

División de Ciencias Básicas e Ingeniería  
Licenciatura en Ingeniería en Computación

Modalidad:

Proyecto Tecnológico

**Asistente para la evaluación de Ontologías**

Reporte final de proyecto terminal

Alumno:

José Francisco Santa Cruz Mora

204304424

Asesora:

Maricela Claudia Bravo Contreras

Departamento de Sistemas

14-Invierno

26 de marzo 2014

## Índice de Contenido

1. Resumen.....	3
2. Introducción, antecedentes y justificación .....	3
3. Objetivo del proyecto.....	5
4. Marco teórico.....	6
5. Desarrollo del Proyecto.....	8
5.1 Interfaz gráfica.....	9
5.2 Almacenar Datos.....	10
5.3 Implementación .....	11
5.4 Prueba del sistema.....	11
5.4.1 Resultados.....	12
6. Conclusión .....	14
7. Referencias Bibliográficas.....	15
8. Apéndice.....	16
8.1 Apéndice A: Código fuente Java.....	16

## Índice de Figuras

Figura 1. Esquema general para obtener y almacenar los datos de una Ontología.....	8
Figura 2. Función de la OWL-API.....	8
Figura 3. Funciones de los razonadores.....	9
Figura 4. Clases relacionadas para mostrar los resultados.....	10
Figura 5. Esquema de la base de datos del asistente para la evaluación de Ontologías.....	10
Figura 6. Diagrama del proceso de almacenamiento.....	11
Figura 7. Ontología después de ser cargada y analizada.....	12
Figura 8. Resultado al cargar una Ontología en la aplicación.....	12
Figura 9. Análisis completo mostrado por la aplicación.....	13

## 1. Resumen.

Es indudable que las ventajas que ofrece Internet son enormes a la hora de buscar información, pero adolece, como se comentará más adelante, de una manera de encontrar información de forma precisa y de poder realizar deducciones con la información. Se trata de convertirla en conocimiento, haciendo referencia a los datos dentro de las páginas web; las Ontologías proporcionan la vía para representar este conocimiento.

Utilizando los principios de diseño de coherencia, eficiencia e inteligibilidad se mejorará la selección en sus búsquedas y se potenciará el desarrollo de aplicaciones de comercio electrónico, ayudando a los desarrolladores o integradores web a tener características de calidad en sus búsquedas.

El sistema se apoyará en OWL-API<sup>1</sup> para la extracción de los datos y elementos correspondientes a cada Ontología así como el razonador de Pellet-API<sup>2</sup> y Protégé-API<sup>3</sup> para resolver el problema de coherencia.

## 2. Introducción, antecedentes y justificación

Las Ontologías permiten trabajar con conceptos, en lugar de palabras clave, en los sistemas de recuperación de información. Considerando a las Ontologías como fuentes de información, estas describen el contenido de los depósitos de datos independientemente de la representación sintáctica de los mismos, posibilitando su integración semántica.

“Una Ontología es una especificación explícita de una conceptualización” [1], según Thomas Gruber. Una conceptualización es una abstracción, una vista simplificada del mundo que queremos representar.

Las Ontologías son acuerdos, en un contexto social, para cubrir una serie de objetivos. Se crean, entre otras cosas, para:

- Permitir el intercambio de datos entre programas.
- Simplificar la unificación (o traducción) de distintas representaciones.
- Facilitar la comunicación entre personas.

Estamos viviendo una época en las ciencias están impulsando una calidad de vida mayor, por ese motivo, es necesario disponer de personal altamente calificado y especializado para crear sistemas inteligentes que obtengan el conocimiento experto y razonar de manera similar a los humanos, ayudando a resolver tareas especializadas.

---

<sup>1</sup> OWL-API: <http://owlapi.sourceforge.net/>

<sup>2</sup> Pellet-API: <http://clarkparsia.com/pellet/>

<sup>3</sup> Protégé-API: <http://protege.stanford.edu/>

En el proceso de construir estos sistemas es necesario verificar la validez o consistencia de esta representación y a partir de ella, inferir nuevo conocimiento consistente y válido.

Actualmente existen buscadores genéricos de Ontologías que nos permiten encontrar información, la cual no se puede diferenciar entre información personal, académica, comercial, etc. Es decir, cuando un buscador realiza una consulta con algunas palabras clave, normalmente aparece información que no es útil porque no corresponde a lo que estamos buscando o no es confiable. Por lo tanto en este proyecto se propone desarrollar una aplicación que se enfocará a obtener Ontologías y evaluar las características de calidad que normalmente ningún buscador considera. Estas características de calidad son: coherencia, inteligibilidad y eficiencia.

Algunos antecedentes a este proyecto son:

#### **a) Proyectos terminales**

El primer trabajo relacionado tiene por título “Sistema interoperable para compartir información entre sistemas web con autocompletado de formularios” [2].

Las semejanzas que existen entre los módulos de los proyectos se encuentran en el desarrollo de una aplicación web que facilite el intercambio de información entre sistemas, la diferencia radica en la función de la aplicación, la cual está enfocada en construir documentación en un formato muy común llamado PDF.

El segundo trabajo relacionado tiene por título “Sistema de recuperación de información semántico” [3].

La semejanza que existe entre los módulos de los proyectos se encuentra en la aplicación de técnicas del procesamiento del lenguaje natural y la diferencia es la aplicación de los resultados para la recuperación de información y extraer una lista de autores de una serie de documentos.

El tercer trabajo relacionado tiene por título “Extracción automatizada y representación de servicios web mediante ontologías” [4].

La semejanza que existe entre los módulos de los proyectos se encuentra en el manejo de Ontologías y en la extracción de información relevante, la diferencia es la manera en que se manejarán las Ontologías para representar servicios web.

#### **b) Software**

Swoogle [5]: permite buscar en una colección que consta de sólo Ontologías, es decir, de documentos que han sido definidos por una clase. Este buscador semántico recoge muchos documentos recolectados de la Web, y día a día sigue creciendo. Mediante su motor de recuperación ayuda a los usuarios a encontrar su consulta en la web semántica.

Semantic Web Search Engine[6]: buscador semántico permite utilizar los vocabularios de la web semántica para delimitar la búsqueda que se desea localizar.

Wolfram Alpha[7]: se trata de un motor de búsqueda de conocimiento semántico que funciona a través del lenguaje natural. Es un nuevo motor de búsqueda inteligente ideado por el matemático británico Stephen Wolfram.

Las semejanzas que existen con el software es la búsqueda de ontologías, las diferencias con la aplicación que se pretende realizar es que los buscadores antes mencionados, no ofrecen resultados de calidad, los cuales puedan ser usados considerando su coherencia, eficiencia e inteligibilidad.

### **3. Objetivo del proyecto**

El objetivo principal es: diseñar e implementar una aplicación web que realice la evaluación asistida de Ontologías, considerando los principios de diseño de: coherencia, inteligibilidad, y eficiencia.

El proyecto se separo en cinco objetivos secundarios para lograr el objetivo principal, los cuales son:

- ⌚ Diseñar e implementar un módulo para la ejecución de tareas de razonamiento.
- ⌚ Diseñar e implementar un módulo para la evaluación de los resultados del razonamiento con lo que se logrará medir la coherencia.
- ⌚ Diseñar e implementar un módulo que permita medir el tiempo de ejecución de las tareas de razonamiento con lo que se logrará medir la eficiencia.
- ⌚ Diseñar e implementar un módulo que permita verificar la existencia de documentación con lo que se logrará medir la inteligibilidad.
- ⌚ Diseñar e implementar la interfaz gráfica que muestre los resultados de cada Ontología.

#### 4. Marco teórico

El término Ontología tiene su origen en la filosofía, proviene de la conjunción de los términos griegos “ontos” y “logos” que significa existencia y estudio, respectivamente. En los últimos años el término a ganado relevancia entre los ingenieros del conocimiento tomando una interpretación particular; En 1995 Guariano y Giaretta proponen utilizar la palabra tal y como la conocemos los ingenieros del conocimiento como: Ontología.

Las Ontologías pueden ser modeladas con diferentes técnicas de modelado de conocimiento y a su vez implementadas con diferentes lenguajes. Estas pueden ser altamente informales, expresadas en lenguaje natural, semi-informales, semi-formales, y rigurosamente-formales si poseen términos definidos meticulosamente con semántica formal, teoremas y propiedades que les dan un sentido completo. Otras maneras que también se han aprovechado para modelar Ontologías, son el lenguaje de modelado unificado (UML) en ingeniería de software y diagramas E/R, utilizados en base de datos para modelar conceptos y las relaciones entre estos.

En 1993, Gruber propuso modelar Ontologías utilizando marcos lógicas de primer orden, identificando cinco clases de componentes de una Ontología:

- ⌚ Clases: son la base de la descripción del conocimiento en las Ontologías, usualmente se organizan en taxonomías pudiéndose aplicar mecanismos de herencia para su manejo.
- ⌚ Relaciones: Representan las interacciones entre los conceptos del dominio. Las Ontologías pueden contener relaciones binarias; el primer argumento de la relación se conoce como el dominio y el segundo el rango.
- ⌚ Funciones: Son un tipo concreto de relación donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de una Ontología.
- ⌚ Instancias: Representan objetos determinados de un concepto.
- ⌚ Axiomas: Se usan para modelar sentencias que son siempre ciertas. Los axiomas permiten, junto con la herencia de conceptos, inferir conocimiento que no esté indicado explícitamente en la taxonomía de la Ontología. También son utilizados para inferir nuevo conocimiento.

Entre los lenguajes para representar Ontologías existen XML, RDF, OIL, DAML + OIL Y OWL, de los cuales la World Wide Web Consortium (W3C) recomienda RDF y OWL [8].

Para representar meta-datos en la web, la W3C nos recomienda el lenguaje RDF. Este proporciona un medio para agregar semántica a un documento sin referirse a su estructura; estructurado por recursos (objetos) y pares de atributos o valores. Un recurso representa cualquier entidad que pueda ser referenciada por un URI (Identificador único de recursos) Los atributos representan las propiedades de los recursos, y sus valores pueden ser entidades atómicas como: strings, enteros u otros recursos.

El lenguaje que utilizamos para elaborar el asistente es el OWL, lenguaje de marcado semántico desarrollado por la W3C para publicar y compartir Ontologías sobre la World Wide Web. Es una extensión del vocabulario de RDF y se deriva de DAML+OIL.

OWL proporciona tres sub lenguajes diseñados para ser utilizados por comunidades específicas de desarrolladores y usuarios. La característica que define a cada lenguaje es su complejidad, se definen como:

OWL Lite: Es el sub-lenguaje con sintaxis más simple, su intención es ser utilizado en situaciones donde se requiera una jerarquía de clases y restricciones simples.

OWL DL : Es mucho más expresivo que el anterior y está basado en lógica descriptiva. Proporciona la máxima expresividad posible sin perder la completitud computacional y está construido de tal forma que toda sentencia pueda ser resuelta en tiempo finito.

OWL FULL: Es el sub-lenguaje más expresivo, su intención es ser utilizado en donde una alta expresividad, este puede contener ciclos infinitos.

Algunas herramientas de razonamiento utilizadas para el desarrollo de aplicaciones que hacen uso de representaciones Ontológicas son: Pellet, Protégé o Jena.

Pellet es una herramienta de razonamiento para Ontologías especificadas en OWL- DL, es implementado en java y de libre distribución, también ofrece una gran variedad de tareas de razonamiento.

Protégé es un framework de código abierto y distribución libre, la aplicación está escrita en java swing para crear su interfaz gráfica y sus archivos se hacen en el lenguaje OWL.

Jena también es un framework de Java para desarrollar aplicaciones de web semántica. Permite manipular Ontologías representadas en RDF, RDFS y OWL.

## 5. Desarrollo del Proyecto

El sistema está constituido por el siguiente esquema general mostrado en la Figura 1.

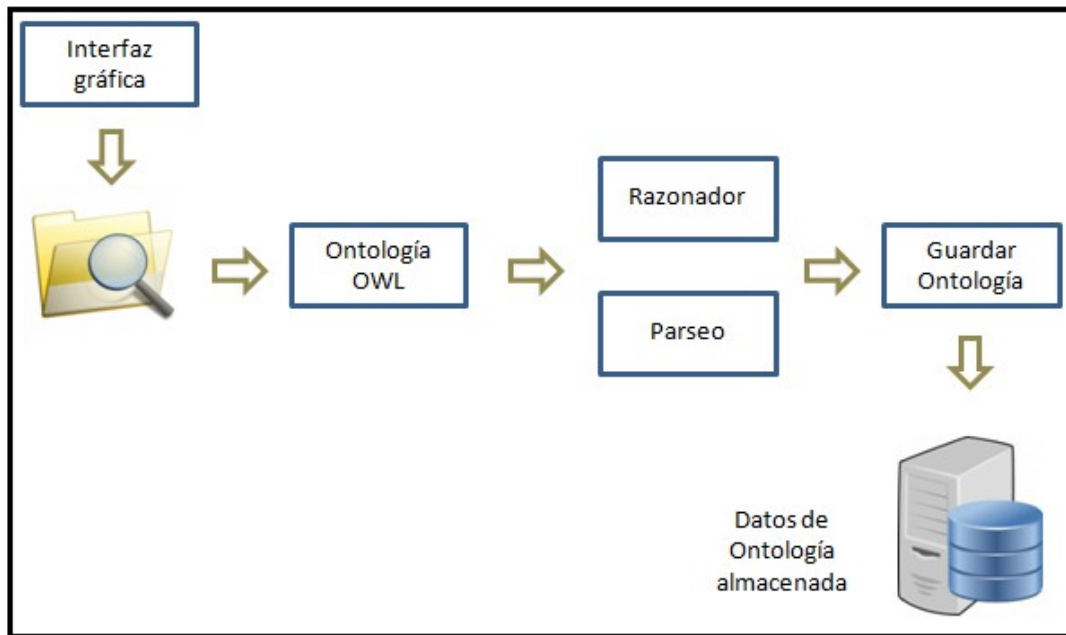


Figura 1. Esquema general para obtener y almacenar los datos de una Ontología.

Podemos ver en la figura anterior, la interfaz gráfica nos permite seleccionar de cualquier directorio almacenado en nuestra PC una Ontología a la cual queremos examinar, está requiere ser analizada por un razonador o por la OWL-API para luego ser almacenada en la base de datos.

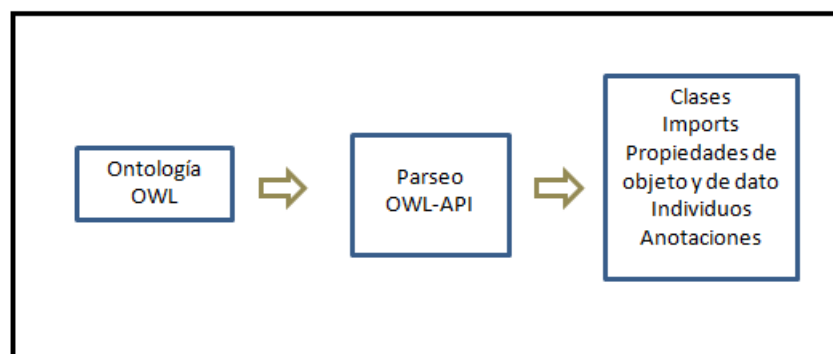


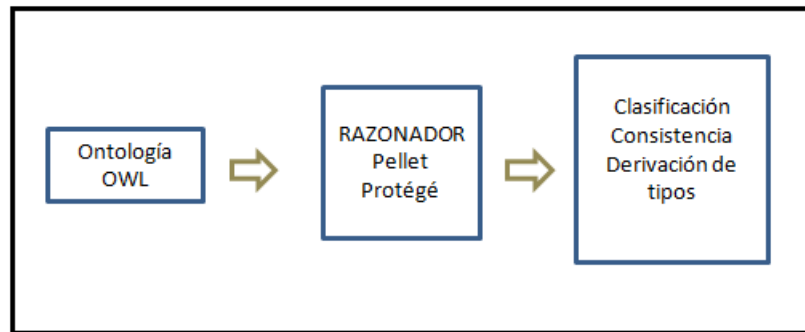
Figura 2. Función de la OWL-API



En la Figura 2 podemos observar la manera en la cual se toma una Ontología OWL y se le asigna a la OWL-API para que extraiga los datos que necesitamos para hacer las diferentes tareas y así poder deducir algunos resultados.

Los razonadores pueden llevar a cabo procedimientos automáticos para inferir y generar nuevas relaciones. Existe un amplio rango de razonadores automatizados disponibles y en general, la inferencia utilizada en este contexto corresponde a lógica de primer orden.

Es una herramienta indispensable para trabajar con Ontologías OWL, por eso utilizamos dos de ellos: Protégé y Pellet, Figura 3.



*Figura 3. Funciones de los razonadores.*

### **5.1 Interfaz gráfica**

Esta interfaz permitirá desplegar la información obtenida por los razonadores y la OWL-API, se podrá observar si la Ontología es inteligible, coherente y que tan eficiente es al momento de buscar los datos elementales para las diferentes tareas.

Las clases relacionadas mostradas en la Figura 4, son las responsables de traer la información mostrada al usuario; Los métodos getters y setters que el lenguaje Java nos proporciona y que son usados por la clase de análisis, son un gran apoyo para mostrar o cargar los resultados.

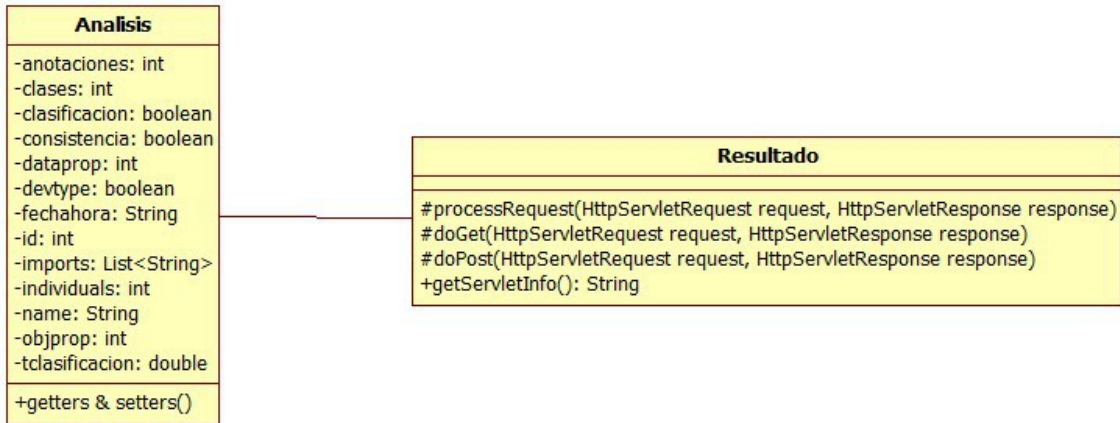


Figura 4. Clases relacionadas para mostrar los resultados.

## 5.2 Almacenar Datos

El sistema se sintetizó en 3 tablas: ontología, análisis e imports, tal como se muestra en la Figura 5.

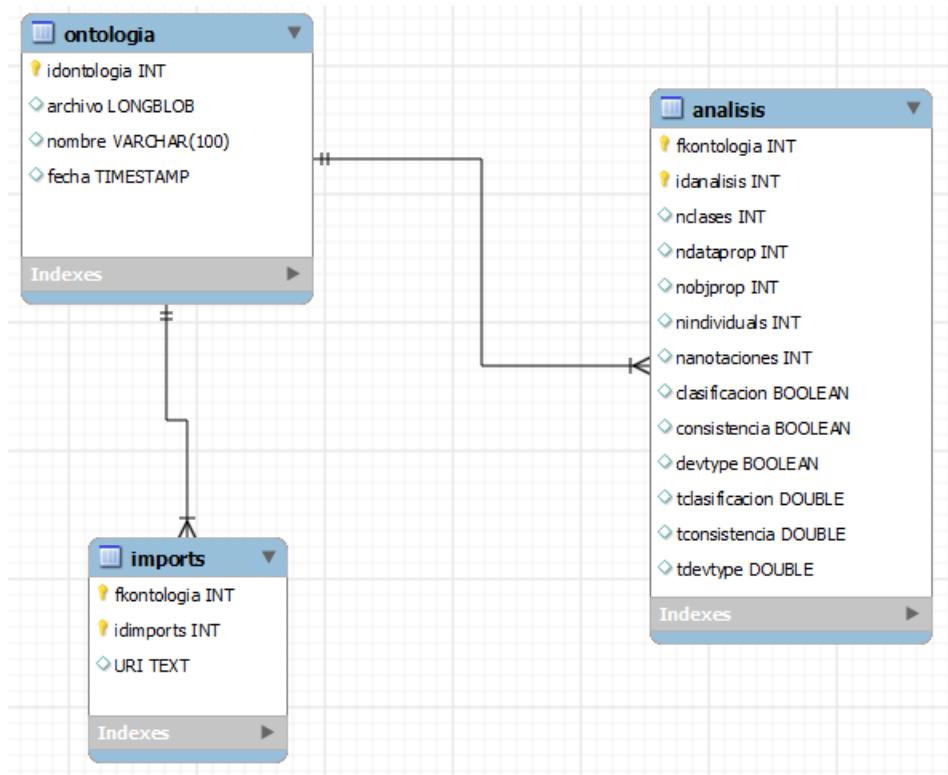
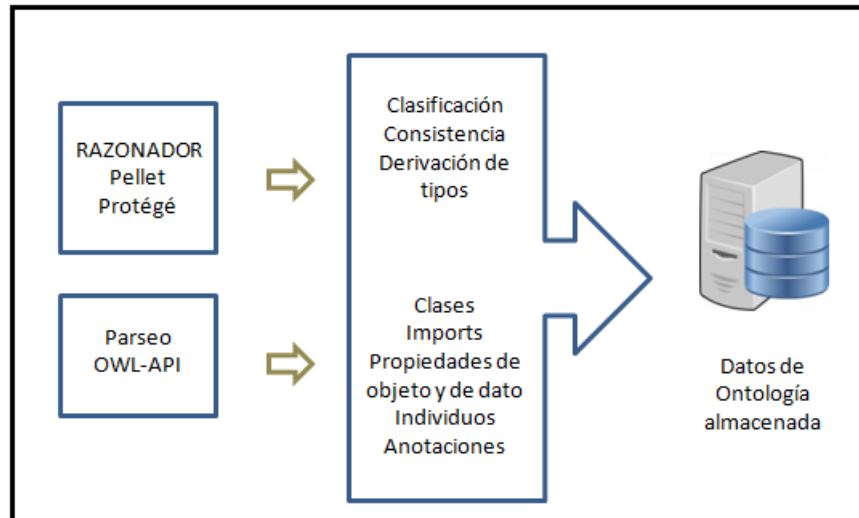


Figura 5. Esquema de la base de datos del asistente para la evaluación de Ontologías.

El proceso para obtener la información que nos interesa es extraída ya sea por el razonador adecuado y por la OWL-API, para después almacenarla en la base de datos y generar los diferentes registros. Figura 6.



*Figura 6. Diagrama del proceso de almacenamiento.*

### 5.3 Implementación

Se necesitaron las herramientas en-listadas a continuación:


- ⌚ MySQL Workbench [9]: Es una herramienta visual unificada para los arquitectos de base de datos, desarrolladores y administradores de base de datos.
- ⌚ Netbeans [10]: Es un proyecto de código abierto que proporciona soporte para varios lenguajes de programación, principalmente para el lenguaje Java.
- ⌚ WampServer [11]: Es un entorno de desarrollo web de Windows. El nos permite crear aplicaciones web con Apache 2,PHP y una base de datos MYSQL.

### 5.4 Prueba del sistema

Es necesario comprobar el correcto funcionamiento de la aplicación web, se realiza una prueba sobre 30 Ontologías para poder obtener la información que nos conviene analizar. En la siguiente Figura 7, podemos apreciar la ventana inicial de la aplicación.

**Archivo de Ontología**  No file selected.



Mostrar  registros por página

Ontología	Fecha de análisis	Ver análisis
<a href="#">pizzas2_2.owl</a>	2014-03-09 18:21:33.0	

Registros 1 de 1 de 1 en total

**Figura 7. Ontología después de ser cargada y analizada.**

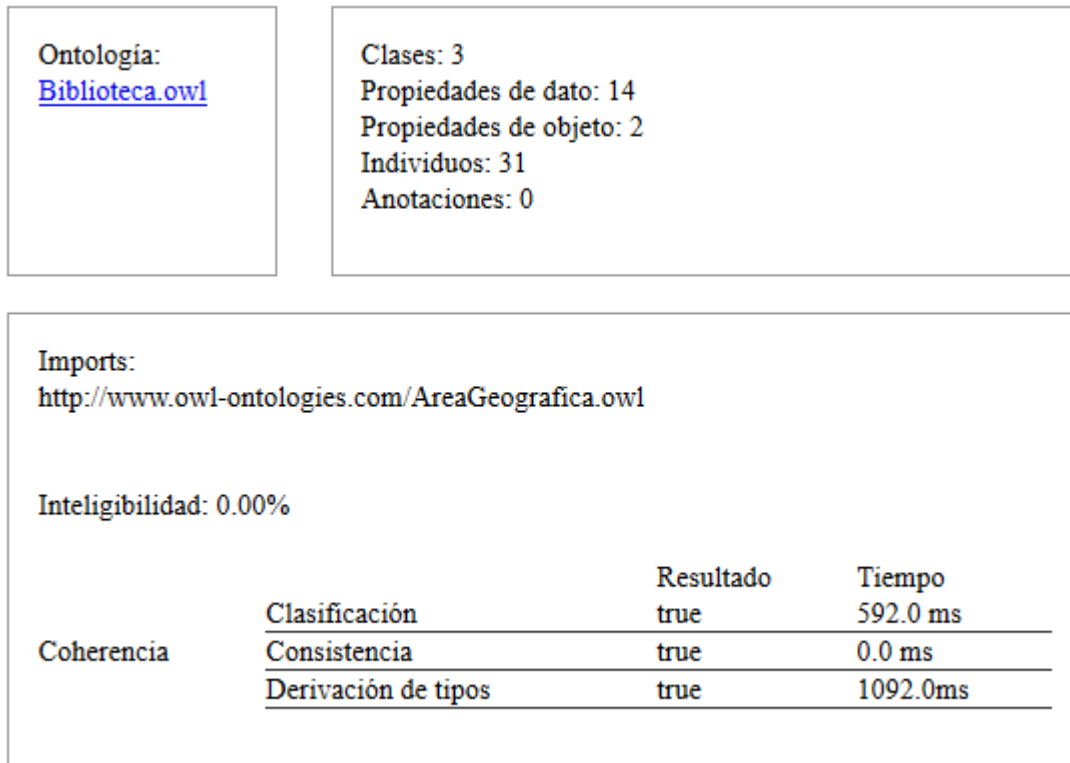
Una vez cargada la Ontología podemos ver el análisis realizado por la OWL-API y el razonador correspondiente, con sólo dar un clic en la lupa, además se muestra el nombre de cada Ontología donde también dándole clic podemos descargar la Ontología seleccionada, nos presenta también la fecha en la cual se realizó el análisis por si el usuario ha realizado algún cambio en la Ontología. Figura 8.

Ontología	Fecha de análisis	Ver análisis
<a href="#">pizzas2_2.owl</a>	2014-03-09 18:21:33.0	
<a href="#">Biblioteca.owl</a>	2014-03-11 18:50:50.0	

**Figura 8. Resultado al cargar una Ontología en la aplicación.**

#### 5.4.1 Resultados

El análisis de cada Ontología cargada nos mostrara: las clases, propiedades de dato, propiedades de objeto, individuos, anotaciones, todos estos datos nos podrán dar la inteligibilidad en un cierto porcentaje, para medir la coherencia tenemos que obtener un valor verdadero para la existencia en: clasificación, consistencia y en derivación de tipos, nos muestra el tiempo en que se realiza el análisis de estas tareas de razonamiento para dejar al usuario inferir si es eficiente; Todos los imports de la Ontología son mostrados. La Figura 9. muestra el análisis completo.



*Figura 9. Análisis completo mostrado por la aplicación.*

#### 5.4.2 Análisis y discusión de resultados

Para la tarea de consistencia, cuando existe, su eficiencia siempre ha sido buena para todas las 30 Ontologías analizadas, marcando un tiempo menor a 0.0 milisegundos(ms). Con este resultado se puede inferir que esta tarea no es de gran relevancia para la eficiencia, siendo esta muy positiva.

La siguiente tarea de razonamiento es la clasificación, su valor en milisegundos(ms), pareciera que no tiene relación alguna con los datos obtenidos con la OWL-API, sólo en un caso se nota una relación, este fue cuando las clases fueron más de 30,000, aquí la eficiencia fue menor ya que el tiempo tomado fue mucho mayor.

La derivación de tipos, en cuanto al punto de vista de la eficiencia, pasa igual que con la tarea anterior, no se nota ninguna relación con los datos que nos manda la OWL-API.

Todas estas tareas de razonamiento nos proporcionan si la Ontología es coherente o no, con sólo una tarea que no cumpla la coherencia no existe.

Para la inteligibilidad nos apoyamos de los datos obtenidos por la OWL-API, aquí podemos inferir si la Ontología es fácilmente entendida o no, esto lo podrías deducir por el porcentaje que arroja la aplicaciones, el dato importante aquí serán los comentarios que tenga contenida, esto nos dirá que tan documentada está y la aplicación nos mostrará que tan inteligible es.

Los imports los obtenemos como datos generales para la aplicación dejando al usuario la decisión de que tan útil serán para él.

## **6. Conclusión**

Las Ontologías nos brindan un vocabulario y una variedad de conceptos con los cuales poder hacer análisis y así poder consultar la información obtenida.

En la actualidad existen varias herramientas que nos pueden ayudar a realizar las tareas de razonamiento, entre ellas están las que se enfocan a un marco más visual, haciendo el manejo de las Ontologías más fácil.

Con lo obtenido por las pruebas en la aplicación podemos considerar como concluidos los objetivos del proyecto; La implementación cumplió con el propósito de hacer uso de los razonadores para extraer la información que más nos interesaba mostrar, para así poder hacer las diferentes conjeturas de los resultados, pudimos percatarnos que algunos datos, como las clases, no tienen relación con los tiempos en los cuales es medida la eficiencia.

Uno de los problemas con los que nos encontramos fueron al usar las diferentes versiones de librerías de los dos razonadores utilizados, ya que la diferencia entre estas, tanto las viejas con respecto de las nuevas, causaban conflictos al momento de hacer los imports a las respectivas librerías.

En la elaboración de este proyecto pude notar lo eficiente que fue el lenguaje de programación Java al combinarla con los diferentes aplicaciones implementadas para las Ontologías, la documentación para el manejo de estas es algo escaso, aún en el lenguaje inglés, algunos casos como el de extracción de los datos inferidos, tomaron más tiempo en ser solucionados debido a esto.

Un dato que considero curioso, es la forma en que se escribe Ontología con la primera letra en mayúscula, para referirse a ella en ingeniería del conocimiento.

## 7. Referencias Bibliográficas

[1] Gruber. T. R. 1993. A Translation Approach to Portable Ontology Specifications [En línea]. Disponible en: <http://tomgruber.org/writing/ontolingua-kaj-1993.htm>

[2] M. Hernández Arias. (2011) “*Sistema interoperable para compartir información entre sistemas web con autocompletado de formularios*”. Proyecto Terminal. Lic. En Ing. En Computación CBI-UAM-Azcapotzalco.

[3] S. M. Ugalde Chávez. (2012) “*Sistema de recuperación de información semántico*”. Proyecto Terminal. Lic. En Ing. En Computación CBI-UAM-Azcapotzalco.

[4] J. Pascual Martínez. (2012) “*Extracción automatizada y representación de servicios Web mediante ontologías*”. Proyecto Terminal. Lic. En Ing. En Computación CBI-UAM-Azcapotzalco.

[5] ebiquitygroup.(20-mayo-2013)*Swoogle*[En línea]. Disponible en: <http://swoogle.umbc.edu/>

[6] *Semantic Web Search Engine*[En línea]. (26-noviembre-2013). Disponible en: <http://swse.org/>

[7] *Wolfram Alpha*[En línea]. (21-noviembre-2013). Disponible en: <http://www.wolframalpha.com/>

[8] "OWL Web Ontology Language - Overview"[En línea], (2004). Disponible en: <http://www.w3.org/2007/09/OWL-Overview-es.html>

[9] MySQLWorkbench (marzo 2014) Disponible en: <http://www.mysql.com/products/workbench/>

[10] NetBeans.org (marzo 2014) Disponible en: <https://netbeans.org/>

[11] WampServer on Windows (marzo 2014) Disponible en: <http://www.wampserver.com/en/>

## 8. Apéndice

### 8.1 Apéndice A: Código fuente Java

- 🕒 `OntoUtils.java`
- 🕒 `RecuperaOntologia.java`
- 🕒 `Analisis.java`
- 🕒 `FileUploadDBServlet.java`
- 🕒 `Resultado.java`