termino origen="Andorrian" destino="andora"/>
...
 </idioma>
```

Tabla 3. 3: Ejemplo de formato de diccionario inglés-esperanto

También hemos generado otros diccionarios con el mismo formato, aunque no son tan ricos en cantidad de términos como es el diccionario de castellano-esperanto (19.006 términos), y el de inglés-castellano (35.213 términos), este último diccionario nos permite comprobar un proceso de identificación ontológica basado en una situación real, partiendo de una ontología descrita en inglés y un contenido web en español.

3.1.3. El proceso de identificación. Métodos.

El proceso de identificación en cada página web se repetirá por cada una de las ontologías que se han añadido al prototipo de sw2sws y que figuran en el RIO (ver Figura 3.5). Si un término del contenido de la página web a identificar, figura en el RIO, de acuerdo al método de identificación que se detallan en los siguientes apartados producirá un incremento en el contador general de la ontología para esa página en concreto. Al finalizar el procesamiento de una página determinada, se revisan los valores de todos los contadores asociados a todas las ontologías de RIO. De forma que, según los criterios de frecuencia que se establezcan, una misma página web puede tener un número de ontologías asociadas de 0 a “n”.

Proponemos tres estrategias o métodos de identificación: la primera, denominada “exacta”, es aplicable a cualquier idioma; la segunda, aunque mucho más optimizada, solo es aplicable a lenguas muy regulares, como regla general serán lenguajes artificiales, ya que son tratables a nivel morfológico con precisión, como es el caso del esperanto. Esta estrategia la hemos denominado “identificación por raiz”; finalmente, proponemos una última estrategía que es un compromiso entre las dos anteriores, pero que es aplicable de forma inmediata a la lengua inglesa. Este método lo llamamos “Identificación basada en WordNetRDF”.

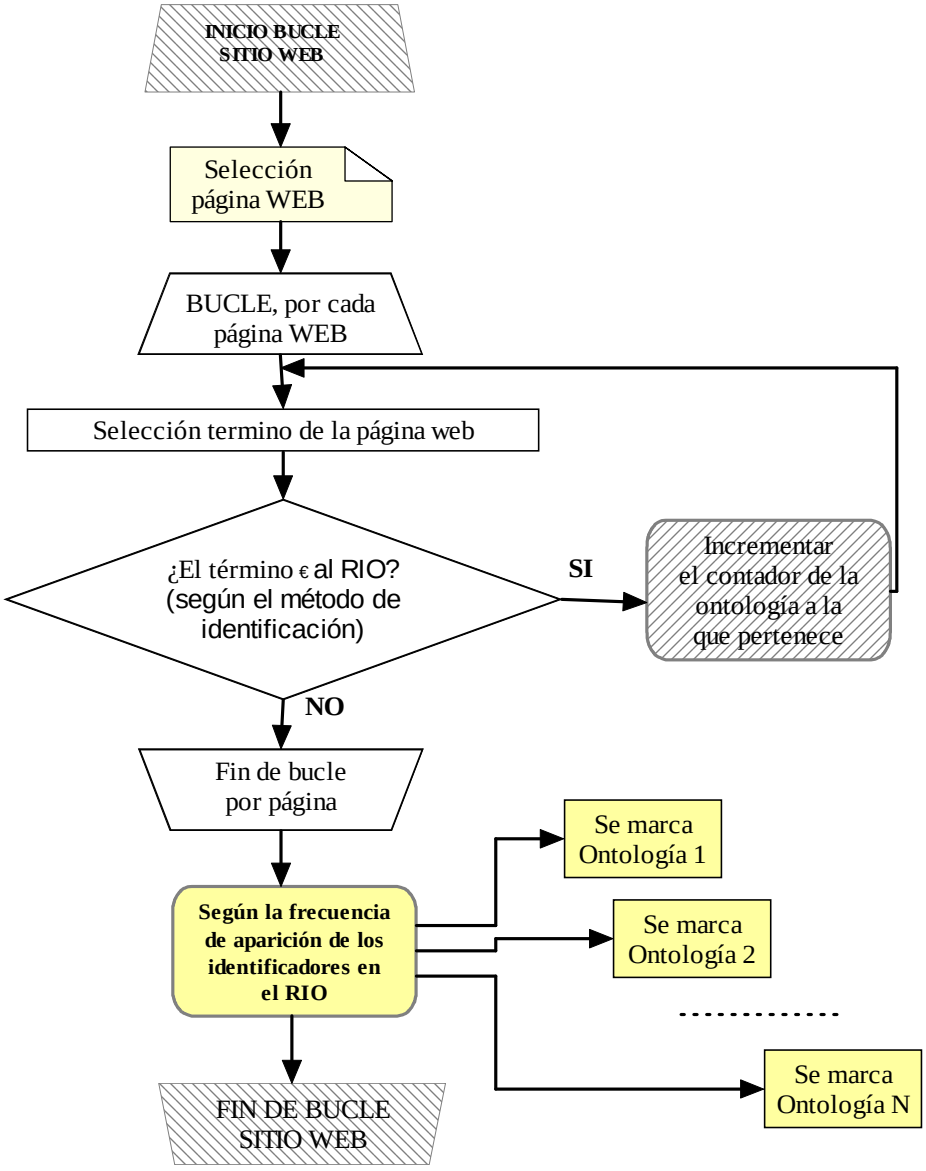


Figura 3. 5: Organigrama de identificación.

3.1.3.1. Identificación exacta

Este método es el más simple que se puede emplear. Se trata de comparar cada término del texto que se desea analizar con el diccionario. Si se encuentra la misma palabra en el diccionario y en el texto entonces el proceso resulta exitoso.

La herramienta que hemos implementado permite averiguar, aplicando este método, a qué ontología se puede asociar un determinado contenido. Para ello se utiliza un fichero de mapeo entre la ontología y el idioma del contenido que hemos generado automáticamente (Tabla 3.3).

Utilizaremos el esperanto, como lenguaje ejemplo, para poder comparar fácilmente las distintas metodologías de identificación usando nuestra herramienta “sw2sws”. El siguiente contenido es el que hemos utilizado en los ejemplos: “La metroo de Montrealo estas sistemo de subtera fervojo de Montrealo, Kebekio, Kanado. Ĝi revoluciigis la desegnon de metrooj monde per sia uzo de pneŭoj sur ĉiuj linioj kaj de la fakto ke ĉiuj stacioj estis desegnitaj de diferenca arkitekto kaj estas riĉe ornamitaj de artverkoj.La metroon kaj la busojn de la insulo Montrealo administras la publika korporacio STM (Société de transport de Montréal - Korporacio de Transporto de Montrealo).”

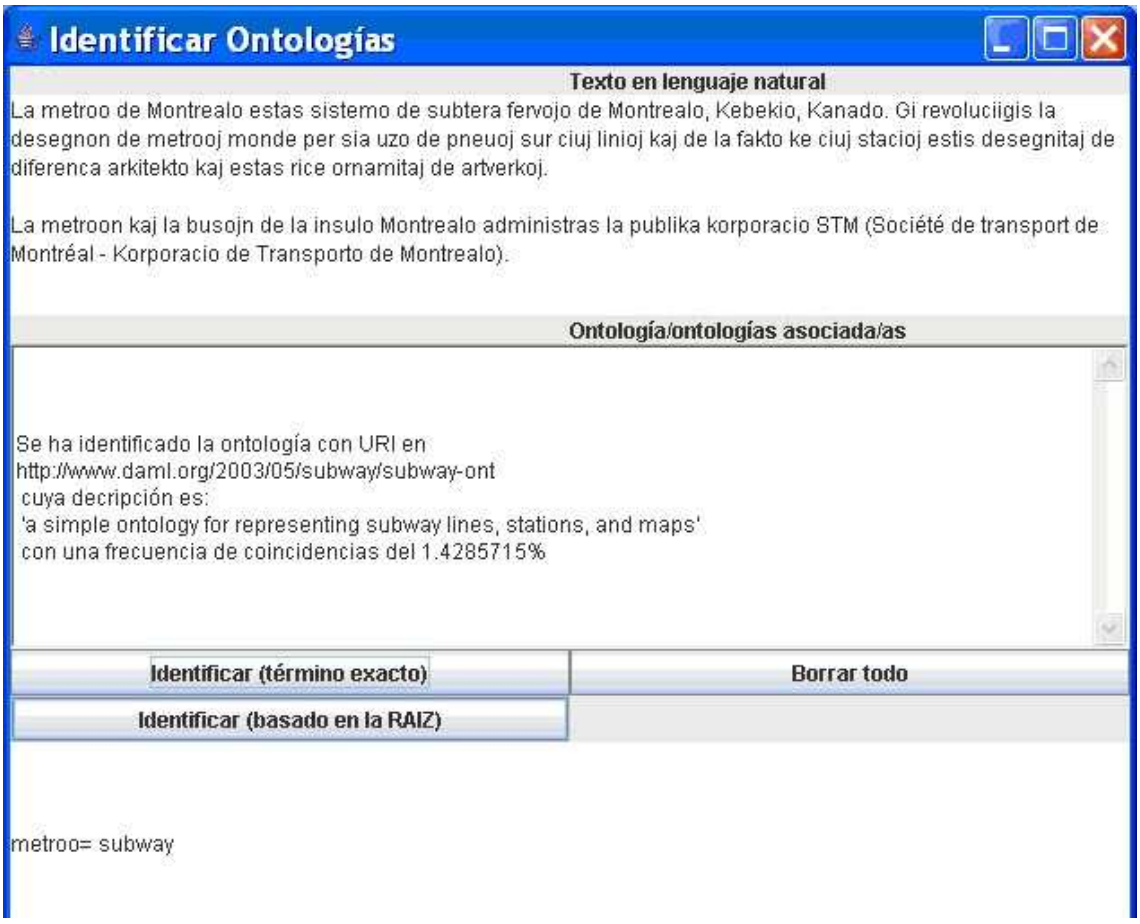


Figura 3. 6: Captura pantalla identificación exacta

En la Figura 3.6 se muestra, una captura de pantalla del prototipo. Obsérvese que en un texto en esperanto el programa reconoce cada término mediante un analizador morfológico (los pormenores se describen en el anexo A). Cada término identificado es comparado con cada ontología. Esta funcionalidad resuelve las preguntas que planteábamos con anterioridad en 3.1.2.2., es decir, ¿se deben generar versiones manuales de cada ontología para los diferentes idiomas?, y si se opta por este camino ¿cuánto tiempo se puede tardar en este proceso?,

¿quedaría algún idioma sin su traducción?. Como puede apreciarse, la solución propuesta, incorpora un mapeo que previamente se ha realizado con la herramienta del usuario activo. Pero no es necesario traducir toda la onotología, solo hace falta construir mapeos entre idiomas.

La herramienta de identificación de ontologías del usuario activo de la web debe conectarse con una determinada frecuencia con uno o varios URLs donde están las ontologías (RO). En estas direcciones se deberá comprobar si hay alguna ontología nueva y si las ontologías antiguas han sufrido modificación; esta comprobación puede implementarse fácilmente aplicando cualquier función hash (MD5, SHA, etc.…). Una vez detectadas las ontologías nuevas o que han sido modificadas se descargarán, y localmente se ejecutará un proceso que sea capaz de actualizar el RIO y que consiste en:

1. Extraer el identificador de todas las clases y propiedades
2. Cada identificador extraído será analizado para detectar las palabras que lo forman
3. Cada palabra, en el idioma original, se traducirá al idioma en que se expresa el contenido a través de un diccionario electrónico. Este proceso se refleja en un fichero XML de mapeo (Tabla 3.3).

En este ejemplo de identificación (Figura 3.4), el fichero de mapeo que hemos utilizado solo incluye dos palabras en inglés, (Tabla 3.4) que son: subway y train. Aún así, al ejecutar el programa se ha obtenido que esta ontología es la única que presenta coincidencias con un índice de frecuencia de aparición del 1,4%

```
<?xml version="1.0" encoding="ISO-8859-15"?>

<!-- puede haber diferentes ointologías -->
<ontologia uri="http://www.daml.org/2003/05/subway/subway-
ont"
descripcion="a simple ontology for representing subway
lines, stations, and maps">


<idioma  origen="ESPERANTO" destino="INGLES" >


     <termino origen="metroo" destino="subway"/>
    <termino origen="strattunelo" destino="subway"/>
          <termino origen="subfervojo" destino="subway"/>
          <termino origen="subtrajno" destino="subway"/>
          <termino origen="subvojo" destino="subway"/>

     <termino origen="akompanantaro" destino="train"/>
          <termino origen="celumi" destino="train"/>
          <termino origen="dresi" destino="train"/>
          <termino origen="kapabligi" destino="train"/>
          <termino origen="obeigi" destino="train"/>
          <termino origen="trajno" destino="train"/>
          <termino origen="trejni" destino="train"/>

</idioma>
</ontologia>
```

Tabla 3. 4: Mapeo de términos ontológicos para el ejemplo

3.1.3.2. Identificación basada en la raíz de términos

Este método es una mejora sobre el anterior, ya que las comparaciones no son estrictas, sino que se comparan las raíces o lexemas de las palabras. Con esto se consigue identificar palabras que quizás no están explícitamente escritas en nuestro diccionario pero que tienen que ver con el contenido de la ontología.

La identificación basada en el lexema o raíz de la palabra permite optimizar el proceso de identificación. Por cada palabra a identificar se realiza un proceso morfológico con el fin de extraer su raíz, dicha raíz se usará para compararla en el fichero XML de mapeo. Por ejemplo, en esperanto la palabra *vino* se dice también “vino”, consiguientemente si la comparación del diccionario es por el método basado en la raíz, no hace falta que palabras como “vinon”, “vinoj” o “vinojn” se incluyan explícitamente en el diccionario, ya que, comparten la misma raíz y el proceso de identificación ontológico basado en la raíz de términos reconocerá cada una de ellas como términos relacionados con la ontología. En esperanto, este análisis morfológico es muy rápido, debido a que este idioma solo tiene 16 reglas sin excepciones. Sin embargo, aplicar esto al castellano o el inglés penalizará el tiempo de proceso, ya que, no serán unas pocas reglas sino conjuntos de procesos complejos, como la lematización, técnica de procesado de lenguaje natural que se ocupa de obtener el lema o raíz de una palabra.

Finalmente, podemos ver (Figura 3.7) que con la misma entrada que utilizamos en el método anterior, ahora la identificación ontológica es más rica, pasando de una frecuencia de coincidencia del 1,4% al 4,3%.

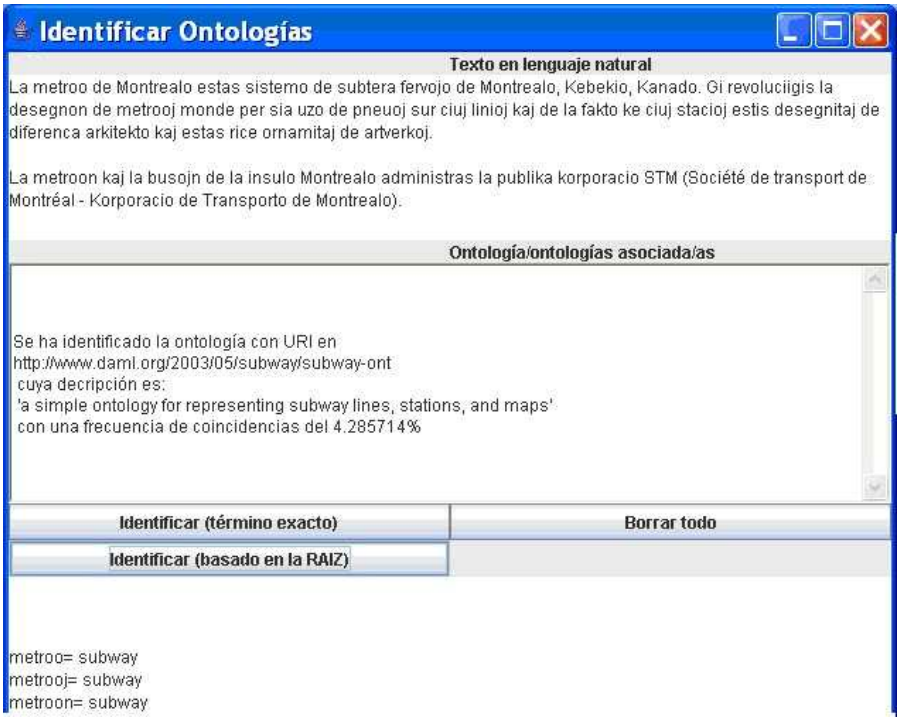


Figura 3. 7: Identificación ontológica basada en raices

3.1.3.3. Identificación basada en WordNetRDF

Este enfoque permite abordar una situación “real” (para el inglés) con resultados razonables, la estrategia consiste en comenzar comparando los términos que aparecen en el contenido con los que aparecen en los identificadores de las clases y propiedades,. Hasta aquí, es una identificación “exacta”. La novedad es que después del primer proceso, estos términos que se han extraído de la ontología se volverán a procesar con WordNetRDF, ideado y desarrollado por Alvaro Graves [21]. Por cada palabra que se procese con WordNetRDF se obtendrá una lista de sinónimos. Esta funcionalidad se consigue realizando una petición del estilo de:

http://pullay.dcc.uchile.cl/~agraves/wordnetrdf.php?word=[PALABRA]&query=Synonym&AskWord=AskWord&word1=&word2=

Donde [PALABRA] representa cualquier término del que se desee obtener un sinónimo. La respuesta del sistema WordNetRDF es una lista de sinónimos expresada en RDF de manera

que el tratamiento es inmediato con el API de Jena. Por ejemplo, si queremos obtener todos los sinónimos de la palabra “subway” tendremos que lanzar la siguiente URL:

http://pullay.dcc.uchile.cl/~agraves/wordnetrdf.php?word=subway&query=Synonym&AskWord=AskWord&word1=&word2=

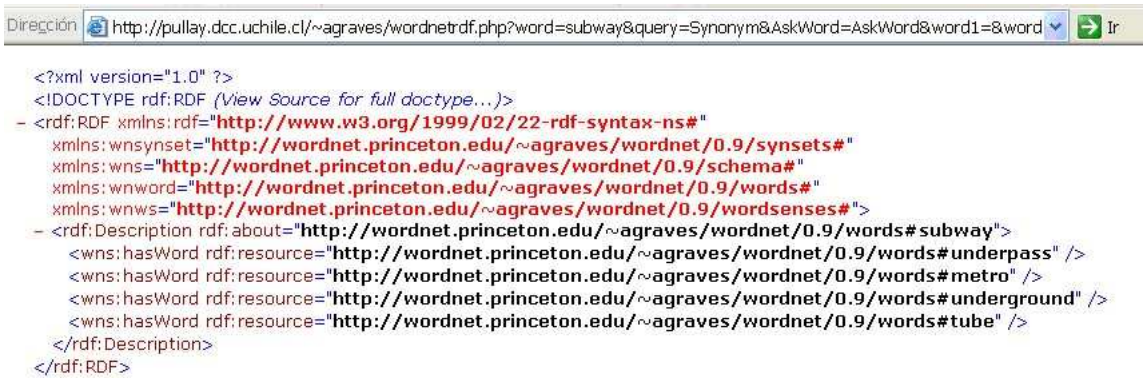


Figura 3. 8: Respuesta on line WordNetRdf

Con esto se consigue identificar palabras que quizás no están explícitamente escritas en nuestro diccionario pero que tienen que ver con el contenido de la ontología. Es decir, estamos en una situación similar a la que permite el esperanto por su singular estructura. WordNetRDF (ver Figura 3.8) nos ha proporcionado los términos como "underpass", "metro", "underground" y "tube" que son sinónimos de "subway”. De esta forma se multiplican las posibilidades de identificar términos relacionados con la ontología. Por consiguiente, el uso de WordNetRDF abre un camino de investigación aplicable al proceso de identificación. El único inconveniente importante que presenta este enfoque es la lentitud de proceso (protocolo http) en una operación que requiere de una extraordinaria velocidad, ya que, se trata de comprobar la frecuencia de aparición de términos con cientos de ontologías.

3.1.4. Comentarios sobre la metodología de identificación

En los ejemplos de identificación exacta y por raíz se ha utilizado el esperanto como lenguaje de contenido, pero es factible realizar procesos de identificación para cualquier contenido en cualquier lenguaje natural (con mayor o menor éxito), ya que la arquitectura que hemos diseñado es parametrizable, de manera que si queremos trabajar en español como lenguaje del contenido y en inglés como idioma de etiquetado en la ontología, bastará usar un fichero de mapeo del estilo de:

<ontologia

url="file:./pizza.owl"

uri="http://www.co-ode.org/ontologies/pizza/2005/05/16/pizza.owl#" descripcion="Ontología de PIZZA, mapeo español-ingles">

<idioma origen="ESPAÑOL" destino="INGLES">

<termino origen="pizza" destino="pizza"/>

<termino origen="queso" destino="cheese"/>

<termino origen="cuatro" destino="four"/>

…..

De forma que al ejecutar la herramienta para identificar la ontología podremos escribir directamente en español, y el sistema traducirá los términos al idioma en que se definió la ontología (en el caso de la ontología “pizza.owl” ha sido descrito en inglés). De manera que si escribimos la frase “pizza de cuatro queso” obtenemos (ver Figura 3.9) un coeficiente de coincidencias del 75%, ya que, hemos utilizado una estrategia de identificación exacta, y las palabras “pizza”, “cuatro”, y “queso” figuran en el diccionario escritas exactamente así.

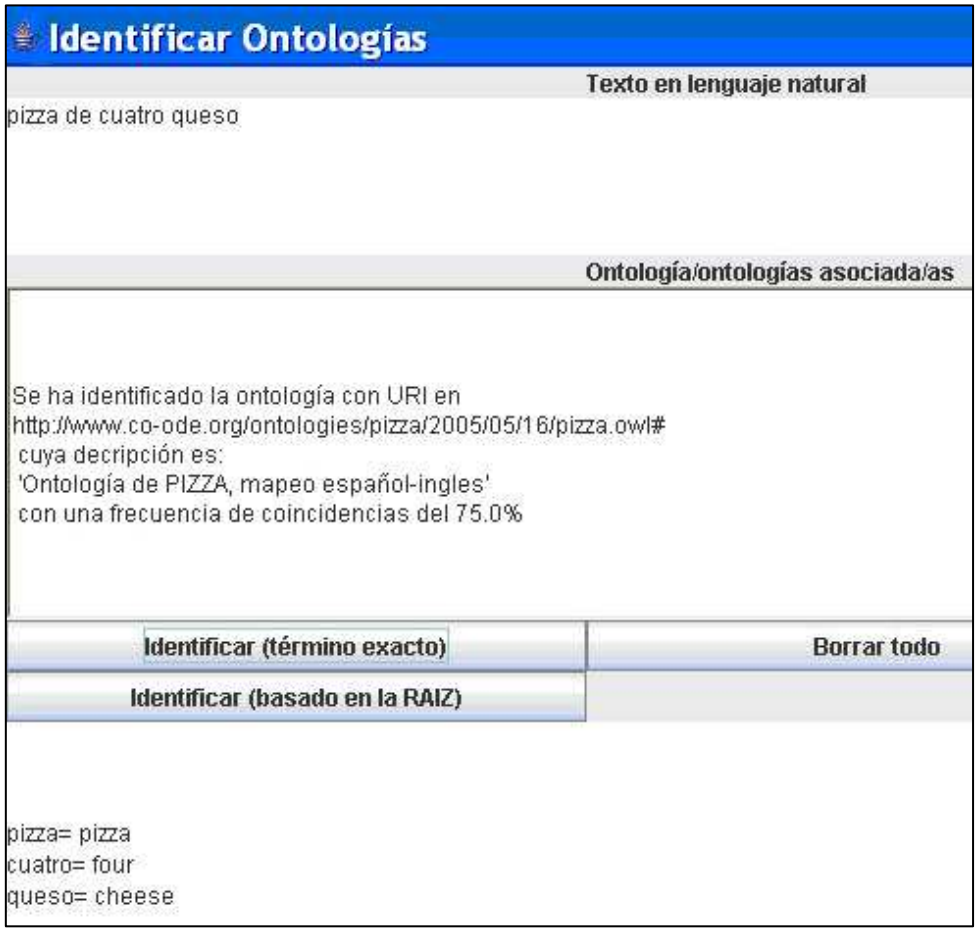


Figura 3. 9: Ejemplo identificación exacta español-inglés

Pero si variamos la frase un poco, solamente escribir “queso” en plural, entonces el sistema obtiene un coeficiente de coincidencias del 50%, y si cambiamos la frase por “pizzas de cuatro quesos”, se reduce a un 25% (Figura 3.10).

Esto es debido a que si no se encuentra el término exacto entonces no se reconoce. En la siguiente pantalla puede verse el resultado de la herramienta cuando escribimos la frase en plural.

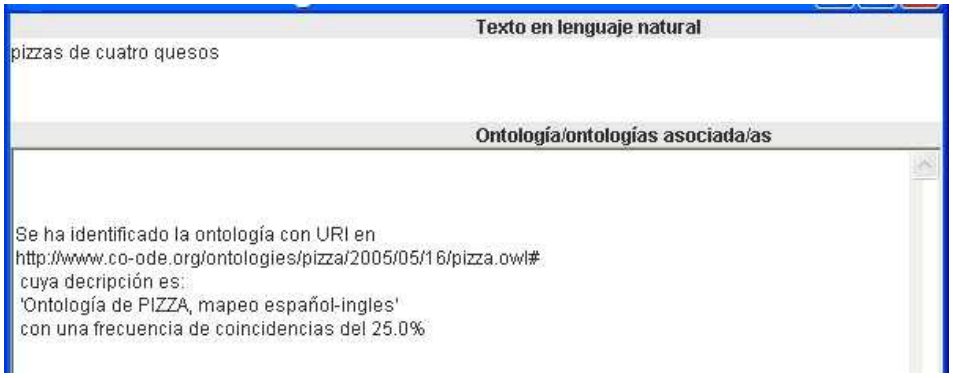


Figura 3. 10: Ejemplo 2 identificación exacta español-inglés

Veamos ahora como el esperanto es un lenguaje que nos permite ser robustos a todas estas pequeñas variaciones, comenzando por el mismo ejemplo “pico kun kvar fromagxo” (pizza de cuatro queso)” (Figura 3.11) y como ocurría en castellano se encuentra un frecuencia de concordancia del 75%.

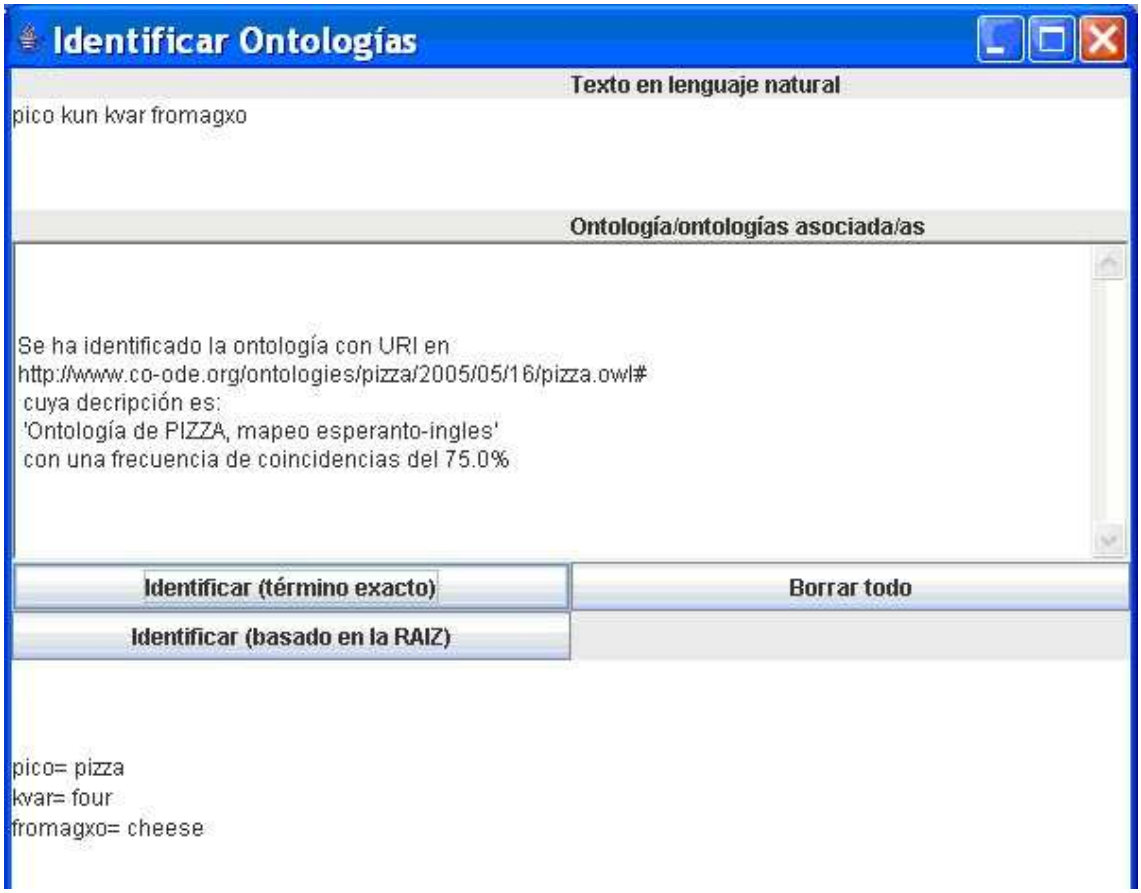


Figura 3. 11: Ejemplo identificación por raiz con esperanto

Si reescribimos la frase en plural (un plural en esperanto se construye añadiendo “j”), se obtiene el mismo resultado del 75%, como puede observarse en la siguiente captura de pantalla (Figura 3.12)

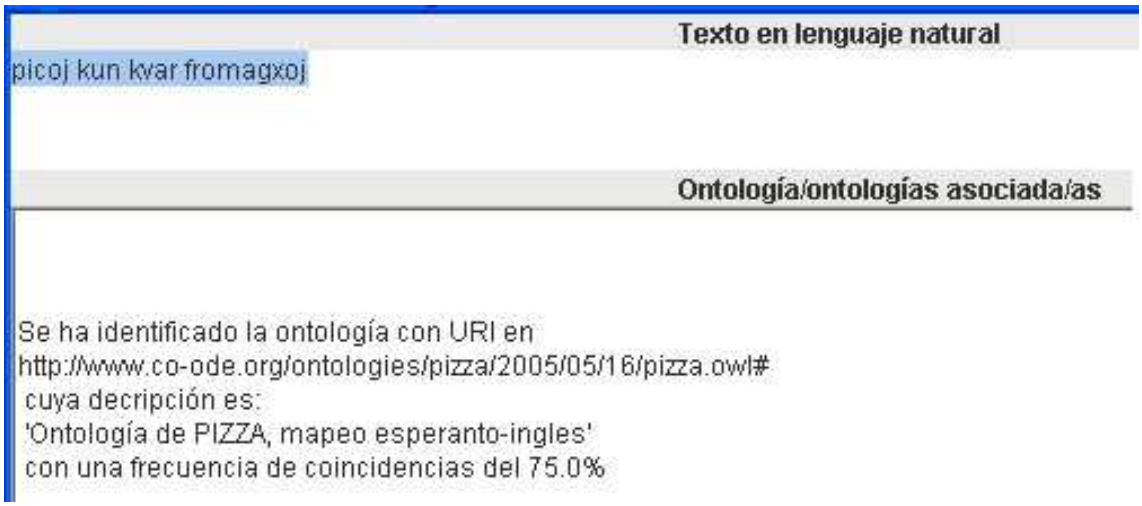


Figura 3. 12: Segundo ejemplo con esperanto

En definitiva, para optimizar el proceso se recomienda aplicar el método de identificación por raíz, ya que este método es el que proporciona mayor robustez. Sin embargo, este método no es aplicable a cualquier idioma si no utilizamos técnicas de procesado de lenguaje natural. El método exacto (aplicable al castellano o al inglés) es menos eficiente, debido a que si un término no coincide rigurosamente entonces no se reconoce.

3.1.5. Herramienta automática para la identificación ontológica de un sitio web

Hasta aqui hemos expuesto los principios de funcionamiento de la identificación tratando solo pequeños textos. En este epígrafe generalizamos para tratar todo el contenido de un sitio web. Como se ha dicho al principio de este capítulo, nuestro prototipo usa solo unas pocas ontologías: tres asociadas al dominio de la gastronomía (food.owl, wine.owl y pizza.owl), una asociada a los vertebrados (vertebrados_es.owl) y por última otra ontología referida al parentesco (gedcom.owl). De estas, solo la ontología de vertebrados, se ha definido originalmente en castellano, pero el resto estan expresadas en inglés.

Como hemos dicho, para que el proceso de identificación funcione es necesario traducir sus términos a castellano, ya que, los sitios webs que se deseaba transformar estaban en castellano. Estos sitios webs con los que hemos probado nuestra herramienta (ver Tabla 3.5.) son:

Sitio Web	número de paginas	Identificadas	tiempo (minutos)
http://www.perrilandia.com/	423	419	2
http://www.todoperros.com	1852	1546	10
http://www.mascotas.com	1060	1028	12
http://www.lomejordelagastronomia.com/	34290	127	145
http://www.mascotasyhogar.com/	6845	874	39
TOTAL	44470	3994	208

Tabla 3. 5: Relación de webs (proceso identificación)

Obsérvese que en media, la velocidad de proceso es de poco más de 213 páginas procesadas al minuto, lo que confirma que el proceso es costoso computacionalmente. Téngase también en cuenta que las pruebas se han realizado con un equipo de tipo medio del que cualquier usuario activo puede disponer hoy en dia (1 Gbytes de RAM, procesador de 3GHz,

189 Gbytes disco y Windows XP). Que las características del equipo parezcan mediocres justifica mejor la decisión de utilizar un repositorio de identificadores de ontologías (RIO) en el proceso de *identificación*, ya que, estos tiempos se pueden considerar pesimistas. Luego es viable que un usuario activo ejecute este tipo de herramientas para transformar su sitio web al nuevo paradigma, ya que solo tendria que esperar unos pocos minutos.

Con sw2sws, para realizar el proceso de identificación sobre un sitio web, se debe definir un fichero de proyecto tipo “properties” en el cual se especifica la URL y el path local entre otros atributos. Luego ejecutamos la herramienta y se selecciona el proyecto con el que se desea trabajar (Figura 3.13), entonces se muestran todas las características de configuración del sitio web. Ademas del camino donde localmente se tiene una copia de la web, se indica la URL donde reside el sitio web. Así como los ficheros que contendrán la relación de páginas identificadas y la lista de las *vistas semánticas* que se generen.

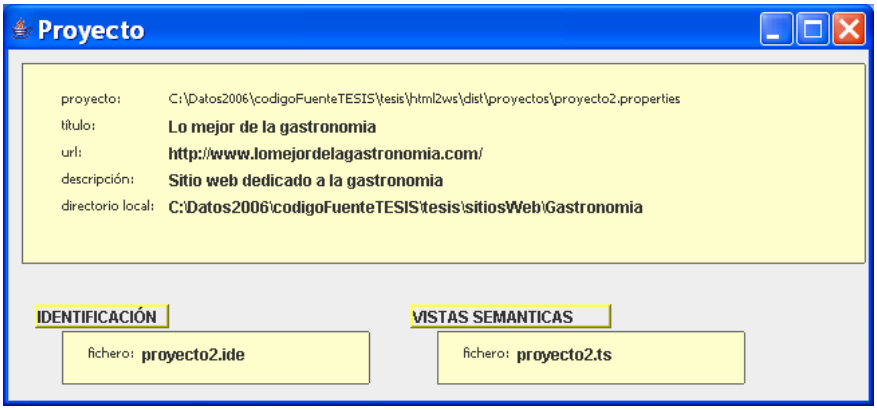


Figura 3. 13: Características del proyecto

Una vez que comienza el proceso, se nos informa por pantalla de la evolución del mismo:

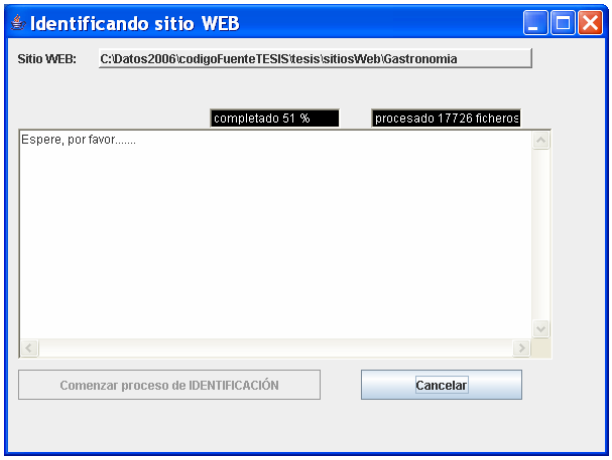


Figura 3. 14: Progreso de identificación

Completado el proceso se visualiza un informe rápido de todos los ficheros analizados (ver Figura 3.14)

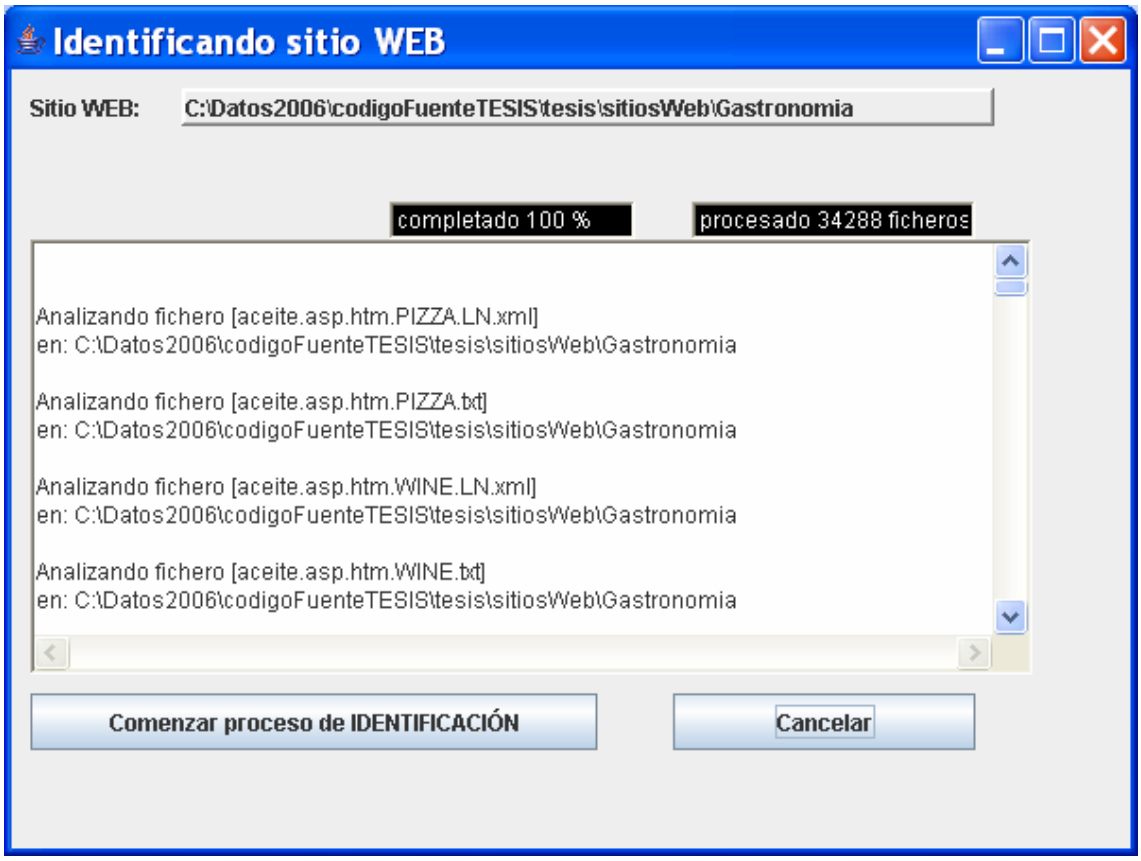


Figura 3. 15: Informe resumen de identificación

Pero es en la opción de “informe ontológico” (Figura 3.15) donde se obtienen estimaciones de las *vistas semánticas* que se podrán generar a partir de este proceso de identificación, asi como información de las ontologías identificadas, porcentajes, umbral del filtro, etc….

3.2. Extracción y análisis

La automatización en la extracción se ha enfocado como pieza aislada del proceso de transformación (ver Figura 3.1, parte numerada como 2) para permitir, que en el futuro, se pueda reemplazar este elemento por implementaciones de expertos en materia de PLN, ya que, el procesamiento de lenguaje natural no es un propósito de nuestra investigación. De ella se ocupan varias comunidades de investigadores, tanto en la rama de la Inteligencia Artificial (PLN, o NLP; Natural Language Processing) como en la rama de la lingüística computacional. Por todo esto, minimizaremos el problema del procesamiento de lenguaje natural lo máximo posible. En nuestro procedimiento hemos tenido que realizar una pequeña implementación para comprobar el funcionamiento general del proceso de transformación de un sitio web a un sitio

web semántico, debido a que sin un módulo de estas características no se podría abordar la migración. El resultado del PLN será filtrado por la ontología en el momento de la interpretación que genera la *vista semántica*, de forma que ambos procesos pueden verse como un procesado de lenguaje natural conducido por ontologías.

La extracción de información consiste en recuperar automáticamente parte de la información que contiene un documento no estructurado y representarlo adecuadamente, de forma estructurada sin intervención de ontologías. El modelo de recuperación de información que se ha seguido se inspira en el descrito por Manning y Schütze. Distinguimos tres etapas consecutivas (ver Figura 3.16):

- PREPROCESADO.
- PROCESADO DE LENGUAJE NATURAL (PLN).
- REPRESENTACIÓN FORMAL DEL LENGUAJE NATURAL EN XML.

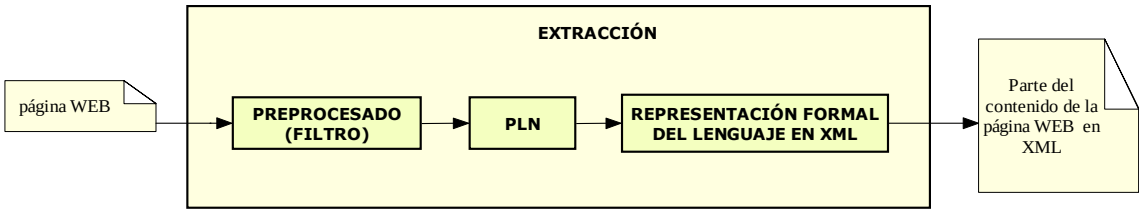


Figura 3. 16: Etapas de extracción.

3.2.1. Preprocesado

La etapa de preprocesado es un filtro que elimina todas las etiquetas HTML generando un fichero de salida de texto que será la entrada para el proceso siguiente, que es la etapa de procesado de lenguaje natural. El preprocesado además realiza algunas funciones de eliminación de ruido, por ejemplo, si una frase es demasiado corta o es una línea en blanco se elimina.

3.3.2. Procesado de Lenguaje Natural (PLN)

Para el desarrollo del módulo de PLN hemos seguido un proceso de evolución. En una primera aproximación lo hemos implementado en esperanto, ya que, los lenguajes artificiales, presentan la característica de tener unas pocas reglas y muy pocas excepciones a éstas, por

lo que se convierten en candidatos ideales en la comunicación hombre-ordenador. Su sencillez y regularidad permiten hacerlos computables con rapidez. Pero cuando hemos avanzado más en nuestra herramienta, hemos considerado usar sitios web en castellano reales, y consecuentemente, hemos avanzado hacia el PLN en castellano. Detallamos a continuación ambos módulos de PLN.

3.2.2.1. PLN en Esperanto

La ventaja de tratar un “lenguaje natural de laboratorio” es que la identificación de términos y de las funciones que desempeñan en una frase es más sencilla que tratando “lenguajes naturales reales”. De forma, que operando con esperanto como si se tratará de un lenguaje natural podemos mostrar el funcionamiento de la Web Semántica.

Para construir una clase (en Java), que nos permitan trabajar con el esperanto, primero hay que conocer las características de esta lengua, de manera que se puedan establecer los requisitos de funcionamiento.

ALFABETO

En Esperanto hay 28 letras, de las cuales 5 son vocales y una es semivocal.

A, B, C, Ĉ, D, E, F, G, Ĝ, H, Ĥ, I, J, Ĵ, K, L, M, N, O, P, R, S, Ŝ, T, U, Ŭ, V, Z.

a b c ĉ d e f g ĝ h ĥ i j ĵ k l m n o p r s ŝ t u ŭ v z.

Cada letra se pronuncia de una única manera, cuando se es capaz de leer en voz alta el alfabeto, ya se puede pronunciar correctamente todas las palabras. Las letras “ñ”, “q”, “w”, “x” e “y” no existen en Esperanto.

Como puede apreciarse en esperanto tenemos una serie de letras difíciles de representar (Tabla 3.6) con los procesadores de textos y los lenguajes de programación actuales, estas son:

TIPO LETRA	LETRA	REPRESENTACIÓN ALTERNATIVA
consonante	Ĉ ĉ	“Cx” “cx”
consonante	Ĝ ĝ	“Gx” “gx”
consonante	Ĥ ĥ	“Hx” “hx”
consonante	Ĵ ĵ	“Jx” “jx”
consonante	Ŝ ŝ	“Sx” “sx”
semivocal	Ŭ ŭ	“Ux” “ux”

Tabla 3. 6: Representación de letras

Hay varias representaciones alternativas de estos símbolos aunque nosotros indicamos la más cómoda de todas ellas. Esta representación alternativa para estos símbolos es la que se usa con frecuencia en el mundo de Internet.

Estos símbolos no aparecen en la tabla ASCII ampliada, por lo que hay que ir a UFT-8 que es un conjunto de tablas que enumeran los caracteres necesarios para todo el mundo. Estas tablas incluyen no sólo las tablas de Latinos, Griego, Ruso, Hebreo, Arábigo, Armenio sino también ideogramas Chino, Japonés y Coreano. En concreto, los códigos que necesitamos para representar estas letras del esperanto lo hemos encontrado en el documento “Latin Extended-A Range: 0100-017F” [25] , la codificación es la siguiente (Tabla 3.7):

Ĉ ĉ	Cx=0108 cx=0109
Ĝ ĝ	Gx=011C gx=011D
Ĥ ĥ	Hx=0124 hx=0125
Ĵ ĵ	Jx=0134 jx=0135

Ŝ ŝ	Sx=015C sx=015D
Ŭ ŭ	Ux=016C ux=016D

Tabla 3. 7: Codificación UTF-8 esperanto

Que tratado en Java hay que expresarlo de la siguiente manera:

```
String cx=”\u0109”;
String Cx=”\u0108”;
```

Para facilitar el trabajo hemos implementado un método con el que filtraremos todos los “Strings”, de forma que si contienen letras especiales de esperanto quedarán representadas correctamente mediante el conjunto de caracteres UFT-8. El método *letrasEsperanto()* requiere como parámetro de entrada el “token” que se va a procesar, esto es, cualquier término del lenguaje. Este término es recorrido letra a letra, de manera que en el momento de identificar alguna de las representaciones alternativas se reemplaza su lugar por la representación adecuda usando los códigos UFT-8 anteriormente comentados. Finalmente el método devuelve el “String” corregido.

FORMACIÓN DE PALABRAS

La mayor parte de las palabras del esperanto (a excepción de elementos indivisibles como preposiciones, cardinales, el artículo, etc..) se forma mediante el esquema que se representa en la Figura 3.14:

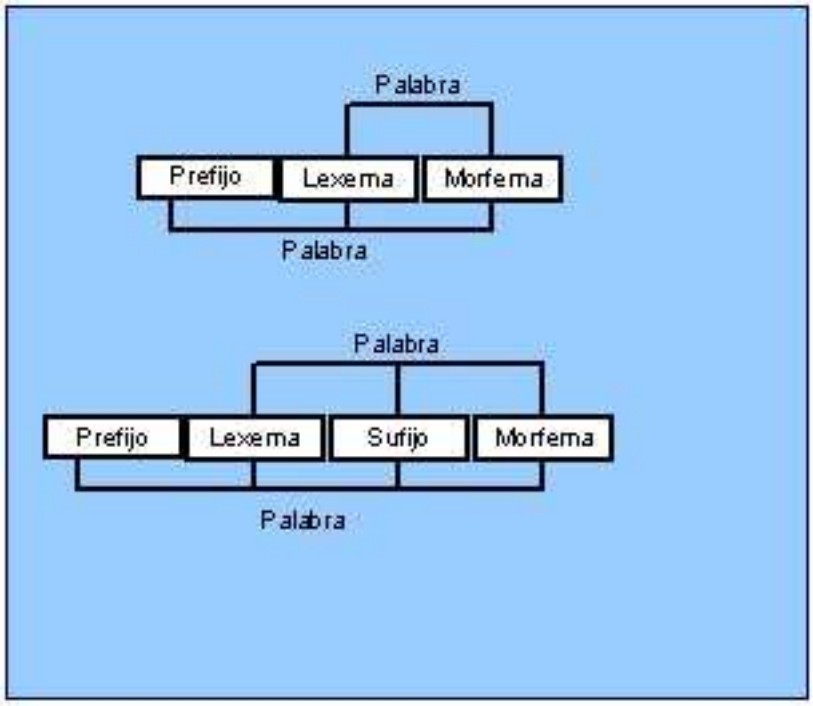


Figura 3. 17: Esquema de formación de palabras en Esperanto

Como ocurre en cualquier lengua [Gutiérrez Araus et al; 2004], la palabra, aun siendo la unidad sintáctica más simple, presenta una estructura compleja: está compuesta por unidades menores, que se clasifican en lexemas, morfemas y afijos (prefijos, sufijos).

El lexema [Gutiérrez Araus et al; 2004] es el segmento de la palabra que aporta el significado (se corresponde con la raíz). El morfema, por el contrario, aporta el valor gramatical (singular/plural, masculino/femenino, tiempo verbal, sustantivo/adjetivo/adverbio y acusativo).

En cuanto a los afijos; son de dos tipos, prefijos y sufijos. En esperanto, como en cualquier otra lengua, los prefijos preceden siempre al lexema. Por otro lado, los sufijos, se añaden detrás del lexema y preceden a los morfemas.

Con todo esto, podemos concluir, que hay una serie de palabras que deben ser analizadas en un orden preciso para identificarlas correctamente, de manera que analizaremos estas palabras en tres etapas:

1. PRIMERO: COMPROBAR LOS MORFEMAS
2. SEGUNDO: COMPROBAR LOS SUFIJOS
3. TERCERO: COMPROBAR PREFIJOS

Estas tres comprobaciones se han incluido en un único método denominado “terminoDivisible()”, al cual se le pasa la palabra a analizar como argumento de tipo “String” y

devuelve un “String” que indica el tipo de palabra que representa.

OTRAS PALABRAS

Pero hay otras palabras especiales que no se componen de estructuras complejas, simplemente son términos indivisibles. Por ejemplo, el término “la” es el único artículo determinado que posee el esperanto.

Otros grupos de palabras indivisibles que posee el esperanto, como ocurre en español, son los pronombres, las conjunciones, los términos interrogativos, ordinales, cardinales, etc… De manera, que este tipo de términos no requieren esfuerzo algorítmico, ya que son términos conocidos e invariantes. El método, de la clase “gramaticaEsperanto” (ver Figura 3.18), que hemos implementado para identificar este tipo de palabras se ha denominado *terminoIndivisible()* y soporta un parámetro tipo “String” que representa la palabra que se desea analizar.

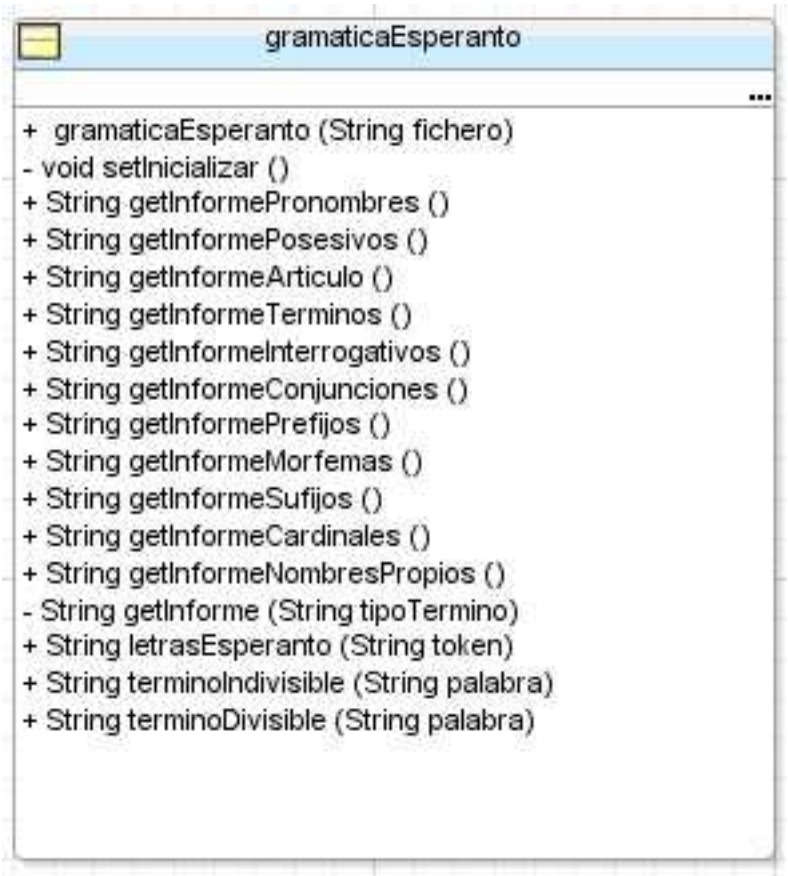


Figura 3. 18: Métodos de la clase gramaticaEsperanto

El método *terminoDivisible()* junto con el método *terminoIndivisible()* nos sirven para realizar un análisis morfológico de las palabras del Esperanto.

REGLAS

La gramática del esperanto es muy simple, pero no por eso incompleta, se basa en 16 reglas que en conjunto son la base del idioma. Las reglas no tienen excepciones. Son tan concretas como esta que escribimos a continuación:

“regla 1: El esperanto sólo posee el artículo determinado [la] igual para todos los géneros, números y casos. Carece de artículo indeterminado. “

El resto de las reglas son del estilo, pero no las escribimos en esta tesis ya que pueden ser consultadas en cualquier libro sobre gramática de este idioma. Lo importante para nosotros ha sido que hemos podido implementar en Java un analizador sintáctico solo teniendo en cuenta estas reglas. Por ejemplo, solo con la terminación de una palabra podemos saber que el término está en modo acusativo (terminan en letra n), que es un verbo en pasado (termina en “is”), etc…

CONCLUSIÓN

Hemos implementado un sencillo procesador de lenguaje esperanto que nos permite identificar cada término, además es posible detectar funciones sintácticas como son el sujeto, verbo y complemento directo. Sin embargo, una demostración de transformación de un sitio web en esperanto a un sitio web semántico en esperanto resulta poco interesante si no se conoce este idioma. En consecuencia nos hemos decidido por integrar en nuestro prototipo un PLN en castellano lo más elemental posible, pero suficiente para poder mostrar un proceso completo de transformación de sitio web a sitio web semántico en castellano.

3.2.2.2. PLN en Castellano

Los sistemas de PLN requieren primero de un análisis morfológico para determinar que tipo de términos se están tratando. Una vez superado este análisis se puede realizar un análisis sintáctico para estudiar las relaciones establecidas entre las palabras que forman unidades superiores como sintagmas y frases. Después, el nivel semántico, donde se estudia el significado

de palabras y frases. Por último se alcanza el nivel pragmático, basado en la relación del lenguaje con el contexto en que es utilizado. En determinadas circunstancias, el sentido de las palabras que forman una frase tiene que interpretarse recurriendo al contexto.

Además de todo esto hay dos problemas añadidos a todos los niveles, estos son la variación y la ambigüedad lingüística. La variación lingüística se refiere a la posibilidad de utilizar diferentes palabras o expresiones para comunicar una misma idea. En cambio, la ambigüedad lingüística se produce cuando una palabra o frase permite más de una interpretación. A continuación se muestran algunos ejemplos [Vallez et al; 2007] que ilustran la repercusión de estos fenómenos en el proceso de recuperación de información:

Nivel morfológico: una misma palabra puede adoptar diferentes roles morfosintácticos en función del contexto en el que aparece, ocasionando problemas de ambigüedad.

Ejemplo: *Deja la comida que sobre sobre la mesa de la cocina, dijo llevando el sobre en la mano.*

La palabra “sobre” es ambigua morfológicamente ya que puede ser un sustantivo masculino singular, una preposición, y también la primera o tercera persona del presente de subjuntivo del verbo sobrar.

Nivel sintáctico: centrado en el estudio de las relaciones establecidas entre las palabras para formar unidades superiores, sintagmas y frases, se produce ambigüedad a consecuencia de la posibilidad de asociar a una frase más de una estructura sintáctica. Por otro lado, esta variación supone la posibilidad de expresar lo mismo pero cambiando el orden de la estructura sintáctica de la frase.

Ejemplo: *María vio a un niño con un telescopio en la ventana* .

La interpretación de la dependencia de los dos sintagmas preposicionales, con un telescopio y en la ventana, otorga diferentes significados a la frase: (i) María vio a un niño que estaba en la ventana y que tenía un telescopio, (ii) María estaba en la ventana, desde donde vio a un niño que tenía un telescopio, y (iii) María estaba en la ventana, desde donde miraba con un telescopio, y vio a un niño.

Nivel semántico: donde se estudia el significado de una palabra y el de una frase a partir de los significados de cada una de las palabras que la componen. La ambigüedad se produce porque una palabra puede tener uno o varios sentidos, es el caso conocido

como polisemia.

Ejemplo: *Luís dejó el periódico en el banco* .

El término banco puede tener dos significados en esta frase, (i) entidad bancaria y (ii) asiento. La interpretación de esa frase va más allá del análisis de los componentes que forman la frase, se realiza a partir del contexto en que es formulada.

Y también hay que tener en cuenta la variación léxica que hace referencia a la posibilidad de utilizar términos distintos a la hora de representar un mismo significado, es decir el fenómeno conocido como sinonimia:

Ejemplo: *Coche / Vehículo / Automóvil.*

Nivel pragmático: basado en la relación del lenguaje con el contexto en que es utilizado. En muchos casos no puede realizarse una interpretación literal y automatizada de los términos utilizados. En determinadas circunstancias, el sentido de las palabras que forman una frase tiene que interpretarse a un nivel superior recurriendo al contexto en que es formulada la frase.

Ejemplo: *Se moría de risa.*

En esta frase no puede interpretarse literalmente el verbo morirse si no que debe entenderse en un sentido figurado.

Otra cuestión de gran importancia es la ambigüedad provocada por la anáfora, es decir, por la presencia en la oración de pronombres y adverbios que hacen referencia a algo mencionado con anterioridad.

Ejemplo: Ella le dijo que los pusiera debajo

La interpretación de esta frase tiene diferentes incógnitas ocasionadas por la utilización de pronombres y adverbio: ¿quién habló?, ¿a quién?, ¿qué pusiera qué?, ¿debajo de dónde?. Por tanto, para otorgar un significado a esta frase debe recurrirse nuevamente al contexto en que es formulada.

Estos ejemplos muestran la complejidad del lenguaje y que su tratamiento automático no resulta fácil ni obvio. Hemos implementado nuestro pequeño módulo de análisis de lenguaje natural en castellano, pero también hemos buscado herramientas de PLN para incorporarlo en nuestro proceso de transformación y poder así comprobar si el uso de elementos más avanzados de PLN mejora el resultado completo del sistema a nivel funcional.

El primer conjunto de herramientas que se pueden utilizar son las que nos permiten reconocer los términos de la lengua, son diccionarios orientados a facilitar procesos automáticos, como por ejemplo, traducciones. WordNet es un diccionario muy popular desarrollado para la lengua inglesa, agrupa las palabras en conjuntos de sinónimos llamados sinsets, proporcionando definiciones cortas y generales, y almacenando las relaciones semánticas entre estos conjuntos de sinónimos. El propósito es doble: por un lado producir una combinación de diccionario y tesauro cuyo uso es más intuitivo, y por otro lado, favorecer el automatismo del análisis automático de textos. La base de datos y las herramientas se han liberado bajo una licencia BSD y pueden ser descargadas y usadas libremente. A partir de WordNet surge EuroWordNet que es una base de datos multilingüe con WordNets para varios idiomas europeos (Holandés, Italiano, Español, Alemán, Francés, Checo y Estonio). Cada idioma diseña su propia WordNet estructurándola de la misma forma que el WordNet de Princeton. Pero estos diccionarios no son suficientes en nuestro trabajo. Los analizadores morfológicos son mucho más interesantes, ya que, además de reconocer que el término pertenece a la lengua nos ayudan a determinar la forma, clase o categoría gramatical de cada palabra de una oración. En castellano tenemos varias herramientas, pero la elección, en nuestro caso, requería cubrir dos necesidades:

1. Que la herramienta fuese gratuita.
2. Conectividad con nuestros procesos.

Hemos evaluado varias herramientas, algunas muy completas, pero demasiado complicadas, lo cual dificultaba la conectividad. Finalmente SVMTool nos ofrece una integración con nuestro sistema muy sencillo. Se trata de comprobar si este elemento en nuestro sistema mejora la funcionalidad del mismo, es por esto que no nos hemos preocupado del rendimiento computacional. El producto puede ser descargado he instalado en un servidor local con lo cual el rendimiento computacional mejorará, pero nosotros perseguimos un objetivo distinto, queremos saber si a medida que introducimos elementos de PLN el proceso de transformación de web a web semántico mejora. Este es el objetivo de nuestro experimento. De

manera, que la integración se realiza directamente con el servicio web on line.

Conociendo la formación de las palabras se puede abordar el análisis morfológico, que nos indica qué tipo de palabra es cada término, pero esto, aunque es sencillo en esperanto, es muy complicado en cualquier lenguaje natural. Hay muchísimos trabajos en esta línea y como resultado de ellos hay implementaciones que proporcionan esta información. Para poder comprender algo de una frase hace falta además realizar un análisis sintáctico. Dicho análisis va a proporcionar la función que cada término realiza en la oración. Trabajaremos solo con oraciones simples, que aunque desde un punto de vista lingüístico una “oración simple” es una estructura sencilla de analizar y comprender, desde el punto de vista computacional se convierte en una tarea tremendamente complicada. Supongamos la frase siguiente: “Pedro escribe una carta a sus tíos”. Para entender lo que dice esta frase (una persona lo hace instantáneamente), lo primero es realizar un análisis morfológico, esto es, identificar en cada término; lexemas, morfemas, etc… esto sirve para conocer los elementos que forman la frase; verbo, adjetivos, pronombres, etc.. Una vez concluido el análisis morfológico deberemos realizar un análisis sintáctico. Ahora estamos en un nivel de abstracción superior y tratamos de comprender cómo se relacionan los términos de la oración, es decir, en este proceso se localiza el sujeto, el verbo, el complemento directo, etc… Por supuesto, en nuestro prototipo ni siquiera intentaremos abordar niveles de mayor complejidad como son el semántico y pragmático.

Una oración simple está compuesta a su vez por dos sintagmas (Figura 3.19), estos son: el sintagma nominal y el sintagma verbal.

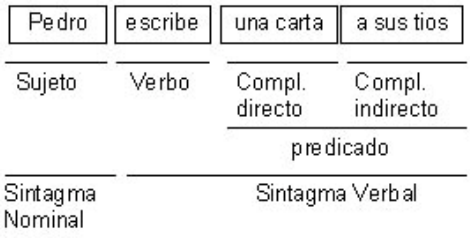


Figura 3. 19: Ejemplo de analisis sintáctico

El Sintagma Nominal se compone normalmente de una o dos palabras, de las cuales una de ellas es el sujeto (núcleo del sintagma nominal). El sujeto será un sustantivo o un pronombre, de manera que para reconocerlo, será suficiente la identificación morfológica que se realizó previamente. Por otro lado, el sintagma verbal se compone de núcleo (verbo) y predicado. El predicado es a su vez una estructura compleja que se compone de complementos. De los cuales vamos a tratar los más importantes: estos son el complemento directo (quien recibe directamente la acción del verbo) y el complemento indirecto (quien recibe indirectamente la acción del verbo).

Para ser operativos en esta tesis y no quedarnos en el plano teórico, hemos implementado dos métodos de procesado de lenguaje natural en castellano.

PROCESADO ELEMENTAL: Se trata de la forma más básica. Sin aplicar técnicas de IA. Consiste en identificar elementos clave, como son el verbo, el sujeto y el complemento directo de una oración simple en castellano. Para ello, el núcleo de análisis, es la clase “extraerFrases” (ver apéndice B para mayor detalle), capaz de extraer dichos elementos directamente del texto. El procesamiento realiza lo siguiente:

1. Se carga en memoria un modelo de estructura de oración. En este modelo se indican partículas que con frecuencia anuncian un complemento indirecto, un sujeto, etc…. En nuestro prototipo solo tratamos frases afirmativas en las que el verbo que interviene es el verbo “ser” en tercera persona del presente de indicativo.
2. Se analiza término a término la oración teniendo en cuenta la posición del mismo en base a unas estructuras modeladas. Cuando se obtienen términos candidatos a ser identificados como verbo, sujeto, CD y/o CI se vuelven a procesar con un método especial llamado “esPosibleEsteTermino()” de la clase “extraerFrases” que devuelve un valor *false* si encuentra alguna incompatibilidad con la función asignada.

Este procesado de lenguaje puede confundir las funciones que asigna a los términos incluso si la frase es demasiado compleja, puede no extraer información alguna.

PLN BASADO EN SVMTool: En este caso usamos un elemento externo al que invocamos para que analice una frase y devuelva el resultado. Para ello hemos construido una clase que comunica por el protocolo http con el servidor de SVMTools llamada de la igual forma (ver apéndice B).

La aplicación SVMT [Giménez et al; 2004], es un etiquetador morfosintáctico, efectivo y eficiente, disponible para su uso público de forma gratuita. SVMT se basa en Support Vector Machines y ofrece un estupendo equilibrio entre varias propiedades, las cuáles lo hacen realmente apropiado para aplicaciones en el campo del procesamiento del lenguaje natural. Siguiendo con nuestra frase de ejemplo “Pedro escribe una carta a sus tíos”, nuestro código lanza una petición http del estilo de:

http://www.lsi.upc.edu/~nlp/SVMTool/demo.php?frase=Pedro+escribe+una+carta+a+sus+tio

s&llengua=es&strategy=0&direction=LR&weightf=0

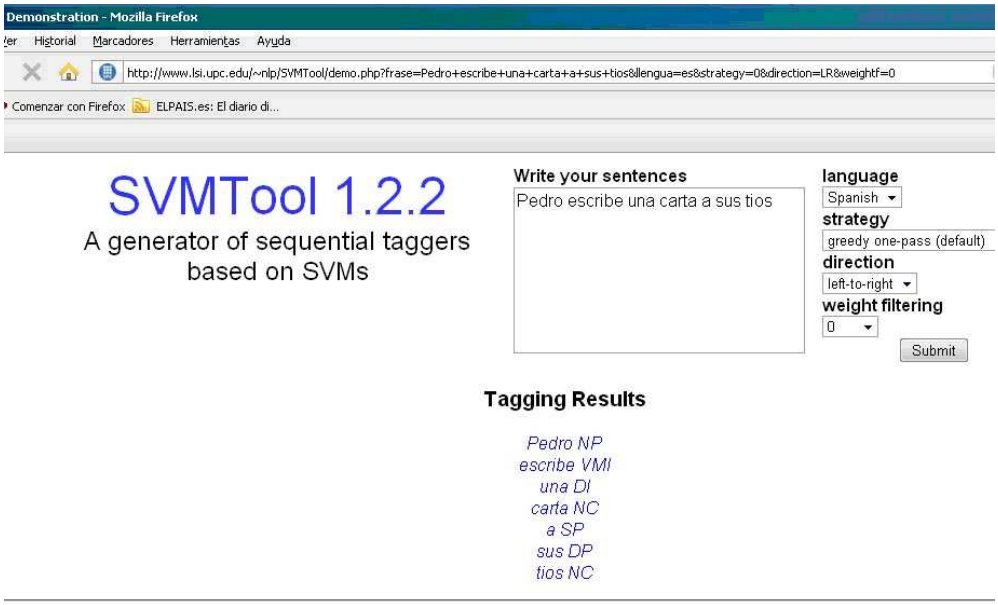


Figura 3. 20: Respuesta de SVMT

De manera, que por cada frase que se desea analizar (ver Figura 3.20) hay que filtar los tags HTML para extraer el análisis que interesa, obteniendo la información de la Tabla 3.8 y Figura 3.21:

Palabra	Tipo
Pedro	NP
escribe	VMI
una	DI
carta	NC
a	SP
sus	DP
tios	NC

Tabla 3. 8: Información obtenida de SVMTools

Con este análisis se conoce cada término, pero no se llega a conocer la función

que cada uno realiza en la frase. Por lo tanto, partiendo de la información que nos aporta el SVMT nuestro código intenta asignar algunos valores sintácticos como son verbo, sujeto y complementos (directo e indirecto). Para ello cargamos en memoria un modelo de estructura de oración simple. Este modelo es el mismo que utilizabamos en el procesado elemental.

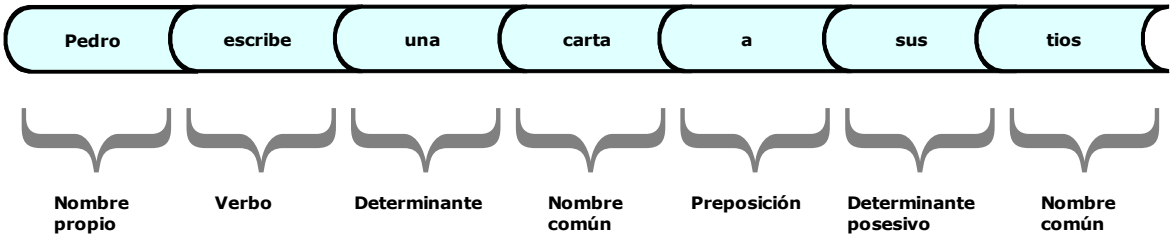


Figura 3. 21: Ejemplo de análisis

En la tesis no abordamos frases demasiado complicadas, así que solo tratamos unas pocas etiquetas (ver Tabla 3.9) de todas las que proporciona la herramienta. Fundamentalmente las etiquetas de verbos, nombres, determinantes y preposiciones.

ADJETIVOS	**PRONOMBRES**
AO → Adjetivo Ordinal	**PO** → Promonbre
AQ → Adjetive Cuantificador	**PD** → Promonbre Demostrativo
CONJUNCIONES	**PE** → Promonbre Exclamativo
CC → Conjunción Coordinativa	**PI** → Promonbre Indefinido
CS → Conjunción Subordinada	**PN** → Promonbre Numeral
ADVERBIOS	**PP** → Promonbre Personal
RG → Adverbio General	**PR** → Promonbre Relativo
RN → Adverbio Negativo	**PT** → Promonbre Interrogativo
PREPOSICIÓN	**PX** → Promonbre Posesivo
SP → Preposición	**VERBOS**
FECHAS	**VAG** → Verbo Auxiliar Gerundio
W → Fecha	**VAI** → Verbo Auxiliar Indicativo
DESCONOCIDO	**VAM** → Verbo Auxiliar Imperativo
X → Descocido	**VAN** → Verbo Auxiliar Infinitivo

DETERMINANTES	**VAP** → Verbo Auxiliar Participio
DA → Determinante Articulo	**VAS** → Verbo Auxiliar Subjuntivo
DD → Determinante Demostrativo	**VMG** → Verbo Principal Gerundio
DE → Determinante Exclamativo	**VMI** → Verbo Principal Indicativo
DI → Determinante Indefinido	**VMM** → Verbo Principal Imperativo
DN → Determinante Numeral	**VNM** → Verbo Principal Infinitivo
DP → Determinante Posesivo	**VMP** → Verbo Principal Participio
DI → Determinante interrogativo	**VMS** → Verbo Principal Subjuntivo
INTERJECCIÓN	**VSG** → Verbo Semi-Auxiliar Gerundio
I → Interjección	**VSI** → Verbo Semi-Auxiliar Indicativo
NOMBRE	**VSM** → Verbo Semi-Auxiliar Imperativo
NC → Nombre Común	**VSN** → Verbo Semi-Auxiliar Infinitivo
NP → Nombre Propio	**VSP** → Verbo Semi-Auxiliar Participio
ABREVIATURA	**VSS** → Verbo Semi-Auxiliar Subjuntivo
Y → Abreviatura	
NÚMEROS	
Z → Figuras	
ZM → Normal	
ZP → Porcentaje	

Tabla 3. 9: Etiquetas herramienta SVMTools (2ª parte)

Conviene aclarar que esta herramienta de procesado morfológico no es infalible, de hecho hemos detectado problemas con frases sencillas. Por ejemplo, no es lo mismo escribir “El Pastor Alemán es un perro.” que escribir “El pastor alemán es un perro.”, ya que en el primer caso “Pastor Alemán” se identifica como nombres propios y en el segundo, el sistema considera que “alemán” es un VSI. En consecuencia, la calidad de las *vistas semánticas* puede verse afectada. Por ejemplo, si en un proceso de PLN se extrae la frase “El pastor alemán es un perro poderoso.” y se procesa con SVMTool, obtendremos el siguiente análisis incorrecto:

El (DA) pastor (NC) alemán (VMI) es (VSI) un (DI) perro (NC) poderoso (AQ)

En base a este resultado, el siguiente proceso intentará asignar roles sintácticos, y en este caso se obtendrá algo también erróneo:

sujeto: pastor

verbo: alemán es

cd: perro

Sin embargo, si la frase es “El Pastor Alemán es un perro poderoso.”, el análisis morfológico es correcto:

El (DA) Pastor (NP) Alemán (NP) es (VSI) un (DI) perro (NC) poderoso (AQ)

Para este análisis morfológico, somos también capaces de realizar un análisis sintactico, orientado a la ontología, correcto:

sujeto: Pastor Alemán

verbo: es

cd: perro

Notesé que otras partículas que forman parte del sujeto, como “El” es filtrado, al igual que ocurre con la partícula “un” que forma parte del complemento directo. La razón es para realizar adecuadamente la anotación OWL en la *vista semántica*.

Después de realizar varias pruebas con SVMTools e intentar procesar un sitio web completo nos hemos encontrado con la dificultad de ser bloqueados, es decir, se nos ha denegado el servicio. Por lo tanto, finalmente hemos usado nuestra implementación.

3.2.3. Representación formal en XML

El proceso de transformación utiliza el fichero de configuración “oracion_simple.xml” que define la gramática de la oración simple y el resultado en lenguaje natural expresado formalmente tiene el siguiente aspecto:

<oracion sujeto="Whippet" verbo="es" CD="perro" CI=" " origen="El Whippet es un perro de distancias cortas "/>

El elemento <oración> tiene cinco atributos, siendo el atributo [origen] la frase original extraida de forma automática. El resto de atributos representan las funciones sintácticas básicas de la frase; sujeto, verbo, complemento directo y complemento indirecto. Una vez ejecutada la opción de “Extraer lenguaje natural”, podremos obtener informes de la transformación de cada fichero. Como puede verse (Figura 3.22) la frase “El Vallhund Sueco es un perro no muy alto” ha sido analizada, de forma automática, identificando “Vallhund Sueco” como sujeto, “es” como verbo y “perro” como complemento directo.

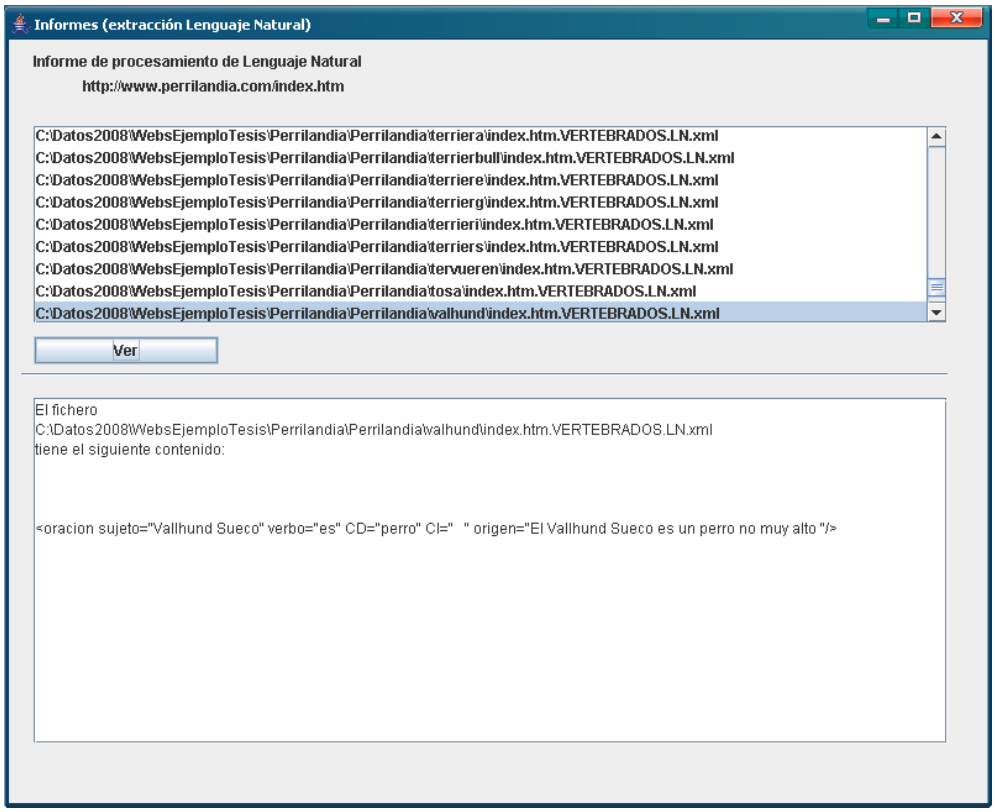


Figura 3. 22: Ejemplo de extracción

Pero puede darse casos (Figura 3.23) en que se produzcan fallos al asignar funciones sintácticas o no se logre asignarlas.

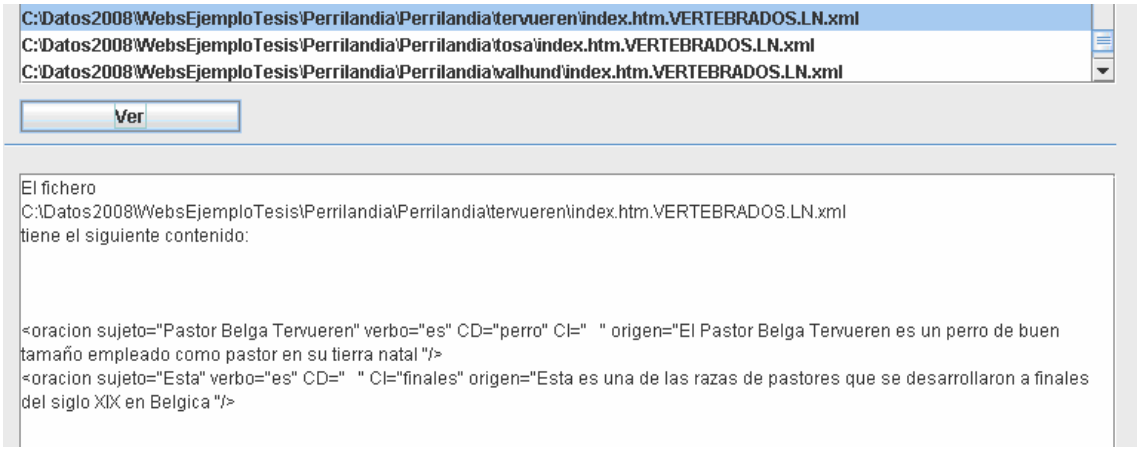


Figura 3. 23: Ejemplo de extracción 2

Esta extracción de lenguaje natural (Figura 3.23) representada en formato XML esta formada por dos frases, la primera perfectamente identificada. En la segunda frase, el proceso de extracción ha sido incorrecto, pero cuando se genere la *vista semántica* se filtrarán las frases incorrectas e incoherentes con la ontología, como es el caso de la captura de pantalla. De manera, que solo las frases que tengan relación con la ontología serán instanciadas.

3.3. Interpretación: Generación de vistas semánticas.

La *interpretación* es la última etapa del proceso de transformación de un sitio web al nuevo paradigma. Esta etapa es la encargada de realizar la anotación semántica de acuerdo al RO, al proceso de *identificación*, y al proceso de *extracción* (ver Figura 3.1).

Como se explico en el capítulo anterior, las *vistas semánticas* son una solución al problema de la anotación, problema que identifica la Agenda Estratégica de Investigación (AEI) propuesta por la plataforma INES [Plataforma Tecnológica Española de Software y Servicios; 2008] en abril del 2008. La AEI considera que, en el campo de la Web Semántica, el primer paso es contar con mecanismos de estructuración y anotación de contenidos lo que permitirá el desarrollo de herramientas de búsqueda y recuperación de información precisas. La anotación, ya sea, manual, automática o semi-automática es necesaria para cualquier proceso posterior que requiera un enfoque semántico. Pero lo interesante es disponer de herramientas que realicen esta anotación en OWL, por los motivos que se han explicado con anterioridad.

3.3.1. Alcance

Una vez que se ha seleccionado la ontología u ontologías y se ha extraído el lenguaje natural expresándolo en XML podemos expresar parte de este contenido en OWL DL, generando así una vista semántica.

Nuestro propósito es que los actores “software” puedan “entender algo” de este contenido. De forma que nos concentraremos en extraer “algo de contenido” en lenguaje natural, procesándolo con una ontología adecuada para poder expresarlo en OWL DL.

Aclarado este alcance, conviene recordar nuevamente que el proceso de “entender algo del contenido” utiliza un enfoque de anotación automática colaborativa que se realizará con cada una de las ontologías identificadas para una página web concreta, es decir, de una única página web se obtendrán anotaciones OWL DL en relación al contenido para una o varias ontologías, generando la vista semántica cuyo orden será igual al número de interpretaciones.

3.3.2. Formalización de una Vista Semántica de primer orden.

Anteriormente definíamos el concepto de *vistas semánticas* (VS) como la estructura que proporciona diferentes visiones de acuerdo a diferentes ontologías, es decir, proporciona diferentes interpretaciones de una misma página web. Recordamos que una interpretación (I) de una página web es un conjunto de instancias de acuerdo a una determinada ontología. De manera que una vista semántica de orden “n” es un conjunto de “n” interpretaciones (cada interpretación puede tener un número de instancias diferente).

Con la herramienta que hemos implementado, como ejemplo, hemos generado automáticamente una interpretación OWL DL que consiste en una instancia “*Boxer*” de la clase perro. Se trata del caso más sencillo, aquel en que la vista semántica de una página web es de primer orden, de manera, que se cumple que la vista solo tiene una interpretación de acuerdo a la ontologia de vertebrados, es decir, el caso de ejemplo es la particularización de la expresión (2.1) para n=1.

$$VS^1 = \{ I_1 \}$$

Esta información se ha guardado en un fichero con un URI único como es:

http://www.criado.info/owl/stand_boxer_3F972AA91EEE3451B64A67FCDA136ACB.owl

El contenido de este fichero puede verse en la figura 3.24

```
<rdf:RDF
    xmlns:j.0="http://www.criado.info/owl/vertebrados_es.owl#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://www.criado.info/owl/stand_boxer_3F972AA91EEE3451B64A67FCDA136ACB.owl#"
  xml:base="http://www.criado.info/owl/stand_boxer_3F972AA91EEE3451B64A67FCDA136ACB.owl">
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://www.criado.info/owl/vertebrados_es.owl#"/>
  </owl:Ontology>
   <j.0:perros rdf:ID="Boxer"/>
   <owl:AllDifferent>
     <owl:distinctMembers rdf:parseType="Collection">
       <perros rdf:about="#Boxer"/>
     </owl:distinctMembers>
   </owl:AllDifferent>
 </rdf:RDF>

<!-- Creado por [html2ws]  http://www.luis.criado.org -->
```

Figura 3. 24: Anotación automática

Como puede apreciarse, en el navegador OWL desarrollado por el grupo MINDSWAP de la Universidad de Maryland, la instancia que hemos generado (ver Figura 3.25), invoca a la ontología madre, en la URI http://www.criado.info/owl/vertebrados_es.owl#,

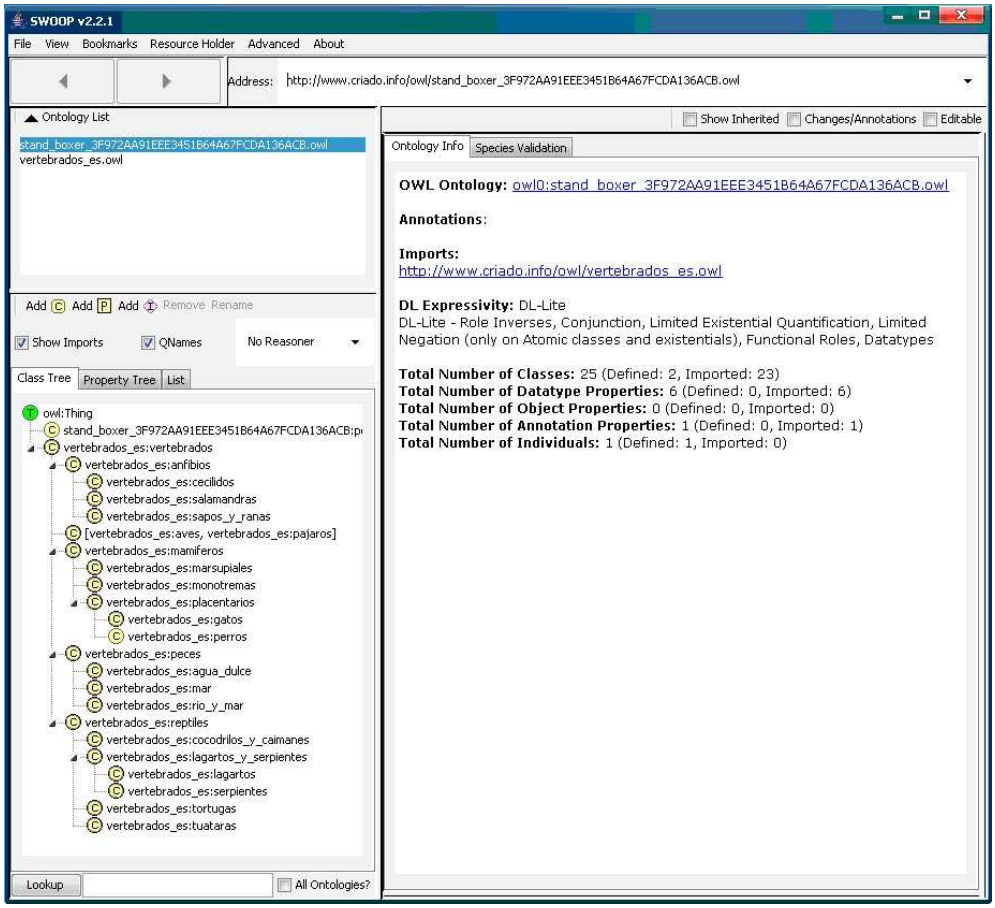


Figura 3. 25: Instancia que invoca a la ontología madre, visor SWOOP

3.3.3. Generación de un sitio web semántico mediante la herramienta sw2sws.

Una vez diseñado un sitio web completo habrá que comprobar por cada página web con que ontologías se relaciona (proceso de identificación), este proceso es posible realizarlo automáticamente con la herramienta que hemos diseñado, tal y como se explicó en “3.1. Identificación Ontológica”. La siguiente fase, también automatizada, consiste en la extracción de lenguaje natural y su representación formal en XML (3.2. Extracción de información). Superadas las etapas de identificación y extracción se puede abordar la etapa de Interpretación, que tiene por objetivo generar las *vistas semánticas* de cualquier orden.

La Figura 3.26 muestra cómo con nuestra herramienta automática se obtienen todas las *vistas semánticas*. Observese como “Keeshond” ha sido instanciado como un tipo de perro.

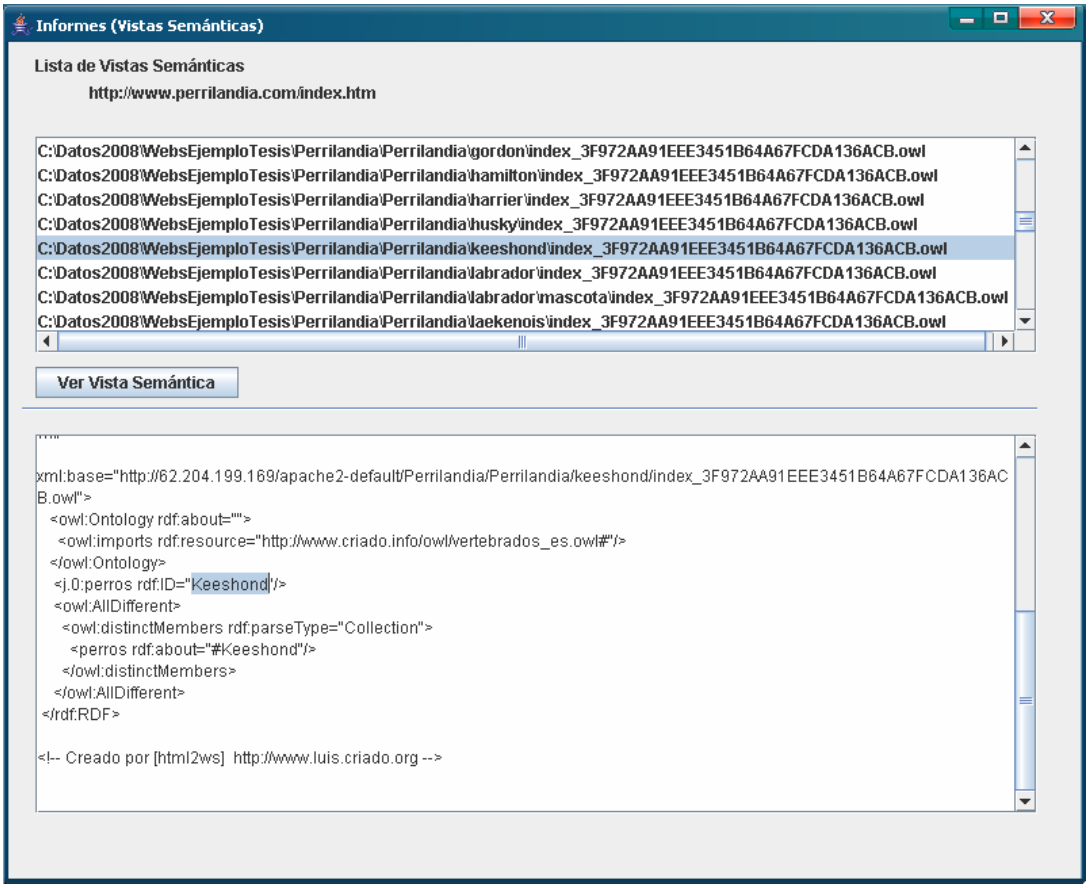


Figura 3. 26: Herramienta html2ws - Vistas Semánticas

3.3.4. Mantenimiento de un sitio web semántico.

El contenido de las páginas web frecuentemente se cambia, de manera que el proceso de transformación de sitios web a sitios web semánticos debe realizarse cada vez que se modifique una página web. Por lo tanto, una herramienta profesional de transformación requiere añadir una funcionalidad de sincronización de ficheros que forman las *vistas semánticas*.

Aunque esta función no la hemos implementado, sí vamos a indicar cómo hacerlo. Obsérvese que en los ejemplos anteriores, el nombre de cada Interpretación que forma la *vista semántica* está formada por el nombre de la página web original seguido de un guión y de una cadena hexadecimal de 32 caracteres. Este valor hexadecimal es el resultado de aplicar el algoritmo MD5 al contenido del archivo original, con lo que se obtiene un identificador único del contenido. De manera, que si el contenido cambia, el MD5 cambia y por lo tanto, el proceso de mantenimiento que se ocupa de la sincronización de contenido entre página web y *vista semántica* puede detectarlo. Lo que acabamos de indicar es una posible solución para el problema de “sincronización de páginas semánticas”, que proponemos como línea de investigación futura en el capítulo 5.

Capítulo 4: *El Buscador Semántico basado en vistas semánticas* **.**

En el capítulo anterior explicamos la herramienta de transformación “sw2sws”. El resultado de ésta es la generación automática de información semántica a partir de un sitio web ordinario que es el próposito de nuestra tesis, pero nos hemos visto en la necesidad de avanzar un poco más para mostrar cómo explotar esta información que ya tiene semántica.

Este capítulo se dedica a utilizar la información semántica generada con “sw2sws”, dicha información semántica se ha expresado en OWL DL organizada en *vistas semánticas* que permite definir el concepto de página semántica (PS), tal y como se especificó en el capítulo segundo, que recordamos es una estructura que debe representar la información de tal forma que tenga sentido tanto para personas como para sistemas informáticos. Pero con un requisito importante, debe ser compatible tanto para un buscador ordinario (actual) como para un buscador semántico (futuro), ya que esto garantiza el proceso de migración al nuevo paradigma de la Web Semántica. La expresión (2.2) indica que una PS es un conjunto formado por dos elementos; uno sencillo, la página web, y el otro, compuesto, ya que es una *vista semántica* de un determinado orden, el mismo que tendrá la página semántica.

El Buscador Semántico basado en *páginas semánticas* (PS), buscará la información haciendo uso de sus *vistas semánticas* (*VS)* y sin embargo, el usuario recibirá los enlaces, como siempre, hacia las páginas web (PW).

4.1. Criterios de filtrado de información en los buscadores no semánticos.

Los motores de los buscadores usan distintos algoritmos para proponer una lista de enlaces que presumiblemente son lo que el usuario necesita. Estos algoritmos no son de dominio público, ya que constituye un punto distintivo entre ellos. Sin embargo, hay cuatro criterios muy comunes:

1. La frecuencia con la que aparecen las palabras extraídas de la pregunta en los documentos encontrados. Cuantas más veces esté presente en el texto una palabra de la pregunta, más significativo es el documento en cuestión.

2. Las palabras asociadas a un diccionario de sinónimos. Por ejemplo, al buscar páginas relacionadas con veleros estaría bien ampliar la pregunta a barcos. Tales términos pueden ponderarse en el momento de la petición para favorecer el término exacto (en este caso veleros), asignándole un peso específico relativamente más importante que el que afecta al término asociado, es decir, al encontrado en el diccionario (en este caso barcos). Este asunto puede resolverse mejor con la Web Semántica, estableciendo sinónimos en las propias ontologías.

3. La proximidad de las palabras buscadas en el documento. Cuanto más cerca estén entre sí las palabras buscadas, más significativas son y, por tanto, más pertinente es el documento. También aquí se puede tener en cuenta los términos asociados a través de un diccionario, siempre ponderándolos. La proximidad es un buen criterio cuando no se tiene otra forma de “entender”. Lógicamente con la Web Semántica este criterio carece de interés.

4. Los índices de los buscadores están poblados de sitios que podrían catalogarse como carentes de interés general para los usuarios. En su afán por ofrecer los mejores

resultados, estos buscadores intentan indexar solamente los sitios que tengan mayor interés potencial. Por esta razón, algunos motores de búsqueda utilizan entre sus criterios de relevancia uno llamado link popularity (o popularidad), que se basa en la cantidad de enlaces que se refieren a una página web por otros sitios webs. Se sabe, que el conocido buscador "Google" utiliza un algoritmo denominado "PageRank" que es una extensión de "link popularity". Dicho algoritmo fue patentado en Estados Unidos en enero de 1998, por Larry Page (uno de los fundadores de Google). El título original es 'Method for node ranking in a linked database'. Pero esto mejorará mucho al incluir ontologías.

El gran problema es que, aunque los buscadores aplican muchísimos criterios para seleccionar adecuadamente un conjunto de páginas o sitios web, el acceso a la información real que el usuario necesita requiere mucho tiempo de dedicación por parte del mismo. Las expectativas futuras no son optimistas, ya que las implementaciones más modernas, en fase experimental, del más conocido y popular buscador de hoy en día, como es "Google" [24], parece que no puede solventar este inconveniente.

En definitiva, entre las dificultades más importantes que un usuario encuentra cuando busca información en Internet se pueden citar las siguientes [Ferreira et al; 1998]

1. Los usuarios deben invertir demasiado tiempo en verificar la exactitud de la información encontrada por los buscadores.

2. Los usuarios desechan generalmente la mayoría de información relacionada con el tema requerido, principalmente debido a que corresponde a una cantidad tal, que es de difícil manejo, por lo que finalmente el interés solo se encuentra en mantener y revisar las primeras referencias.

3. Los usuarios sólo tienen la posibilidad de buscar o recuperar información haciendo uso de palabras claves.

4. La búsqueda de la información solicitada no toma en cuenta aspectos lingüísticos relacionados con el tema, propios de la naturaleza de los documentos textuales, lo cual puede ayudar a entregar una mayor precisión sobre éste.

5. Hay poca interacción o retroalimentación en el proceso de búsqueda/filtrado de

información. Las decisiones dependen directamente de las consideraciones del sistema.

4.2. ¿Cómo se imagina desde el Consorcio World Wide Web los Buscadores Semánticos?

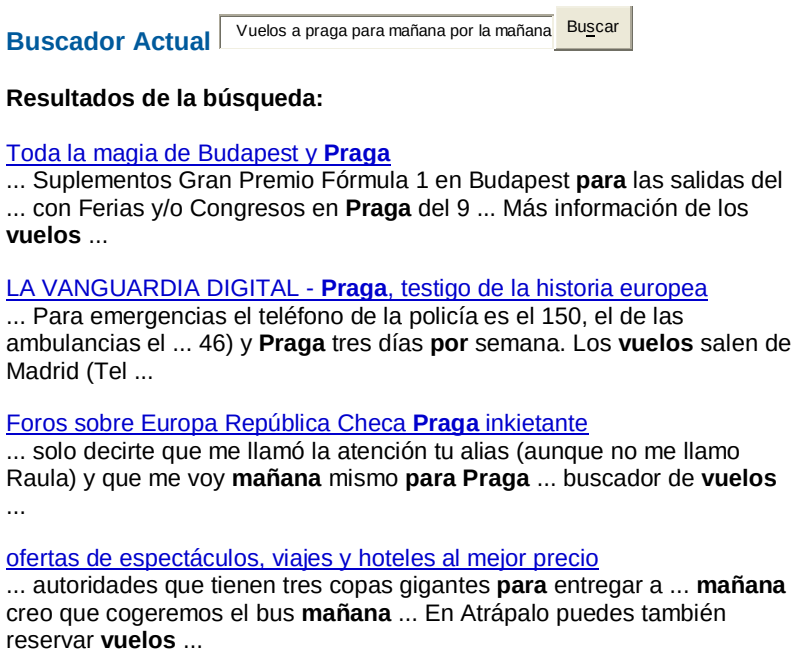


Figura 4. 1: Buscador Actual

Supongamos un usuario, en su intento, por ejemplo (este es el ejemplo del Consorcio de la W3C), por encontrar todos los vuelos a Praga para mañana por la mañana, qué resultados obtendría con un buscador actual. Probablemente (ver Figura 4.1) [23] resultados inexactos, cualquier buscador ofrecería información variada sobre Praga, pero que no es lo que realmente el usuario buscaba. El paso siguiente por parte del usuario es realizar una búsqueda manual entre esas opciones que aparecen, con la consiguiente dificultad y pérdida de tiempo. Con la incorporación de semántica a la Web los resultados de la búsqueda serían exactos. La Figura 4.2 muestra los resultados obtenidos a través de un buscador semántico. Estos resultados ofrecen al usuario la información exacta que estaba buscando.

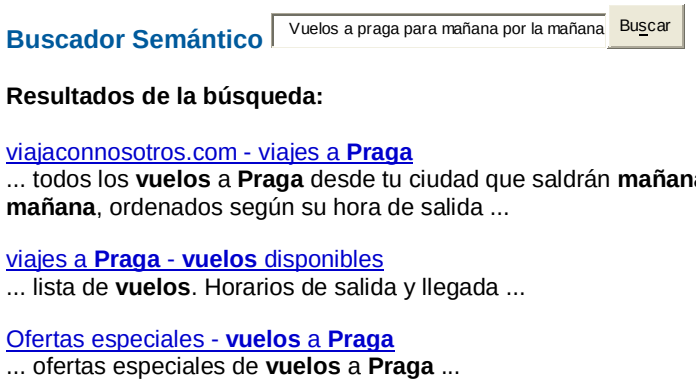


Figura 4. 2: Buscador semántico

La idea de una buscador semántico es muy atractiva y las posibilidades como base tecnológica y económica para futuras aplicaciones son espectaculares, tal vez por esto tanta gente une esfuerzos en materializarla. Como puede verse en este ejemplo recogido de la Oficina Española del Consorcio World Wide Web (W3C) (ver Figura 4.2) [23], un buscador semántico de esta naturaleza implica para su funcionamiento dos cosas: la existencia de ontologías y cierto nivel de compresión del lenguaje natural, por mucho que se negara en el pasado, el avance en las técnicas y métodos del procesamiento del lenguaje natural actuará positivamente en el desarrollo de la Web Semántica, sencillamente por el hecho de que el buscador semántico, como mínimo, "tiene que saber de qué le están hablando", es decir, cuál es el contexto, o en términos tecnológicos, cuál es el dominio de conocimiento sobre el que actuar, que modelaremos con una ontología o familia de ellas.

4.3. Situación de los buscadores semánticos a principios del 2009

Hemos identificado buscadores semánticos verticales, que son buscadores especializados en una actividad concreta, como es el caso del buscador del Ayuntamiento de Zaragoza desarrollado por iSOCO (2006) [42] o el buscador semántico de la School of Electronic and Computer Science de la Universidad de Southampton [22], que se lanzó en febrero del 2005. Pero también hay prototipos de buscadores semánticos genéricos o de propósito general. Aunque los detalles de funcionamiento de cada uno de ellos se lleva con un gran secretismo, si es posible conocer los enfoques generales de funcionamiento. En este epígrafe exploraremos las características de algunos de los más importantes.

Casi todos los buscadores, que se presentan como buscadores semánticos o de tercera generación no son más que buscadores de ontologías, es decir, buscan ficheros en RDF, microformatos y algunos en OWL, pero son herramientas inadecuadas para usuarios finales. Estos buscadores son útiles para expertos y probablemente, para una vez procesada toda la información, poder ofrecer servicios nuevos, tal y como argumenta Sindice [48] en su web.

Hay muy pocos buscadores semánticos de propósito general que ya funcionen y que esten orientados a usuarios finales, tan pocos, que después de analizar varios de ellos que se autodefinen como buscadores semánticos, solo los casos de Hakia [51], True Knowledge [50] y Powerset [49] nos parecen cercanos a la idea de buscador semántico o de tercera generación de propósito general como comentaremos en el apartado 4.3.2. De manera, que distinguimos dos categorías de buscadores semánticos: los no orientados a usuarios y los que si estan orientados a usuarios.

4.3.1 Buscadores semánticos no orientados a usuarios

Cuando se comienza a estudiar el estado del arte de los buscadores semánticos, al principio da la impresión de que son muchos los que han desarrollado motores orientados al nuevo paradigma. Sin embargo, haciendo un estudio detallado se deduce que son realmente escasos los buscadores semánticos, entendiendo por buscador semántico aquellos que utilizan técnicas de Web Semántica y lenguaje natural en sus motores internos para ofrecer al usuario los enlaces que éste desea. Lo que si se encuentra en gran número son buscadores de información ontológica no orientados a usuarios, estos buscadores son interesantes para expertos pero no son útiles para usuarios finales. En esta categoría de buscadores incluimos a la mayoría; como SWSE [47], Swoogle [41], Watson [52], Falcons [53] y Sindice.

Swoogle está orientado a proporcionar documentos en RDF, OWL, etc. Pero no está orientado a proporcionar documentos HTML que son los documentos que requieren los usuarios finales, es un buscador de ontologías. Ha sido desarrollado por la Universidad de Maryland, Baltimore County (UMBC), en concreto por el grupo de investigación del departamento de Ciencias de Computación e Ingenieria Electrica (CSEE), conocido como UMBC eBiquity. La UMBC eBiquity [41] se compone de profesores y estudiantes, y está financiado por empresas y administraciones, que incluyen la NSF, DARPA, la NASA, el Departamento de Defensa de EEUU, HPL, IBM y Fujitsu.

Según SWSE (Semantic Web Search Engine), ya existe un gran volumen de datos que se ajusta a la propuesta de las normas de la Web Semántica (por ejemplo, hay información basada en vocabularios RDF). La gente puede publicar las descripciones sobre sí mismo utilizando FOAF, los proveedores de noticias pueden utilizar RSS (RDF Site Summary) incluso las imágenes están siendo anotadas utilizando diversos vocabularios RDF. La cantidad de datos formalmente expresados disponibles está creciendo de manera constante. Este buscador permite un medio para encontrar esta información. Lo mismo ocurre con Sindice, buscador beta, que indexa archivos RDF y microformatos, que también se basa en la idea de que hay unos 10 millones de piezas semánticas, entre RDF y microformatos, distribuidos en 100 millones de páginas web. De manera que Sindice se ocupa de indexar toda esta información con el objetivo de utilizar estos datos para ofrecer nuevos servicios. Sindice realiza un razonamiento sofisticado que aumenta la reutilización de estos datos y proporciona búsqueda de precisión sobre esta información. Al igual que Swoogle no está pensado para un usuario final. Este buscador es un proyecto de investigación en DERI y cuenta con el apoyo de la Science Foundation Ireland (SFI) en virtud de la subvención No SFI /02/CE1/I131 y de la Iniciativa OKKAM.

El Semantic Web Search desarrollado por Intellidimension Inc. El buscador semántico "Watson" implementado por Knowledge Media Institute (KMi) de la Open University (UK) o Falcons, del Institute of Web Science, Southeast University, P.R. China, son más ejemplos de buscadores semánticos no orientados a usuarios, aunque Falcons dispone de una de las visualizaciones más cómodas

4.3.2 Buscadores semánticos orientados a usuarios.

Naveganza [42] es un buscador español de tercera generación desarrollado por iSOCO, que se basa en la comprensión de las preguntas tecleadas por los usuarios y en los documentos indexados. Se trata de una línea de productos que facilita la interacción de las personas con la inmensa cantidad de información no estructurada que se da en un entorno profesional. La arquitectura del sistema distingue componentes que son independientes e interoperables, permitiendo una fácil integración con otras soluciones.

Knowledge Index (Figura 4.3) es el componente central de Naveganza, encargado de indexar y extraer los contenidos en diferentes formatos, para su posterior recuperación y presentación.

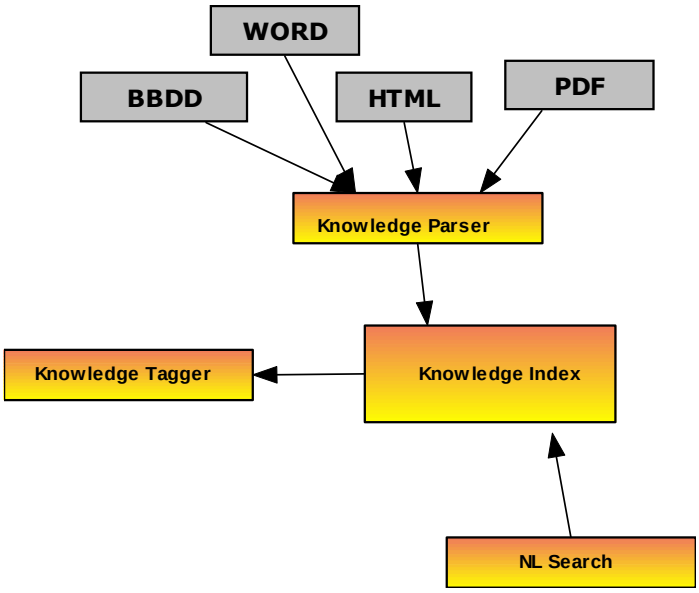


Figura 4. 3: Arquitectura de Naveganza

La herramienta para la captación de información, procedente de páginas web, en formato word, en formato pdf u otros, es el Knowledge Parser. El componente NL Search se encarga del análisis y procesamiento de lenguaje natural, unido al resto de componentes, permite realizar búsquedas en lenguaje natural, aunque también funciona con palabras claves. Finalmente el Knowledge Tagger es el módulo encargado de anotar el contenido indexado y relacionarlo con una ontología de conocimiento específico. Esta anotación y posterior procesado permite realizar una búsqueda o comparación conceptual, al establecerse un modelo de objetos relacionados. Actualmente este producto se ha implantando como solución de buscador semántico vertical.

Hakia, que comenzó desarrollando varios buscadores semánticos verticales, tiene ahora un buscador semántico de propósito general [51]. La calidad de los resultados depende simultáneamente tres criterios:

(1) fuentes dignas de confianza (verticales)

(2) información reciente

(3) relevante para la consulta

El motor de búsqueda actualmente funciona en modo beta, pero el análisis y desarrollo continúa avanzando. .

El 7 de enero de 2008, Hakia anuncio que ha recaudado 21 millones de dólares. Según el Dr. Riza Berkan, fundador y director general de Hakia, están llegando al final de la primera fase de desarrollo. Lo que da idea de la envergadura e importancia que ya tiene el proyecto de la Web Semántica. Hakia trabaja para tener un buscador semántico de propósito general, pero sus efectos, a fecha de la redacción de esta tesis, no son aún espectaculares, realmente parece un buscador corriente.

Powerset es un buscador que se fundo en el 2005 y fue comprada por Microsoft el 1 de agosto del 2008. Aunque por el momento solo soporta inglés, este buscador ofrece primero una pequeña respuesta o resumen informativo sobre la cuestión que se consulta y después ofrece enlaces.

El caso de "True Knowledge" [50], parece muy prometedor, pero en el momento de redactar esta tesis es publicidad de lo que hará, este buscador utiliza un video que muestra como funcionará y desde luego es espectacular. Lo interesante es que no solo proporciona una lista de enlaces, además proporciona respuestas a una pregunta concreta. El ejemplo del video, es muy clarificador, ante la pregunta "¿está Jennifer López soltera?" el buscador responde que "no" y además nos ofrece un enlace con información relacionada con su matrimonio.

4.3.3 Conclusión.

Parece razonable pensar que los buscadores semánticos mejoren la capacidad de compresión de las preguntas formuladas por el usuario y prorporcionen enlaces de calidad, ya que se basarán en las técnicas de Web Semántica y en PLN. Pero al no existir aún la Web Semántica, pues los sitios webs no incorporan anotaciones OWL. Los buscadores semánticos de propósito general no pueden incorporar las funcionalidades propias del nuevo paradigma Web, por lo que tienen que limitarse al PLN en el interface con el usuario. Es decir, la causa de que los buscadores semánticos que han aparecido no obtengan mejores resultados que Google y Yahoo, es debido a que no operan con anotaciones semánticas. Simplemente no existen *sitios web semánticos*

En esta tesis, se propone un procedimiento para realizar la tarea de anotación semántica

en base a la colaboración de los usuarios activos, obteniendo así sitios web semánticos, lo que permitirá la construcción de verdaderos buscadores semánticos. Sin embargo, es muy difícil que funcione la Web Semántica si no se cuida la coherencia de la anotación respecto al contenido. También conviene señalar que de nada sirve definir ontologías si no las usan un número significativo de la comunidad de Internet, es por esto, que hay que proporcionar herramientas para que los usuarios activos de Internet puedan "usar estas ontologías" y en consecuencia "puedan generar páginas semánticas" y creemos, que en este trabajo hemos logrado la primera herramienta de este tipo, dicha herramienta, basada en una estrategia de anotación automática colaborativa (capitulo 3) que deberá evolucionar, marca una estrategia que permite construir y mantener la anotaciones semánticas que indexan los buscadores semánticos, ya sean, verticales como de propósito general.

4.4. Contexto de la busqueda.

Un buscador semántico genérico o de propósito general tiene que disponer de mecanismos para resolver el problema de la selección de contexto. Es decir, se requiere que el buscador "sepa" en que dominio de conocimiento se está trabajando. Se trata de entender lo que el usuario escribe. Es por esta razón que el interface del buscador semántico con el usuario debe incorporar algún mecanismo de procesado de lenguaje natural. Esto debe ser el objetivo final, pero actualmente plantea problemas difíciles de resolver como el identificado por Nenad Stojanovic respecto a la manera de formular preguntas por parte del usuario. Según Nenad [Stojanovic; 2005] los usuarios suelen hacer preguntas muy cortas (2-3 términos) y tienden a considerar sólo los diez primeros resultados. De manera, que ante preguntas tan cortas resulta prácticamente imposible deducir con certeza el contexto sobre el cual el usuario plantea la cuestión. Dejando esta dificultad a un lado, teóricamente sí se puede enunciar las alternativas posibles de funcionamiento de un buscador semántico para establecer el contexto de trabajo, pero actualmente no tenemos una solución práctica al problema:

1. Selección manual de Dominio/Ontología

   Antes de usar el buscador semántico el usuario seleccionaría el "contexto" para que el sistema "sepa" qué ontología deberá usar. Esta estrategia es usada actualmente por Intellidimension [18] en su buscador semántico y requiere que el usuario seleccione el tipo de "formato semántico" que desea tratar; RSS, FOAF, RDF (este buscador no trata OWL). En cualquier caso, esta forma de actuar no parece acertada, ya que estamos creando dificultades al usuario final y seguramente le será díficil decidir qué ontología escoger, sobre todo cuando el número de ontologías aumente.

2. Selección automática de Dominio/Ontología

   El propio buscador semántico seleccionaría el "contexto" basándose en un análisis mediante PLN. Desde nuestro punto de vista, hay que resolver este problema para poder plantear el desarrollo de buscadores genéricos o de propósito general. En nuestro prototipo, al procesar varios sitios webs, nos hemos encontrado con información semántica de una misma temática, por lo tanto, *Vissem* funciona como un buscador vertical o especializado, de manera que no presenta el problema de la selección de contexto.

4.5. Criterios de filtrado de información en los buscadores semánticos

Lo interesante es que el buscador semántico pudiera ofrecer soluciones a todas las deficiencias actuales, que tan claramente se describe en el trabajo de Anita Ferreira y John Atkinson [Ferreira et al; 1998]. Sabemos que el empleo de ontologías aplicado en el buscador semántico resolverá los principales problemas, como son:

- Exactitud de la información encontrada, que implica que los usuarios NO INVIERTAN tiempo en verificar las ofertas de enlaces.

- Los usuarios tendrán la posibilidad de buscar o recuperar información haciendo uso de ontologías, lo que significa que el buscador tendrá en cuenta el contexto de los documentos, es decir, el dominio de conocimiento. Las ontologías aportan soluciones al problema de la cercanía semántica: sinónimos, palabras relacionadas, conceptos similares definidos en otras ontologías, etc….

Sin embargo, parece adecuado que los buscadores semánticos continuen usando algunos criterios de los buscadores tradicionales, por ejemplo, pensamos que resultará muy útil conservar el concepto de link popularity que sirvirá al buscador semántico para ordenar ofertas en igualdad de condiciones.

Si además de todo esto añadimos capacidad de interacción o retroalimentación en el proceso de búsqueda de información con la finalidad de mejorar la calidad de las ofertas, el resultado será magnífico. Esta retroalimentación se puede realizar mediante un mecanismo de

afinidad de perfil. Sería el concepto de roles de usuario adaptado a la situación semántica, que consiste en identificar la utilidad de los enlaces propuestos respecto a los intereses de los usuarios y utilizar la propia ontología para conducir el "aprendizaje" que el sistema realizaría del comportamiento de usuarios con las mismas inquietudes, conociendo así las páginas realmente útiles para los usuarios que utilizan la misma ontología. El objetivo es que el buscador "conozca", dentro de las páginas semánticas desarrolladas en base a una determinada ontología, cuales de ellas resultan más interesantes a usuarios afines sin que resulte una molestia para ellos, de forma que el buscador semántico ordene sus ofertas que tengan condición igualitaria por un criterio de interés de usuarios afines. Tal vez, se debe pensar en una estrategia mixta entre link popularity y el comportamiento del usuario ante la oferta. En cualquier caso, discutir de los criterios que utilizarán los buscadores semánticos basados en vistas semánticas es adelantar demasiado y habrá que tenerlo en cuenta cuando se disponga de sitios web semánticos.

4.6. Breve comentario sobre el modelo de datos del Buscador Semántico

El modelo de datos de un buscador semántico, basado en vistas, no requiere especificar los atributos tradicionales que hasta ahora incorporaban explícitamente en sus BBDD, como son:

1. URL de la página web
2. título de la página para mostrar en el buscador
3. descripción de la página para mostrar en el buscador

Solo se requiere incorporar tablas que operen con tripletas, en ellas irá embebida la información detallada con anterioridad. Estas tablas que operan con tripletas incluyen dos tipos de información; por un lado, las ontologías y por otro lado, la población de éstas, es decir, las instancias que se han generado en el proceso de anotación semántica.

4.7. Arquitectura teórica del Buscador Semántico basado en vistas semánticas

El buscador semántico se compone de cuatro etapas (ver Figura 4.4), estas son:

1. Análisis Lexicográfico: Permitirá al buscador extraer todos los elementos de la pregunta del usuario para, al analizarlo, tratar de optimizar su "entendimiento". Idealmente el analizador lexicográfico enviará toda la información extraída a una capa de procesamiento de lenguaje natural.

2. Selección de contexto: Esta capa permite al buscador semántico decidir con qué ontología o conjuntos de ontologías trabajar.

3. Motor Inferencial OWL-DL: Es el corazón del buscador semántico, ya que una vez establecida la ontología objeto de la consulta se realiza una búsqueda mediante SPARQL o similar en todas las páginas semánticas construidas en base a dicha ontología.

4. Oferta de Enlaces: Esta capa se ocupa de ordenar los resultados en condiciones de igualdad mediante métodos tradicionales adaptados a la Web Semántica

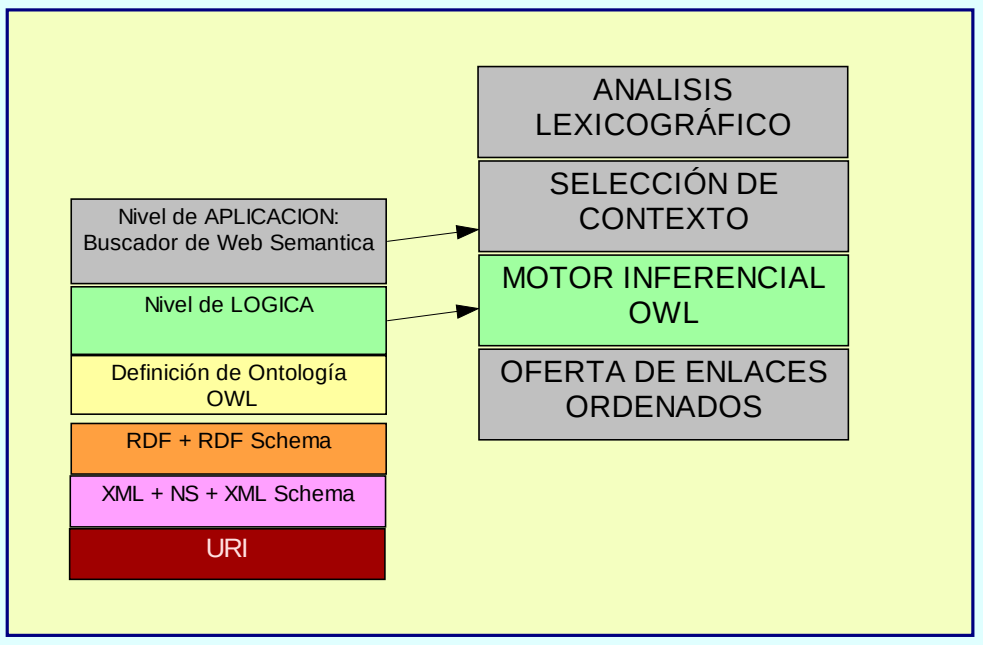


Figura 4. 4: Arquitectura del buscador semántico

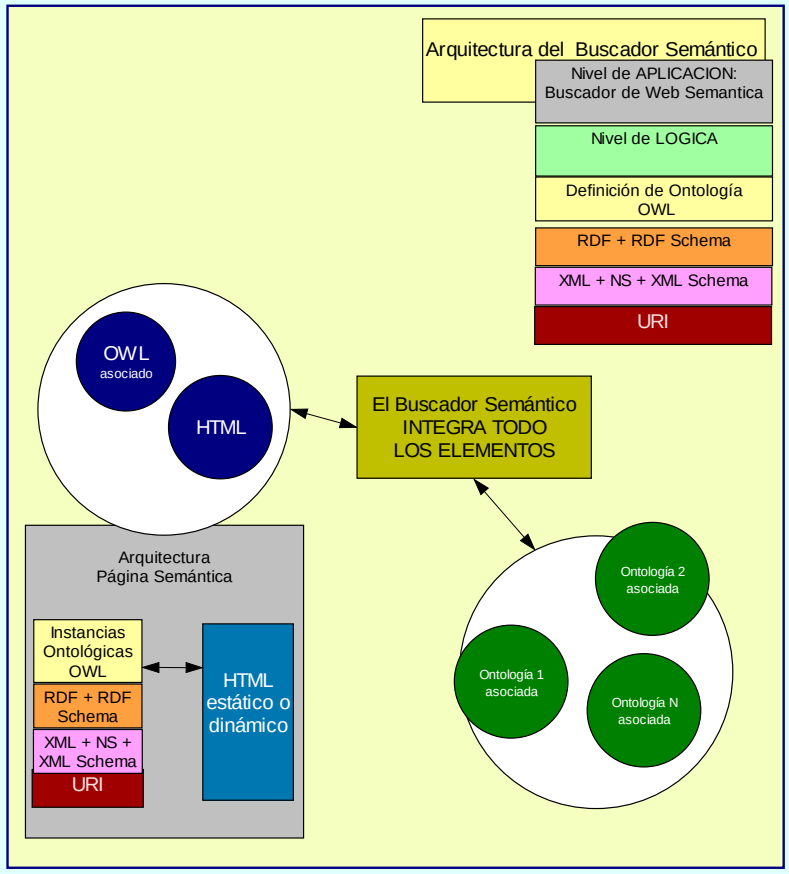


Figura 4. 5: Funcionamiento de las aplicaciones en la Web Semántica

El buscador semántico, al igual que cualquier otra aplicación semántica, debe dar coherencia a todos los elementos (ver Figura 4.5): ontologías, paginas webs, *vistas semánticas*, etc...

En el escenario general un usuario de la Web interactúa con diferentes elementos. Sin embargo, éste se relaciona directamente con el buscador y sólo cuando decide que la oferta del enlace le interesa se conectará con el hosting que contiene la página que busca. Lo mismo ocurrirá sí el usuario web accede a un buscador semántico.

4.8. Prototipo de Buscador: VISSEM

Los capítulos anteriores se han dedicado a resolver el problema de cómo transformar la Web en Web Semántica, proponiendo una forma de transformación basada en tres fases (identificación, extracción e interpretación) que por un lado garantiza la compatibilidad con los buscadores actuales y por otro proporciona información semántica que puede ser explotada por los buscadores del futuro, los buscadores semánticos que operan con anotaciones semánticas. La herramienta de transformación de un sitio web a un sitio web semántico que hemos implementado proporciona información semántica para la nueva generación de buscadores. De forma, que hemos implementado también un primer prototipo de buscador al que hemos llamado *Vissem* y que está basado en *vistas semánticas*, aunque *Vissem* soporta unas pocas ontologías, permite diferencias interesantes respecto a los buscadores actuales, tanto los tradicionales, como los autodenominados semánticos.

4.8.1. Inicializando la BBDD del buscador VISSEM.

Antes de poder utilizar el buscador semántico es indispensable que incorpore en su BBDD las tripletas que representa cada ontología. Esta utilidad de inicialización se ha implementando usando Jena. Este framework requiere construir el modelo de la ontología en memoria para después incorporarlo a la BBDD, es por ello que es poco escalable pero sirve perfectamente para volcar un fichero OWL en un modelo persistente basado en tripletas. Se recomienda ampliar la memoria de la máquina virtual java cuando se ejecute la herramienta de carga de ontologías. En el apartado A.8 del apéndice A se explica cómo configurar y compilar la utilidad para la carga de ontologías en el modelo de BBDD del buscador semántico. Esta carga

previa es necesaria antes de incorporar las vistas semánticas de un sitio web, ya que las vistas son instancias respecto a las primeras. Una vez compilado y particularizada la IP de la BBDD de su buscador semántico busque el directorio “crearOntologiaEnBBDD” en la carpeta de “distribución” y ejecute el comando siguiente:

java -Xms1024m -Xmx1024m -cp .\;.\antlr-2.7.5.jar;.\arq.jar;.\commons-logging.jar;.\concurrent.jar;.\icu4j_3_4.jar;.\iri.jar;.\jakarta-oro-2.0.8.jar;.\jena.jar;.\jenatest.jar;.\json.jar;.\junit.jar;.\log4j-1.2.12.jar;.\stax-api-1.0.jar;.\wstx-asl-2.8.jar;.\xercesImpl.jar;.\xml-apis.jar;.\ontologiaSPARQL.jar;.\terica.jar;.\classes12_g.zip tesis.owl.jenaSPARQL

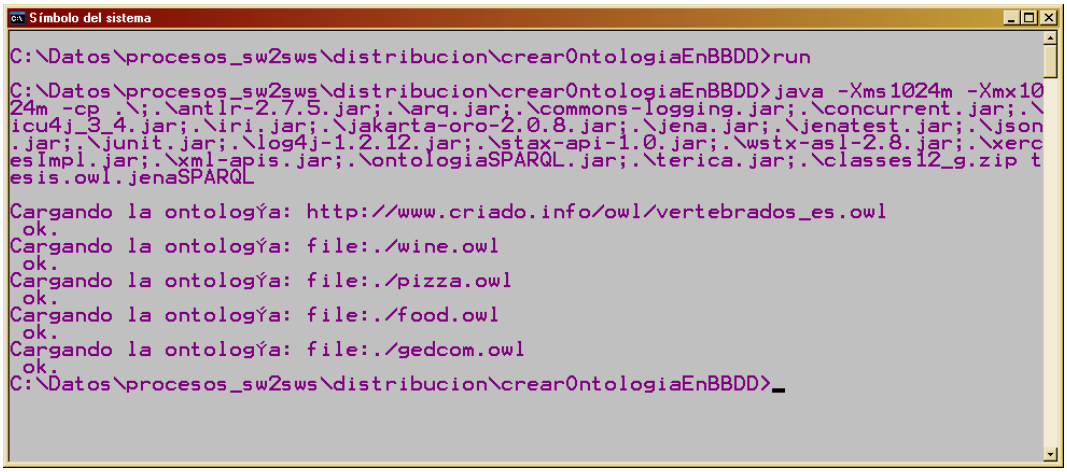


Figura 4. 6: Inicialización de ONTOLOGIAS en la BBDD del buscador semántico

El resultado de esta operación (Figura 4.6) es que en la BBDD tendrá una relación de tablas similar a la Figura 4.7:

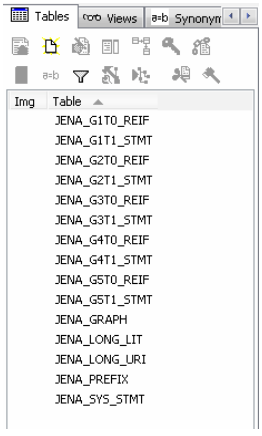


Figura 4. 7: Tablas en la BBDD que representan las ontologías

4.8.2. Mantenimiento de vistas semánticas en el buscador semántico.

En el apartado "3.3.4. Mantenimiento de un sitio web semántico" hemos indicado cómo mantener la coherencia entre el sitio web y el sitio web semántico en base a detectar si el contenido web ha cambiado. En cuyo caso se generará de nuevo las *vistas semánticas* que sean necesarias. Pero, por otra parte, las aplicaciones de la Web Semántica que usan la información de un sitio web semántico, deben mantener las representaciones OWL DL coherentes con las páginas web que representan. Es decir, si una *vista semántica* o un conjunto de *vistas semánticas* han sido modificadas, la anotación en OWL DL que se realizó en la BBDD de la aplicación semántica, como puede ser el buscador semántico, deberá ser modificada. En consecuencia, cualquier aplicación de la Web Semántica que incorpore una BBDD donde se registran *vistas semánticas* tendrá que incorporar los mecanismos para mantener esta información coherente respecto el sitio web semántico. En esta tesis hemos llegado a desarrollar la funcionalidad de incorporar por primera vez las *vistas semánticas* de un sitio web semántico en nuestro prototipo de buscador semántico, el detalle se describe en el epígrafe "A.9" del anexo A. La sincronización de páginas semánticas, tanto la herramienta de transformación de un sitio web a un sitio web semántico, como desde el punto de vista de las aplicaciones semánticas se identifica en el capítulo 5 como una de las posibles líneas de investigación futuras.

4.8.3. ¿Cómo se utiliza el buscador VISSEM?.

En el epígrafe 4.4. indicábamos que en nuestro prototipo, el buscador selecciona el contexto sin intervención del usuario, de manera, que el buscador identifica automáticamente que dominio a utilizar. Esto es sencillo en el prototipo, ya que, soporta unas pocas ontologías y la información volcada en la BBDD no es demasiada, pero es un problema complejo al que se le debe dar importancia (véase líneas futuras de investigación).

Para mostrar como funciona Vissem vamos a comparar las busquedas de ejemplo con los dos principales buscadores como son Google y Yahoo, y además con los buscadores Hakia y Powerset, que como hemos visto anteriormente incorporan capacidades de PLN en el interface con el usuario. El primer ejemplo que analizamos es preguntar al buscador lo siguiente "¿ El Bengala es un gato ?". Obsérvese como en la Figura 4.8, nuestro buscador responde a la

pregunta formulada, con “Si” y ademas proporciona un enlace directo a Bengala. Seguidamente muestra la lista de gatos que conoce. Compruebe, que ante esta pregunta Google (Figura 4.8) y Yahoo (Figura 4.9) proporcionan buenos enlaces, al igual que los buscadores semánticos Hakia y Powerset..

Figura 4. 8: Comparación con Google, ejemplo 1

Obsérvese como en la parte superior izquierda (Figura 4.8) se especifica el dominio general sobre el que está operando el buscador *Vissem*. El propio buscador ha decidido que es la ontología que debe utilizar en base a la pregunta realizada.

Figura 4. 9: Comparación con Yahoo, ejemplo 1

En el ejemplo siguiente (Figura 4.10) realizamos la pregunta "¿ Es el Boxer un gato ?". Nuestro buscador contesta "No", proporcionando un enlace directo a un Boxer y una lista de perros en general. Pero impresiona comprobar como Google, el buscador más popular de la Web, proporciona unos enlaces completamente desacertados.

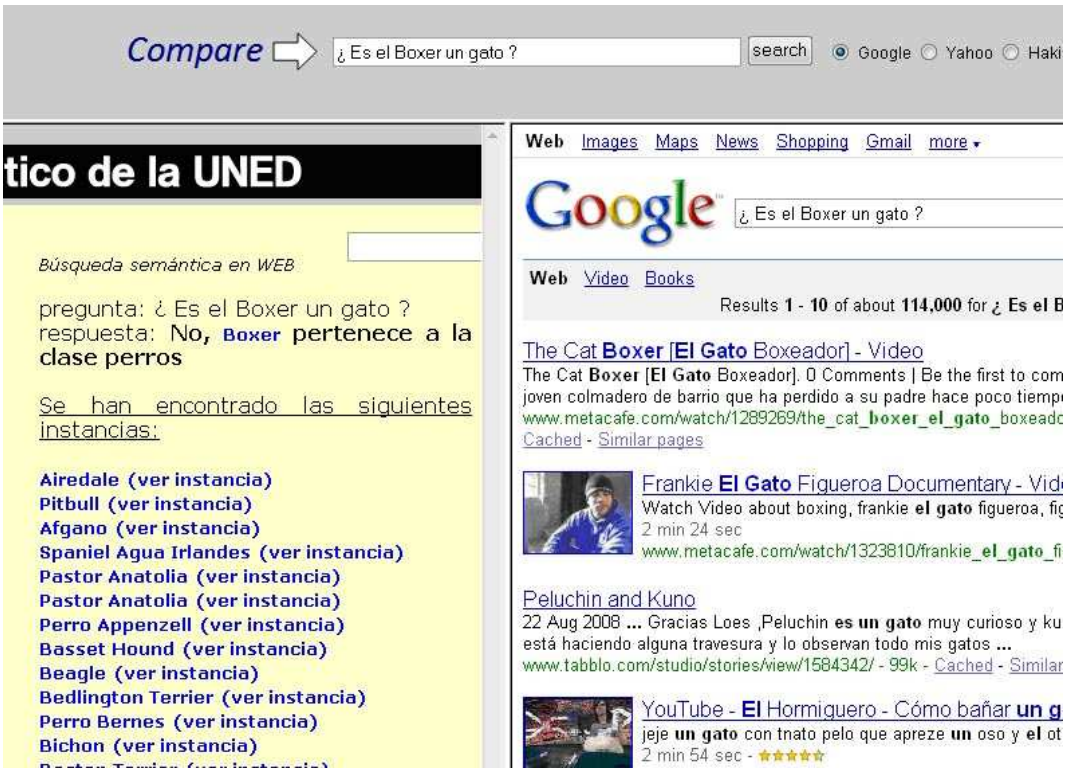


Figura 4. 10: Comparación con Google, ejemplo 2

Sin embargo, Yahoo (Figura 4.11), en sus primeras referencias, lo hace bien. Hakia falla (ver Figura 4.12), confundiendo, al igual que Google, un video de un gato que al parecer boxea con una raza de gatos, lo que no ocurriría si el buscador además del PLN en el interface incorporase *vistas semánticas.*

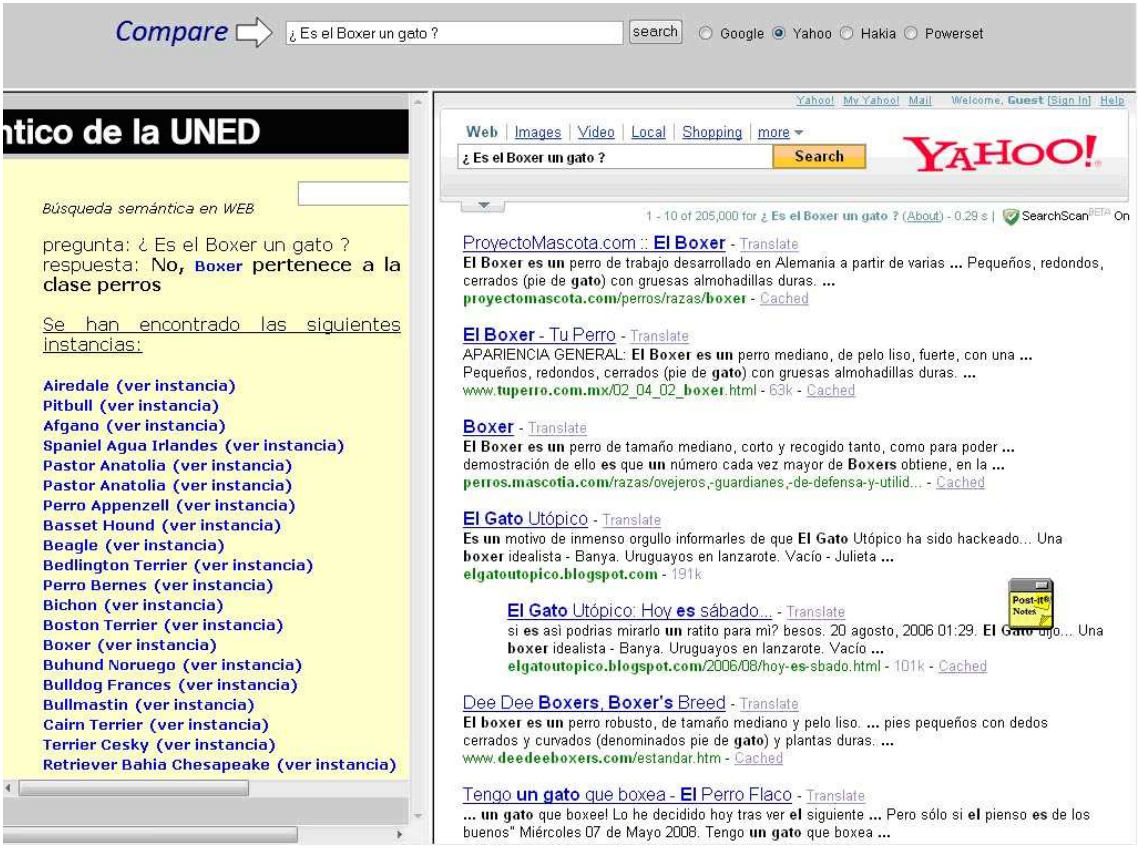


Figura 4. 11: Comparación con Yahoo, ejemplo 2

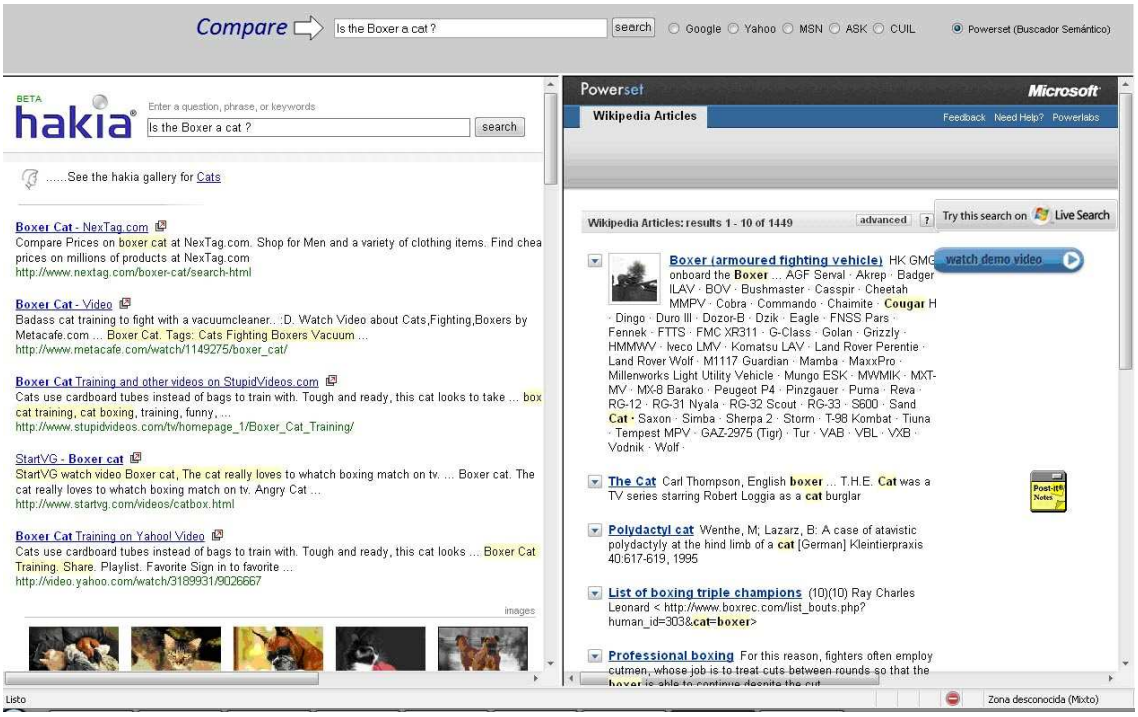


Figura 4. 12: Comparación Hakia y Powerset (ejemplo 2)

Nuestro buscador semántico contesta a preguntas sencillas en base a las vistas semánticas que incorpora en su BBDD y presenta al usuario enlaces en HTML, aunque como es un prototipo también podemos ver la interpretación de la vista semántica que utiliza y que lo vincula con la página web fomando una página semántica. Por ejemplo, si preguntamos "¿ Que es un Pitbull ?", el buscador nos contesta que pertenence a la clase perros (Figura 4.13).

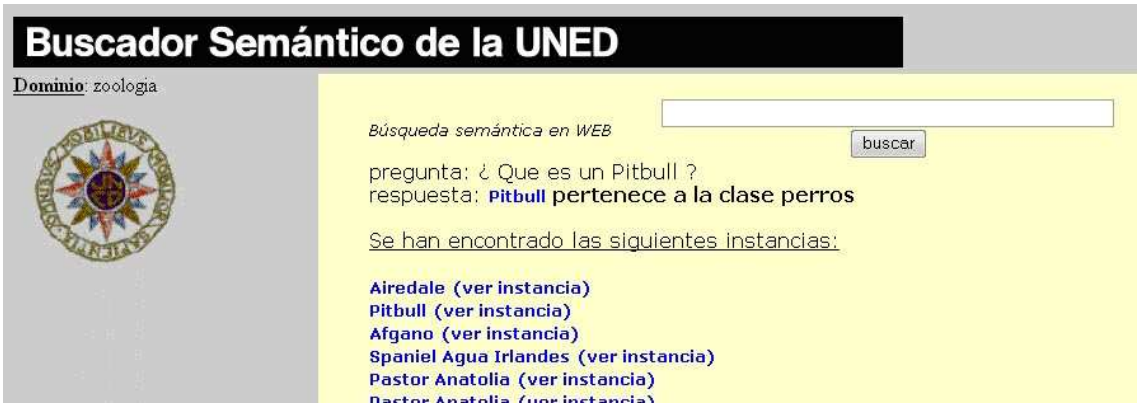


Figura 4. 13: Ejemplo 3, buscador UNED

El enlace "(ver instancia)" se suprimirá en un entorno de producción, ya que apunta a la instancia, de forma que es lo que entiende el sistema, en la figura 4.14 se puede ver la instancia

que permite al buscador comprender la pregunta. El usuario accederá solo al enlace directo a Pitbull (página web)

```
<rdf:RDF
    xmlns:j.0="http://www.criado.info/owl/vertebrados_es.owl#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://213.139.18.202/owl/mascotas/26-12-2007/perros/el-perro-american-staffordshire-terrier/default_3F972AA91EEE3451B64
  xml:base="http://213.139.18.202/owl/mascotas/26-12-2007/perros/el-perro-american-staffordshire-terrier/default_3F972AA91EEE3451B6
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://www.criado.info/owl/vertebrados_es.owl#"/>
  </owl:Ontology>
   <j.0:perros rdf:ID="Pitbull"/>
   <owl:AllDifferent>
     <owl:distinctMembers rdf:parseType="Collection">
       <perros rdf:about="#Pitbull"/>
     </owl:distinctMembers>
   </owl:AllDifferent>
</rdf:RDF>

<!-- Creado por [html2ws]  http://www.luis.criado.org -->
```

Figura 4. 14: (ver instancia) de Pitbull

4.8.4. Requisitos funcionales en el diseño de buscadores semánticos.

A lo largo de esta investigación hemos insistido en que antes de hablar de aplicaciones semánticas, como son los buscadores semánticos, es necesario disponer de información semántica expresada en OWL, para lograr esto, propusimos e implementamos un procedimiento de transformación de un sitio web a un sitio web semántico.

Los problemas principales a los que se enfrentan los buscadores semánticos son los que van a permitir las diferencias entre ellos. El modo de resolver estos problemas pueden convertirse en las verdaderas ventajas empresariales que popularicen un buscador semántico frente a otro. Estos problemas a resolver son los siguientes:

Contexto: El contexto sobre el que se formula una pregunta es uno de los mayores problemas, tal y como hemos descrito en el epígrafe "4.4. Contexto de la busqueda", ya que el sentido de las palabras que forman una frase tiene que interpretarse recurriendo a una ontología. Aunque los usuarios de los buscadores actuales suelen hacer preguntas muy cortas (2-3 términos), con la Web Semántica probablemente se utilicen más términos, ya que se podrán elaborar preguntas sofisticadas que los buscadores semánticos entenderan y esto puede facilitar la tarea de la selección de contexto. En cualquier caso, lograr identificar el dominio de conocimiento es fundamental para el buen funcionamiento del buscador semántico, que propondrá enlaces al usuario en base a esta identificación de contexto.

Interface de PLN conducido por ontologías: El procesado de lenguaje natural es una disciplina compleja. El objetivo es recuperar información directamente del lenguaje y para hacer esto bien, hay que comprender las frases en cierta grado.

El lenguaje natural, entendido como la herramienta que utilizan las personas para expresarse, posee propiedades que merman la efectividad de los sistemas de recuperación de información textual. Estas propiedades son la variación y la ambigüedad lingüística. La variación es la posibilidad de utilizar diferentes palabras o expresiones para comunicar una misma idea y la ambigüedad lingüística, donde una palabra o frase permite más de una interpretación. Ambos problemas afectan a todos los niveles (dichos niveles se comentaron en el apartado "3.2.2.2. PLN en Castellano"). Resultando especialmente complejo en el nivel pragmático, que permitirá interpretar el significado en base al contexto en que es utilizado.

Ahora bien, en un contexto concreto relacionado con una ontología determinada, el problema de la ambigüedad se miminiza y el problema de la variación puede desaparecer, ya que las ontologías pueden incorporar sinónimos, tanto para propiedades (equivalentProperty) como para clases (equivalentClass).

El interfaz debe ser capaz de extraer de las preguntas del usuario los objetos relacionados con las clases, propiedades e instancias que se incluyen también en la ontología, para de esta forma consultar las tripletas OWL de la BBDD del buscador semántico mediante un razonador.

El razonador del buscador semántico.

El razonador es una pieza clave y debe ser muy eficiente. Los buscadores semánticos deberán esforzarse en proporcionar la mejor información. De forma que si un webmaster ha generado su sitio web semántico mediante una herramienta del estilo “sw2sws” los buscadores semánticos parten de la misma calidad de información que incorporarán a sus bases de datos, pero la oferta de enlaces al usuario puede ser mejor si los propios buscadores dedican esfuerzos a potenciar ciertos aspectos de las ontologías aceptadas como de uso público.

Por ejemplo, en nuestro prototipo del buscador semántico *Vissem* hemos visto que responde bien frente a preguntas sencillas del estilo: ¿ el Bengala es un gato ?, ¿ el periquito es un perro ?, ¿ es el Airedale un gato ? o ¿ que es un Pitbull ?. Este conocimiento ha sido generado automáticamente por el webmaster mediante nuestra herramienta *sw2sws*, de

manera que el buscador ha añadido las *vistas semánticas* a su BBDD. Pero también podría ser que el buscador semántico añadiese su propia información, como por ejemplo, que el doberman es un perro peligroso o que un galgo es el perro más rápido pero no tanto como un caballo.

4.9. Anotación semántica de los buscadores semanticos.

En nuestra exposición hemos defendido que el objetivo de los buscadores semánticos no es transformar la información existente en información semántica, que es inviable, de acuerdo a esto hemos propuesto en el capítulo 2 y 3 un procedimiento de transformación basado en la colaboración, de manera que cada responsable de sus contenidos transformará su web en web semántica. Sin embargo, es fácil imaginar, que los buscadores semánticos encontrarán interesante diferenciarse unos de otros para lograr mayor popularidad. De la misma forma que hoy Google domina el ranking de los buscadores en base a su algoritmo PageRank (desarrollado por Larry Page y Sergey Brin), en el futuro será en base a nuevos elementos, como por ejemplo; la identificación automática del contexto, el interface de PLN conducido por ontologías, anotaciones semánticas generadas por el propio buscador y el razonador del buscador semántico.

Para comprobar como puede mejorar un buscador, variando solo alguno de los aspectos que acabamos de enumerar, hemos experimentando con *Vissem* dotándolo de mayor conocimiento en la ontología de vertebrados que utiliza, introduciendo clases genéricas con sus propiedades y valores de las mismas, pero el coste es pasar de OWL DL a OWL FULL. En la Figura 4.15 se puede ver como desde PROTÉGÉ hemos creado diferentes clases de perros, que para *Vissem* serán clases genéricas que definen características de algunos perros. Observesé como hemos cuantificado el valor de las propiedades asociadas, por ejemplo, hemos definido que el valor de rápidez del “caniche” es de 0.2 y su peligrosidad la hemos fijado en 0.1. Estos valores, aunque subjetivos, que pueden recordar a los valores multievaluados de la lógica difusa, van a permitir al razonador distinguir entre perros más y menos peligrosos.

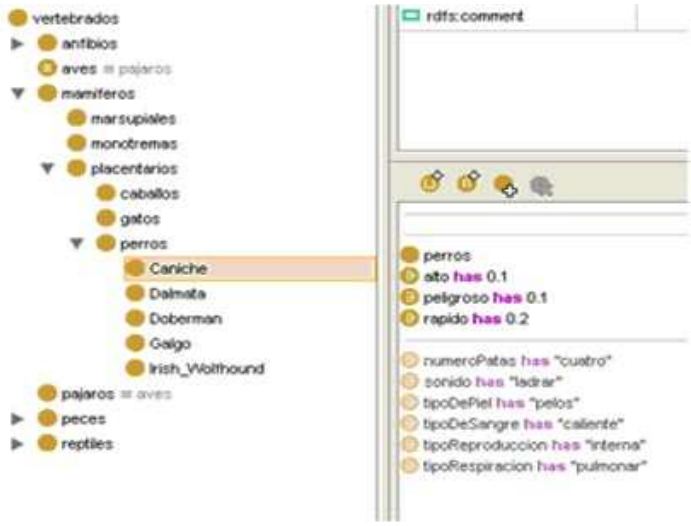


Figura 4. 15: Clase genérica caniche.

Para indicar que el "doberman" es el más peligroso de los perros, hemos especificado en la ontología el valor de 1.0 para la propiedad "peligroso" (ver Figura 4.16).

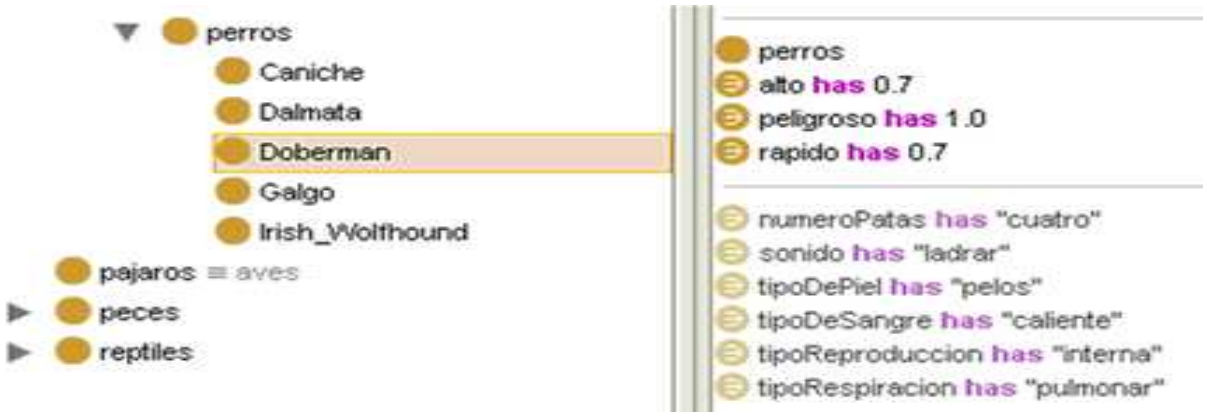


Figura 4. 16: Clase genérica doberman.

Una vez enriquecida la ontología de vertebrados la hemos incorporado a Vissem por el procedimiento comentado con anterioridad (apartado "4.8.1. Inicializando la BBDD del buscador semántico").

4.10. Vissem enriquecido con su propia anotación semántica.

Añadiendo las modificaciones indicadas en el epígrafe anterior a la BBDD del buscador semántico se puede plantear preguntas más complejas de las vistas en el apartado “4.8.3. ¿Cómo se utiliza el buscador VISSEM?”.

El interface de PLN conducido por ontologías del buscador determina si la pregunta contiene clases, instancias y propiedades incluidas también en la BBDD y en base a esta información genera una consulta al motor de base de datos. Observesé que no utilizamos el razonador de Jena por considerarlo inestable cuando desarrollamos nuestro prototipo (ver apartado "1.5.1.9. Interfaces de programación").

El algoritmo que se ha implemtado en *vissem* para el PLN conducido por ontologías se basa en un esquema de tripletas. El buscador opera de la siguiente manera:

1. Cada término de la pregunta se busca en la tripletas de la BBDD de *vissem*, si *vissem* conoce el término se clasifica en función de que sea una propiedad, una instancia o una clase.
2. Cada pregunta relaciona un número determinado de propiedades y/o instancias y/o clases. En función de estas combinaciones se realiza una consulta concreta a la BBDD.

Para el caso más sencillo, preguntas del estilo “¿ que es un Pitbull ? ”, *vissem* busca en las tripletas de la BBDD que puede ser cada uno de los términos que forman la misma y solo encontrará “Pitbull”, de manera, que está claro que el usuario quiere información sobre esta instancia. *Vissem* primero responde la pregunta y después proporciona enlaces que justifican su respuesta.

Supongamos ahora una pregunta del estilo “¿ es el Airedale un gato ?”. Al analizar esta frase se encuentran dos términos que están en las tripletas de vissem “AIREDALE” y “GATO”, de manera que *vissem* clasifica dichos términos determinando que uno es una instancia y otro es una clase. Por lo tanto, la pregunta es clara y se debe entender que se esta buscando una relación entre instancia y clase. *Vissem* comprobará si la instancia pertenece a la clase, en este caso “AIREDALE” pertenece a la clase “PERROS”, de manera que *vissem* puede contestar que “no” y proporcionar enlaces a “AIREDALE” y otros perros. Esta sencilla pregunta ocasiona problemas a cualquier buscador actual como google y yahoo,

incluidos los nuevos buscadores semánticos como son Powerset, true Knowledge y Hakia.

Vamos a probar con preguntas un poco más avanzadas, como puedan ser del tipo "¿ cual es el perro más peligroso ?" (ver Fgura 4.17). Para contestar a esta pregunta, que relaciona una clase y una propiedad hemos tenido que enriquecer la anotación semántica de *vissem*. El buscador semántico dispone de clases de perros tipo en la cual se ha especificado valor a sus propiedades. En consecuencia *vissem* puede obtener el valor de peligrosidad asociado a cada perro y determinar cúal es el más y menos peligroso. Obsérvese como *vissem* contesta que el perro más peligroso es el "DOBERMAN" y además proporciona un enlace sobre esta raza de perro al usuario.

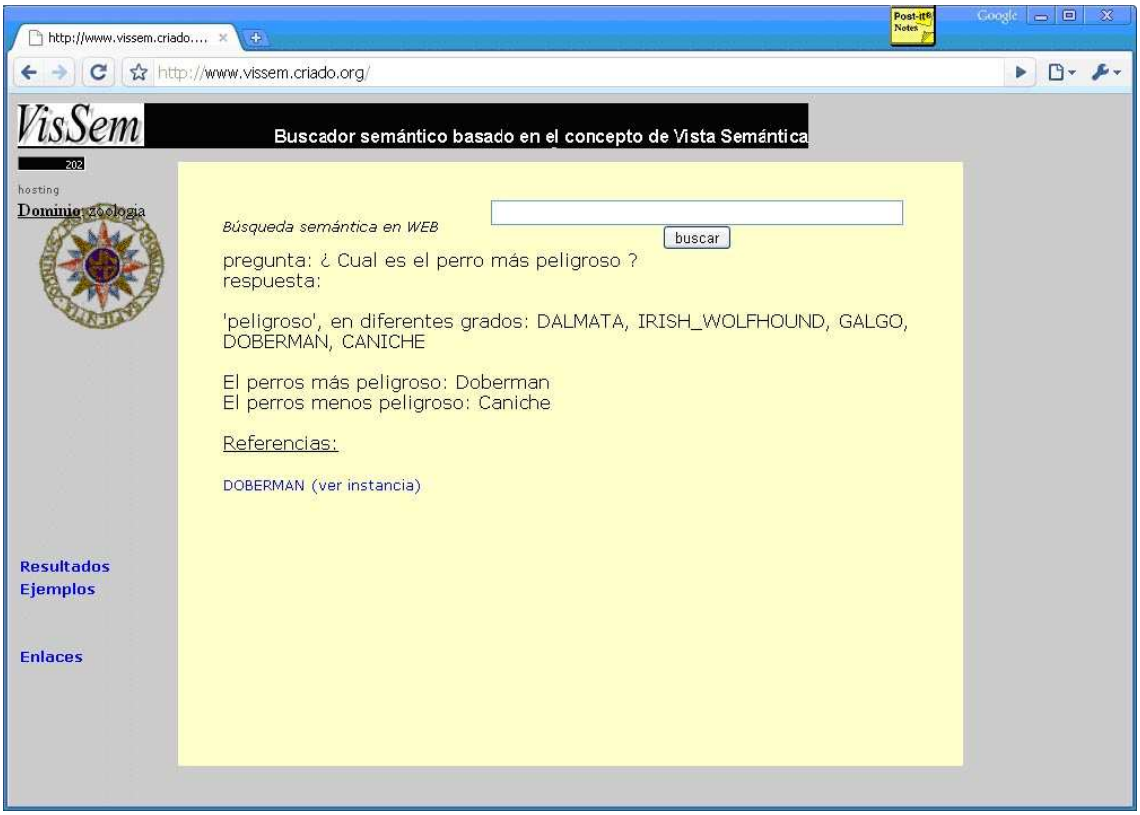


Figura 4. 17: ¿ Cúal es el perro más peligroso ?

Para una pregunta como "¿ cúal es el animal más rápido ?" (ver Figura 4.18) téngase en cuenta que "animal" no figura en la ontología, de forma, que *vissem* obtendrá el valor de todas las clases con propiedad rápido. En consecuencia, la respuesta incluira cualquier vertebrado del cual tenga información. En este caso de caballos y perros.

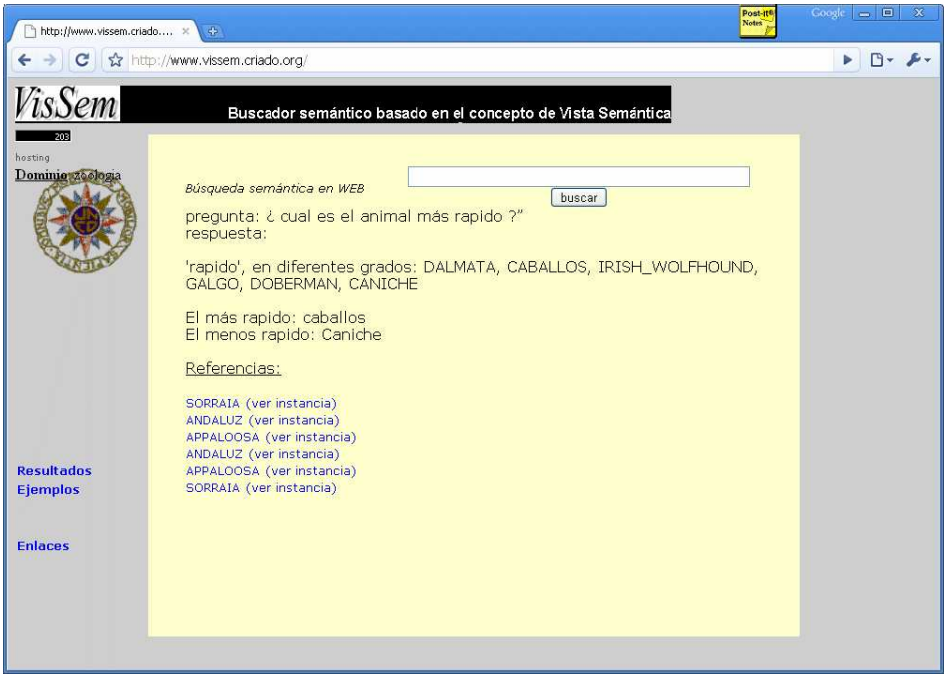


Figura 4. 18: ¿ Cúal es el animal más rápido ?

También hemos implementado la posibilidad de comparar una cualidad, de forma que ante la pregunta "¿ que es más peligroso un caniche o un doberman ?", *vissem* es capaz de determinar que el dobermás es más peligroso.

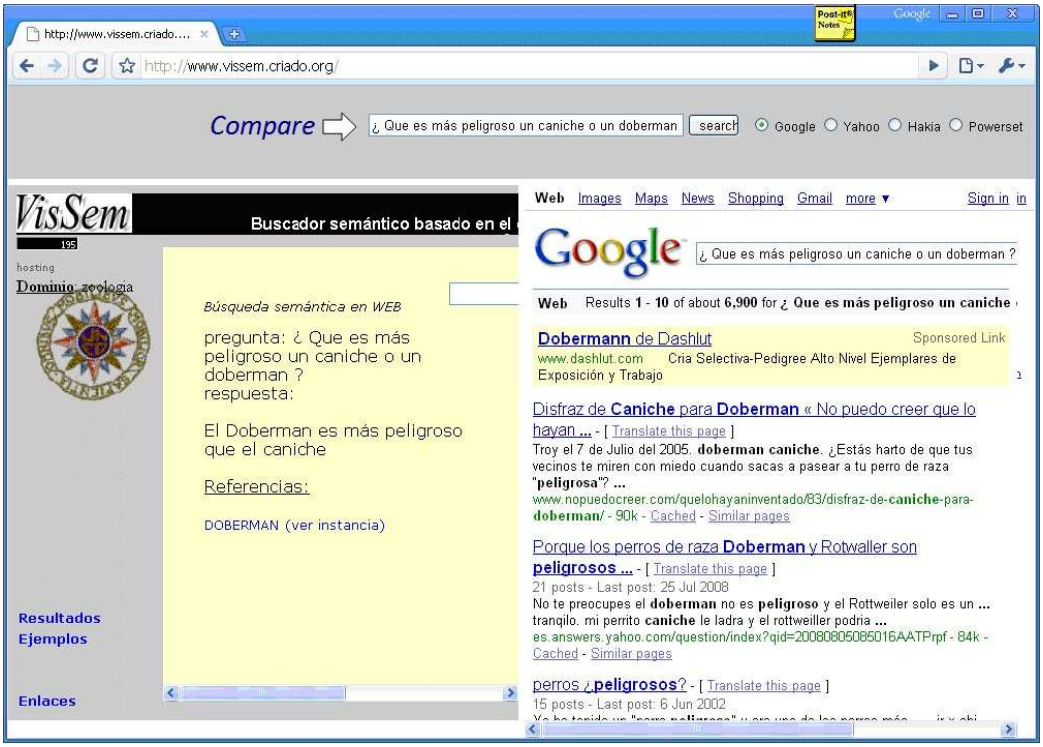


Figura 4. 19: ¿Qué es más peligroso un caniche o un doberman ?

Capítulo 5: Valoración, conclusiones y trabajos futuros.

Se han analizado las debilidades, amenazas, fortalezas y oportunidades de la Web Semántica respecto de la Web actual. La oportunidad principal para implantar la Web Semántica es, en realidad, la debilidad de la Web actual, pero indicabamos que esto no sería posible si no se disminuye la complejidad de la tecnología, que es la principal debilidad de la Web Semántica para que los usuarios activos pudieran adaptarse a ella. Actores que, repetidas veces se ha dicho en esta tesis, son los responsables del contenido web y, por lo tanto, los que deben adaptarse al nuevo paradigma. Si tenemos actores que generen páginas semánticas entonces podremos construir aplicaciones semánticas, entre las que son esenciales los buscadores semánticos de propósito general.

Hicimos también hincapié en que dichos actores deben ocuparse, a nuestro juicio, de tres acciones de gran trascendencia para el éxito de la Web Semántica. Estas son:

- Identificación del dominio de conocimiento al que pertenece la página o sitio web.
- Anotación formal de contenidos.
- Actualización y mantenimiento (coherencia de contenido con anotación semántica)

Sin embargo, pusimos de manifiesto que se necesitaban herramientas que facilitasen todo este trabajo a los usuarios activos, ya que si para este proceso se dispone de herramientas adecuadas de anotación, sean semi-automáticas o automáticas, entonces parece factible que se generasen páginas semánticas sin demasiado esfuerzo y, por consiguiente, se fomentará la aparición de sitios webs semánticos cuya información formalmente anotada pueda ser explotada por las aplicaciones semánticas. Esto es lo que hemos procurado hacer en esta tesis.

En este último capítulo vamos resumir las aportaciones para, a continuación, dar una valoración de las mismas y establecer las conclusiones correspondientes. Termina el capítulo con una propuesta amplia de trabajos futuros que surgen a partir de nuestra investigación.

5.1. Aportaciones

La primera aportación es de tipo metodológico, un procedimiento para contribuir en la extensión de la población de ontologías, que facilita a un usuario activo el etiquetado semántico de la información que gestiona, y que ya ha descrito en texto en su página HTML. Hemos definido unas etapas de transformación que deben realizarse de forma secuencial.

- *Identificación*, que permite asociar la ontología u ontologías que son más afines al contenido de la página web.
- *Extracción*, que procesa el texto a nivel morfológico y sintáctico.
- *Interpretación*, que lleva a cabo la anotación semántica en base a los procesos de identificación y extracción.

En nuestro trabajo se tiene muy en cuenta la posibilidad de que el contenido a etiquetar pueda hacer referencia a diferentes temas o pueda interpretarse desde diferentes puntos de vista, distintas ontologías proporcionan diferentes formas de "ver" el contenido de una página web. El proceso puede "poblar" diferentes ontologías desde el mismo contenido, lo que en este trabajo denominaremos generar diferentes "*vistas semánticas*".

Pero además un sitio web semántico debe ser compatible con la Web actual, es decir, el proceso de anotación no debe afectar al funcionamiento actual de cualquier buscador. En consecuencia, al transformar un sitio web en un sitio web semántico se obtendrá funcionalidades semánticas que podrán ser explotadas por un buscador semántico, pero cuando sea tratado por un buscador ordinario existirá compatibilidad total y el buscador ordinario lo tratará como si fuera un sitio web más. También en esta tesis se ha tenido en cuenta esta exigencia, las *vistas*

semánticas se mantienen diferenciadas de la página HTML, accesibles pero sin afectar a los buscadores habituales.

En el desarrollo, la metodología empleada se ha basado en simplificar la problemática sin perder la categoría conceptual para poder abarcar todo el ámbito de la propuesta, compuesta por una secuencia de procesos que se desarrollan a lo largo de la tesis. Es decir, se ha planteado un escenario simplificado que recrea los elementos fundamentales de la Web actual para proponer una estrategia de migración o transformación hacia la Web Semántica. Las conclusiones alcanzadas son el resultado de un proceso de autocorrección experimental. Hemos implementado por completo la propuesta de esta tesis que puede ser verificada por cualquier investigador siguiendo las indicaciones del anexo A.

Para realizar esta transformación o migración, se ha implementado una herramienta prototipo (*sw2sws*) que automatiza las tres etapas que hemos presentado. Se ha probado sobre sitios webs reales. Nuestra herramienta prototipo automatiza el proceso de anotación con las ontologías usadas en la tesis, pero es fácilmente adaptable para soportar otras. Además nuestro enfoque acepta la posibilidad de intervención del usuario (proceso semiautomático) que complete o mejore cualquiera de las fases del proceso global. Para probar el proceso, hemos realizado nuestro propio módulo de PLN que permite mostrar la viabilidad para usuarios activos.

Finalmente hemos construído un prototipo de buscador semántico (VISSEM), cuyas búsquedas se basan en las *vistas semánticas* y cuyos resultados siguen siendo en el formato web al que el usuario está acostumbrado. De manera, que un usuario de un buscador semántico no ve la Web Semántica pero percibe sus efectos en una mejora del interfaz, en cuanto la expresión natural de las preguntas al buscador, y de la calidad de la información recibida, mucho más precisa, más ajustada como respuesta a cuestión planteada.

5.2. Valoración de la propuesta

Es difícil pensar en una evaluación objetiva del procedimiento de tranformación de la Web en Web Semántica mediante una estrategia de anotación semi-automático colaborativa, ya que, necesitaríamos usuarios que probasen nuestra implementación y posteriormente realizar una encuesta.

Pero es cierto también, que actualmente un usuario activo, como un webmaster no puede crear un sitio web semántico sin esfuerzo. De forma, que con una herramienta, como la del prototipo de esta tesis, un webmaster podría transformar su web en menos de una horas (ver Tabla 5.1). Los resultados obtenidos, aunque modestos, son gratificantes, ya que, hemos podido automatizar el proceso completo de transformación, que era el objetivo principal de la tesis. En nuestras pruebas, hemos logrado transformar, de forma automática, cuatro sitios web a sitios web semánticos. Estos primeros sitios web semánticos son aún pobres, ya que el sistema ha generado muy pocas *vistas semánticas* respecto a la información que realmente contienen. Pero lo importante es que esta información semántica puede ser tratada por el buscador semántico aplicando lógica descriptiva SROID(D). Obteniendo así respuestas para el usuario de mayor calidad.

Los sitios transformados han sido descargados completamente de Internet y gestionados por nuestra aplicación, dependiendo de la complejidad del lenguaje se obtienen resultados mejores o peores (ver Tabla 5.1). Conviene matizar que todo se ha realizado de forma automática, sin conocer de antemano la información a la que el sistema se enfrenta y que el proceso se realiza localmente por el webmaster, de manera que el tiempo no es crítico.

Sitio Web	número de paginas	Identificadas	tiempo
http://www.perrilandia.com/	423	419	2 minutos
http://www.todoperros.com	1852	1546	10 minutos
http://www.mascotas.com	1060	1028	12 minutos
http://www.mascotasyhogar.com/	6845	874	39 minutos
Sitio Web	número de paginas	Extracción	tiempo
http://www.perrilandia.com/	423	170	57 segundos
http://www.todoperros.com	1852	101	1 minuto
http://www.mascotas.com	1060	1023	2 minutos
http://www.mascotasyhogar.com/	6845	1161	5 minutos
Sitio Web	número de paginas	Vistas Semánticas	tiempo
http://www.perrilandia.com/	423	98	< 1 minuto
http://www.todoperros.com	1852	3	< 1 minuto
http://www.mascotas.com	1060	5	< 1 minuto
http://www.mascotasyhogar.com/	6845	67	< 1 minuto

Tabla 5. 1: Resultados de la transformación

Obsérvese como en la primera etapa se logra identificar la mayoría de las páginas web y como en la etapa siguiente disminuye considerablemente el volumen de páginas en las que se logra extraer contenido. La última etapa interpreta el contenido extraído de acuerdo a la

ontología, de manera, que funciona como un filtro y solo la información que tiene sentido en la misma se anota. Es poca la información que se logra anotar automáticamente, pero si se tiene en cuenta los datos de las encuestas de Netcraft de mayo del 2009 [56] (ver epígrafe 2.1 y 3.1), el tamaño de la web superficial se cuantifica en más de 236 millones de sitios web, lo que supone, solo teniendo en cuenta la web superficial, aproximadamente $26{,}9 * 10^9$ páginas, que equivaldría (siguiendo las referencias de Bergman) a más de 94 millones de libros, en el peor de los casos y a 188 millones en el caso más favorable. Por lo tanto, no es aventurado pensar que aunque las herramientas de anotación automáticas seán básicas, la repercusión en la mejora de la calidad de la información experimentará un gran impulso. El beneficio de herramientas del estilo *sw2sws* parece evidente, pues cada sitio web será transformado, por cada webmaster, contribuyendo a transformar la Web. Es decir, se trata de transformar la Web en Web Semántica desde la cooperación.

La calidad de la anotación obtenida depende de varios factores; como son la propia calidad de la ontología con respecto a la que anota (afinidad, precisión, estandarización, completitud, etc), la claridad del contenido y la capacidad de extracción y análisis, condicionada, en gran medida, al procesado de lenguaje natural (PLN).

Nuestra propuesta representa una solución de las debilidades y amenazas de la Web Semántica actual (ver Figura 5.1). Estas debilidades y amenazas se convierten en fortalezas en una Web Semántica basada en *vistas semánticas*.

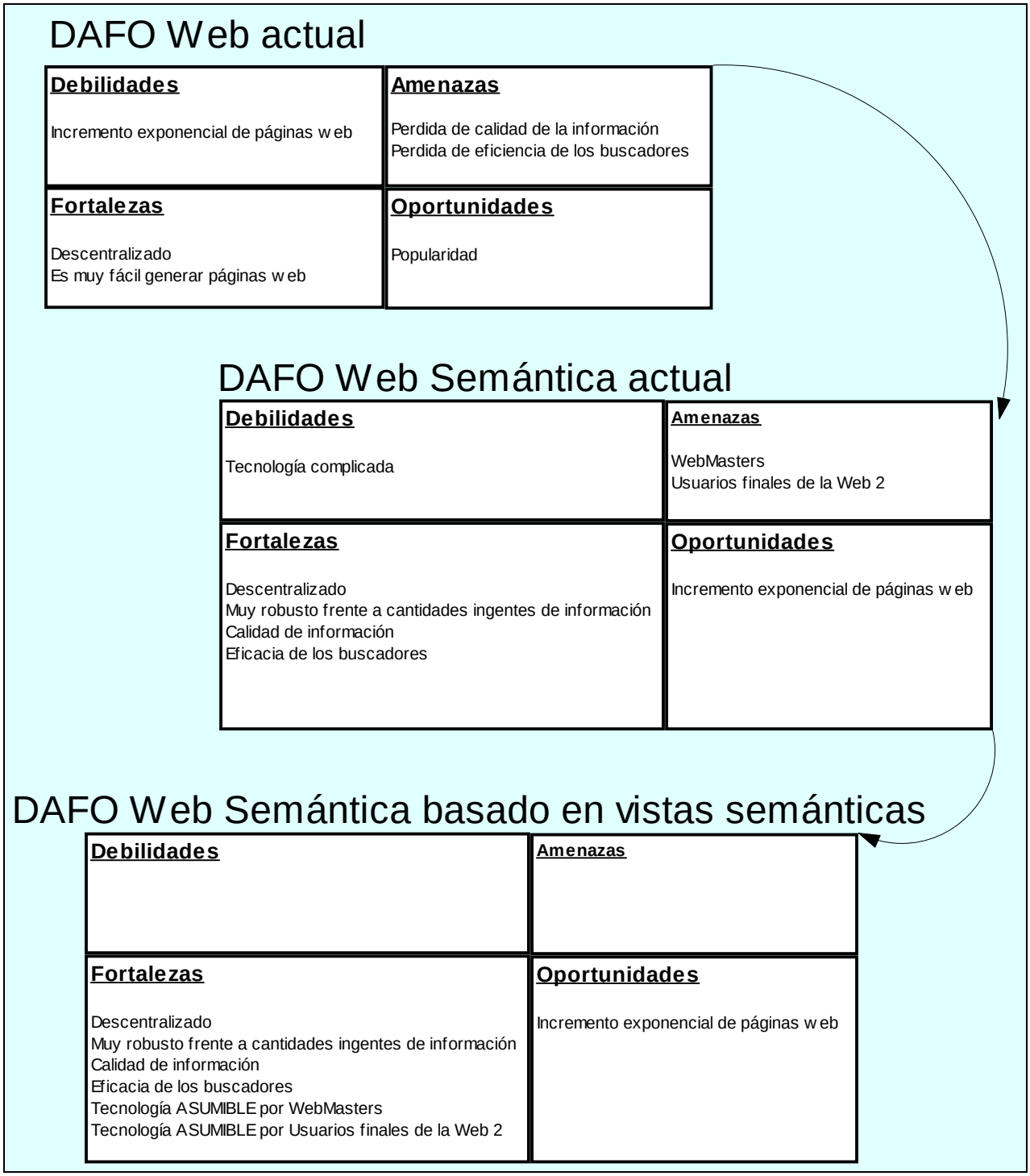


Figura 5. 1: DAFO de la Web Semántica basada en vistas semánticas

.

5.3. Conclusiones

Hemos dicho en varias ocasiones, que estamos dando los primeros pasos hacia la Web Semántica. En el 2004 se definieron las especificaciones OWL DL que toda la Comunidad Científica hoy usa para la descripción de las ontologías. Pero las especificaciones no son suficientes para hacer realidad la Web Semántica, hacen falta muchas herramientas. Nosotros pensamos que una herramienta importante para impulsar la Web Semántica son las herramientas de anotación automáticas y consideramos que las *vistas semánticas* pueden ser un medio para lograr este tipo de herramientas. Pero para construir estas herramientas de uso masivo, es necesario avanzar en mejorar las técnicas de PLN.

Identificamos tres factores fundamentales que son necesarios para implantar la Web Semántica:

- El PLN: Fundamental para que las herramientas de anotación automáticas sean eficaces y para el interface de los buscadores semánticos con el usuario. Además es el enlace con el motor lógico del buscador semántico.

- La cooperación de los usuarios activos: generarán y mantendrán la información semántica con sus herramientas automáticas de anotación.

- La explotación de la información semántica: Las aplicaciones semánticas tendrán que tratar la información semántica de forma inteligente. Los motores de estas aplicaciones utilizarán, en principio, lógica descriptiva para obtener la información más adecuada, por lo que parece razonable aplicar técnicas de IA, ya que es posible avanzar a lógicas más complejas como pueda ser la lógica difusa. De manera que, por ejemplo, el motor de los buscadores semánticos vendrá muy condicionado por las capacidades lógicas que admita para tratar su información semántica y probablemente sea esto lo que marque diferencias en los buscadores del futuro. De la misma manera que en el pasado lo marco el algoritmo PageRank de Google.

5.4. Trabajos futuros

Detallamos en primer lugar, las líneas futuras de investigación que presumiblemente requieren varios años de dedicación y, en segundo lugar, señalamos posibles trabajos de innovación, que aunque requieren implementación de software, se estima que el tiempo será del orden de meses. Finalmente nos hemos aventurado a imaginar algunas oportunidades que presenta este nuevo paradigma al mundo empresarial.

5.4.1. Líneas de Investigación

En este epígrafe se describe trabajos, que por su importancia y complejidad, requieren mayor dedicación.

5.4.1.1. Sincronización de Páginas Semánticas.

La generación de páginas semánticas implica varios procesos como son: identificación, extracción e interpretación. El resultado de la ejecución de estos procesos es la transformación de un sitio web en un sitio web semántico, generando páginas semánticas basadas en *vistas semánticas*, o lo que es lo mismo, páginas que "entienden" las "máquinas". Pero este proceso no basta realizarlo una sola vez, sino cuando el contenido de cualquier página web del sitio transformado a sitio web semántico cambie. De esta forma se lanza, de nuevo, los procesos de transformación (identificación, extracción e interpretación) garantizando la integridad de cada página semántica, es decir, cada vista semántica debe ser coherente con el contendo de la página web que representa.

También en el ámbito de este trabajo sería oportuno profundizar en cómo esta información puede ser actualizada por las aplicaciones semánticas, como por ejemplo buscadores semánticos.

5.4.1.2. Optimización de los procesos de transformación basados en PLN.

Esta investigación debería abarcar desde el proceso de identificación hasta la generación y actualización de *vistas semánticas* mediante métodos morfológicos. Como se ha descrito en esta tesis, el método de identificación basada en la raíz ha sido aplicado, con resultados muy positivos, al caso del esperanto. Este método tiene por objetivo localizar los lexemas de los vocablos para identificar otras palabras relacionadas. Sin embargo, en el caso del castellano o el inglés este método no tiene aplicación inmediata.

Se considera una línea de gran interés incorporar en el proceso de identificación alguna técnica ligera de PLN. La identificación no puede ser computacionalmente pesada, ya que, es un proceso que debe comprobar si la página web que se esté analizando puede ser interpretada por miles o millones de ontologías. Nos parece que SCOGEME [Carreras Riudavets; 2002] y la versión europea de WordNet conocida como EuroWordNet [46] (que incluye castellano) son buenas opciones a tener en cuenta. De todas formas, no debemos olvidar herramientas profesionales (no gratuitas) como DataLexica [45]. Se recomienda potenciar mediante PLN el proceso de la generación de la *vistas semánticas* asociada a la página web, logrando una página semántica más rica que las alcanzadas en esta tesis.

5.4.1.3. Contenido Semántico de naturaleza Dinámica

Aunque las *vistas semánticas* se han definido como una estructura que soporta contenido de texto estático y dinámico, es evidente que el contenido dinámico es muy complejo de tratar, ya que las páginas web dinámicas no existen en realidad y se generan con una consulta a la Base de Datos (BBDD) en tiempo real; es por esto que se intuye que generar *vistas semánticas* sincronizadas con contenido dinámico representa uno de los principales problemas de la Web Semántica.

En esta investigación habrá que definir un procedimiento para permitir generar y modificar *vistas semánticas* en tiempo real. Probablemente incluso se tenga que proponer especificaciones de cómo hacer esto. Como referencia para comenzar esta investigación sería muy útil examinar propuestas como R2O y D2R. En concreto la tesis de Jesús Barrasa (enero 2007), que propone el lenguaje R2O y el procesador ODEMapster, que parece que puede servir de base para plantear soluciones al problema de la generación de *vistas semánticas* a partir de contenido dinámico (información en BBDD).

5.4.1.4. Algoritmos para la extracción de identificadores y traducción al idioma del contenido

La tarea de asociar una o varias ontologías con un contenido en lenguaje natural requiere diseñar un proceso que permita buscar, dentro de la ontología que se desea comparar, todas las palabras "claves" que la forman en el lenguaje natural de la ontología, esto es, los identificadores que se utilizaron para nombrar las clases y propiedades. Se debe tener en cuenta que la busqueda proporciona identificadores que deberán ser tratados adecuadamente, por ejemplo, la mayoría de los identificadores estarán formados por palabras concatenadas que habrá que separar. Una vez que se logra identificar y separar palabras, habrá que traducirlas al idioma natural que se use en el contenido, es decir, el idioma que el webmaster utiliza en sus páginas con el objetivo de poder comprobar si estas palabras traducidas aparecen en el contenido del webmaster calculando así la frecuencias de aparición. A mayor frecuencia de coincidencia mayor probabilidad que el contenido se asocie con una determinada ontología. Esta descripción del procedimiento deberá repetirse para cada ontología con la que se desee comparar el contenido. Lo que conduce a plantear también la coherencia y sincronismo de estos procesos con las ontologías originales de las que se parte. Se requiere que estos algoritmos sean eficientes computacionalmente.

5.4.1.5. Estrategias para el tratamiento masivo de ontologías en el proceso de identificación

Se requiere de un método que computacionalmente sea aceptable para decidir qué ontologías utilizar en el proceso de identificación del contenido de una página web. Téngase en cuenta, que una vez identificado la o las ontologías relacionadas con el contenido, la *vistas semánticas* se generán de inmediato, de manera que el proceso de identificación es una tarea bloqueante. Sirva como ejemplo ilustrativo del problema, que el buscador de ontologías SWOOGLE [41] actualmente, y todavía la Web Semántica es un proyecto, cuenta con aproximadamente 10.000 ontologías. Como es lógico, las herramientas de los usuarios activos para identificar automáticamente el tipo de contenido de sus páginas web, no puede comparar dicho contenido con 10.000 o más por motivos obvios de tiempo.

5.4.1.6. Selección automática de contextos para aplicaciones semánticas

El problema del contexto en aplicaciones semánticas concretas, como es el caso de los buscadores semánticos determinará, en cierta medida, la calidad de la busqueda y la sencillez para el usuario. Es fundamental desarrollar estrategias para seleccionar el "contexto" de forma automática. De esta forma, el buscador semántico podrá conocer el dominio de conocimiento que debe usar y por consiguiente las ontologías implicadas para sus consultas lógicas.

5.4.1.7. Motor de BBDD basado en lógica

Los buscadores no se diferenciaran en las tripletas, ya que este trabajo de anotación lo realizarán los usuarios activos, pero si en como explotar esta información. Por tanto, será necesario diseñar nuevos algoritmos de busqueda basados en las posibilidades de la Web Semántica, esto es en lógica descriptiva y también avanzar hacia otros tipos de lógicas más completas que se puedan generar a partir de ésta.

5.4.2. Trabajos de Innovación Tecnológica

Los trabajos que se indican en este epígrafe son recomendables para proyectos de fin de carrera.

5.4.2.1. Optimización del Buscador Semántico: VISSEM

No se debe olvidar que un buscador semántico deberá incorporar todas las prestaciones actuales mejorádolo con las ventajas semánticas. En esta tesis se ha construido un prototipo de buscador semántico que puede ser mejorado en varios aspectos:

- Mejora del interface de PLN
- Mejora de funciones lógicas
- link popularity
- Incorporación de más ontologías

5.4.2.2. Web Service para RO y RIO

La mayoría de las ontologías están etiquetadas en inglés y esto supone un freno para el desarrollo de la Web Semántica en castellano. Si queremos incorporarnos al nuevo paradigma de la Web Semántica debemos tener sistemas capaces de adaptar todas las ontologías existentes y futuras a nuestro idioma en tiempo real.

Proponemos la implementación de un conjunto de Web Services donde se ofrezca, por un lado, un repositorio de identificadores de ontologías (RIO) descrito en el capítulo 3, traducido al castellano, y por otro lado un repositorio de ontologías (RO) coherente con el RIO, de manera que las herramientas de anotación automática de los usuarios activos se conecten a estos repositorios que determinarán el número de ontologías sobre las cuales podrán realizar anotaciones semánticas. De esta forma se resuelven las limitaciones de la herramienta "Sw2sws" en cuanto al número de ontologías que soporta.

5.4.3. Oportunidades Empresariales

La Web Semántica está en pleno desarrollo, hoy apenas ha salido de las universidades, pero es un buen momento para acercarse a este nuevo conocimiento y tomar posiciones ventajosas, se trata de un sector virgen. Proponemos un par de ideas para las empresas.

5.4.3.1. Herramienta profesional para generación de páginas semánticas basadas en ontologías concretas.

Es muy probable que los propios buscadores semánticos proporcionen sus propias herramientas gratuitas de transformación para promover y promocionar la nueva generación de buscadores y posicionarse en el mercado, pero estas herramientas estarán enfocadas a contenidos de propósito general, debido a que un buscador tiene por objetivo ser útil para cualquier usuario. En consecuencia, se abre un mercado a herramientas profesionales para automatizar los procesos de los webmasters. Herramientas que representen ventajas relevantes frente a las herramientas gratuitas. Por supuesto, a todos interesará que esto ocurra, ya que estas herramientas tenderán a especializarse en ontologías concretas o en conjuntos de ontologías y ofrecerán porcentajes de extracción de conocimiento superiores que las herramientas gratuitas. Dicho en otras palabras, es interesante y recomendable que surgan herramientas especializadas

para sectores, por ejemplo, si somos webmasters de empresas de alimentación nos interesará la herramienta que genere las *vistas semánticas* especializadas en el dominio de conocimiento de la alimentación, es decir, que extraiga de las páginas web que mantenemos habitualmente, la máxima información, garantizando el mantenimiento y la integridad de las páginas semánticas.

5.4.3.2. Diseño específico de Ontologías.

Por muchas ontologías que existan siempre aparecerán singularidades que no se ajusten al contenido de una entidad, en algunos casos se tendrán que ampliar las ontologías existentes, en otros habrá que diseñarlas específicamente. De cualquier forma, parece un tipo de negocio perfecto para las Consultoras. No se debe olvidar nunca que, en este caso, interesa siempre compartir las ontologías, pues se trata que los buscadores semánticos proporcionen los mejores resultados que conduzcan al navegante de Internet a la web de nuestro cliente. No debe preocuparnos que copien parte de nuestras ontologías, ya que el negocio es inagotable, simplemente hay que preguntarse; ¿cuántas empresas tienen presencia en Internet?.

Bibliografía

Libros, Tesis y Artículos

Se presenta la relación de todos los textos de interes, ordenados primero por año de publicación y como segundo criterio el primer apellido del autor.

2008

[Cheng *et al*; 2008]
Gong Cheng, Weiyi Ge y Yuzhong Qu, *Falcons: Searching and Browsing Entities on the Semantic Web*. Beijing,China: Institute of Web Science, School of Computer. Science and Engineering. Southeast University, Nanjing 210096, PR China, 2008. 2 p.

[García Castro; 2008]
Raúl García Castro, *TESIS DOCTORAL: "Benchmarking Semantic Web technology"*; Dra. Asunción Gómez Pérez (director). Madrid. España: UNIVERSIDAD POLITECNICA DE MADRID, Facultad de Informática, 2008. Disponible en Web: http://delicias.dia.fi.upm.es/~rgarcia/

[Plataforma Tecnológica Española de Software y Servicios; 2008]
INES Plataforma Tecnológica Española de Software y Servicios, *Agenda Estratégica de Investigación*. ver. 3.0, abril 2008. on line: INES, 2008. 96 p. Disponible en Web: http://www.ines.org.es/

[*Rolando Beltrán*; 2008]
Rolando Beltrán A.. (2008). *Adaptación de contenido para la Web Semántica: Anotaciones semánticas, estado del arte*. Disponible en web. http://rolandobeltran1.googlepages.com/Entrega_Final.pdf

2007

[Attardi; 2007]
Giuseppe Attardi, *Búsqueda en la Web del futuro (MONOGRFÍA).*; Manuel Maña López y Agustín Palomar Olid (traductor/es).; artículo extraido de la revista "Novática: revista de la Asociación de Técnicos de Informática". Núm. 185. Barcelona: ATI, 2007. 6 p. ISBN: 0211-2124. Disponible en Web: http://www.ati.es/novatica/

[Barrasa Rodríguez; 2007]
Jesús Barrasa Rodríguez, *Modelo para la definición automática de correspondencias semánticas entre ontologías y modelos relacionales.*; Asunción Gómez Pérez (director/es). Madrid: Universidad Politécnica de Madrid. Facultad de Informática, 2007. 384 p.

[Danger Mercaderes; 2007]
Roxana Maria Danger Mercaderes, *TESIS DOCTORAL: "Extracción y análisis de información desde la perspectiva de la Web Semántica".*; Dr. Rafael Berlanga Llavorí (director). Castellón. España: UNIVERSITAT JAUME I , Departamento de Lenguajes y Sistemas Informáticos, 2007. Disponible en Web: http://krono.act.uji.es/PhDs/RoxPhD

[Galicia Haro *et al*; 2007]
Sofía N. Galicia Haro, Alexander Gelbukh, *INVESTIGACIONES EN ANÁLISIS SINTÁCTICO PARA EL ESPAÑOL.* México: Publicaión realizada con el apoyo de CONACyT. proyectos R420219-A y 50206, 2007. 347 p. ISBN: 970-36-0265-7. Disponible en Web: http://gelbukh.com/libro-investigaciones/LibroSint.pdf

[Huff *et al*; 2007]
Markus Huff, Stephan Schwan and B. Garsoffky, *The Spatial Representation of Dynamic Scenes - An Integrative Approach.* Berlin Heidelberg: Springer-Verlag, 2007. 16 p.

[Vallez *et al*; 2007]
Mari Vallez, Rafael Pedraza-Jimenez, *El Procesamiento del Lenguaje Natural en la Recuperación de Información Textual y áreas afines.* núm. 5,. en línea: Hipertext.net, 2007. 12 p. Disponible en Web: http://www.hipertext.net

2006

[Adida *et al*; 2006]
Ben Adida, Mark Birbeck, *Embedding RDF in XHTML.* on line: W3C, 2006. Disponible en Web: http://www.w3.org/TR/2006/WD-xhtml-rdfa-primer-20060310/

[Boag *et al*; 2006]
Scott Boag, Don Chamberlin, IBM Almaden Research Center, Mary F. Fernández, AT&T Labs, Daniela Florescu, Oracle, Jonathan Robie, DataDirect Technologies, Jérôme Siméon, IBM T.J. Watson Research Center, *XQuery 1.0: An XML Query Language*. on line: W3C, 2006. Disponible en Web: http://www.w3.org/TR/xquery/

[Cimiano *et al*; 2006]
P. Cimiano, J. Völker and R. Studer, *Ontologies on Demand? - A Description of the State-of-the-Art, Applications, Challenges and Trends for Ontology Learning from Text*, 2006. 86 p

[Fresno Fernández; 2006]
Víctor Diego Fresno Fernández, *Representación Autocontenida de Documentos HTML: una propuesta basada en Combinaciones Heurísticas de Criterios*.; Dra. Da. Raquel Martínez Unanue, Dra. Da. Angela Ribeiro Seijas, Tutor: Dr. D. José María Cañas Plaza (director/es). Madrid: Universidad Rey Juan Carlos. Escuela Superior de Ciencias Experimentales y Tecnología. Departamento de Ingeniería Telemática y Tecnología Electrónica, 2006. 225 p.

[Lamarca Lapuente; 2006]
María Jesús Lamarca Lapuente, *TESIS DOCTORAL, "HIPERTEXTO: EL NUEVO CONCEPTO DE DOCUMENTO EN LA CULTURA DE LA IMAGEN"*.; Dr. Félix del Valle Gastaminza (director/es). Facultad de Ciencias de la Información de la Universidad Complutense de Madrid, 2006. Disponible en Web: http://www.hipertexto.info/, http://www.hipertexto.info/documentos/owl.htm,

[Manzano-Macho *et al*; 2006]
David Manzano-Macho, Asunción Gómez-Pérez y Daniel Borrajo
, *HOLA: A Hybrid Ontology Learning Architecture.* Podebrady, Czech Republic: EKAW 2006 - 15th International Conference on Knowledge Engineering and ..., 2006. 2 p.

[Prud'hommeaux *et al*; 2006]
Eric Prud'hommeaux, Andy Seaborne , *SPARQL Query Language for RDF*. on line: W3C, 2006. Disponible en Web: http://www.w3.org/TR/rdf-sparql-query/

[Rivera; 2006]
Alicia Rivera, *Hacia una 'web' que entienda al usuario.* on line: El País, 2006. Disponible en Web: http://vistoyleido.blogspot.com/2006_02_01_vistoyleido_archive.html, http://www.elpais.es/articulo/elpfutpor/20060222elpepifut_1/Tes/futuro/web/entienda/usuario,

[Russell *et al*; 2006]
Stuart Russell, Peter Norvig, *Inteligencia Artificial. Un enfoque moderno.* Obra original: *Artificial inteligence a modern approach*; Luis Joyanes Aguilar (coordinador del equipo de traducción y revisión técnica) (traductor/es). España: ed. Prentice Hall, 2006. 1211 p. ISBN: 84-205-4003-X.

[Sabou *et al*; 2006]
Marta Sabou, V. Lopez, Enrico Motta y Victoria Uren, *Ontology Selection: Ontology Evaluation on the Real Semantic Web.* Proceedings of the 1. Edinburgh: KMI & Centre for Research in Computing of the Open University, UK., 2006. 8 p. Disponible en Web: http://www2006.org/, http://www2006.org/tracks/semweb.php,

[Wu *et al*; 2006]
X Wu, Zhang, L., and Yu, Y, *Exploring social annotations for the semantic web.* Edinburgh: Proceedings of the 15th International Conference on World Wide Web, 2006. 10 p. Disponible en Web: http://www2006.org/programme/item.php?id=4071

2005

[Abián; 2005]
Miguel Ángel Abián, *El futuro de la web. XML,RDF/RDFS, ontologías y la Web Semántica.* 2005. Disponible en Web: http://www.javahispano.org/

[Alesso *et al*; 2005]
H. Peter Alesso, Craig F. Smith, *Developing Semantic Web Services.* 2005. 445 p. ISBN: 1-56881-212-4.

[Barrón Cedeñoa; 2005]
Alberto Barrón Cedeñoa, Temas selectos del web. on line: 2005. 3 p.

[Carvin; 2005]
Andy Carvin, *Tim Berners-Lee: Weaving a Semantic Web.* 2005. Disponible en Web: http://www.digitaldivide.net/articles/view.php?ArticleID=20

[Charlet *et al*; 2005]
Jean Charlet, Bruno Bachimont and Marie-Christine Jaulent, *Buildingmedical ontologies by terminology extraction from texts: An experiment for the intensive care units.* Paris, Francia: ELSEVIER, 2005.

[Harold *et al*; 2005]
Elliotte Rusty Harold, W. Scout Means, *XML; imprescindible.* Obra original: *XML in a Nutshell*; Patricia Scott Peña (traductor/es). Edición 2005. Madrid: Anaya multimedia, 2005. 831 p. ISBN: 84-415-1812-2.

[Juárez Herrero; 2005]
José Manuel Juárez Herrero, *La Web Semántica . Problemática y arquitectura.*; artículo extraido de la revista "Ingeniería Informática del Colegio de Ingenieros en Informática de la Región de Murcia, publicación trimestral (abril 2005)". Murcia: ed. InforMAS (on line), 2005. Disponible en Web: http://www.cii-murcia.es/informas/abr05/articulos/La_web_semantica_problematica_y_arquiectura.php

[Lima Díaz; 2005]
Felipe Lima Díaz, *Manual Avanzado de Java 2 v5.0.* Madrid: Ed. Anaya, 2005. 494 p. ISBN: 84-415-1946-3.

[Noy *et al*; 2005]
Natalya F. Noy, eborah L. McGuinness, *Desarrollo de Ontologías-101: Guía Para Crear Tu Primera Ontología.* Obra original: *Ontology Development 101: A Guide to Creating Your First Ontology*.; Erick Antezana (traductor/es). Stanford University, 2005. 29 p. Disponible en Web: http://protege.stanford.edu/publications/ontology_development/ontology101-es.pdf, http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html,

[*Reeve et al*; 2005]
Reeve, L. & HAN, H. (2005). *Survey of semantic annotation platforms*.

[Richardson *et al*; 2005]
W. Clay Richardson, Donald Avondolio, Joe Vitale, Scot Schrager, Mark W. Mitchell and Jeff Scanlon, *Profesional Java 2 v5.0.*; José Manuel Gómez Pastor (traductor/es). Ed. Anaya, 2005. 702 p. ISBN: 84-415-1855-6.

[*Sánchez Fernández et al*; 2005]
Luis Sánchez Fernández y Norberto Fernández Garcia, *La Web Semántica: fundamentos y breve "estado del arte" (MONOGRFÍA).* artículo extraído de la revista "Novática: revista de la Asociación de Técnicos de Informática". Núm. 178. Barcelona: ATI, 2005. 6 p. Disponible en Web: http://www.ati.es/novatica/

[Stojanovic; 2005]
Nenad Stojanovic, *Approach for defining relevance in the ontology-based information retrieval*.; artículo extraido de la revista "The 2005 IEEE/WIC/ACM International Conference on Volume , Issue , Page(s): 359 - 365". 19-22 Sept. 2005. University of Karlsruhe. Germany: IEEE, 2005. 7 p. ISBN: 0-7695-2415-X/05. Disponible en Web: http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel5/10179/32502/01517872.pdf?arnumber=1517872

[Valencia García; 2005]
Rafael Valencia García, *TESIS DOCTORAL: "Un Entorno para la Extracción Incremental de Conocimiento desde Texto en Lenguaje Natural"*.; Dr. Rodrigo Martínez Béjar y Dr. Jesualdo Tomás Fernández Breis (director/es). Murcia: UNIVERSIDAD DE MURCIA, Departamento de Ingeniería de la Información y las Comunicaciones, 2005. Disponible en Web: http://www.tdr.cesca.es/TESIS_UM/AVAILABLE/TDR-1123105-140512//rvalencia.pdf

2004

[Antoniou *et al*; 2004]
Grigoris Antoniou, Frank van Harmelen, *A Semantic Web Primer.* Ed. Massachusetts Institute of Technology, 2004. 238 p. ISBN: 0-262-01210-3.

[*Cimiano et al*; 2004]
Cimiano, P., Handschuh, S. & Staab, S, *Towards the self-annotating web* . Proceedings of the 13th World Wide Web Conference.

[Contreras Cino; 2004]
Jesús Contreras Cino, *TESIS DOCTORAL: "Incremento Crítico del Contenido de la Web Semántica mediante Poblado Automático de Ontologías".*; Dra. Ana M. García Serrano y Dr. V. Richard Benjamins (director/es). Madrid. España: UNIVERSIDAD POLITECNICA DE MADRID, Facultad de Informática, 2004. Disponible en Web: http://oa.upm.es/258/01/10200411.pdf

[Corcho *et al*; 2004]
Oscar Corcho, Gómez-Pérez A, Barrasa J, *R2O, an Extensible and Semantically Based Database-to-ontology.* Toronto. Canada: Second Workshop on Semantic Web and Databases (SWDB2004), 2004. 18 p.

[Escribano Alcaide *et al*; 2004]
Raúl Escribano Alcaide, Antonia García Rodríguez, Mónica Julia Alcañiz y Ana Belén Marcos Fernández, *Sistemas de Representación y Procesamiento Automático del Conocimiento. ONTOLOGIAS EN LA WEB SEMÁNTICA.* Facultad de Informática, Universidad Politécnica de Valencia, 2004. Disponible en Web: http://personales.upv.es/ccarrasc/doc/2003-2004/Onto_WS/OntologiasenlaWebSemantica.htm

[Giménez *et al*; 2004]
Jesús Giménez, Lluís Márquez. Obra original: *SVMTool: A general POS tagger generator based on Support Vector Machines.* Barcelona: TALP Research Center, LSI Department. Universitat Polit`ecnica de Catalunya, 2004. 4 p. Disponible en Web: http://www.lsi.upc.edu/~nlp/SVMTool/

[Gómez Pérez *et al*; 2004]
Asunción Gómez Pérez, Mariano Fernández López y Oscar Corcho, *Ontological Engineering.* ed. Springer, 2004. 403 p. ISBN: 1-85233-551-3.

[Gutiérrez Araus *et al*; 2004]
Mª Luz Gutiérrez Araus, Manuel Esgueva Martínez, Mario Garcia-Page Sánchez, Paloma Cuesta Martínez, Ana-Jimena Deza Enríquez, Ángeles Estévez Rodríguez, Mª Antonieta Andino Herrero y Pilar Ruiz-Va Palacios, *Introducción a la Lengua Española.* 1.ª edición. Madrid: Ed. Ramón Areces, 2004. 448 p. ISBN: 84-8004-679-1.

[Hunter *et al*; 2004]
Jane Hunter, John Drennan and Suzanne Little, *Realizing the Hydrogen Economy through Semantic Web Technologies.* IEEE Computer Society, 2004.

[Passin; 2004]
Thomas B. Passin, *Explorer's Guide to the Semantic Web.* USA: Ed. Manning, 2004. 281 p. ISBN: 1-932394-20-6.

[Smith *et al*; 2004]
Michael K. Smith, Chris Welty and Deborah L. McGuinness, *OWL Web Ontology Language Guide.* on line: W3C, 2004. Disponible en Web: http://www.w3.org/TR/owl-guide/

[Valencia García *et al*; 2004]
Rafael Valencia García, Juana María Ruiz Sánchez, Pedro José Vivancos Vicente, Jesualdo Tomás Fernández Breis y Rodrigo Martínez Bejar, *An incremental approach for discovering medical knowledge from texts.* Murcia, España: Departamento de Ingeniería de la Información y las Comunicaciones, Facultad de Informática, Universidad de Murcia., 2004.

2003

[Alani *et al*; 2003]
Harith Alani, Sanghee kim, David E. Millard, Mark J. Weal, Wendy Hall, Paul H. Lewis and Nigel R. Shadbolt, *Automatic Ontology Based Knowledge Extraction from Web Documents.*; artículo extraido de la revista "IEEE Computer Society, january/february 2003". University of Southampton, 2003.

[Atanas *et al*; 2003] Atanas Kiryakov, Borislav Popov, Damyan Ognyanoff, Dimitar Manov, Angel Kirilov and Miroslav Goranov, *Semantic Annotation, Indexing, and Retrieval. Human Language Technologies Workshop at the 2nd International Semantic Web Conference* (ISWC2003); October 2003. Disponible en: http://www.ontotext.com/publications/SemAIR_ISWC169.pdf (3/07/04).

[Bilenko *et al*; 2003]
Mikhail Bilenko, Raymond Mooney, William Cohen, Pradeep Ravikumar and Stephen Fienberg, *Adaptive Name Matching in Information Integration.* IEEE Computer Society, 2003.

[Castells; 2003]
Pablo Castells, *La web semántica.* España: Ediciones de la Universidad de Castilla - La Mancha, 2003. 195 p. ISBN: 84-8427-352-0. Disponible en Web: http://www.ii.uam.es/~castells/publications/castells-uclm03.pdf, http://www.ii.uam.es/~castells/publications/, http://www.ii.uam.es/~castells/

[Chang *et al*; 2003]
Chia-Hui Chang, Harianto Siek, Jiann-Jyh Lu, Chun-Nan Hsu and Jen-Jie Chiou , *Reconfigurable Web Wrapper Agents.* IEEE Computer Society, 2003.

[Criado; 2003]
Luis Criado, *Filtrado de información en Internet basado en roles.* Madrid: UNED, 2003. 23 p. Disponible en Web: http://www.ia.criado.org/

[Cruz *et al*; 2003]
Isabel F. Cruz, Afsheen Rajendran, *Semantic Data Integration in Hierarchical Domains.* IEEE Computer Society, 2003.

[Deaño; 2003]
Alfredo Deaño, *Introducción a la lógica formal.* 3ª reimpresión. Madrid: Alianza Editorial, 2003. 424 p. ISBN: 84-206-8681-6.

[*DILL et al*; 2003]
DILL, S., EIRON, N., GIBSON, D., GRUHL, D., GUHA, R., JHINGRAN, A., KANUNGO, T., MCCURLEY, K.S., RAJAGOPALAN, S., TOMKINS, A., TOMLIN, J.A. & ZIEN, J.Y., *A case for automated large-scale semantic annotation . J. Web Sem.*,

[Fensel *et al*; 2003]
Dieter Fensel, James Hendler, Henry Lieberman and Wolfgang Wahlster; foreword by Tim Berners-Lee, *Spinning the Semantic Web.* Ed. Massachusetts Institute of Technology, 2003. 479 p. ISBN: 0-262-06232-1.

[Gómez Pérez *et al*; 2003]
A. Gómez Pérez, D. Manzano-Macho, *OntoWeb. A survey of ontology learning methods and techniques*, 2003. 86 p

[Kopena *et al*; 2003]
Josepk Kopena, William C. Regli, *DAMLJessKB: A Tools for Reasoning with the Semantic Web.* IEEE Computer Society, 2003.

[Masuoka *et al*; 2003]
Ryusuke Masuoka, Yannis Labrou. Bijan Parsia and Evren Sirin, *Ontology-Enabled Pervasive Computing Applications.* IEEE Computer Society, 2003.

[O'Neill *et al*; 2003]
Edward T. O'Neill, Brian F. Lavoie, Rick Bennett.
Obra original: *Trends in the Evolution of the Public Web.*
Volume 9 Number 4. Web: D-Lib Magazine, 2003. ISBN: ISSN 1082-9873. Disponible en Web: http://www.dlib.org/dlib/april03/lavoie/04lavoie.html,

[Powers; 2003]
Shelley Powers, *Practical RDF. Solving Problems with the Resource Description Framework.* First Edition. USA: O'Reilly, 2003. 350 p. ISBN: 0-596-00263-7.

[Schwartz; 2003]
David G. Schwartz, *From Open IS Semantics to the Semantic Web: The Road Ahead.* IEEE Computer Society, 2003.

[Schweiger *et al*; 2003]
Ralf Schweiger, Simon Hoelzer, Dirk Rudolf, Joerg Rieger and Joachim Dudeck
, *Linking clinical data using XML topic maps.* ELSEVIER, 2003.

[Shamsfard *et al*; 2003]
Mehrnoush Shamsfard, Ahmad Abdollahzadeh Barforoush, *The state of the art in ontology learning: a framework for comparison.*; artículo extraido de la revista "The Knowledge Engineering Review.". Volume 18 , Issue 4. New York, NY, USA: Cambridge University Press, 2003. 24 p.

Anterior a 2003

[Carreras Riudavets; 2002]
Francisco Carreras Riudavets, *TESIS DOCTORAL: "Sistema Computacional de Gestión Morfológica del Español (SCOGEME)"*.; Dr. Octavio Santana Suárez y Dr. José R. Pérez Aguiar (director/es). Universidad Las Palmas de Gran Canaria, 2002. 387 p. Disponible en Web: http://www.gedlc.ulpgc.es/~paco/scogeme.pdf, http://www.gedlc.ulpgc.es/docencia/doctorado/tespaco.html,

[Castells; 2002]
Pablo Castells Madrid: Escuela Politécnica Superior. Universidad Autónoma de Madrid, 2002.

[Fallside; 2002]
David C. Fallside, *Esquema XML Parte 0: Fundamentos.* Obra original: *XML Schema Part 0: Primer.*; Jose Manuel Alonso. (traductor/es). on line: W3C, 2002. Disponible en Web: http://www.w3c.es/Traducciones/es/TR/2001/REC-xmlschema-0-20010502/

[Haustein *et al*; 2002]
S. Haustein, J. Pleumann, *Is Partici-pation in the Semantic Web too Difficult?.* on line. Cerdeña, Italia: International Semantic Web Conference (ISWC'2002), 2002. Disponible en Web: http://iswc2002.semanticweb.org/

[Vargas-Vera *et al*; 2002]
Maria Vargas-Vera, Enrico Motta, John Domingue, Mattia Lanzoni, Arthur Stutt and Fabio Ciravegna, *The 13th International Conference on Knowledge Engineering and Management.*; artículo extraido de la revista "EKAW 2002". Springer Verlag, 2002. 14 p. Disponible en Web: http://kmi.open.ac.uk/projects/akt/MnM/

[Aranda Almansa *et al*; 2001]
Joaquín Aranda Almansa, José Luis Fernández Marrón, José Jiménez González y Fernando Morilla García, *Fundamentos de Lógica Matemática.* 2ª reimpresión. Madrid: ed. Sanz y Torres S.L., 2001. 321 p. ISBN: 84-88667-45-0.

[Bergman; 2001]
Michael K. Bergman Obra original: *The Deep Web: Surfacing Hidden Value.* vol. 7, no. 1. Journal of Electronic Publishing, 2001. Disponible en Web: http://quod.lib.umich.edu/cgi/t/text/text-idx?c=jep;view=text;rgn=main;idno=3336451.0007.104, http://hdl.handle.net/2027/spo.3336451.0007.104,

[Berners-Lee *et al*; 2001]
Tim Berners-Lee, James Hendler and Ora Lassila, *The Semantic Web.* on line: Scientific American, 2001. Disponible en Web: ttp://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21

[Brickley *et al*; 2001]
Dan Brickley, R.V. Guha, *Resource Description Framework. (RDF) Especificación del Esquema 1.0.* Obra original: *Resource Description Framework (RDF) Schema Specification 1.0.*; Eva Méndez (traductor/es). on line: W3C, 2001. Disponible en Web: http://www.sidar.org/recur/desdi/traduc/es/rdf/rdfsch.htm

[de Diego; 2001]
Fernando de Diego, *Nuevo método de Esperanto.* 3ª edición. Madrid: Gram Ediciones, 2001. 221 p. ISBN: 84-88519-07-9.

[Gruber; 2001]
T. R. Gruber, *What is an Ontology?.* Stanford: Knowledge Systems Laboratory. Stanford University, 2001. Disponible en Web: http://www-ksl.stanford.edu/kst/what-is-an-ontology.html

[*Kogut et al*; 2001]
P. Kogut, W. Holmes., *AeroDAML: applying information extraction to generate DAML annotations from web pages,* in: Proceedings of the Workshop on Knowledge Markup and Semantic Annotation at 1st International Conference on Knowledge Capture (K-CAP 2001), Victoria, B.C., Canada, 2001.

[Lassila *et al*; 2001]
Ora Lassila, Ralph R. Swick, *Resource Description Framework(RDF). Especificación del Modelo y la Sintaxis.* Obra original: *Resource Description Framework (RDF) Model and Syntax Specification.*; Eva Méndez (traductor/es). W3C. on line: 2001. Disponible en Web: http://www.sidar.org/recur/desdi/traduc/es/rdf/rdfesp.htm

[León Fariña; 2001]
María del Amor León Fariña, *PROYECTO FIN DE CARRERA: SISTEMA INTELIGENTE DE REGISTRO EN BUSCADORES.*; Jesús Sánchez Allende (director/es). UNIVERSIDAD ALFONSO X EL SABIO. ESCUELA POLITÉCNICA SUPERIOR INGENIERÍA DE TELECOMUNICACIÓN, 2001. 147 p.

[Mira Mira *et al*; 2001]
José Mira Mira, Ana Esperanza Delgado, Jesús González Boticario, Francisco Javier Díez, *Aspectos básicos de la Inteligencia Artificial.* 1ª reimpresión. Madrid: ed. Sanz y Torres, S.L., 2001. 604 p. ISBN: 84-88667-13-2.

[Wuwongse *et al*; 2001]
Vilas Wuwongse, Chutiporn Anutariya, Kiyoshi Akama and Ekawit Nantajeewarawat , *XML Declarative Description: A Language for the Semantic Web.* IEEE Computer Society, 2001.

[Froufe; 2000]
Agustín Froufe, *Java 2. Manual de usuario y tutorial.* 2ª edición. Madrid: Ed. Ra-Ma, 2000. 682 p. ISBN: 84-7897-429-6.

[Moreno Ortiz; 2000]
Antonio Moreno Ortiz, *WordNet.* on line: 2000. ISBN: ISSN: 1139-8736, Depósito Legal: B-35510-2000. Disponible en Web: http://elies.rediris.es/elies9/2-4-2.htm

[Beza *et al*; 1999]
Ricardo Baeza Yates y Berthier Ribeiro Neto, *Modern Information Retrieval.* ACM Press. Addison-Wesley. 1999. 501 p.

[Zukowski; 1999]
John Zukowski, *Programación en Java 2.*; Carlos Anaya Álvarez (traductor/es). Edición española. Madrid: ed. ANAYA, 1999. 798 p. ISBN: 84-415-0948-4.

[Berners-Lee; 1998]
Tim Berners-Lee, *What the Semantic Web can represent.* on line: 1998. Disponible en Web: http://www.w3.org/DesignIssues/RDFnot.html

[Berners-Lee 1; 1998]
Tim Berners-Lee 1, *Semantic Web Road map.* 1998. Disponible en Web: http://www.w3.org/DesignIssues/Semantic.html

[Ferreira *et al*; 1998]
Anita Ferreira, John Atkinson, *UN MODELO DE AGENTE DE BÚSQUEDA Y FILTRADO DE INFORMACIÓN INTELIGENTE APOYADO POR INTERACCIONES EN LENGUAJE NATURAL.*; artículo extraido de la revista "FACULTAD DE INGENIERIA, U.T.A. (CHILE), VOL. 5,". Chile: 1998.

[Clocksin *et al*; 1993]
W.F. Clocksin, C.S. Mellish, *Programación en Prolog.* Obra original: *Programming in Prolog.* 2ª edición. Barcelona: Gustavo Gili, S.A., 1993. 270 p. ISBN: 84-252-1339-8.

[Gruber; 1993]
Thomas Gruber, *A Translation Approach to Portable Ontology Specifications.* California: KNOWLEDGE SYSTEMS LABORATORY. Computer Science Department, 1993. 27 p. Disponible en Web: http://tomgruber.org/writing/ontolingua-kaj-1993.pdf

[Miller; 1986]
George Miller, *Dictionaries in the Mind.* USA: Department of Psychology. Princeton University, 1986. 10 p. Disponible en Web: http://acl.ldc.upenn.edu/P/P85/P85-1038.pdf

Referencias de interés

[1] Semantic Web Architecture

http://www.w3.org/2001/09/21-orf/hagino-sw/slide8-0.html

[2] OWL-QL Projectfor the Stanford Knowledge Systems Laboratory

http://ksl.stanford.edu/projects/owl-ql/

[3] ProductsSemanticWorks SemanticWorks Semantic Web tool - Visual RDF and OWL editor

http://www.altova.com/products/semanticworks/semantic_web_rdf_owl_editor.html

[4] Enlace histórico sobre la gestión de la información que planteaba Tim Berners-Lee entre 1989-1990

http://www.w3.org/History/1989/proposal.html

[5] An Introduction to RDF and the Jena RDF API

http://jena.sourceforge.net/tutorial/RDF_API/index.html

[6] Tim Berners-Lee

http://www.w3.org/People/Berners-Lee/

[7]

http://tomgruber.org/

[8] Tom Gruber's Space on the Web

http://tomgruber.org/

[9] OWL API

http://sourceforge.net/projects/owlapi

[10]

http://www.openrdf.org/

[11] Proporciona una capa de inferencia de alto rendimiento para el tratamiento de OWL basado en SESAME.

http://www.ontotext.com/owlim/index.html

[12] Jastor, Ontology Driven RDF Access from Java

http://jastor.sourceforge.net/

[13] NeOn

http://www.neon-project.org/web-content/,
http://www.atosorigin.es/noticias/2006_noticia_16.htm,
http://www.bureaudeprensa.com/es/view.php?bn=bureaudeprensa_software&key=1148391976&pattern=

[14] coordinador del proyecto NeOn

http://kmi.open.ac.uk/people/motta/

[15] The Open University

http://www3.open.ac.uk/about/

[16] Especificación de RSS 2,0

http://blogs.law.harvard.edu/tech/rss,
http://www.w3c.es/Divulgacion/Guiasbreves/WebSemantica,

[17] FOAF

http://www.foaf-project.org/,

http://f14web.com.ar/inkel/2003/01/27/foaf.html,
http://www.w3c.es/Divulgacion/Guiasbreves/WebSemantica

[18] Semantic Web Search

http://www.semanticwebsearch.com/

[19] Webposible: Microformatos Dublin Core

http://www.webposible.com/microformatos-dublincore/

[20] Marco Schorlemmer

http://documenta2.blogsome.com/category/buscadores/, http://www.iiia.csic.es/~marco/,

[21] Alvaro Graves
http://wordnet.princeton.edu/~agraves/

[22] School of Electronics and Computer Science
http://beta.mspace.fm/,
http://www.ecs.soton.ac.uk/about/,

[23] Consorcio World Wide Web (W3C): oficina española
http://www.w3c.es/, http://www.w3c.es/Divulgacion/Guiasbreves/WebSemantica,

[24] Google Labs: el laboratorio tecnológico de Google
http://labs.Google.com/

[25] Latin Extended-A Range: 0100-017F
http://www.unicode.org/

[26] Taller de Programación (Pascal), sobre la notación EBNF
http://www.geocities.com/v.iniestra/apuntes/tall_prog/

[27] Epsilon Eridani - XML - especificación 1.0
http://www.epsilon-eridani.com/docs/disenoweb/xml/index.html

[28] Tutorial en castellano sobre XML y tecnologías relacionadas: XSL, XQL, RDF, DOM, XLINK, XPOINTER, DSSSL, etc
http://www.programacion.net/html/xml/principal.htm

[29] DTDs y XML Esquema
http://www.hipertexto.info/documentos/dtds.htm

[30] Resource Description Framework (RDF) / W3C Semantic Web Activity
http://www.w3.org/RDF/

[31] Wikipedia
http://es.wikipedia.org/wiki/Web_sem%C3%A1ntica

[32] W3C, World Wide Web Consortium

http://www.w3.org/2001/sw/Europe/events/200406-esp/trabajo-final-extratesauros/node5.html

[33] W3C, World Wide Web Consortium

http://www.w3c.es/Prensa/2004/nota040210.html

[34] EL NUEVO CONCEPTO DE DOCUMENTO EN LA CULTURA DE LA IMAGEN

http://www.hipertexto.info/documentos/owl.htm

[35] Blog sobre avances tecnológicos

http://www.euroresidentes.com/Blogs/avances_tecnologicos/2005_03_01_archive.html

[36] WordNet

http://wordnet.princeton.edu/

[37] Memex de Vannevar Bush. Pionero del Hipertexto

http://www.lacoctelera.com/lapalestra

[38] Breve historia de la World Wide Web

http://html.conclase.net/articulos/historia

[39] Historia de los servicios web

http://www.desarrolloweb.com/articulos/1883.php?manual=61

[40] La Web Semántica: una visión crítica

http://www.biomeds.net/biomedia/d02030203.htm

[41] Buscador de Ontologías

http://swoogle.umbc.edu/

[42] Buscador Semántico Zaragoza

http://www.isoco.com/noticias/06/06_07_12.html,
http://www.zaragoza.es:82/buscadorsemantico/,

[43] Editor Ontológico Protégé

http://protege.stanford.edu/

[44] JENA: framework for building Semantic Web applications
http://jena.sourceforge.net/

[45] DataLexica
http://www.bitext.com/datalexica.asp

[46] EuroWordNet
http://www.illc.uva.nl/EuroWordNet/

[47] SWSE Semantic Web Search Engine
http://www.swse.org/, http://swse.deri.org/,

[48] Buscador Semántico: sindice
http://sindice.com

[49] Buscador Semántico Powerset
http://www.powerset.com/

[50] Buscador Semántico “True Knowledge”
http://www.trueknowledge.com/

[51] Buscador Semántico: Hakia
http://www.hakia.com/

[52] Buscador Semántico: Watson
http://watson.kmi.open.ac.uk/WatsonWUI/

[53] Buscador Semántico: Falcons
http://iws.seu.edu.cn/services/falcons/objectsearch/index.jsp

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

[55] Gleaning Resource Descriptions from Dialects of Languages (GRDDL)
http://www.w3.org/2001/sw/grddl-wg/,
http://www.w3.org/TR/grddl/, http://www.w3.org/TR/grddl-tests/

[56] Netcraft

http://news.netcraft.com/

[57] Nova Spivack, Radar Networks

http://www.radarnetworks.com/

[58] Blekko

Rich Skrenta (rich@skrenta.com)

http://www.skrenta.com/about.html

http://www.skrenta.com/2008/01/why_search.html

http://translate.google.com/translate?u=http%3A%2F%2Fwww.skrenta.com%2F2008%2F01%2Fwhy_search.html&hl=es&ie=UTF8&sl=en&tl=es

[59] Xanadu & Ted Nelson

http://www.xanadu.com.au/ted/

[60] primeras especificaciones DAML+OIL

http://www.daml.org/2000/12/daml+oil-index.html

[61] articulo crítico: La Web Semantica está muerta

http://yahooresearchberkeley.com/blog/2007/05/16/the-emerging-semantics-web-the-semantic-web-is-dead/

[62] The Nielsen Company, ranking buscadores

http://www.marketwire.com/press-release/The-Nielsen-Company-812173.html

[63] 16th Internacional World Wide Web Conference - May 8-12, 2007 - Alberta,CANADA

http://www2007.org/

[64] SWOOP, navegador OWL desarrollado por el grupo MINDSWAP de la Universidad de Maryland http://www.mindswap.org/2004/SWOOP/downloads/

[65] CRISOL: Una aproximación a la generación automática de instancias para una Web Semántica

http://www3.uji.es/~berlanga/Chronology/jisbd2003_rox.pdf

[66] Técnicas y Lenguajes para la Representación del Conocimiento (Patxi Echart)

http://www.eslomas.com/index.php/archives/2006/12/14/tecnicas-y-lenguajes-para-la-representacion-del-conocimiento/

[67] MORFEO: Experimento semántico para definir contextos y recursos

https://forge.morfeo-project.org/wiki/index.php/D_2.4_Experimento_sem%C3%A1ntico_para_definir_contextos_y_recursos

[68] razonador DL KAON2

http://kaon2.semanticweb.org/

[69] razonador DL PELLET

http://pellet.owldl.com/

[70] El proyecto Ask-It

http://www.ask-it.org/

[71] Ontologias de Ask-It

http://askit.iti.gr/ontology/

[72] tipos de anotación semantica. Resultados del análisis del proyecto RODA

http://roda.ibit.org/herramientas.cfm

[73] anotación MnM

http://kmi.open.ac.uk/projects/akt/MnM/index.html

[74] anotación Ontoprise

http://www.ontoprise.de/products/ontoannotate

[75] anotación OntoMat-Annotizer

http://annotation.semanticweb.org/tools/ontomat/index.html

[76] anotación SHOE

http://www.cs.umd.edu/projects/plus/SHOE/KnowledgeAnnotator.html

[77] anotación SMORE

http://www.mindswap.org/~aditkal/editor.shtml

[78] anotación Melita
http://nlp.shef.ac.uk/melita/

[79] anotación GATE
http://gate.ac.uk/

[80] anotación COHSE
http://cohse.semanticweb.org/

[81] anotación Annotea
http://www.w3.org/2001/Annotea/

[82] anotación Annozilla
http://annozilla.mozdev.org/

[83] anotación Yawas
http://www.univ-savoie.fr/labos/syscom/Laurent.Denoue/yawas/

[84] Annotation System
http://www.ncb.ernet.in/groups/dake/annotate/details.shtml#system

[85] Anotación Trellis Web
http://www.isi.edu/index.php

[86] Euroworksafe
http://www.euroworksafe.org/

[87] Sitios web que usan los estándares de W3C

http://www.elgrupoinformatico.com/las-webs-cumplen-estandar-w3c-segun-opera-t4257.html, http://www.diarioti.com/gate/n.php?id=19905

[87] SEKT
http://www.sekt-project.com

[88] Dot.Kom
http://kmi.open.ac.uk/projects/dotkom/

[89] X-Media
http://www.x-media-project.org

[90] Abraxas
http://nlp.shef.ac.uk/abraxas/

[91] Ontology Learning
http://semanticweb.org/wiki/KWTR:_Ontology_Learning

[92] Sistema Amilcare
http://www.dcs.shef.ac.uk/~fabio/Amilcare.html

[93] Armadillo
http://www.dcs.shef.ac.uk/~sam/armadillo.html

[94] AKTive Media
http://www.dcs.shef.ac.uk/~ajay/html/cresearch.html

Anexo A: Instalación de las herramientas implementadas.

El objetivo de este apéndice es presentar todas las herramientas desarrolladas en esta tesis para permitir a cualquier investigador comprobar la viabilidad de todas las propuestas expuestas, así como reproducir todos los resultados que se obtienen.

A.1. Herramientas implementadas.

Se ha implementado la librería "terica.jar" que contiene varios subpaquetes, base de nuestro sistema, comentamos los grupos más significativos:

terica.file: Utilidad para buscar ficheros. Lo utilizamos, por ejemplo, para buscar todos los ficheros HTML que tenemos que procesar. De manera, que conociendo el directorio local del sitio web se procesan todas las páginas que la componen.

terica.firmas: Contiene dos funciones hash (md5 y sha). Estas funciones tienen como propiedad que dada una entrada; siendo esta una cadena, un párrafo o un fichero de cualquier tipo y tamaño, se obtiene una salida única con una probabilidad muy alta. Además las funciones hash solo operan en un sentido, por lo que no es posible obtener la entrada a partir de la salida. Usamos estas técnicas para solucionar dos cuestiones:

1. Como mecanismo para detectar modificaciones en el contenido de la web, ya que en

este caso hay que volver a realizar las anotaciones semánticas adecuadas, es decir, si una página web es modificada habrá que modificar la *vista semántica* asociada

2. Para la definición de URI

terica.jena: Conexión con las librerias de jena, que permiten insertar las *vistas semánticas* en la base de datos.

terica.pln: Compuesto por varias clases que permiten abordar las fases de identificación y extracción por completo y la generación de *vistas semánticas*.

En base a la mencionada librería y a las librerias de Jena hemos implementado una herramienta gráfica, denominada "sw2sws" que permite transformar un sitio web en un sitio web semántico mediante el procedimiento propuesto y descrito en esta tesis, es decir, contempla las etapas de identificación, extracción e interpretación. Finalmente hemos implementado un prototipo de buscador semántico de propósito general que utiliza las *vistas semánticas* generadas con "sw2sws".

A.2. Descarga de herramientas.

Todo el código fuente ha sido probado en Java (version 1.5.0_06) y está disponible, junto con la versión ejecutable, en tres archivos comprimidos en: http://descargas.criado.org/. Para acceder a la descarga seleccione la opción "tesis/herramientas" del menú e introduzaca estos datos de identificación:

Login: lcriadof

Password: Sw2sws2009

Una vez que este autentificado tendrá opción de descargar:

- Librería terica.jar y todos las herramientas de pruebas (archivo procesos_sw2sws.rar)
- Herramienta para transformar su sitio web en un sitio web semántico (sw2sws)
- Demo de Buscador Semántico VISSEM.

Seguidamente se detalla el contenido de cada paquete:

procesos_sw2sws.rar: Contiene la librería "terica.jar" y todos las herramientas de pruebas. Una vez descargado y descomprimido se dispondrá de cuatro directorios principales. Estos son:

- **.\apache-ant-1.6.5**

  Distribución de la herramient ANT para realizar comodamente la tarea de compilación (construcción-build), que a su vez se divide en otros cuatro directorios (bin,docs,etc y lib)

- **.\src**

  Directorios con cada funcionalidad necesaria para probar cada proceso necesario para la transformación de Web a Web Semántica:

  1_ExtraerPalabrasOntologia

  2_MapeoEntreDosIdiomas

  3_GramaticaEsperanto

  4_GenerarMapeoDeUnaOntologia

  5_ExtraerContenido

  7_GenerarVistaSemantica

  8_identificarSitioWeb

  9_html2pl

10_pl2vistasSemanticas

11_crearOntologiaEnBBDD

12_crearInstanciasEnBBDD

13_urlPOST

- **.\lib**

  Librerias necesarias para la compilación. El fichero "terica.jar" ha sido desarrollado para abordar las fases de identificación, extracción e interpretación. Este "jar" es utilizado para la construcción de la herramienta semi-automática que permite transformar un sitio web en un sitio web semántico.

Sw2sws.rar: Contiene la herramienta que permitirá a cualquier usuario activo transformar su web en un sitio web semántico. Se trata de una herramienta gráfica, funciona con las cinco ontologías mencionadas en este trabajo. Una vez descomprimido, se obtiene la siguiente jerarquia de directorios:

- .\lib
  - Librerias jena y terica
- .\src
  - Programa fuente
- .\dist
  - En este directorio esta el ejecutable. Si tiene como sistema operativo Windows bastará hacer doble clic sobre html2ws.jar

Vissem.rar: En este fichero se encuentra un demo de Buscador Semántico de propósito general basado en el concepto de *vistas semánticas*. Esta aplicación la hemos instalado y probado sobre un servidor Tomcat versión 6.0 con una base de datos Oracle xe 10g.

DIRECTORIOS DEL SERVIDOR:

----index_archivos

----servlets

----logs

----WEB-INF

----classes

----lib

----oracle

----driver10_2_0_1_0

----driver8_1_7

En el directorio “classes” estan los servlets y un fichero por lotes para compilar y copiar al directorio donde se ejecutarán por el servidor Tomcat, que es el directorio denominado “servlets”. Todas las librerias necesarias están en el directorio “lib”.

Si desea usar las webs de ejemplo que se han utilizado en la tesis para la comprobación de resultados puede bajarlas accediendo a la url: http://descargas.criado.org/. Seleccione la opción “tesis/Webs de ejemplo” del menú e introduzaca estos datos de identificación

usuario:
lcriadof@yahoo.es
clave:
Sw2sws2009

A.3. Cómo añadir una nueva ontología en sw2sws.

Para añadir una nueva ontología a la herramienta sw2sws es preciso realizar dos tareas previas:

1. Extracción de todas las clases y propiedades
2. Traducción de las clases y propiedades al lenguaje natural con el que se va a describir el contenido

A.3.1. Extracción de todas las clases y propiedades

Para poder realizar el proceso de identificar un contenido con respecto una ontología lo primero que necesitaremos es un archivo que contenga todos los identificadores de clases y propiedades de cada ontología.

Esta funcionalidad puede verificarse de la siguiente forma:

PARA COMPILAR:

1. Situarse en el directorio .\ procesos_sw2sws\apache-ant-1.6.5\bin
2. Ejecutar: **ant exe1**

CONFIGURACIÓN NECESARIA

1. Fichero: ontologia.properties

PARA EJECUTAR:

1. Situarse en el directorio

   .\ procesos_sw2sws\distribucion\ExtraerPalabrasOntologia

2. Ejecutar: **run**

Pero antes de ejecutar el programa deberemos realizar adecuadamente la configuración:

Configurar el fichero "ontologia.properties"

Este fichero contiene la referencia a la ontología de la que queremos extraer la lista de clases y propiedades, es muy importante comprobar en el navegador que la URL que se asigna en el atributo "ontologia.fuente" responde con un fichero en formato XML sin hacernos ninguna pregunta.

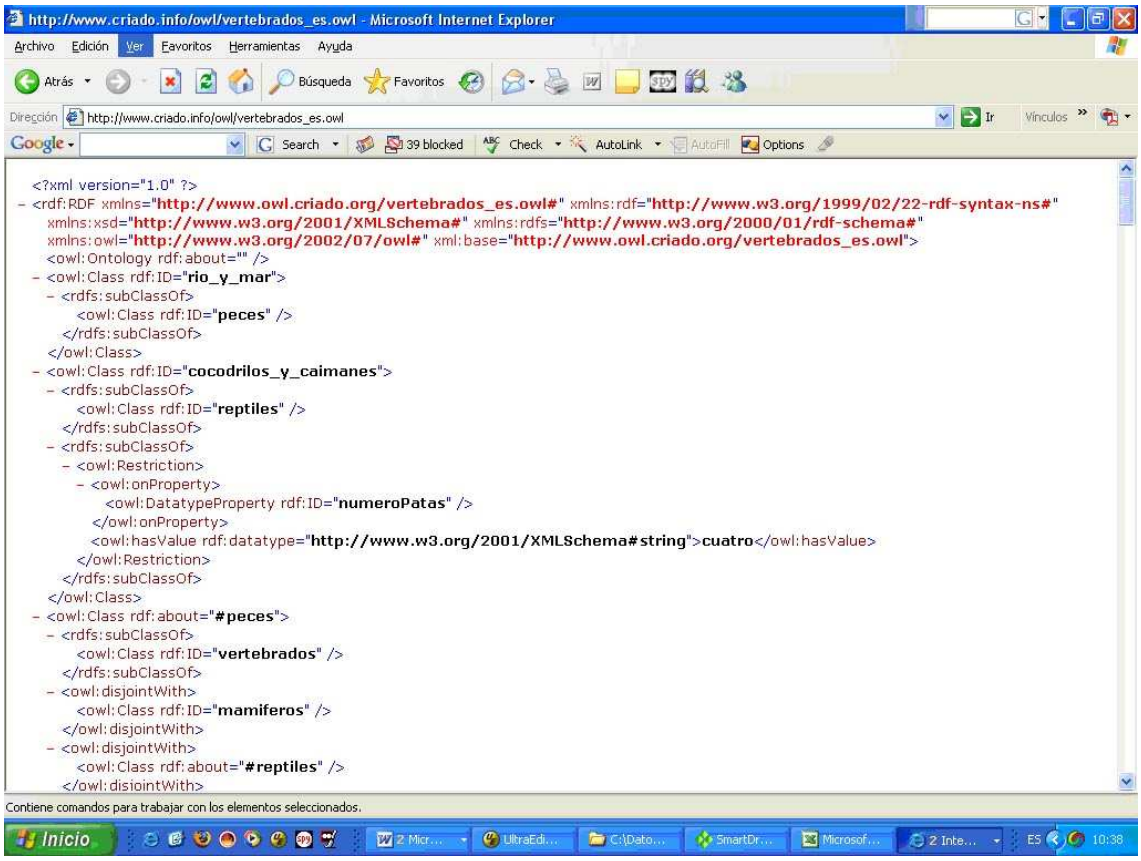


Ilustración 1: ontología en OWL

Un ejemplo de contenido de este fichero puede ser por ejemplo:

ontologia.fuente = http://www.criado.info/owl/vertebrados_es.owl
ontologia.namespace = http://www.owl.criado.org/vertebrados_es.owl#
ontologia.idioma = español
ontologia.ficheros.palabras.clave= claves1.xml

Otro ejemplo:

ontologia.fuente = http://reliant.teknowledge.com/DAML/Government.owl
ontologia.namespace = http://reliant.teknowledge.com/DAML/Government.owl#
ontologia.idioma = ingles
ontologia.ficheros.palabras.clave = claves2.xml

Si hemos descargado en nuestro ordenador la ontología podemos configurar el fichero de la siguiente forma:

ontologia.fuente = file:./pizza.owl
ontologia.namespace = http://www.co-ode.org/ontologies/pizza/2005/05/16/pizza.owl
ontologia.idioma = ingles
ontologia.ficheros.palabras.clave = claves_pizza.xml

Ejecutar el proceso de extraer palabras de la ontologia

Abierta una consola y situados en el directorio de ejecución

(.\ procesos_sw2sws\distribucion\ExtraerPalabrasOntologia), teclearemos “run” y se ejecutará un archivo por lotes que invocará al programa java

El programa generará un fichero de salida con el nombre declarado en el atributo “ontologia.ficheros.palabras.clave” que contiene todas las palabras clave de la ontología y que será necesario traducir al castellano.

El fichero resultante tiene el siguiente aspecto:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<ontologia url="file:./pizza.owl" uri="http://www.co-
ode.org/ontologies/pizza/2005/05/16/pizza.owl#">
  <idioma  origen="ingles">
      <termino origen="cajun"/>
```

```
        <termino origen="rosa"/>
        <termino origen="peperoni"/>
        <termino origen="sausage"/>
        <termino origen="topping"/>
…. ….. ……
        <termino origen="olive"/>
        <termino origen="fruit"/>
</idioma>
</ontologia>
```

A.3.2. Traducción de identificadores al lenguaje del contenido

Una vez obtenido el fichero de identificadores de clases y propiedades, para poder realizar el proceso de identificar un contenido con respecto una ontología necesitaremos un archivo que contenga todos los identificadores de clases y propiedades, en el idioma del contenido, de cada ontología con la que se comparará el contenido que se desea clasificar.

Esta etapa requiere también del uso de un diccionario. De manera, que hemos preparado tres diccionarios que utilizaremos según nuestras necesidades. Estos diccionarios han sido expresados en XML y son los siguientes:

1. de inglés a español
2. de inglés a esperanto
3. de español a esperanto

La razón de que sean de inglés a otro idioma es que casi todas las ontologías están escritas en inglés. Todos los diccionarios tienen el siguiente formato:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<!-- diccionario esperanto-ingles -->
<idioma origen="INGLES" destino="ESPERANTO">
…..
        <termino origen="pizza" destino="pico"/>
        <termino origen="pizzeria" destino="picejo"/>
        ….
</idioma>
```

Esta funcionalidad puede verificarse de la siguiente forma:

PARA COMPILAR:

- Situarse en el directorio .\ procesos_sw2sws\apache-ant-1.6.5\bin
- Ejecutar: **ant exe4**

CONFIGURACIÓN NECESARIA

- Fichero: generar.properties

  Este fichero contiene la referencia al diccionario con el cual se realizará la traducción, el fichero que contiene todos los nombres de clases y propiedades de la ontología que se desea traducir y por último, el nombre del fichero resultante que contendrá cada identficador de clase y propiedad traducido al idioma determinado por el diccionario.

  Un ejemplo de contenido de este fichero puede ser:

```
diccionario = diccionarioInglesEsperanto.xml
clases.propiedades = claves_pizza.xml
mapeo.ontologico = mapeo_pizza.xml
```

otro ejemplo, puede ser

diccionario = diccionarioInglesSpain.xml
clases.propiedades = claves_pizza.xml
mapeo.ontologico = mapeo_pizza_es.xml

PARA EJECUTAR:

- Situarse en el directorio

  .\ procesos_sw2sws\distribucion\ GenerarMapeoDeUnaOntologia

- Ejecutar: **run**

El resultado generará un fichero como el siguiente:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<ontologia url="file:./wine.owl" uri="http://www.w3.org/2001/sw/WebOnt/guide-src/wine#">
<idioma origen="CASTELLANO" destino="INGLES">
  <termino origen="dulce" destino="sweet"  esclase="si" instanciablecon="SweetWine" />
  <termino origen="blanco" destino="white"  esclase="si" instanciablecon="WhiteWine" />
  <termino origen="azúcar" destino="sugar"  esclase="si" instanciablecon="WineSugar" />
  <termino origen="rosa" destino="rose"  esclase="si" instanciablecon="RoseWine" />
  <termino origen="uva" destino="grape"  esclase="si" instanciablecon="WineGrape" />
  ....
</idioma>
</ontologia>
```

El elemento [idioma] posee atributos para indicar el idioma origen de la ontología y el idioma que se encontrará en el contenido a identificar. Cada elemento [termino] especifica como transformar una palabra, indicando si representa una clase y el nombre de la clase de la ontología que se utilizará para realizar la instancia.

Para tratar este fichero resultante se ha implementado una clase pública denominada “mapeoIdiomas” que se incorpora en el paquete “terica.pln” y que trata por completo la estructura. Esta clase proporciona métodos para localizar el URI de la ontología [*getUri()*], conocer el nombre de la clase para instanciar [*getClaseInstancia()*], traducir un término [*getTraducción()*], obtener la lista de términos que se soportan para su traducción [*getOrigen()*], etc….

Si desea usar el fichero generado anteriormente, puede hacerlo de la siguiente forma:

- Copie el fichero resultante XML con el nombre de mapeo.xml al directorio

  .\ procesos_sw2sws\src\ MapeoEntreDosIdiomas

- Compile de la siguiente forma:

  Situarse en el directorio .\ procesos_sw2sws\apache-ant-1.6.5\bin

  Ejecutar: **ant exe2**

- Para ejecutar:

  Situarse en el directorio

  .\ procesos_sw2sws\distribucion\ MapeoEntreDosIdiomas

  Ejecutar: **run**

A.4. Proceso de identificación de un sitio web.

Los epígrafes anteriores muestran cómo construir los ficheros necesarios para poder abordar el proceso de identificación. Una vez que se logra obtener un mapeo entre los identificadores de las ontologías (en su idioma original) y el idioma del contenido se puede comprobar, página por página web, si aparecen estos identificadores. Todo este proceso se ha empaquetado en una clase denominada “identificarSitioWEB” que está en el paquete “terica.pln”.

Se puede probar fácilmente el proceso de la siguiente manera:

PARA COMPILAR:

- Situarse en el directorio .\ procesos_sw2sws\apache-ant-1.6.5\bin
- Ejecutar: **ant exe8**

PARA EJECUTAR:

- Situarse en el directorio
    - .\ procesos_sw2sws\distribucion\ identificarSitioWeb
- Ejecutar: **run [path del sitio web]**

A.5. Extracción de Lenguaje Natural y su representación formal en XML.

En los capítulos 2 y 3, se detalló el método de identificación por raíz, que era sin duda el mejor de las tres opciones propuestas. Sin embargo, era apropiado para lenguajes como el esperanto, pero difícil de aplicar a idiomas como el castellano o el inglés. Durante bastante tiempo, esta tesis se basó en el esperanto para todos los prototipos, ya que con el esperanto los resultados que se obtienen son mucho más espectaculares, incluso hemos implementado un pequeño módulo morfológico en esperanto que puede compilarse (ant exe3) y ejecutarse mediante el fichero por lotes “run” en:

.\ procesos_sw2sws\distribucion\ GramaticaEsperanto

Una vez se decidió que la tesis deberia reproducir lo mejor posible la situación actual, hemos tenido que implementar un pequeño simulador de procesador de lenguaje natural, que es muy limitado y solo sirve para pruebas de laboratorio. Pero era necesario disponer de este elemento para poder completar nuestra propuesta. La clase “html2ln” del paquete “terica.pln” permite generar, a partir de todas las páginas HTML de un sitio web, un fichero en Lenguaje Natural (LN) asociado a cada pagina HTML expresa en formato XML (esta clase genera internamente también un fichero plano por cada HTML en el que se filtran todos los TAGS que es previo a generar el fichero de LN).

Se puede probar el proceso de la siguiente forma:

PARA COMPILAR:

- Situarse en el directorio .\ procesos_sw2sws\apache-ant-1.6.5\bin

- Ejecutar: **ant exe9**

PARA EJECUTAR:

- Situarse en el directorio .\ procesos_sw2sws\distribucion\ html2pl
- Ejecutar: **run**

El fichero por lotes lanza la orden:

java -cp .\;.\terica.jar;.\probar.jar probar proyecto1.ide

donde el fichero “proyecto1.ide” contiene la referencia a las páginas HTML que deben ser tratadas para extraer de ellas lenguaje natural expresado formalmente en XML. Para localizar estos ficheros expresados formalmente en lenguaje natural, compruebe el fichero resultado de la ejecución “proyecto1.ide” , en el cual se indicarán los ficheros de entrada y de salida, como puede verse en este ejemplo:

```
<ln>
<fichero ontologia="VERTEBRADOS"
html="C:\Datos2006\codigoFuenteTESIS\tesis\sitiosWeb\Perrilandia\Perrilandia\whippet\index.htm"
ln="C:\Datos2006\codigoFuenteTESIS\tesis\sitiosWeb\Perrilandia\Perrilandia\whippet\index.htm.VERTEBRADOS.LN.xml"/>
<fichero ontologia="VERTEBRADOS"
html="C:\Datos2006\codigoFuenteTESIS\tesis\sitiosWeb\Perrilandia\Perrilandia\wolfhound\index.htm"
ln="C:\Datos2006\codigoFuenteTESIS\tesis\sitiosWeb\Perrilandia\Perrilandia\wolfhound\index.htm.VERTEBRADOS.LN.xml"/>
</ln>
```

El proceso de transformación utiliza el fichero “oracion_simple.xml” que define la gramática de la oración simple y el resultado en lenguaje natural expresado formalmente tiene el siguiente aspecto:

<oracion sujeto="Whippet" verbo="es" CD="perro" CI=" " origen="El Whippet es un perro de distancias cortas "/>

El elemento <oración> tiene cinco atributos, siendo el atributo [origen] la frase original extraida

de forma automática, el resto de atributos representan las funciones sintácticas básicas de la frase; sujeto, verbo, complemento directo y complemento indirecto.

Hasta aquí se ha descrito el proceso global, es decir, el ejemplo 9, permite comprobar que dada una lista de ficheros HTML que han sido identificados con una o varias ontologías, se generará otra lista de ficheros en el que se expresará las frases en lenguaje natural para posteriormente ser tratadas en la creación de *vistas semánticas*. Pero no se ha detallado como se realiza el PLN.

En el capítulo 3, presentamos dos estrategias: PROCESADO ELEMENTAL y PLN BASADO EN SVMTool. Ejemplo 13 y 14.

El ejemplo 13, se conecta a herramientas externas como son SVMTool y SCOGEME. Sitúese en el directorio .\procesos_sw2sws\src\13_urlPOST y ejecute el fichero run.bat. Es posible que al ejecutar varias veces el sitio web de SCOGEME y SVMTool le deniegen el servicio a no ser que contacte con ellos. De manera, que podrá realizar el experimento unas pocas veces. También puede comprobar el PROCESADO ELEMENTAL, que hemos mejorado sustancialmente a lo largo de nuestra investigación. Para ello ejecute el fichero run.bat en el directorio .\procesos_sw2sws\src\14_plnPropio. Podrá verificar que tanto SCOGEME como SVMTool ofrecen mejores resultados.

A.6. Generación de Vistas Semánticas.

A partir de las frases extraidas es posible generar *vistas semánticas* de diferentes órdenes de acuerdo a las cinco ontologías tratadas en esta tesis. Se puede ejecutar el ejemplo 10 para verificar esta funcionalidad.

PARA COMPILAR:

- Situarse en el directorio .\ procesos_sw2sws\apache-ant-1.6.5\bin
- Ejecutar: **ant exe10**

PARA EJECUTAR:

- Situarse en el directorio
    - .\ procesos_sw2sws\distribucion\ pl2vistasSemanticas
- Ejecutar: **run**

A.7. Herramienta para transformar un sitio web en un sitio web semántico (sw2sws).

Esta herramienta es la más intuitiva de todas, dispone de un entorno gráfico y está pensada para un usuario final, un usuario activo. La jerarquía de directorios es la siguiente:

```
-------build
   -------classes
-------dist
   -------identificacion
   -------imagenes
   -------lib
   -------ontologias
   -------proyectos
   -------tmp
   -------vistas
-------lib
   -------lib_jena
-------nbproject
   -------private
-------src
-------test
```

Como en las demás herramientas, en el directorio “src” se encuentran los fuentes y en “lib” las librerias necesarias. El directorio “dist” contiene el ejecutable, pero este directorio a su

vez se ramifica en otros que van a permitir por un lado, configurar ciertos parámetros y por otro, almacenar estadísticas y ubicación de archivos como ficheros html y *vistas semánticas*.

La aplicación permite configurar la ubicación de todos los directorios, esto se hace modificando el fichero "proyectos.properties", que es el único fichero de configuración que se encuentra directamente en "dist". Desde aquí se podrá indicar el directorio donde el sistema encontrará las ontologías con sus respectivos mapeos, generados a partir del proceso comentado con anterioridad, así como el directorio de trabajo para el proceso de identificación y para el cálculo de las correspondientes estadísticas. Este directorio se utiliza también como fuente de información en el proceso de extracción, ya que este proceso extrae solo información de los ficheros si previamente se sabe que está relacionado con alguna ontología. También se puede definir un directorio para los procesos de generación de *vistas semánticas*, en este directorio no se guardan las vistas en si mismas, solo referencias a que vistas se han generado, cómo se han nombrado y donde están.

El objetivo de la herramienta es generar *vistas semánticas*, pero estas vistas pueden ser usadas por cualquier aplicación semántica que funcione en base a este concepto. En nuestro prototipo de buscador semántico hemos tenido que crear una herramienta que incorpora esta información en su base de datos.

A.8. Carga de Ontologías en la BBDD del Buscador Semántico.

El prototipo implementado solo dispone de cinco ontologías, estas son:

- http://www.co-ode.org/ontologies/pizza/2005/05/16/pizza.owl
- http://www.w3.org/2001/sw/WebOnt/guide-src/food.owl
- http://www.w3.org/2001/sw/WebOnt/guide-src/wine.owl
- http://www.daml.org/2001/01/gedcom/gedcom.owl
- http://www.criado.info/owl/vertebrados_es.owl

Después de descargar el fichero "lcriadof.rar" e instalarlo en tomcat, deberá definir la conexión con la base de datos oracle, con arreglo a su IP, instancia de BBDD y usuario. Para ello modifique en el servlet "BuscadorSemantico" que está en el directorio lcriadof\WEB-INF\classes, las variables que correspondan:

String idMaquina="192.168.1.3";
String instanciaBBDD="pruebas";
String passwd="websem",userName="websem";

Para compilar utilice el fichero por lotes “compila.bat” que encontrará en el mismo directorio. Este BAT además de compilar, copia el servlet a su lugar de ejecución. Ahora solo falta inicializar la BBDD con las cinco ontologías que se utilizan en el prototipo. De manera que tendrá que cargarlas desde el ejemplo 11, que se descargo en el fichero comprimido “procesos_sw2sws.rar”. El procedimiento es el siguiente:

- Situese en el directorio .\ procesos_sw2sws\src\ 11_crearOntologiaEnBBDD
- Modifique el fichero jenaSPARQL.java de manera acorde con lo que modifico en el servlet, las variables que deberá modificar son:
    - String DB_URL="jdbc:oracle:thin:@192.168.1.3:1521:pruebas"; // URL of database @maquina:puerto:instancia
    - String DB_USER = "websem"; // database user id
    - String DB_PASSWD = "websem"; // database password
    - String DB = "Oracle"; // database type
- Sitúese en .\ procesos_sw2sws\apache-ant-1.6.5\bin y ejecute “ant exe11”
- Ejecute el fichero “run.bat” del directorio: “.\procesos_sw2sws\distribucion\crearOntologiaEnBBDD”

A.9. Carga de Vistas Semánticas en la BBDD del Buscador Semántico.

Cuando un webmaster transforma su sitio web en un sitio web semántico tiene que subir toda la información a su hosting mediante cualquier programa FTP. Una vez que el hosting está actualizado, es el momento de incorporar las *vistas semánticas* a la BBDD del buscador semántico de propósito general. Para ello solo tenemos que abrir una consola y compilar el ejemplo 12 con “ant exe12”. Pero asegúrese antes, que los parámetros de conexión son los correctos, para ello compruebe los valores de las variables:

- String DB_URL = "jdbc:oracle:thin:@10.10.25.166:1521:pruebas
- String DB_USER = "websem"; // database user id
- String DB_PASSWD = "websem"; // database password
- String DB = "Oracle"; // database type

Para ejecutar esta utilidad, sitúese en el directorio ““.\procesos_sw2sws\distribucion\crearInstanciasEnBBDD”. El comando del fichero por lotes “run.bat” es el siguiente:

```
java -Xms1024m -Xmx1024m -cp .\;.\antlr-2.7.5.jar;.\arq.jar;.\commons-
logging.jar;.\concurrent.jar;.\icu4j_3_4.jar;.\iri.jar;.\jakarta-oro-
2.0.8.jar;.\jena.jar;.\jenatest.jar;.\json.jar;.\junit.jar;.\log4j-1.2.12.jar;.\stax-api-1.0.jar;.\wstx-asl-
2.8.jar;.\xercesImpl.jar;.\xml-apis.jar;.\ontologiaSPARQL.jar;.\terica.jar;.\classes12_g.zip
tesis.owl.jenaSPARQL %1
```

Donde %1 representa la lista de *vistas semánticas*, por ejemplo, “C:\Datos\html2ws\html2ws\dist\vistas\proyecto1.ts_VistasSemanticas.indice”. Esta información es generada automáticamente por nuestra herramienta (html2ws) en el momento de transformar el sitio web en un sitio Web Semántico.

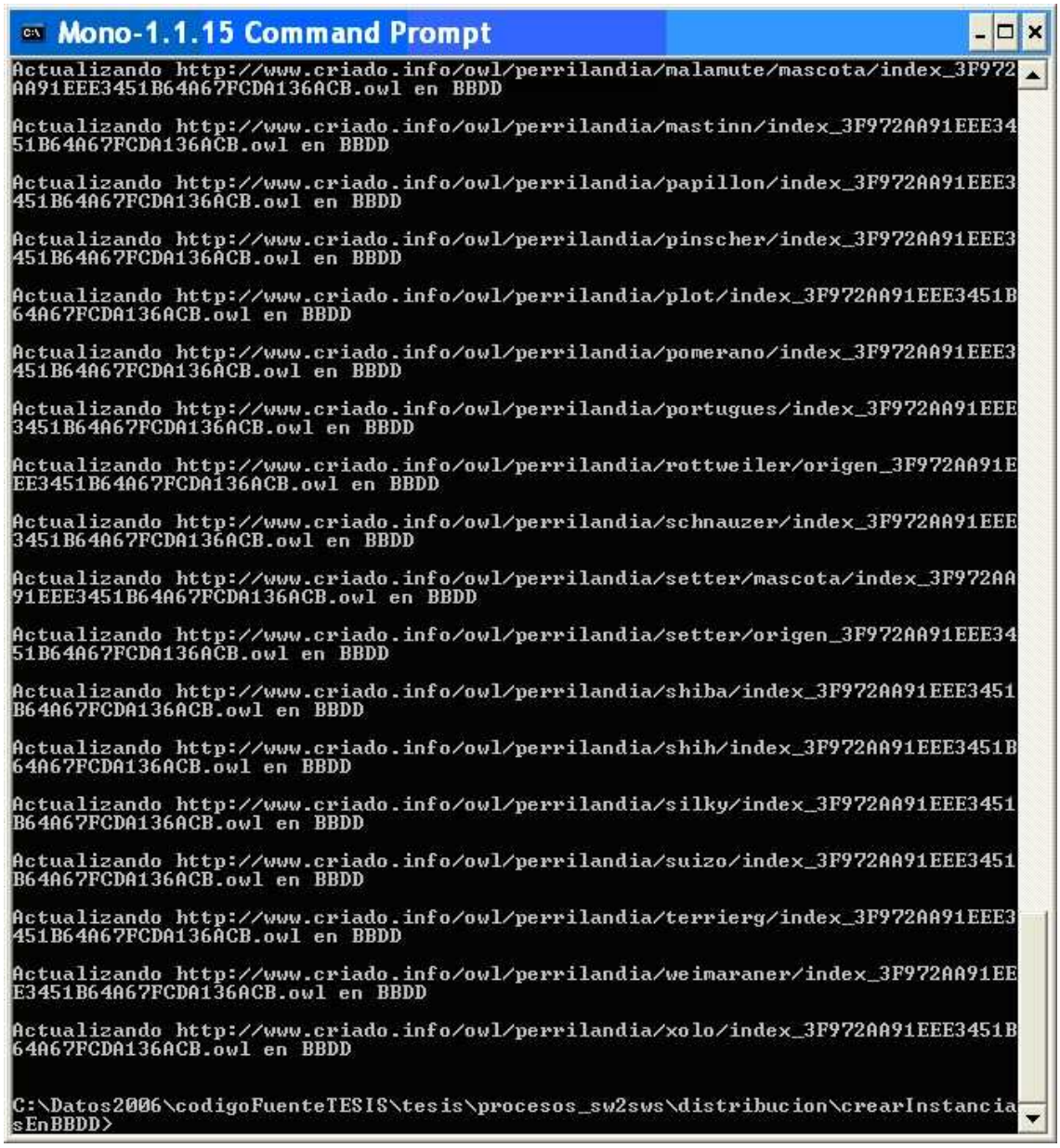
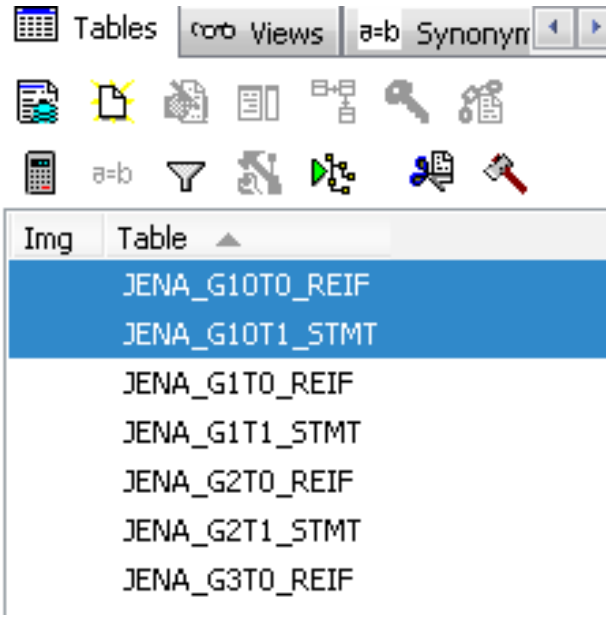


Ilustración 2: Incorporación de vistas semánticas al buscador semántico

La captura de pantalla es un ejemplo del funcionamiento de la utilidad que permite incorporar todas las *vistas semánticas* de un sitio Web Semántico al buscador de propósito general.

A.10. Migración del modelo de Jena a VISSEM.

Cuando realice la carga de ontologías (A.8) e instancias (A.9) verá que en la BBDD aparecen dos tablas por cada una de ellas, que siguen la siguiente nomenclatura:



- JENA_G[secuencia]T1_STMT: Donde [secuencia] es un contador único por ontología e instancia.

- JENA_G[secuencia]T0_REIF: Donde [secuencia] es un contador único por ontología e instancia.

El buscador VISSEM en principio operaba con todas estas tablas simultáneamente, pero a medida que se añade información, la penalización computacional se incrementa mucho. De forma, que finalmente hemos optado por trabajar con una única tabla que hemos denominado "JENA_TERICA". Sobre esta tabla se añaden todos los registros de las tablas del tipo "JENA_G[secuencia]T1_STMT", las tablas del tipo "JENA_G[secuencia]T0_REIF" no se necesitan para esta versión de VISSEM.

El script con las tripletas que usara el buscador VISEM es el siguiente:

```
CREATE TABLE JENA_TERICA
(
 SUBJ     VARCHAR2(250)                NOT NULL,
 PROP     VARCHAR2(250)                NOT NULL,
 OBJ      VARCHAR2(250)                NOT NULL,
 GRAPHID  INTEGER
);
CREATE INDEX JENA_TERICA1 ON JENA_TERICA (SUBJ, PROP);
CREATE INDEX JENA_TERICA2 ON JENA_TERICA (OBJ);
CREATE INDEX JENA_TERICA3 ON JENA_TERICA (SUBJ, OBJ);
```

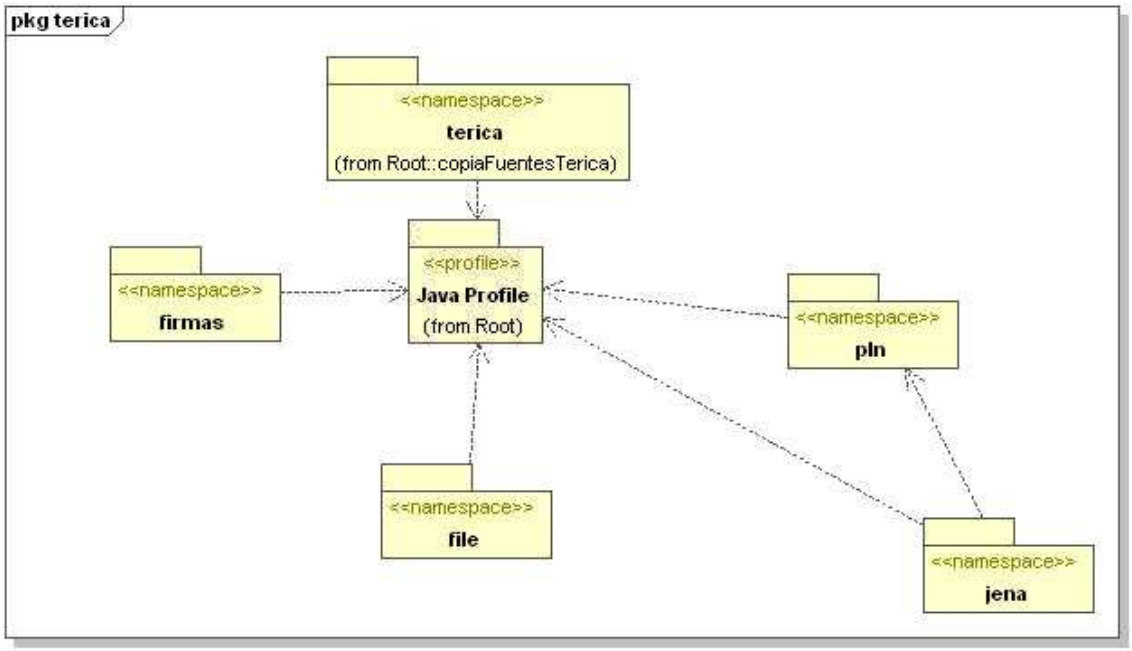
La migración de las tablas que genera Jena a VISEM tendra que ser del estilo siguiente:

```
insert into JENA_TERICA
select * from JENA_G1T1_STMT
UNION
select * from JENA_G2T1_STMT
UNION
select * from JENA_G3T1_STMT
UNION
select * from JENA_G4T1_STMT
UNION
select * from JENA_G5T1_STMT
```

Anexo B: Diagramas de las aplicaciones

Terica.jar

La librería “terica.jar” es el núcleo de nuestra implementación, ya que esta librería permite a la herramienta del webmaster realizar el proceso de transformación completo, es decir, esta librería permit automatizar el proceso de identificación, extracción e interpretación.

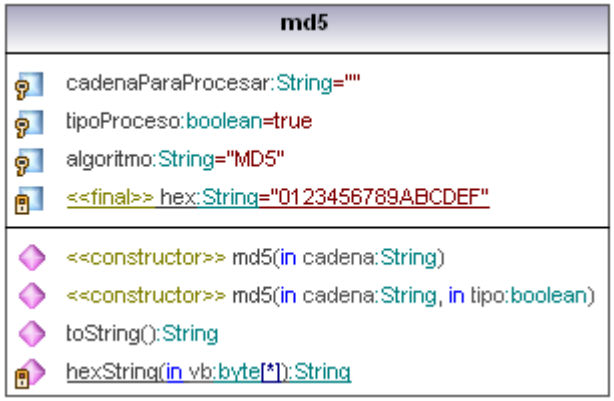


A continuación se detallan todas las clases implementadas, indicando los atributos, métodos que incorporan y a qué paquete pertenecen.

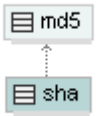
Clase **lista**

diagrama	
paquete	**terica.file**

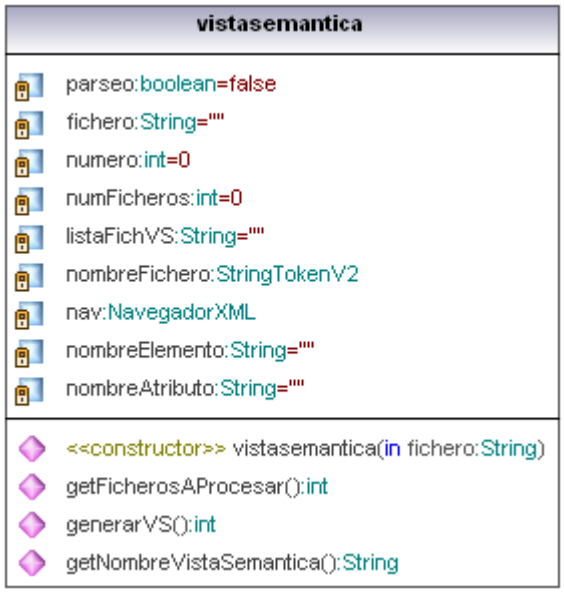
Clase **md5**

diagrama	
paquete	**terica.firmas**

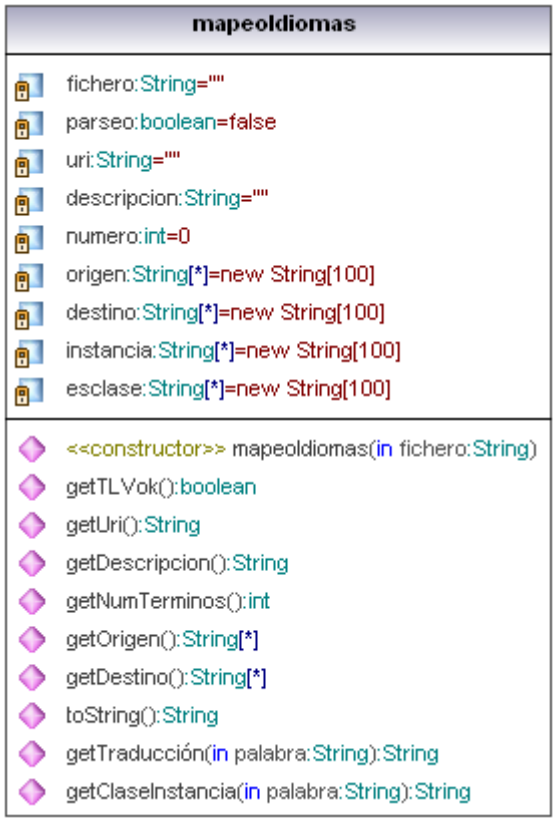
Clase **sha**

diagrama	
Clase padre	
paquete	**firmas.sha**

Clase **vistasemantica**

diagrama	
paquete	terica.jena

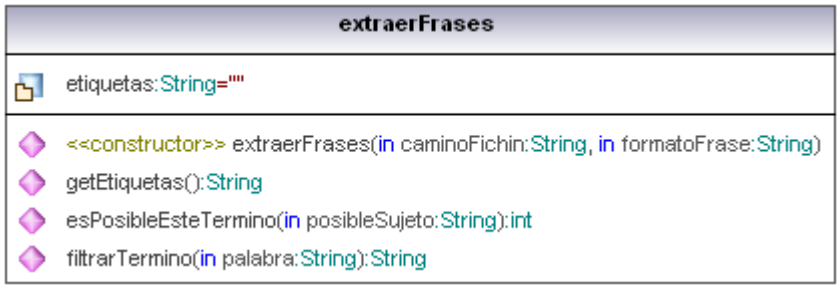
Clase **mapeoldiomas**

diagrama	
paquete	terica.pln

Clase **estructuraOracionSimple**

diagrama	
paquete	terica.pln

Clase **extraerFrases**

diagrama	
paquete	terica.pln

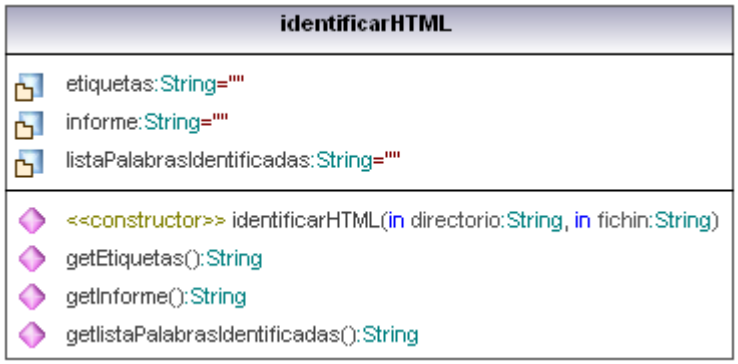
Clase **html2ln**

diagrama	
paquete	terica.pln

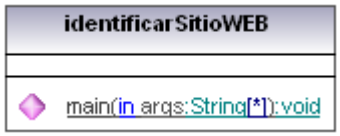
Clase **html2txt**

diagrama	
paquete	terica.pln

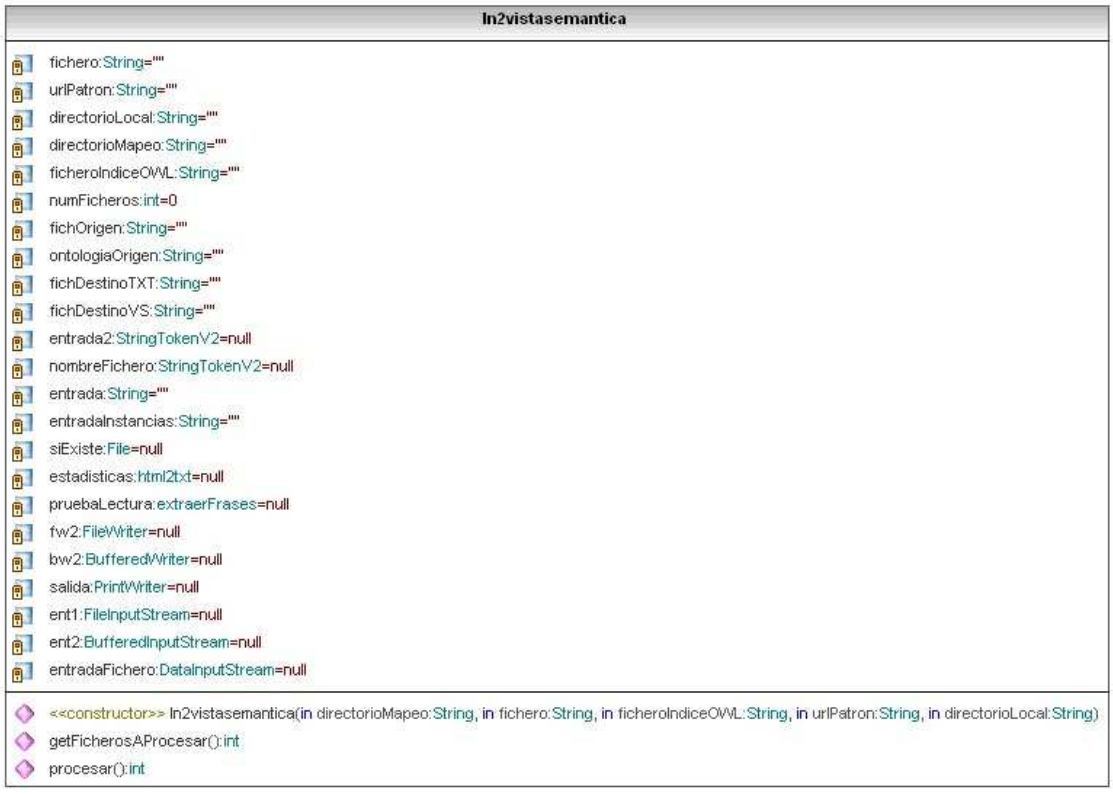
Clase **identificarHTML**

diagrama	
paquete	terica.pln

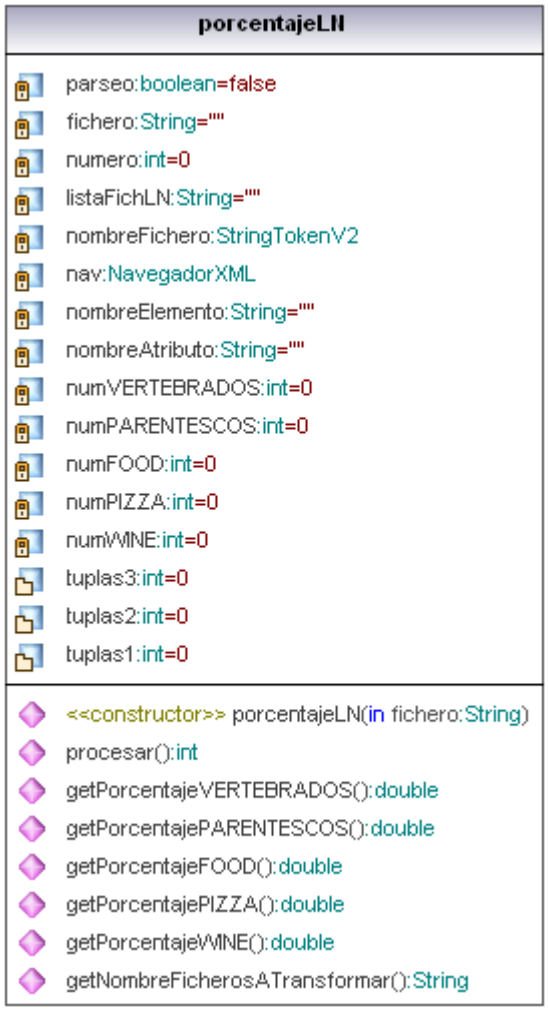
Clase **identificarSitioWEB**

diagrama	
paquete	terica.pln

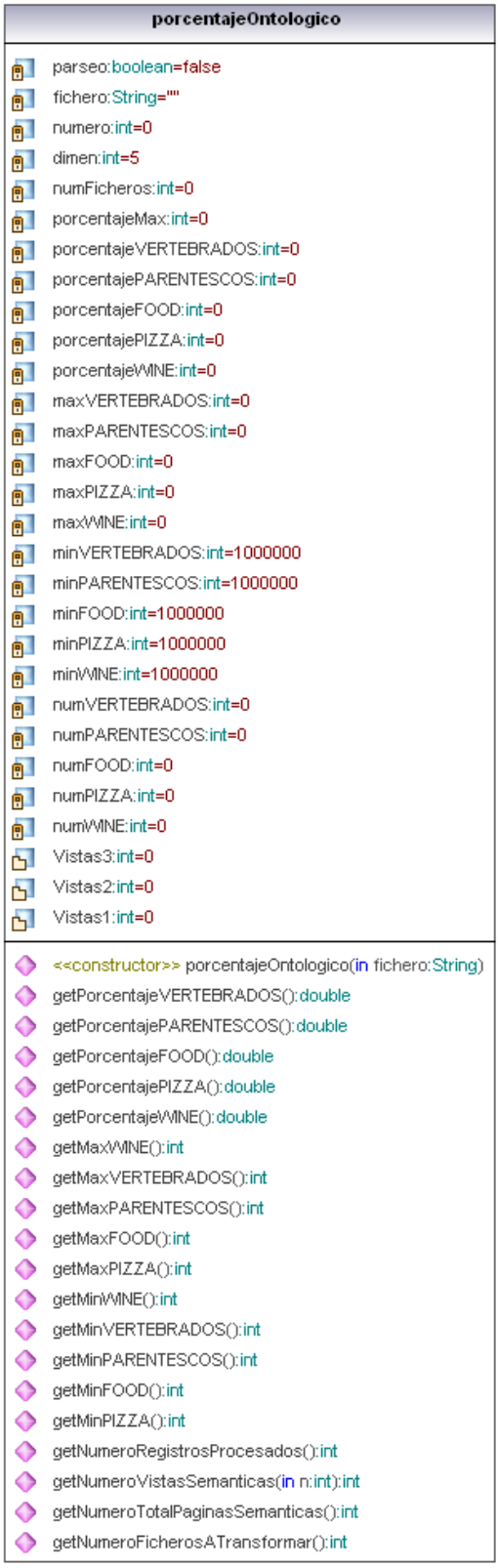
Clase **In2vistasemantica**

diagrama	
paquete	terica.pln

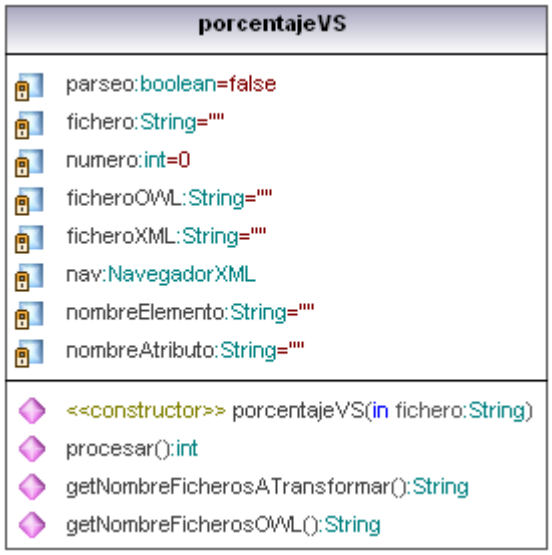
Clase **porcentajeLN**

diagrama	
paquete	**terica.pln**

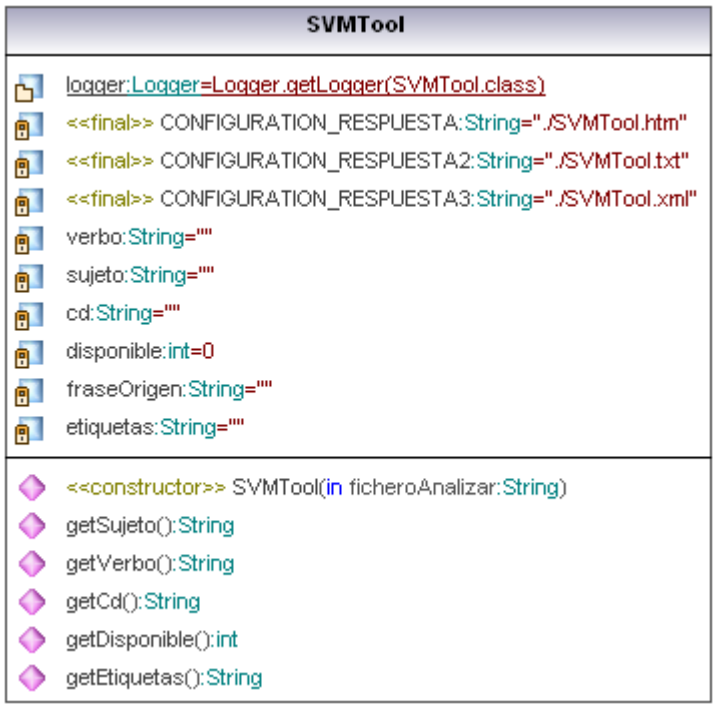
Clase **porcentajeOntologico**

diagrama	
paquete	**terica.pln**

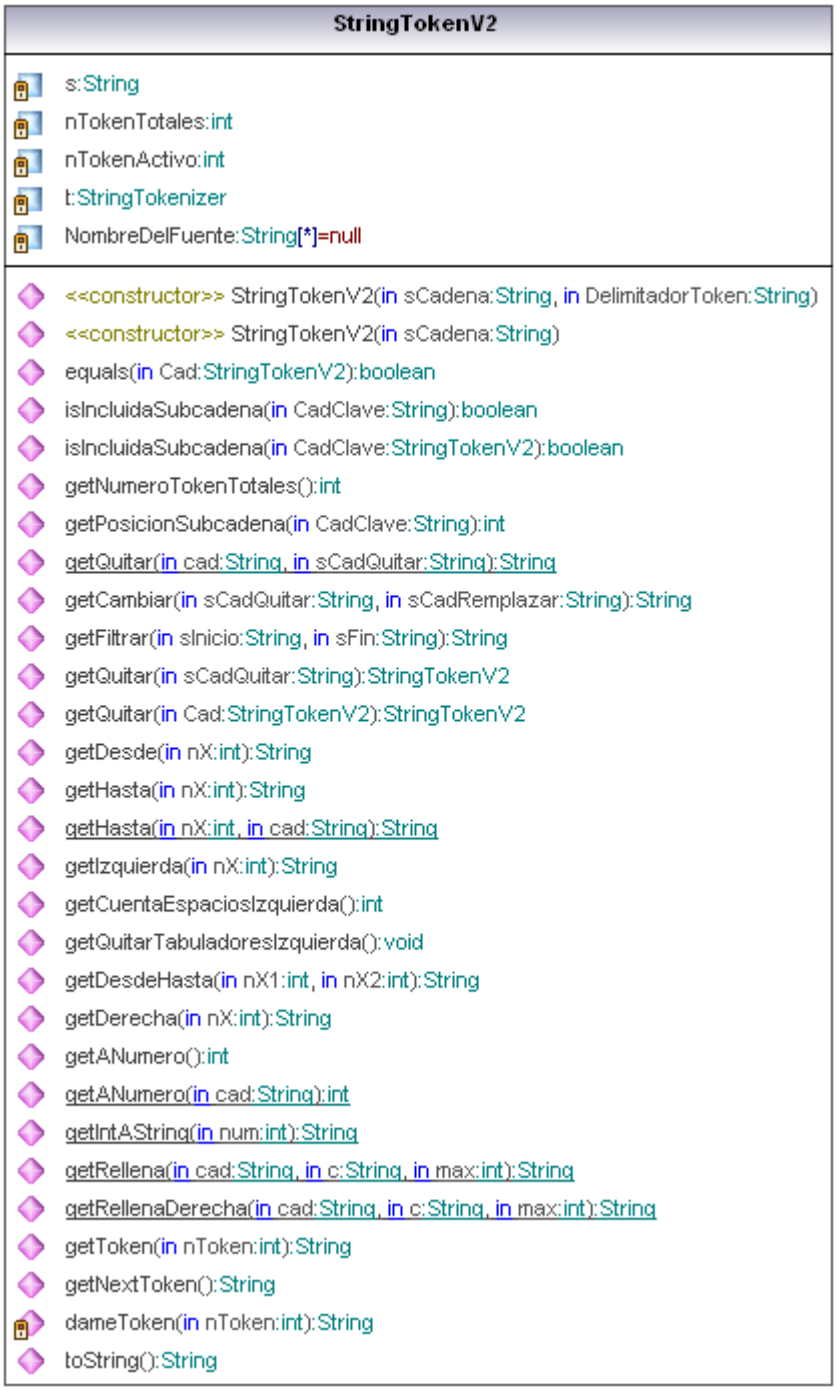
Clase **porcentajeVS**

diagrama	
paquete	**terica.pln**

Clase **SVMTool**

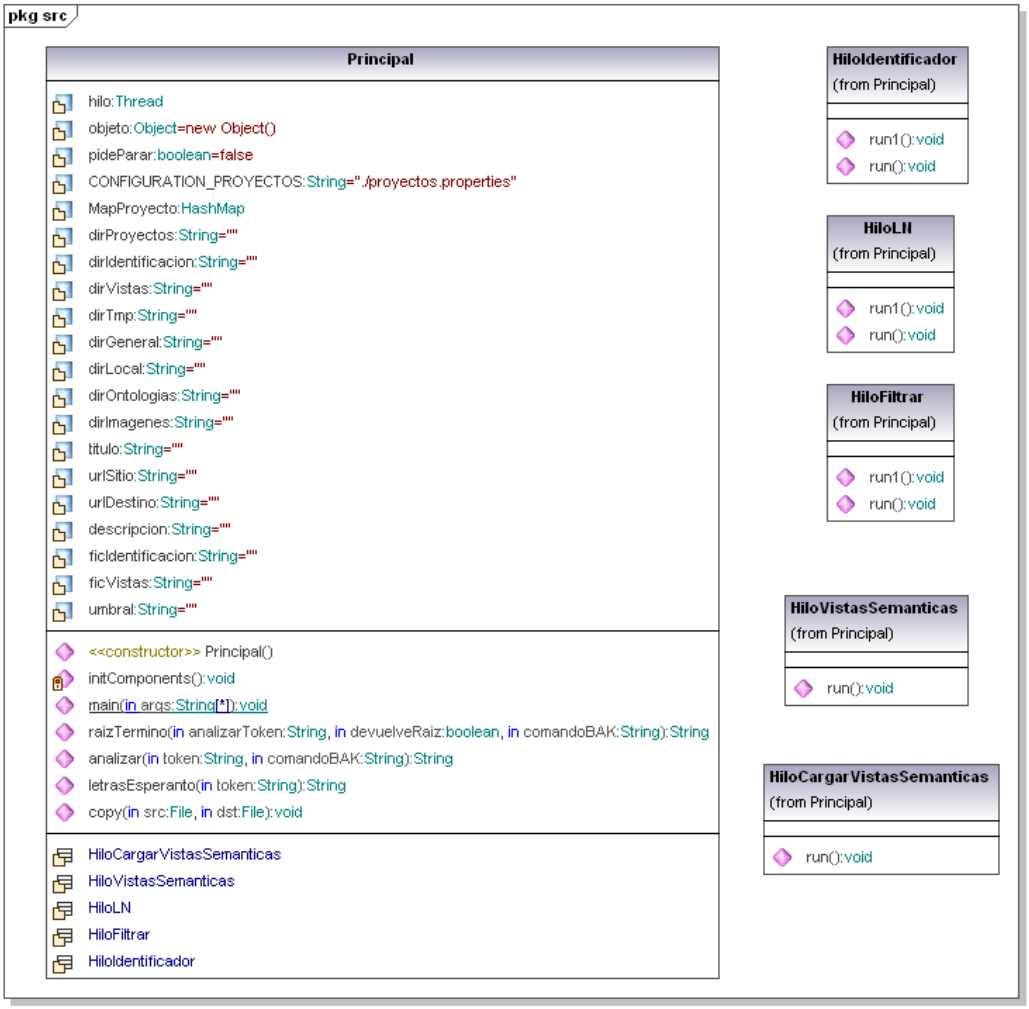
diagrama	
paquete	**terica.pln**

Clase **StringTokenV2**

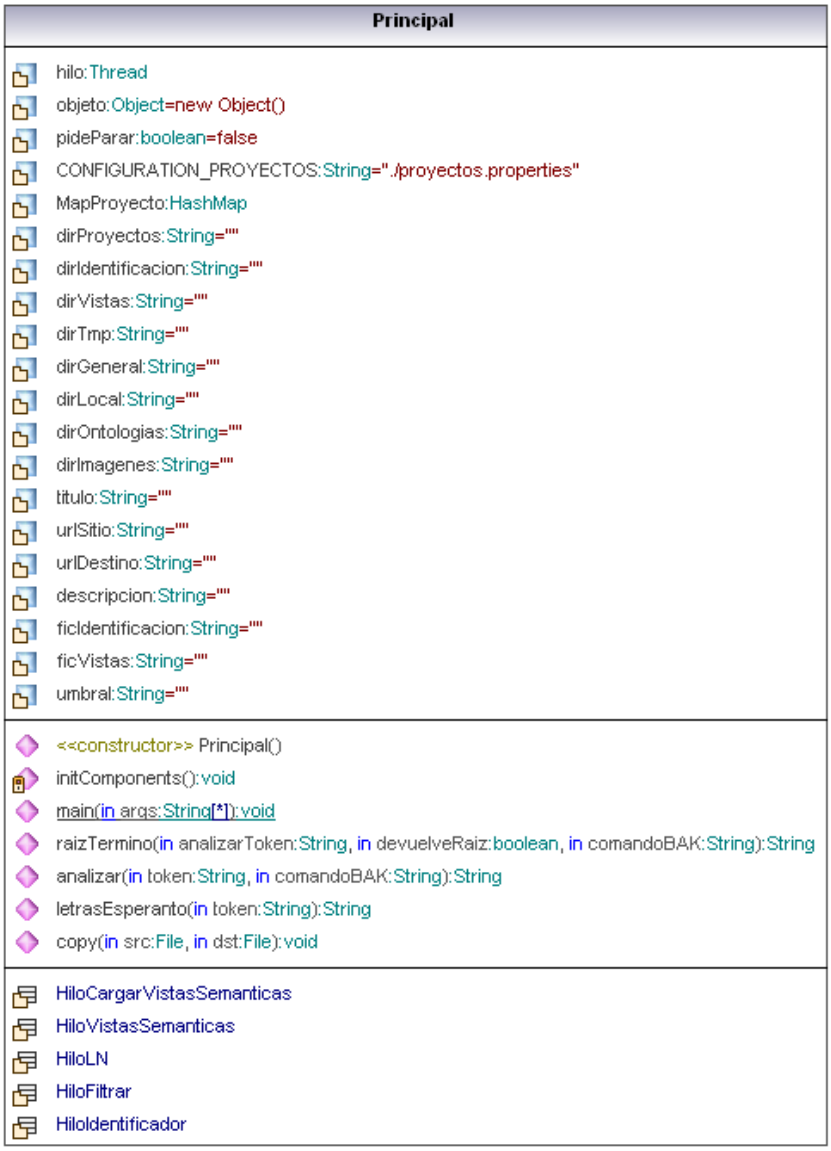
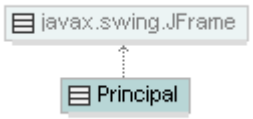
diagrama	
paquete	terica.pln

Herramienta de transformación sw2sws

El diagrama siguiente representa las clases de control sobre las funciones de alto nivel de la aplicación que permite transformar un sitio web en un sitio web semántico. La clase "Principal" lanza el programa. Los Thread que se lanzan desde principal son: HiloCargarVistasSemanticas, HiloFiltrar, HiloIdentificador, HiloLN y HiloVistasSemanticas. Todos ellos utilizan las clases y métodos de la librería "terica.jar".

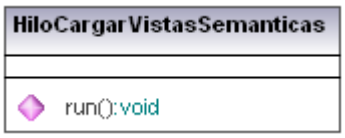


Class **Principal**

diagram	
hierarchy	
owner	**src**

properties	qualified name: src::Principal visibility: public leaf: false abstract: false active: false code file name: Principal.java code file path: C:\tmp\sw2sws\src\Principal.java <<annotations>>: false <<static>>: false <<final>>: false <<strictfp>>: false
ownedMember	**analizar CONFIGURATION_PROYECTOS copy descripcion dirGeneral dirIdentificacion dirImagenes dirLocal dirOntologias dirProyectos dirTmp dirVistas ficIdentificacion ficVistas hilo HiloCargarVistasSemanticas HiloFiltrar HiloIdentificador HiloLN HiloVistasSemanticas initComponents letrasEsperanto main MapProyecto objeto pideParar Principal raizTermino titulo umbral urlDestino urlSitio**
general	**javax.swing.JFrame**
target of relation	ComponentRealization **src**
shown on diagram	**Content of src** **Content of src and all subpackages**

Class **Principal::HiloCargarVistasSemanticas**

diagram	
hierarchy	
owner	**Principal**

properties	qualified name	src::Principal::HiloCargarVistasSemanticas
	visibility	package
	leaf	false
	abstract	false
	active	false
	<<annotations>>	false
	<<static>>	false
	<<final>>	false
	<<strictfp>>	false
ownedMember	**run**	
general	**Thread**	
shown on diagram	**Content of src** **Content of src and all subpackages**	

Class **Principal::HiloFiltrar**

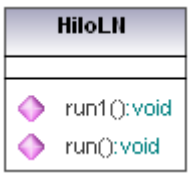
diagram	
hierarchy	
owner	**Principal**
properties	qualified name: src::Principal::HiloFiltrar visibility: package leaf: false abstract: false active: false <<annotations>>: false <<static>>: false <<final>>: false <<strictfp>>: false
ownedMember	**run run1**
general	**Thread**

shown on diagram	**Content of src** **Content of src and all subpackages**

Class **Principal::HiloIdentificador**

diagram	
hierarchy	
owner	**Principal**
properties	qualified name: src::Principal::HiloIdentificador visibility: package leaf: false abstract: false active: false <<annotations>>: false <<static>>: false <<final>>: false <<strictfp>>: false
ownedMember	**run run1**
general	**Thread**
shown on diagram	**Content of src** **Content of src and all subpackages**

Class **Principal::HiloLN**

diagram	

hierarchy	
owner	**Principal**
properties	qualified name: src::Principal::HiloLN visibility: package leaf: false abstract: false active: false <<annotations>>: false <<static>>: false <<final>>: false <<strictfp>>: false
ownedMember	**run run1**
general	**Thread**
shown on diagram	**Content of src** **Content of src and all subpackages**

Class **Principal::HiloVistasSemanticas**

diagram	
hierarchy	
owner	**Principal**

properties	qualified name: src::Principal::HiloVistasSemanticas visibility: package leaf: false abstract: false active: false <<annotations>>: false <<static>>: false <<final>>: false <<strictfp>>: false
ownedMember	**run**
general	**Thread**
shown on diagram	**Content of src** **Content of src and all subpackages**

Class **File**

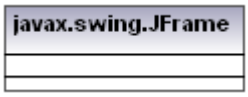
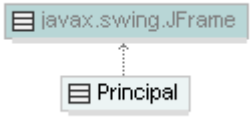
diagram	
owner	**Unknown Externals**
properties	qualified name: Unknown Externals::File visibility: public leaf: false abstract: false active: false
typedElements	Class **Principal** Operation **copy**

Class **HashMap**

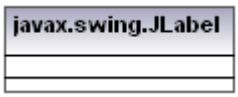
diagram	
owner	**Unknown Externals**
properties	qualified name: Unknown Externals::HashMap visibility: public leaf: false abstract: false

typedElements	Class **Principal** Property **MapProyecto**

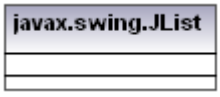
Class **javax.swing.JFrame**

diagram	
hierarchy	
owner	**Unknown Externals**
properties	qualified name: Unknown Externals::javax.swing.JFrame visibility: public leaf: false abstract: false active: false
specific	**Principal**

Class **javax.swing.JLabel**

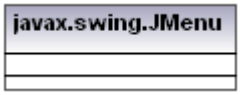
diagram	
owner	**Unknown Externals**
properties	qualified name: Unknown Externals::javax.swing.JLabel visibility: public leaf: false abstract: false active: false

Class **javax.swing.JList**

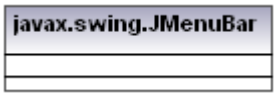
diagram	

owner	**Unknown Externals**
properties	qualified name: Unknown Externals::javax.swing.JList visibility: public leaf: false abstract: false active: false

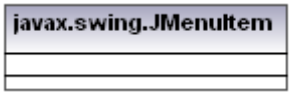
Class **javax.swing.JMenu**

diagram	
owner	**Unknown Externals**
properties	qualified name: Unknown Externals::javax.swing.JMenu visibility: public leaf: false abstract: false active: false

Class **javax.swing.JMenuBar**

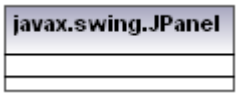
diagram	
owner	**Unknown Externals**
properties	qualified name: Unknown Externals::javax.swing.JMenuBar visibility: public leaf: false abstract: false active: false

Class **javax.swing.JMenuItem**

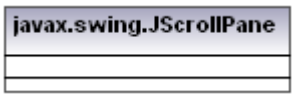
diagram	
owner	**Unknown Externals**

properties	qualified name	Unknown Externals::javax.swing.JMenuItem
	visibility	public
	leaf	false
	abstract	false
	active	false

Class **javax.swing.JPanel**

diagram		
owner	**Unknown Externals**	
properties	qualified name	Unknown Externals::javax.swing.JPanel
	visibility	public
	leaf	false
	abstract	false
	active	false

Class **javax.swing.JScrollPane**

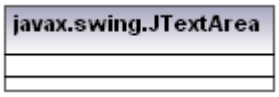
diagram		
owner	**Unknown Externals**	
properties	qualified name	Unknown Externals::javax.swing.JScrollPane
	visibility	public
	leaf	false
	abstract	false
	active	false

Class **javax.swing.JSeparator**

diagram		
owner	**Unknown Externals**	

properties	qualified name	Unknown Externals::javax.swing.JSeparator
	visibility	public
	leaf	false
	abstract	false
	active	false

Class **javax.swing.JTextArea**

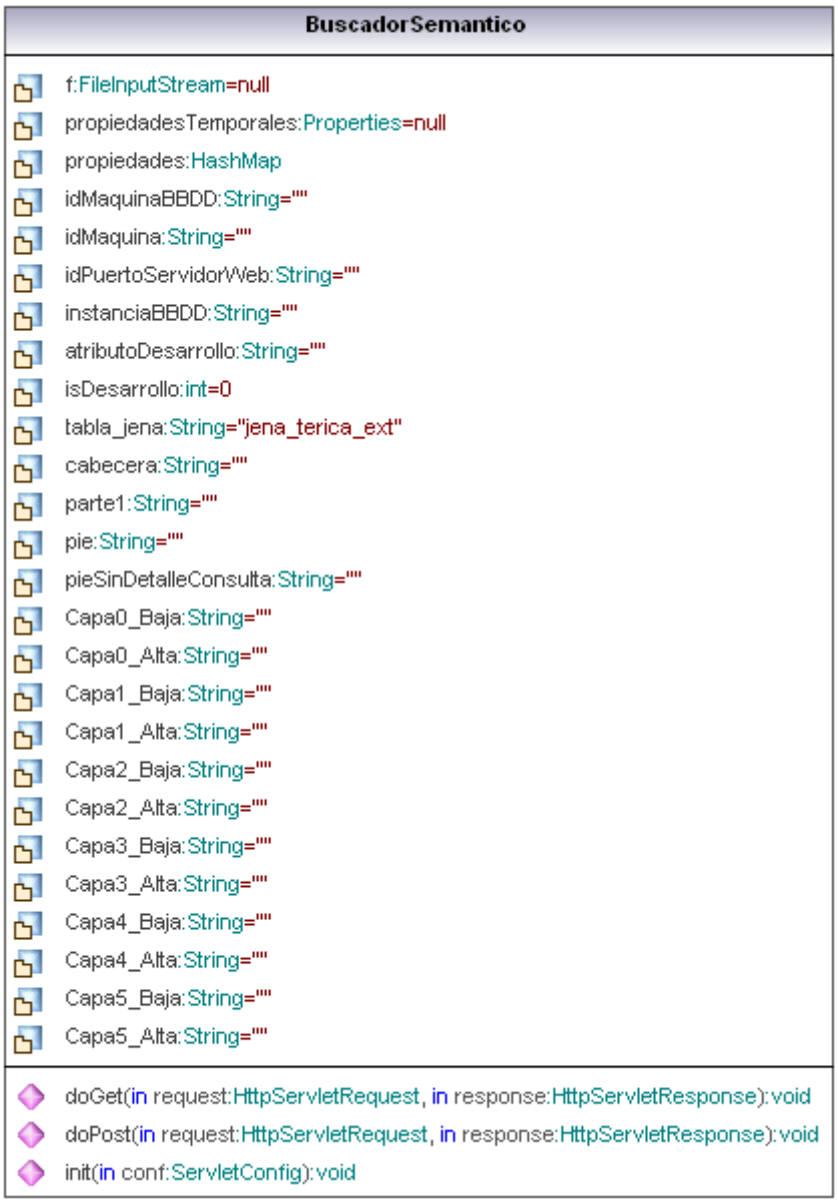
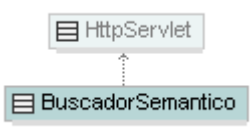
diagram		
owner	Unknown Externals	
properties	qualified name	Unknown Externals::javax.swing.JTextArea
	visibility	public
	leaf	false
	abstract	false
	active	false

Class **Thread**

diagram		
hierarchy		
owner	Unknown Externals	
properties	qualified name	Unknown Externals::Thread
	visibility	public
	leaf	false
	abstract	false
	active	false
specific	HiloCargarVistasSemanticas HiloFiltrar HiloIdentificador HiloLN HiloVistasSemanticas	
typedElements	Class Principal Property hilo	

Buscador Vissem

La clase “BuscadorSemantico” es el servlet con el que se contruye vissem

diagram	
hierarchy	
owner	WEB-INF

properties	qualified name: WEB-INF::BuscadorSemantico visibility: public leaf: false abstract: false active: false code file name: BuscadorSemantico.java code file path: C:\tmp\vissem\WEB-INF\classes\BuscadorSemantico.java <<annotations>>: false <<static>>: false <<final>>: false <<strictfp>>: false
ownedMember	**atributoDesarrollo cabecera Capa0_Alta Capa0_Baja Capa1_Alta Capa1_Baja Capa2_Alta Capa2_Baja Capa3_Alta Capa3_Baja Capa4_Alta Capa4_Baja Capa5_Alta Capa5_Baja doGet doPost f idMaquina idMaquinaBBDD idPuertoServidorWeb init instanciaBBDD isDesarrollo parte1 pie pieSinDetalleConsulta propiedades propiedadesTemporales tabla_jena**
general	**HttpServlet**
target of relation	ComponentRealization **classes**
shown on diagram	**Content of WEB-INF Content of WEB-INF and all subpackages**

El servlet utiliza la clase HttpServlet, HttpServletRequest y HttpServletResponse.

Class **HttpServlet**

diagram	
hierarchy	
owner	**Unknown Externals**
properties	qualified name: Unknown Externals::HttpServlet visibility: public leaf: false abstract: false active: false
specific	**BuscadorSemantico**

Class **HttpServletRequest**

diagram	
owner	**Unknown Externals**
properties	qualified name: Unknown Externals::HttpServletRequest visibility: public leaf: false abstract: false active: false
typedElements	Class **BuscadorSemantico** Operation **doGet doPost**

Class **HttpServletResponse**

diagram	
owner	**Unknown Externals**
properties	qualified name: Unknown Externals::HttpServletResponse visibility: public leaf: false abstract: false active: false
typedElements	Class **BuscadorSemantico** Operation **doGet doPost**