UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

ESCUELA DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

TEMA:

Estudio comparativo y evaluación de razonadores para Lenguajes Ontológicos en la Web Semántica.

Trabajo de fin de carrera previa a la obtención del título de Ingeniera en Sistemas Informáticos y Computación

AUTOR:
Rosario Elizabeth Guamán Tandazo

DIRECTORA:
Ing. María del Carmen Cabrera

CODIRECTORA:
Ing. Audrey Romero

Loja - Ecuador
2010

CERTIFICACIÓN

Ingeniera.

María del Carmen Cabrera

DIRECTOR DE TESIS

CERTIFICA:

Haber dirigido y supervisado el desarrollo del presente proyecto de tesis previo a la obtención del título de INGENIERA EN SISTEMAS INFORMÁTICOS Y COMPUTACIÓN, y una vez que este cumple con todas las exigencias y los requisitos legales establecidos por la Universidad Técnica Particular de Loja, autoriza su presentación para los fines legales pertinentes.

Loja, Noviembre del 2010

Ing. María del Carmen Cabrera

DIRECTORA DE TESIS

CERTIFICACIÓN

Ingeniera.

Audrey Romero

CODIRECTORA DE TESIS

CERTIFICA:

Haber dirigido y supervisado el desarrollo del presente proyecto de tesis previo a la obtención del título de INGENIERA EN SISTEMAS INFORMÁTICOS Y COMPUTACIÓN, y una vez que este cumple con todas las exigencias y los requisitos legales establecidos por la Universidad Técnica Particular de Loja, autoriza su presentación para los fines legales pertinentes.

Loja, Noviembre del 2010

Ing. Audrey Romero

CODIRECTORA DE TESIS

AUTORÍA

El presente proyecto de tesis con cada una de sus observaciones, análisis, evaluaciones, conclusiones y recomendaciones emitidas, es de absoluta responsabilidad del autor.

Además, es necesario indicar que la información de otros autores empleada en el presente trabajo está debidamente especificada en fuentes de referencia y apartados bibliográficos.

.....

Rosario Elizabeth Guamán Tandazo

CESIÓN DE DERECHOS

Yo, Rosario Elizabeth Guamán Tandazo declaro ser autora del presente trabajo y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales.

Adicionalmente declaro conocer y aceptar la disposición del Art. 67 del Estatuto orgánico de la Universidad Técnica Particular de Loja que su parte pertinente textualmente dice: "Forman parte del parte del patrimonio de la universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado que se realicen a través, o con el apoyo financiero académico o institucional (operativo) de la universidad".

_____

Rosario Elizabeth Guamán Tandazo

DEDICATORÍA

La realización de este proyecto, va dedicado a mis queridos padres José Miguel Guamán y Lupe Elizabeth Tandazo por todo el apoyo incondicional que siempre me han brindado en las buenas y las malas, a mis queridos hermanos David Miguel y Cindy Jazmín; quienes siempre me han animado a seguir adelante y para quienes espero ser ejemplo.

Luego a Dios por darme la capacidad de culminar con éxito el desarrollo de este proyecto de Tesis, en fin a toda mi familia que de una u otra manera han estado siempre a mi lado dándome su apoyo y ánimo para seguir adelante y jamás rendirme.

Rosario Guamán.

AGRADECIMIENTO

Mi agradecimiento más profundo a Dios por estar conmigo en todo momento con su guía y ejemplo, y de esta manera ayudarme a culminar con éxito mi proyecto de tesis, y con esto dar paso a una nueva vida llena de retos mayores pero esta será la base para seguir y superarme.

A mis padres, hermanos y demás familiares, por estar conmigo siempre en cada uno de los momento más difíciles de mi vida, y que gracias a ellos he podido superar.

A la Ing. María del Carmen Cabrera Directora de Tesis, por guiarme con la ayuda de sus conocimientos y experiencias. Así como el apoyo y ánimo incondicional que siempre la ha caracterizado y que me ha impulsado a seguir siempre adelante a pesar de las adversidades.

A los Docentes de la Escuela de Ciencias de la Computación, quienes impartieron el conocimiento necesario para responder de la mejor manera en mi vida profesional. A mis amigos y compañeros de clase que siempre están ahí conmigo en todo momento y siempre tuvieron para mí una palabra de aliento porque siempre salga adelante.

Rosario Guamán

INDICE DE CONTENIDOS

RESUMEN

La web semántica proyecta introducir información que sea interpretada por agentes inteligentes, con el objetivo de incrementar su base de conocimiento y realizar inferencias que permitan ayudar a los procesos que actualmente los usuarios las efectúan de forma manual. Así mismo se da surgimiento a los lenguajes ontológicos para la web, siendo el más destacado el OWL (Web Ontology Lenguage) y los razonadores semánticos como: FaCT++, Pellet, Racer Pro, etc.

El presente trabajo se centra en la investigación de los razonadores como herramientas útiles para la evaluación de ontologías, realizando un análisis desde las diferentes perspectivas de su usabilidad.

El primer paso que se realizó es la investigación, la que permitió establecer criterios de evaluación tales como: Eficacia, Eficiencia, Facilidad de Instalación y Disponibilidad de la Información; los resultados obtenidos de este proceso son analizados y comparados con la finalidad de obtener el mejor razonador que cumpla con los requerimientos establecidos en la investigación.

Además se detalla el proceso de evaluación y comparación de los razonadores, empezando por los fundamentos teóricos, herramientas utilizadas, descripción de cada razonador, la fase de evaluación, conclusiones, recomendaciones y anexos que respaldan los resultados; con esto se espera sirva de base para investigaciones futuras relacionadas.

INTRODUCCIÓN

Con el desarrollo de esta investigación se desea evaluar y comparar los razonadores semánticos que mejores prestaciones presenten en la verificación de una ontología. Para alcanzar dicho objetivo se definen criterios que ayuden a realizar una evaluación de los diferentes aspectos que caracterizan a un razonador, analizados en los Capítulos del I al III, los mismos que se desarrollaron con el fin de aportar conocimiento valioso y útil para determinar el proceso de comparación y evaluación.

En el Capítulo I se realiza una introducción a aspectos relacionados con la web semántica y ontologías, donde se hace referencia a los conceptos básicos y necesarios para empezar con su estudio; además aquí se adquiere la necesidad de utilizar herramientas que permitan evaluar el conocimiento representado en las ontologías; en este caso el uso de un razonador.

En el Capítulo II se estudia la lógica de razonamiento que utilizan los razonadores semánticos, permitiendo de ésta manera profundizar el conocimiento en su poder de expresividad.

Una vez comprendida la lógica de razonamiento en el Capítulo II se estudia los razonadores como herramienta de manera global tomando en cuenta aspectos como: características, arquitectura, razonamiento, licencia, ventajas y desventajas; estos brinda una pauta para una mejor valoración.

Para el Capítulo IV se realiza una evaluación de los razonadores en base a los aspectos considerados en toda la investigación, para esto se define un esquema de las fases a seguir en el proceso de evaluación, así existirá confiabilidad en los resultados obtenidos.

Finalmente, se presentan las conclusiones y recomendaciones en donde se evidencian los resultados obtenidos durante toda la investigación, siendo una base para trabajos futuros. También se anexa la documentación que se generó en el proceso de evaluación así como: aplicación de pruebas, manual de instalación de los razonadores, glosario de términos y las referencias bibliográficas que aportan en toda la investigación.

CAPITULO I
WEB SEMÁNTICA Y ONTOLOGÍAS

Introducción

Hoy en día la Word Wide Web (www), es una herramienta de trabajo, consulta, comunicación, información de uso cotidiano para nuestra sociedad, si se la compara con otros medios que le antecedieron como la radio quien tuvo que esperar 38 años para tener 50 millones de usuarios, la televisión 13 años, y la internet con la www, solo 4 años, está última les lleva una gran ventaja.

Al mismo tiempo que la web ha ido evolucionado, lo han hecho las tecnologías que la conforman; las primeras fueron HTML y HTTP, y algunas de las más recientes: CGI, Java, JavaScript, ASP, JSP, PHP, Flash, XML, etc.; que hacen posible una web mejorada, más eficiente y de un mantenimiento fácil.

Cada día avanza la web y su evolución no termina, una de las tendencias más recientes y que sobresalen para cambiar el futuro de la www, surge a finales de los 90, lo que se llama "Web Semántica", corriente creada por Tim Berners-Lee, el propio inventor de la Web y presidente del consorcio W3C, que busca que las máquinas puedan entender y comprender el contenido de la web. Estás máquinas estarían conformada por agentes capaces de realizar actividades por nosotros de tal manera que se ahorre procesos de trabajo mejorando los resultados para optimizar recursos.

Para alcanzar este fin la web semántica propone describir los recursos de la web como algo entendible, no sólo para las personas sino también para los programas que puedan asistir, representar o sustituir en tareas rutinarias al hombre.

Las tecnologías que dan forma a la web semántica busca desarrollar una web donde sea más fácil encontrar, compartir e integrar información y servicios; para así fortalecer más los recursos disponibles a través de www.

Objetivos

- ✓ Desarrollar un fundamento teórico de la web semántica y los factores que han intervenido en su evolución en los últimos tiempos.
- ✓ Establecer las ventajas y desventajas de la web semántica, de tal manera que permita una visión objetiva para impulsar el desarrollo de la presente investigación.
- ✓ Estudiar las ontologías, sus componentes, aspectos más importantes y el papel que desempeña dentro de la web semántica.

1. Web Semántica

1.1. Historia sobre WWW

El término “Word Wide Web” fue creado en 1989 por Tim Berners-Lee; quien desarrolló el primer servidor para Word Wide Web, httpd; así como el primer programa de cliente (un navegador y un editor) en octubre de 1990. (Jacobs, 2001)

Tim Berners-Lee, es el director del Consorcio W3C desde sus inicios tuvo dos visiones; la primera dirigida a tener un sistema que permita consultar información sin importar en donde se encuentre así como compartir documentos. La segunda visión es la creación de la Web Semántica. (Ortiz,s.f)

La web abarca cuatro fases de crecimiento dentro del Internet, la primera es la Web 1.0 que conecta la información, en la Web 2.0 conecta las personas poniendo al “YO” en usuario de interfaz y el “nosotros ”dentro de la Web para una participación social. Como siguiente fase, la Web 3.0 en la actualidad está empezando, basada en la representación de los significados, uniendo conocimientos y añadiendo otros que hacen de la experiencia del internet algo novedosos, eficiente y útil. Después de algún tiempo se prepara la Web 4.0; donde se unen las personas, las cosas razonaran y existirá una comunicación entre ellos. (Millis, 2008)

1.2. La Web 2.0

La Web 2.0 es un término que apareció a finales del 2004, en sus inicios intenta agrupar nuevos servicios y tecnologías que surgen de la web. Actualmente casi todo está representado de una u otra forma en la www, con la ayuda de un buen buscador se puede optimizar los resultados, además brinda una gran capacidad para almacenar, leer y visualizar los contenidos, pero no interpretar el sentido de la información.

Una de las limitantes que tiene; es que un software no sustituye a una persona, puede generar, buscar, transportar la información a los usuarios; pero la máquina no interpreta esta información. (O’Reilly, 2005)

En los últimos años se ha expandido aún más la web 2.0, debido a la aparición de una gran variedad de portales Web y APIs para la creación de nuevos servicios. Hay factores que han influido en este cambio, como la convergencia de dominios en el internet y las telecomunicaciones alrededor del protocolo IP; de tal manera que los desarrolladores incorporan funciones de las telecomunicaciones a la composición de servicios (*mashups*) de Internet. Los Mashups dan la oportunidad a los usuarios de construir sus propios servicios en base a servicios ya existentes mediante el uso de portales y editores. (Cuevas, 2006, p.75-81)

La infraestructura de la web 2.0 es compleja y evoluciona diariamente, aquí influyen algunas técnicas como las que a continuación se presentan:

- ***CSS, marcado XHTML***.- Hojas de estilo en cascada es un lenguaje formal usado para definir la presentación de un documento escrito en HTML o XML. El W3C es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegador.
- Técnicas de aplicaciones ricas no intrusivas (***como AJAX***).- Técnica de desarrollo web para crear aplicaciones interactivas; de manera que es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.
- ***Java Web Start***.- Es la implementación de referencia de la especificación JNLP (Java Network Launching Protocol) y está desarrollado por Sun Microsystems, permitiendo arrancar aplicaciones Java que están en un servidor web de aplicaciones comprobando previamente si el cliente tiene la versión actualizada de dicha aplicación
- ***XUL.-*** Es una aplicación de XML a la descripción de la interfaz de usuario en el navegador Mozilla.
- ***Redifusión/Agregación de datos en RSS/ATOM***.- RSS es un formato de fuente web codificados en XML, utilizado para suministrar a suscriptores de información actualizada frecuentemente. ***ATOM***.- es un formato de redifusión, alternativa RSS.
- ***URLs sencillas con significado semántico.-*** Son aquellas que dentro de lo que cabe son entendibles para el usuario. Están formadas de palabras relacionadas con el contenido de la página y fáciles de recordar.
- ***JCC y APIs REST o XML.***- Hace referencia a las técnicas de programación que, utilizando objetos JSI en el navegador (en el cliente y no en el servidor).
- ***JSON.-*** Es un formato ligero para el intercambio de datos. Es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML.

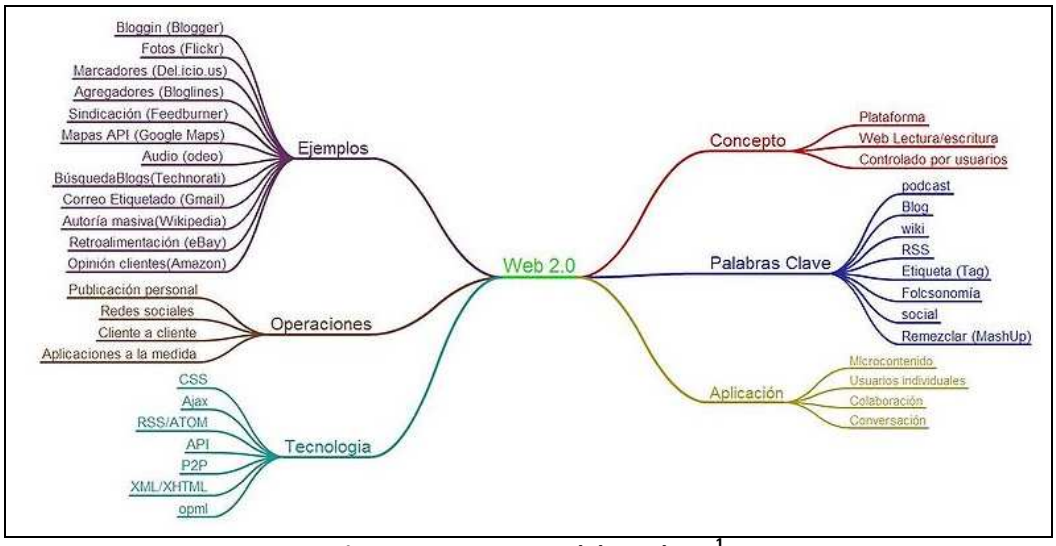

Figura 1. Mapa Mental de Web 2.0[1]

La web 2.0 en sus inicios trata de agrupar un conjunto de nuevos servicios; basándose en algunos criterios como:

- ✓ Toda acción que se realiza sobre los sistemas se hacen vía web, por medio de navegadores estándar, sin instalar nada; lo que deja atrás lo clásico de tener que instalar programas en los ordenadores.

  Una ventaja es que se dispone de un servicio actualizado, que permite a cada a cada usuario un servicio al que se puede dar de alta o baja inmediatamente. (O'Reilly, 2005, p.4)

- ✓ Aplicaciones en las cuales, sus usuarios participan en la construcción de los contenidos; un ejemplo son los blogs. Del mismo modo ayudan con aplicaciones colaborativas, de escritura colaborativa como wiki. Dentro de estás colaboran varias redes de tal manera que cada una se alimenta de los datos de la otra.

[1] http://galeria.sld.cu

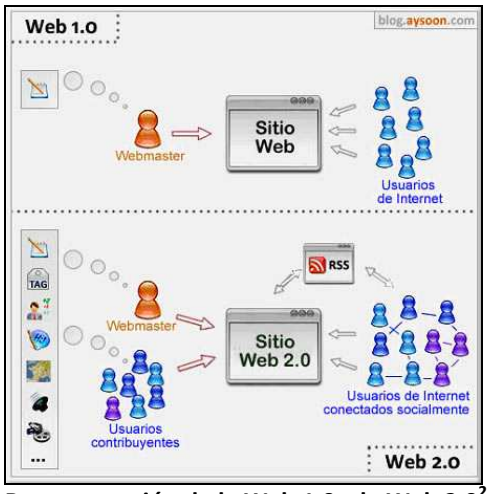

Figura 2. Representación de la Web 1.0 y la Web 2.0[2]

Una diferencia entre las fases de la web:

- Web 1.0 - Personas conectadas a la Web
- Web 2.0 - Personas conectadas a personas - redes sociales, wikis, colaboración, posibilidad de compartir.
- Web 3.0 - Aplicaciones web conectadas a aplicaciones web, a fin de enriquecer la experiencia de las personas.
- Web 4.0 – Une a las personas, las cosas razonan y existe una relación estrecha entre ellos.

1.3. Web Semántica

Definición

Es una corriente originada por el propio inventor de la Web y presidente del consorcio W3C, Tim Berners-Lee, cuya finalidad es lograr que las máquinas puedan entender y utilizar los contenidos y servicios disponibles en la red, sin importar el idioma en que se encuentren escritos. (Goméz & Pérez, 2006)

Berners-Lee, Hendler, y Lassila, (2001) define *"La web Semántica pretende añadir significado a los datos, en forma de* ***metadatos****, de modo que se pueda interpretar de una mejor manera la información que existe en la web".*

Según estos criterios se desea que la información contenida en la web tenga un significado de tal manera que sea interpretada y comprendida por las personas y las máquinas.

[2] http://www.4esoconsolacionpsospedra.blogspot.com/

Para lograr esta finalidad se aplica dos aspectos relevantes ***la lógica*** y ***la confianza en los metadatos***, la primera es una disciplina que estudia los principios del razonamiento, mientras la segunda da un significado de importancia a los metadatos, por lo tanto la confianza en que éstos sean correctos es primordial ya que de ellos depende el éxito del proceso.

Objetivos de la Web Semántica

- ✓ Crear un medio universal para el intercambio de información basado en la representación del significado de los recursos de la Web, de manera entendible para las máquinas.
- ✓ Mejorar la web ampliando la interoperabilidad entre los sistemas informáticos y reducir la necesaria intervención de operadores humanos.

Ventajas de la Web Semántica

- La web semántica permite recuperar información de tal manera que el usuario puede dejar al software tareas de razonamiento o procesamiento de un determinado contenido.
- Al poner significado a una Web se mejora la eficiencia de los buscadores, y se optimiza la búsqueda de la información.
- Realiza una interacción colectiva entre todos los recursos que el internet proporciona de esta forma se puede tener un mejor aprovechamiento de la información que se busca.
- El diseño que posee asegura una integración de otros recursos, vía XML, RDF, OWL, etc.

Desventajas de la Web Semántica

- Los documentos que se encuentran en la red no tiene el formato que propone la web semántica, por tal razón habría que realizar una restructuración para poder ser interpretados de una manera correcta en los ordenadores.
- Los idiomas son un inconveniente, ya que cada uno tiene una semántica diferente, lo que puede conllevar a hacer buscadores semánticos por idiomas.

Metadatos

Se menciona que la web semántica quiere dar significado a los datos mediante los ***metadatos***, su finalidad es proveer una categorización semántica de su contenido, que le permite razonar automáticamente sobre la información.

En general es un objeto que describe o dice algo sobre otro objeto de información. Una de las funciones principales es su uso para la búsqueda de información precisa. (Gilliland, 2000, p.2)

La recuperación de información con metadatos

Anteriormente las consultas que se realizaban pretendían que el texto o el objeto de la información tengan la capacidad de entregar un contenido detallado de un recurso. Este proceso es útil, pero al momento de requerir una consulta con mayor precisión, éste no es eficiente.

Por esta razón la existencia de estructuras de metadatos que hace posible la creación de herramientas para la extracción de recursos que involucran aspectos semánticos de las consultas.

Los resultados de las búsquedas mejoran en un ambiente en el que existan los *metadatos,* definen múltiples interrelaciones entre objetos de información, brinda la estructura semántica que ellos necesitan para realizar una recuperación eficaz; y como resultado se obtiene un mejor rendimiento dentro de las consultas complejas.

Los metadatos permiten el acceso a los recursos de manera controlada al conocer con precisión el objeto descrito; por lo tanto es posible establecer un sistema de filtrado, generar bases para una autenticación y mecanismos para definir un grado de confianza sobre fuentes de información. (Gilliland, 2000, p.9)

Estructura de la web semántica

Para aprovechar la web semántica, sé necesita lenguajes semánticos potentes, que sean capaces de comprender el conocimiento basándose en el uso de los *metadatos y ontologías.*

Debido a la información que se almacena en las *ontologías* se obtiene automáticamente los datos de la web, y sacar de ellas conclusiones que ayudan a la toma de decisiones.

Según Berners-Lee con ayuda del uso de anotaciones RDF y RDFSchema, la arquitectura de la Web Semántica se representar de la siguiente manera. (Pérez, 2007)

Figura 3. Web Semántica (Berners, 2008)

En la Figura 3. Se muestra la estructura de la Web Semántica y en la Tabla 1. Descripción de las Capas de la Web Semántica. Se realiza una descripción de cada una de sus capas para una mayor especificación.

Tabla 1. Descripción de las Capas de la Web Semántica

Característica	Definición
Unicode	El alfabeto. Conjunto de caracteres cuya finalidad es proporcionar el medio por el cual un texto en cualquier forma e idioma puede ser codificado. (Unicode)
URI	La referencia. Son cadenas que permiten acceder a cualquier recurso que se encuentre en la Web. (Uri)
XML+NS+mlschema	Es una capa técnica donde se encuentran almacenadas las diferentes tecnologías que posibiliten la comunicación entre agentes. (Xml)
RDF+dfschema	Define el lenguaje universal con el que se expresa ideas en la web semántica. (Rdf)
Vocabulario de Ontologías	Ayuda a clasificar la información, a su vez permite extender la funcionalidad de la Web Semántica. (Owl, 2007)
Lógica	Además de ontologías se precisan reglas de inferencia.
Pruebas	Se intercambiaran “pruebas” escritas en el lenguaje unificado de la web semántica.

Confianza	Debe comprobarse de forma segura las fuentes de información, los agentes deberán ser escépticos acerca de lo que leen en la web semántica. (OpenPGP)
Firma Digital	Sirve para verificar que la información obtenida sea una fuente fiable.

En suma, el objetivo de la Web Semántica es que la Web pase de ser una colección de documentos a convertirse en una base de conocimiento **Figura 4**.

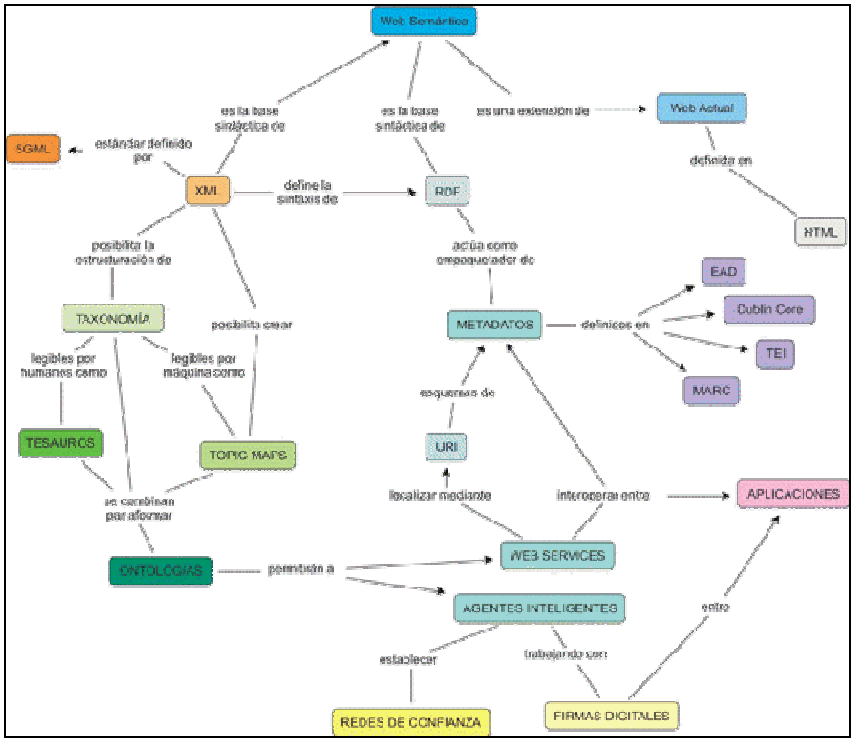

Figura 4. Mapa Conceptual de la Web Semántica (Rodrigo., 2005)

1.4. Ontología

Definición

Existen algunas definiciones de ontologías, aquí las más relevantes:

"*Una ontología es un vocabulario acerca de un dominio: términos + relaciones + reglas de combinación para extender el vocabulario*". Neches (1991:90)

Esta definición hace una comparación de la ontología con un vocabulario, se identifica los términos, relaciones entre ellos y las reglas que tiene que tomar en cuenta para su combinación, incluyendo el conocimiento que se genera de la misma.

Gruber (1993:199) expreso: "*Una ontología es la especificación de una conceptualización*".

Está definición se convirtió en una de las citadas en la comunidad de la ontología, y sirvió de base para otras definiciones que se propusieron.

Borst (1997:12) "*Una ontología es una especificación formal de una conceptualización compartida*".

Esta definición se basó en la propuesta por Gruber un poco más definida.

"*Una ontología es una base de datos que describe los conceptos generales o sobre un dominio, algunas de sus propiedades y cómo los conceptos se relacionan unos con otros*". (Weingand, 1997)

La definición que integra varios criterios de los autores antes mencionados, para entregar un concepto más completo, simple y definido.

(Muñoz, 2009) define a la ontología dentro de Web Semántica como: "*Las ontologías son un punto común de investigación en varias comunidades, como la investigación del conocimiento, procesamiento de lenguaje natural, sistemas de información cooperativos y gestión del conocimiento, permiten una compresión compartida y cocensuada del conocimiento de un dominio que puede ser comunicada entre personas y sistemas heterogéneas*."

Todos estos autores dan una definición de lo que es Ontología desde su punto de vista; en pocas palabras *las ontologías definen conceptos y relaciones de algún dominio de manera compartida y conceptualizada y que está sea representado de una manera formal, legible y utilizable por los ordenadores*.

Características

- La multiplicidad de la representación permite que un concepto se lo pueda representar de diferentes maneras.
- Un mapeo de ontologías, establece relaciones entre elementos de una o más ontologías; para establecer conexiones, especializaciones, generalizaciones, etc.

Tipos de Ontología según Gruber (Ovejero & Martín , 2001)

- **Ontología de la Aplicación:** Usadas por la aplicación. Ejemplo: ontología de procesos de producción, diagnostico de fallas.
- **Ontología del dominio:** Especificas para un tipo de artefacto. Ejemplo: ontología del proceso de producción.
- **Ontologías técnicas básicas:** Describe características generales de artefactos. Ejemplo: componentes, procesos.

- **Ontologías genéricas:** Describe la categoría de más alto nivel.

Criterios de Diseño según Gruber

- **Claridad.-** Debe comunicar claramente el significado de sus términos, las definiciones deben ser objetivas y comentadas en lenguaje natural.
- **Coherencia.-** Permitir hacer inferencias que sean acordes a las definiciones.
- **Extendible.-** Anticipar el uso y permitir extensiones.
- **Sesgo de Codificación mínimo.-** Especificar el nivel de conocimiento sin depender de una codificación particular a nivel de símbolo.
- **Mínimo compromiso ontológico.-** Menos cantidad de "pretensiones" acerca del mundo modelado.

Metodología para la Construcción de una Ontología.

Methontology

Metodología desarrollada por el grupo de Ontología de la Universidad Politécnica de Madrid brinda una construcción de ontología a nivel de conocimiento. Tiene sus raíces en las principales actividades identificadas por el proceso de desarrollo del software. (Goméz & Pérez, 2006)

Está metodología incluye:

- Identificación del proceso de desarrollo de la ontología.
- Un tipo de vida basado en la evolución de los prototipos.
- Técnicas para llevar a cabo cada actividad en la gestión.

Methontology propone su construcción de ciclo de vida basado en el *desarrollo de prototipos* que permite añadir, cambiar y remover términos en cada nueva versión de prototipo. Las siguientes tareas: (Fernandez, Gomez, & Pazos, 1999)

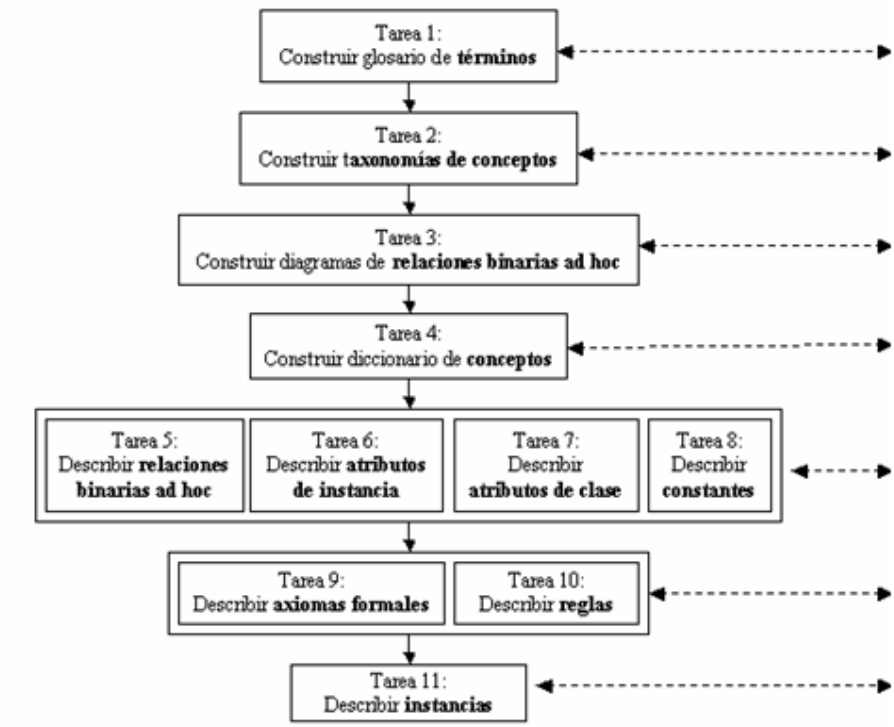

Figura 5. Modelado Conceptual (Corcho, Fernández, & Goméz, 2004)

Construcción de Glosario de Términos: identifica los términos relevantes del dominio (conceptos, instancias, atributos, relaciones entre conceptos, etc.) así como sus descripciones en lenguaje natural, sus sinónimos y acrónimos. (Corcho, Fernández, & Goméz, 2004)

Ejemplo:

Tabla 2. Ejemplo de Glosario de términos

Nombre	Sinónimos	Acrónimos	Descripción	Tipo
TipoCifrado	-	-	Una propiedad de la Criptografía	Propiedad
Mensaje	-	-	Texto que será encriptado	Propiedad

Construir taxonomías de conceptos: Selecciona del glosario los términos que son *conceptos*. La taxonomía define la jerarquía entre conceptos del dominio.

Ejemplo:

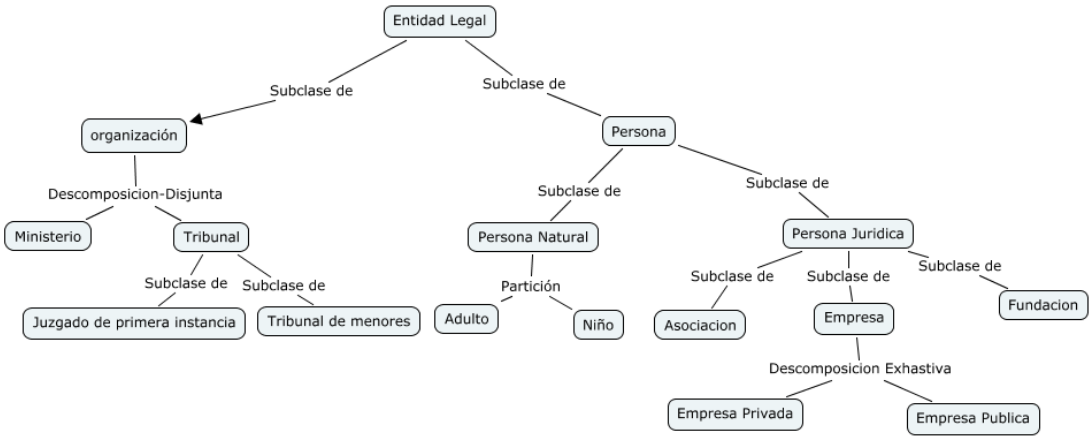

Figura 6. Ejemplo de construcción de Taxonomías de conceptos (Corcho, Fernández, & Goméz, 2004)

Construir diagramas de relaciones binarias ad hoc: El objetivo es establecer las relaciones existentes entre conceptos de la ontología y sus inversas.

Ejemplo:

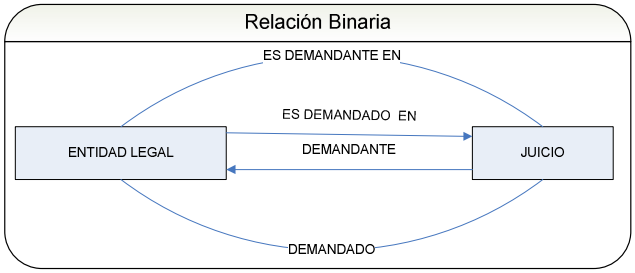

Figura 7. Ejemplo de Construcción de Relaciones Binarias

Construir diccionario de conceptos: De cada concepto de taxonomía se especifican sus propiedades, las relaciones del diagrama de relaciones binarias y las instancias de cada uno de los conceptos. (Corcho, Fernández, & Goméz, 2004, p.6-7)

Ejemplo:

Tabla 3. Ejemplo de Construcción de Diccionario de Conceptos

Nombre del Concepto	Instancias	Atributos de Clase	Atributos de Instancias	Relaciones
Tribunal	Tribunal Supremo	--	Número de miembros ubicación	realiza
Juicio	--	--	--	Demandante Demandado Se realiza en

Una vez que el diccionario de concepto se construye, el ontólogo debe definir en detalle cada uno de las relaciones provisionalmente binarias, atributos de instancia y clase que los atributos identificaron en el diccionario de concepto, así como también las constantes principales de ese dominio.

Describir relaciones binarias ad hoc: Por cada relación binaria se debe especificar: nombre, nombre de concepto origen y destino, cardinalidad y relación inversa en caso de tener.

Ejemplo:

Tabla 4. Ejemplo de Detalle de Relaciones Binarias

Nombre de la relación	Concepto origen	Cardinalidad máxima	Concepto destino	Relación inversa
demandante	juicio	N	persona	es demandante en
demandado	juicio	N	persona	es demandado en

Describir atributos de instancias: Para cada atributo de instancia especificar: nombre, concepto al que pertenece, tipo de valor, rango de valores y cardinalidad.

Ejemplo:

Tabla 5. Ejemplo de Atributos de Instancias

Nombre (Atributo de Instancia)	Concepto	Tipo de valor	Rango de valores	Cardinalidad
número de miembros	tribunal	Entero	1 ..	(1, 1)
ubicación	tribunal	Cadena de caracteres	--	(1, 1)

Describir atributos de clase: Para cada atributo especificar: nombre del atributo, nombre del concepto donde el atributo se define, tipo de valor, valor(es) y cardinalidad.

Ejemplo:

Tabla 6. Ejemplo de Detalle de Atributos de Clase

Nombre del Atributo Clase	Concepto	Tipo de valor	Cardinalidad	Valores
Tipo de Control	Compañía privada	[privado, público]	(1,2)	privado

Describir constantes: Por cada constante de términos se debe especificar: nombre, tipo de valor, valor, unidad de medida (para constantes numéricas).

Ejemplo:

Tabla 7. Ejemplo de Detalle de Constantes

Nombre	Tipo de valor	Valor	Unidad de medida
Edad de mayoría de edad	Cardinalidad	18	año

Una vez que los conceptos, taxonomías, atributos y relaciones se han definido, el ontólogo deberá construir los axiomas formales (*Describir Axiomas Formales*) y las reglas (*Describir Reglas*) que sirven para las restricciones y para ir infiriendo valores para los atributos.

Y solo opcionalmente los ontólogos deben introducir información acerca de instancias (*Describir Instancias*). (Corcho, Fernández, & Goméz, 2004, p.8-11)

Componentes de la Ontología

El acoplar ontologías comunes para todos los que participen dentro de la web semántica es una buena idea ya que de esta forma al trabajar (contribuyendo o consumiendo) los recursos serán compatibles.

Las Ontologías poseen los siguientes componentes que serán de ayuda para la representación del conocimiento de algún dominio: (Lozano, 2001)

Conceptos: Ideas básicas que se intenta formalizar, estos son: clases de objetos, métodos, planes, estrategias, etc.

Ejemplo: La música -- La salsa -- Merengue : son conceptos dentro de los Géneros de música.

- ✓ ***Relaciones:*** Representan la interacción y enlace entre los conceptos del dominio. Suelen formar la taxonomía del dominio.

  Ejemplo: Algunos ejemplos pueden ser: conectado_a, subclase_de, parte_de, etc.

  Merengue --es subclase_de → **Música**

- ✓ ***Funciones:*** Son un tipo concreto de relaciones donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología.

  Ejemplo: cant_productos → devuelve la cantidad de productos.

- ✓ ***Instancias:*** Se utilizan para representar objetos determinados de un concepto.

  Ejemplo: *Música* → Clásica → Romántica → Opera

- ✓ ***Axiomas:*** Teoremas que se declaran sobre relaciones que deben cumplir los elementos de la ontología.

Ejemplo: Aplicaciones → nombre Todas las aplicaciones deben tener un nombre.

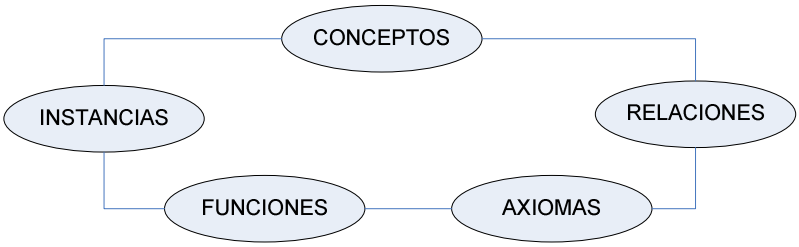

Figura 8. Componentes de una Ontología

1.5. Editor de Ontologías

En gran parte las herramientas de edición de ontologías no satisfacen totalmente las necesidades de explotación y manejo de las características de los lenguajes. Pero gracias a la alta expresividad de los lenguajes semánticos se soluciona este inconveniente. (García, Samper, & Castillo, 2006, p.4)

PROTÉGÉ

Es una herramienta de sistemas basados en el conocimiento y para el desarrollo de ontologías, para la solución de problemas y toma de decisiones en dominios particulares. Soporta frames, XML schema, RDF, OWL. Es una herramienta de código abierto desarrollada por Stanford Medical Informatics. Esta herramienta está separada en dos partes, la primera el modelo donde se presenta el mecanismo de representación interna de ontologías y bases de conocimiento; la segunda de visualización donde se brinda al usuario una interfaz para manipular el modelo. (Floréz, 2008, p.5)

Tareas que realiza:

- Creación de una ontología.
- Optimización de datos de entrada de formularios.
- Inserción de datos.

Plugin OWL

Permite editar una ontología al mismo tiempo; así como editar archivos y base de datos OWL. Una característica clave de las ontologías que se describen mediante OWL–DL es que pueden ser revisadas por un razonador, uno de los servicios que tiene es probar si una clase es sub-clase de otra clase. (Knublauch, 2005)

Al realizar estas pruebas con todas las clases de la ontología es posible calcular la jerarquía de la clase a inferirse. Otro servicio es comprobar la consistencia de una ontología, con

base en la descripción de una clase se comprueba si la clase es posible con todas sus instancias. Soporta tres sub-lenguajes Owl Lite, Owl DL y Owl Full. (Floréz, 2008, p.4)

Gracias a la interfaz DIG (estándar o protocolo que se introduce para proporcionar una interfaz común para los razonadores DL) que posee Protégé, se realiza la prueba con los diferentes razonadores. (Bechhofer, 2003)

Razonamiento con Protégé

Cada Razonador de Lógica de Descripción puede proporcionar los siguientes servicios de inferencia (Papataxiarhis, 2007)

- **Verificar la consistencia:** Controlar la consistencia entre las propiedades de objetos y restricciones asignadas a las clases; de tal manera que no contenga hechos contradictorios.

- **Clasificación de taxonomías:** Busca la jerarquía de inferencia de la ontología basada en la jerarquía de aserción que es la clasificación que se obtiene en la construcción de la ontología. En la clasificación de taxonomías genera una jerarquía de inferencia en la ontología que es diferente de la jerarquía de inserción (adición de una subclase a una clase)

- **Computación de tipos inferidos**: Permite computar los tipos de objetos dentro de la ontología.

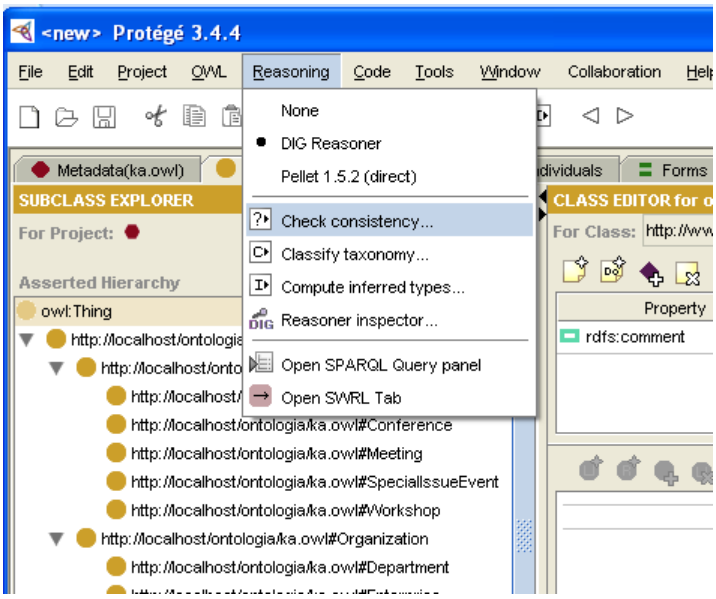

Figura 9. Tareas para el Razonamiento con Protégé

En estas tareas se habla sobre dos conceptos: Jerarquía de inferencia y Jerarquía de Inserción.

Jerarquía de inserción: Son las clases y subclases que va creando el usuario, como se muestra en la Figura 10.

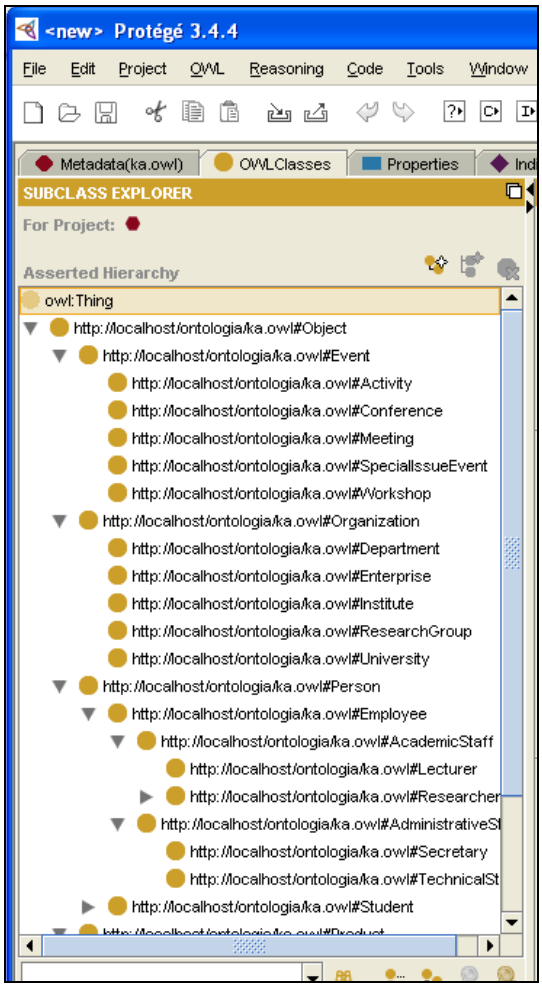

Figura 10. Jerarquía de Inserción

Jerarquía de Inferencia: Una vez realizada la tarea de clasificación de la taxonomía a cada clase se aplican las restricciones que se le imponen a cada clase de tal manera que se organizan las clases y subclases.

Almacenamiento de la ontología

1. Ficheros de texto estándar
2. RDF Schema
3. Base de datos JDBC

Almacenamiento en base de datos

Soporta cualquier base de datos con un driver JDBC, en la práctica se han probado: Oracle, MySQL, Microsoft SQL Server y Microsoft Access.

Formatos de importación/exportación de la ontología

RDF(S), XML Schema, RDB schema a través del plug-in Data Genie.

Interfaz Gráfica

Visualización de clases y propiedades globales con el plug-in GraphViz y vista de gráficos anidados con el plug-in Jambayala.

Creación de la Ontología

Pasos básicos para la creación de una ontología.

1. Creación de un proyecto.
2. Creación de clases y subclases
3. Creación de slots(atributos de las clases)
4. Creación de instancias(datos reales del conocimiento base)

Una vez terminada la ontología se almacena en la estructura de directorios con extensión RDF, uno contiene los datos y el segundo la estructura de la jerarquía de clases. (Goméz & Pérez, 2006, p.6)

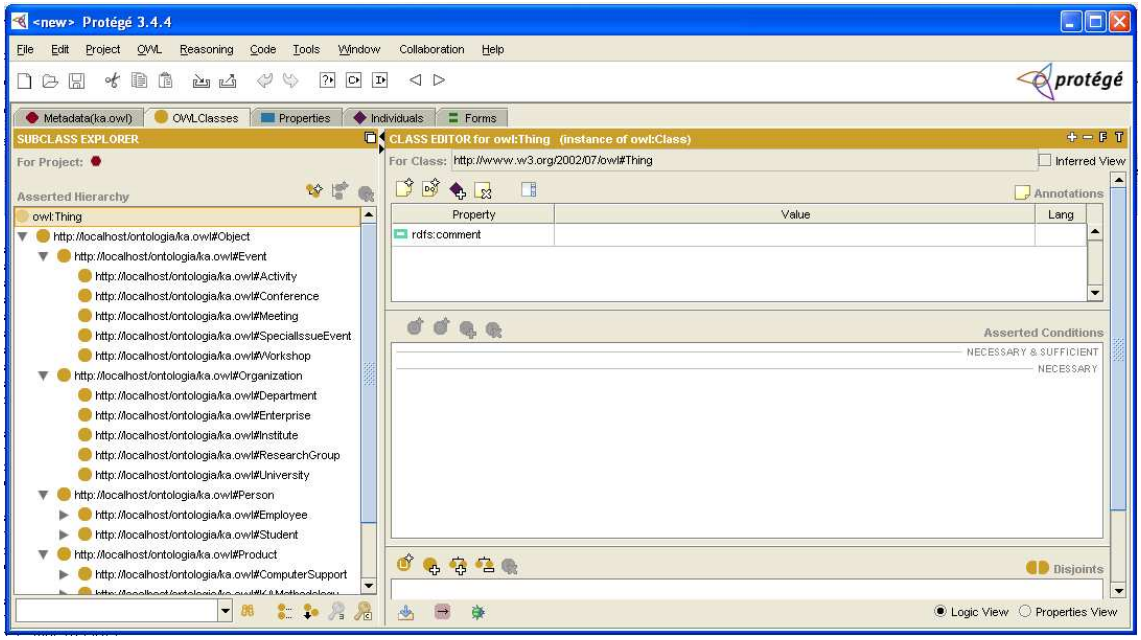

Figura 11. Editor de Ontología Protégé

Además esta herramienta hace posible la descripción de conceptos así como otras facilidades como tener un conjunto de operadores (and, or, y la negación); basado en un modelo lógico que garantiza que los conceptos están bien descritos, este modelo permite usar un razonador que chequea la consistencia descrita en la ortología garantizando que la jerarquía sea correcta.

ONTOEDIT

Es una herramienta que apoya el desarrollo y mantenimiento de las ontologías utilizando medios gráficos en un entorno web. Permite la representación semántica de lenguajes

conceptuales y estructuras mediante conceptos, jerarquía de conceptos, relaciones y axiomas.

Es libre, y se acopla a las necesidades del usuario, sus datos son manipulaos en una base de datos relacional. (OntoEdit, 2004)

Además esta herramienta tiene la posibilidad de convertir la ontología a diferentes lenguajes como EDF y DAML +OIL. Su visualización la realiza mediante un grafo dirigido con lo que permite identificar sus clases, subclases y relaciones entre ellas. (Chang & Gramajo, 2007)

Plug-In de OntoEdit

Plug-in de Conceptos y Relaciones

Permite la creación de la ontología donde se especifica las clases, subclases y sus relaciones (propiedades); así como sus respectivos atributos de cada clase. En este plugin se puede crear, editar y elimar las clases y sus respectivas relaciones; aquí se incluye la respectiva documentación del porque de la clase. (Chang & Gramajo, 2007, p.82)

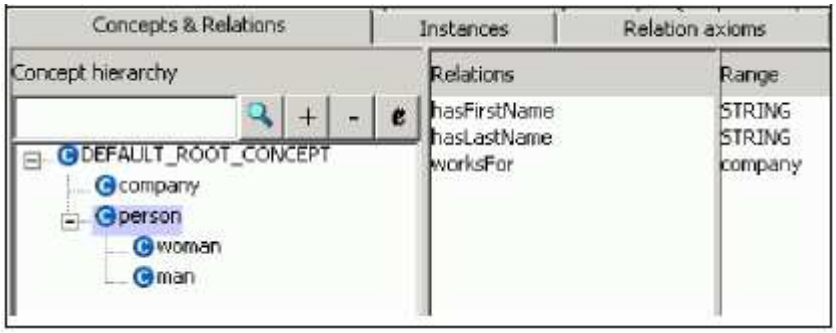

Figura 12. Plug-in de Conceptos y Relaciones

Plug-in de Instancias

Para esta sección las instancias son creadas, modificadas o eliminadas según lo requiera el usuario, cabe destacar que las instancias en cada clase tienen que ser creadas en el nivel más bajo de la jerarquía de la clase.

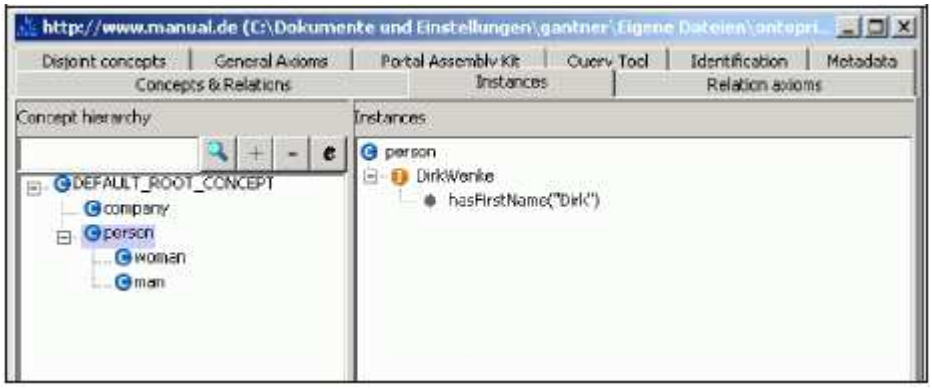

Figura 13. Plug-in de Instancias

Plug-in de Axiomas

Los axiomas representan un agregado más para la ontología, aquí se agrega reglas de pertenencia para las diferentes relaciones creadas. Se debe mencionar que los valores que se generen por cada axioma deben ser par.

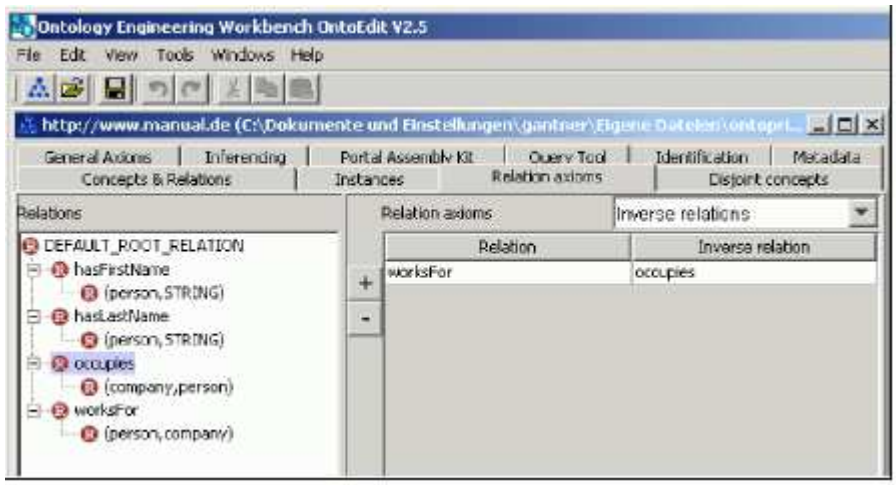

Figura 14. Plug-in de Axiomas

Plug-in Visualizador y de Consultas

El visualizador da la opción de editar los componentes de una ontología, así como poder realizar búsquedas, así como mostrar la ontología en forma de grafo dirigido. Este plug-in se puede ver en la Figura 15. Plug-in de Visualizador. (Chang & Gramajo, 2007, p.83)

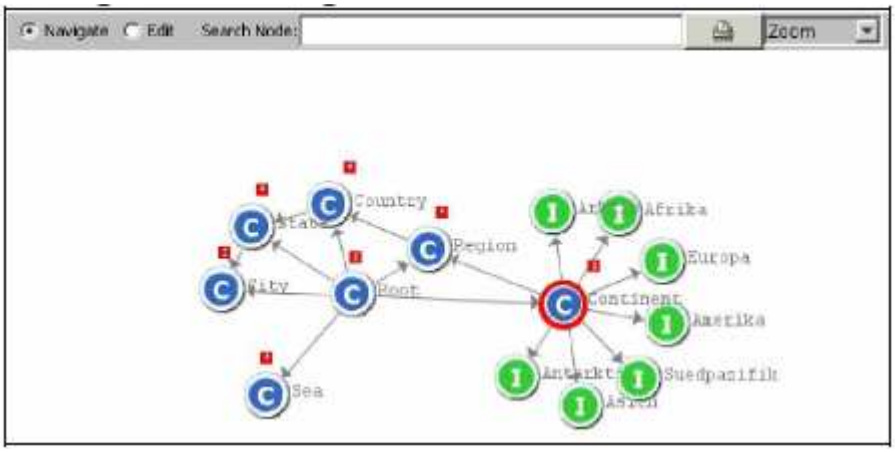

Figura 15. Plug-in de Visualizador

Para realizar las consultas la herramienta OntoEdit tiene un plug-in específico para las consultas sobre la ontología como se muestra en la Figura 16. Plug-in Consultas

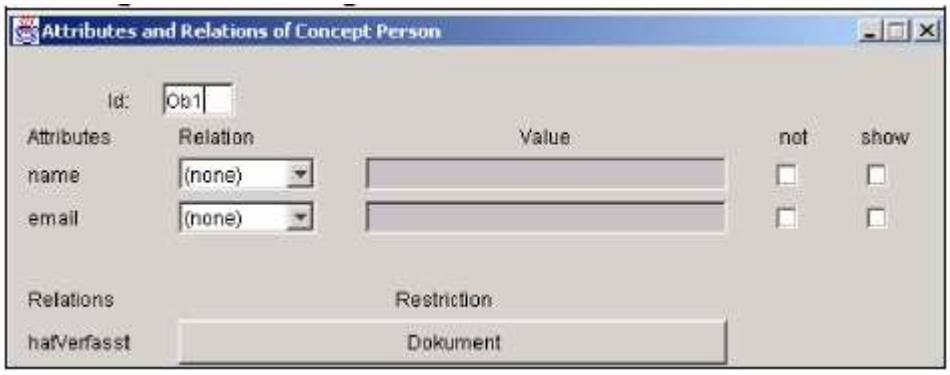

Figura 16. Plug-in Consultas

Tareas: (OntoEdit, 2004)

- Importar las estructuras de directorio
- Importar tablas de Excel
- Construir reglas graficas
- Dispone de reglas que eliminan fallos en la visualización del gráfico.
- Visualiza y edita ontologías en un gráfico.

Arquitectura de OntoEdit

Sigue el modelo cliente- servidor

- Conjunto de aplicaciones: Se manipula ontologías en la web, comprende un modelo conceptual y su implementación lógica y física.
- Conjunto de Datos: Datos manipulados por la herramienta y gestionada por un SGBD relacional.
-

Exportación/Importación

Utiliza el formato DAML+OIL para exportar o importar ontologías como se ilustra en la Figura 17. Exportación/Importación de OntoEdit.

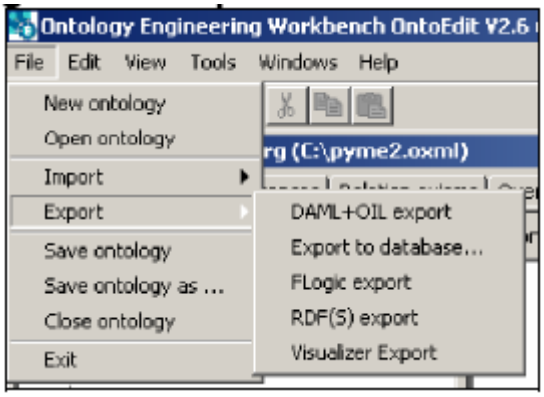

Figura 17. Exportación/Importación de OntoEdit

1.6. Lenguajes para Ontologías

A continuación algunos lenguajes:

El lenguaje XML: Es una forma de escribir datos, independiente de lenguajes, plataformas y herramientas dando una estructura sintáctica para que sean interpretados por computadoras. (Lamarca, 2009)

SHOE: *Simple HTML Ontology Extensions*. Es el primer lenguaje de etiquetado para diseñar ontologías en la Web. Este lenguaje nació antes de que surgiera la Web Semántica. Las ontologías y las etiquetas se insertan en archivos HTML. Permite definir clases y reglas de inferencia, pero no negaciones o disyunciones.

OIL: *Ontology Inference Layer*. Este lenguaje está basado en SHOE. Utiliza la sintaxis del lenguaje XML y está definido como una extensión de RDFs. Se basa en la lógica descriptiva (declaración de axiomas) y en los sistemas basados en frames (taxonomías de clases y atributos). ***OIL*** posee varias capas de sub-lenguajes, entre ellas destaca la capa base que es RDFS, a la que cada una de las capas subsiguientes añade alguna funcionalidad y mayor complejidad. Una desventaja de este lenguaje es la falta de expresividad para declarar axiomas.

DAML Y OIL: Este lenguaje se creó con la colaboración de OIL y DARPA que une los lenguajes DAML (DARPA's Agent Markup Language) y OIL (Ontology Inference Layer). Se basa en estándares del W3C. El lenguaje DAML se desarrolló como una extensión del lenguaje XML y de Resource Description Framework (RDF) y para extender el nivel de expresividad de RDFS. ***DAML- OIL*** hereda muchas de las características de OIL, pero se aleja del modelo basado en clases (frames) y potencia la lógica descriptiva. Es más potente

que RDFS para expresar ontologías. Sin embargo, este lenguaje presenta algunas carencias debido a su complejidad conceptual y de uso, complejidad que se intentó solventar con el desarrollo de OWL.

OWL: *OWL Web Ontology Language* o Lenguaje de Ontologías para la Web es un lenguaje de etiquetado semántico para publicar y compartir ontologías en la Web. Es recomendado por W3C, y puede usarse para representar ontologías de forma explícita, es decir, permite definir el significado de términos en vocabularios y las relaciones entre aquellos términos (ontologías). En realidad, OWL es una extensión del lenguaje RDF y emplea las tripletas de RDF, aunque es un lenguaje con más poder expresivo que éste. Se trata de un lenguaje diseñado para usarse cuando la información contenida en los documentos necesita ser procesada por programas o aplicaciones, en oposición a situaciones donde el contenido solamente necesita ser presentado a los seres humanos

KIF: *Knowledge Interchange Format*es un lenguaje para representar ontologías basado en la lógica de primer orden. KIF está basado en la lógica de predicados con extensiones para definir términos, meta-conocimiento, conjuntos, razonamientos no monotónicos, etc.; y pretende ser un lenguaje capaz de representar la mayoría de los conceptos y distinciones actuales de los lenguajes más recientes de representación del conocimiento. Se trata de un lenguaje diseñado para intercambiar conocimiento entre sistemas de computación distintos, diferentes lenguas, etc.; y no para la interacción entre seres humanos.

FOAF: aunque no es exactamente un lenguaje de ontologías ya que se trata de un vocabulario con definiciones que usa el lenguaje RDFS/OWL, FOAF hace más fácil que el software procese los términos del vocabulario FOAF para describir documentos. FOAF permite crear una base de datos unificada de información al normalizar una forma de definir categorías, tipos de relaciones, etc.

1.7. Áreas de Aplicación

- ***Gestión del conocimiento y modelado en las organizaciones***.- Se plasma una visión general de la gestión del conocimiento empresarial, de sus diferentes elementos relacionados a sus activos de conocimiento y al modelado, para después dar un breve bosquejo de web semántica, donde existen algunas herramientas, tecnologías y software relacionado a lo misma. (Barceló, Guzmán, & Pérez, 2006)

- ***Procesamiento del lenguaje natural.-*** Trata de la gestión automatizada de cualquier fuente escrita o hablada en general no estructurada, es decir, textos y conversaciones de cualquier origen y dominio de aplicación. La finalidad es

conseguir desarrollar herramientas que lleguen a la compresión del lenguaje y, por lo tanto, a la comunicación de ideas y al razonamiento.

- Dentro de la web semántica en donde ha tenido su mayor apertura es en la creación de ***buscadores semánticos en español***.

- Otra de las ramas en donde está incursionando es la ***educación***, de tal manera que ayude a los investigadores a reunir información útil, rápida y eficientemente; gracias a las redes de aprendizaje personal, compuestas por personas y agentes conectados a través de la información semántica, haciendo del aprendizaje algo más dinámico y mucho más fácil.

  Además con el aprendizaje y la organización educativa, como cursos, clases, pruebas, grupos de trabajo, será más fácil de organizar con los agentes que de forma automática hagan un seguimiento de estas tareas, dando como resultado un mejor control y organización en todas las tareas.

1.8. Aprendizaje del Capítulo

- La web semántica es una extensión de la web actual, en la que se trata de dar una estructura semántica a los contenidos de la web, que sean interpretados por agentes inteligentes y realizar tareas complejas. Para lograr esto se necesita de lenguajes que expresen datos y reglas para razonar sobre los datos, dando así una flexibilidad a los sistemas de representación de conocimiento.

- Dentro de la web semántica las ontologías cumplen un papel muy importante ya que son las encargadas de representar el conocimiento, estas definen conceptos y relaciones de un dominio específico de manera compartida; esta debe ser expresada de manera formal, legible y que sea utilizada por las computadoras.

- Para la creación de ontologías se debe seguir un proceso que asegure el buen desarrollo y funcionamiento de la misma, hoy en día existen muchas metodologías que ayudan al buen desenvolvimiento, la investigación ha llevado a definir a Methontology como una de las metodologías que mejores beneficios presta para la creación de ontologías.

- Una vez que la ontología ha sido creada se requiere de una herramienta que permita su desarrollo, en este caso Protégé tiene las mejores características para la creación, gestión y pruebas de la ontología. Existe buena información además de un buen soporte para la utilización de la misma; así como ejemplos que guían el buen desempeño en el uso de la herramienta.

- Las tecnologías de la web semántica buscan desarrollar una web más interrelacionada, donde sea fácil encontrar, compartir e integrar la información y los servicios; para esto las computadoras deben tener acceso a estructuras de información y conjunto de reglas de inferencia para que tengan un razonamiento automático, y esto lo logra gracias a las ontologías.

- Existen numerosas áreas de aplicación para la web semántica, conforme pasa el tiempo y la tecnología avanza factores como la poca información y escases de herramientas ya no son un limitante para incursionar en estas áreas y así desarrollar investigaciones de alto nivel.

CAPITULO II
LÓGICA Y RAZONAMIENTO PARA LOS RAZONADORES SEMÁNTICOS

Introducción

En el presente capítulo se revisará conceptos básicos de la lógica y razonamiento que utilizan los distintos razonadores semánticos, esto conlleva al estudio de la Lógica Descriptiva que es un lenguaje para la representación del conocimiento, y utilizados como base para los lenguajes ontológicos.

A pesar de haberse dado una introducción de los lenguajes ontológicos ya en el Capítulo I en esté se profundizar un poco más sobre las potencialidades de cada uno. Aquí se incluye los algoritmos de razonamiento en los que se basan los razonadores semánticos.

Con el desarrollo de estás temáticas se pretende desarrollar conocimiento de OWL DL dentro de las diferentes lógicas que existen, dando a conocer así cuales tienen un buen contenido semántico.

Objetivos

- ✓ Estudiar los conceptos fundamentales de lógica y razonamiento para los lenguajes ontológicos.
- ✓ Investigar sobre los tipos de lógica que existen y cuál de ellas poseen características semánticas.
- ✓ Desarrollar y Estudiar sobre los métodos con los que trabajan los razonadores semánticos y su razonamiento.

2. Sistemas de Web Semántica Actual

Los sistemas de la web semántica actual se basan, en una Base de Conocimiento (***KB*),** y en algún Motor de Inferencia (***IE***):

- ➢ **Base del Conocimiento**

(García, 2009:26) definió base del conocimiento como: “La unión del conjunto de aserciones y conjunto de reglas; contiene el conocimiento que el sistema experto maneja, es decir; una formulación simbólica, automáticamente manipulable, del área de conocimiento sobre el cual el sistema es experto. La función de la **KB** es suministrar al **IE**, información sobre la naturaleza del problema a manejar.”

Conocida también como un tipo de base de datos para gestionar el conocimiento, brinda los medios necesarios para poder recolectar, organizar y recuperar el conocimiento computarizado. Para esto tiene un conjunto de sentencias en un lenguaje formal, que es aprovechado por los agentes inteligentes que reúnen varias sentencias sobre el tema que necesitan para construir su base de conocimiento. (Romero, 2007, p.61)

En conclusión *la base de conocimiento es similar a una base de datos donde se gestiona conocimiento, y tiene los medios necesarios para que esta información sea computarizada, mediante un conjunto de aserciones y conjunto de reglas. Su función es dar al motor de inferencia información sobre un problema determinado*.

- **Motor de Inferencia**

Permite al agente obtener conclusiones de la base del conocimiento y resolver un problema determinado, de la misma forma se obtiene nuevas sentencias a partir de sentencias ya existentes en la base de conocimiento, así por ejemplo; es más factible determinar si una sentencia es deducible a partir de sentencias de su base del conocimiento. Es importante recordar que un lenguaje lógico es un lenguaje formal para representar información de la que se deduce conclusiones y dispone de una sintaxis y una semántica. (Romero, 2007, p.62)

Analizando el concepto dado por Romero, *para que un motor de inferencia funcione necesita obtener información de la base de conocimiento; y parte de sentencias nuevas, basadas en sentencias ya existentes, para hacer más fácil la deducción de una sentencia.*

2.1. Definición de Razonamiento

Se declara un modelo, como un mundo estructurado que es tomado como base y da la facilidad de poder evaluar la veracidad o la falsedad de algún hecho. Un ejemplo sería: "salgo todos los días" y "voy al trabajo"; una deducción podría ser, que salgo todos los días y sea verdad, y que salga al trabajo puede ser falso o viceversa.

Así como esta deducción se dan algunos modelos más como posibilidades de un sistema para llegar a algunas contradicciones. (Franconi, 2002)

Si se habla de una ***KB*** y la frase **α (o función lógica, como combinación de frase)**

$$KB \models \alpha \qquad \text{[Fórmula. 2.1]}$$

Si **α** es verdad en todos los modelos (mundos posibles) de KB.

$$KB \models \alpha \text{ si } M(KB) \subseteq M(\alpha) \qquad \text{[Fórmula. 2.2]}$$

Siendo:

M (KB) Conjunto de todos los modelos de KB.

M (α) Conjunto de todos los modelos de que **α** puede ser verdadera.

Esto indica que la frase **α** es derivable de la base de conocimiento, ya que no es posible un modelo de KB que no sea modelo de **α.**

Una frase o función lógica **α** se deriva (deduce o infiere) de la KB por medio del ***procedimiento*** (**p).**

$$KB \mid -_p \alpha$$ [Fórmula. 2.3]

Esto es:

1. Una conclusión a la que se llega es una hipótesis de la KB que da; sólo conclusiones que son correctas.

Siempre que $KB \mid -_p$ **α**, es cierto que ***KB*|= α** [Fórmula 2.4]

2. Si algo se puede concluir de una KB, se puede inferir por medio del procedimiento ***p***. Es decir, si algo es correcto se lo obtiene por medio de **p.**

Siempre que ***KB*|= α** *es cierto que* $KB \mid -_p$ α [Fórmula 2.5]

En estos casos se espera que los procedimientos den conclusiones correctas y que siempre obtengan una conclusión. La lógica proposicional tiene procedimientos seguros y completos, pero también existen otras lógicas mucho más expresivas que también tienen estas características como por ejemplo la lógica del primer orden (**LPO**). (Areces & Rijke, 2001)

Otra de las características que interesa es que sus procedimientos sean eficientes computacionalmente hablando, por lo tanto se ven obligados a restringirlos LPO en forma de DL, por esto algunas veces se conforman con procedimientos que sean seguros, pero no completos.

2.2. Lógica Proposicional (LP)

Es un sistema lógico muy simple, basado en proposiciones atómicas de las que solo sé permite indicar algo Verdadero o Falso. (Barceló P. , 2008, p.3)

SINTAXIS DE LP

La lógica proposicional basa su sintaxis en un conjunto de proposiciones atómicas $\sum = \{A, B, C.....\}$ individuales e independientes entre sí, que se podrán combinar entre ellas a modo de fórmulas lógicas. (Romero, 2007, p.87)

Tabla 8. Sintaxis de Lógica Proposicional

Símbolo	Denominación
V	Verdadero
F	Falso
$\neg A$	Negación
$A \wedge B$	Conjunción
$A \vee B$	Disyunción

$A \rightarrow B$	Implicación
$A \leftrightarrow B$	Equivalencia

SEMÁNTICA DE LP

En esta lógica, se llama ***Interpretaciones I*** sobre un conjunto de átomos ∑ a una función que dará un valor verdadero o falso por cada proposición atómica.

$$\mathbf{I:}\ \sum \rightarrow \{T, F\} \qquad \text{[Fórmula. 2.6]}$$

La interpretación para **x** se representa de la siguiente manera:

$$\mathbf{I(x)\ o\ también\ x^I} \qquad \text{[Fórmula. 2.7]}$$

De esta manera se llega a la semántica; está es aplicable a las proposiciones atómicas.

Tabla 9. Semántica Lógica Proposicional.

Sintaxis	Significado	Explicación
$I\models A$	A^I	El átomo A es cierto en la interpretación I
$I\models \neg A$	**A** no se satisface en **I**	**A** es falso en la **I(**interpretación**)**
$I\models A \wedge B$	I|=A y además I|=B	A y B son ciertos en la interpretación I
$I\models A \vee B$	I|=A o I|=B	Por lo menos uno de los dos A o B es verdadero en la interpretación
$I\models A \rightarrow B$	Si I|=A, entonces I|=B	No es posible que B sea cierto en I y no lo sea A
$A \leftrightarrow B$	I|=A, si y sólo si I|=B	A y B son verdaderas o Falsas a la vez en **Interpretaciones**
I|=V	Es "verdadero" en cualquier interpretación I	
I|≠F	Es falso en cualquier interpretación I	

Equivalencias:

Tabla 10. Equivalencias

Formulas	Equivalencias
$A \rightarrow B$	$\neg(A \wedge \neg B)$
$\neg(A \vee B)$	$(\neg A \wedge \neg B)$

$\neg(A \wedge B)$	$(\neg A \vee \neg B)$
$A \vee (A \wedge B)$	A
$A \wedge (A \vee B)$	A
$\wedge, \vee, \leftrightarrow$ *son conmutativas y asociativas*	
$\wedge$ *es distributiva frente a* $\wedge$	
$\vee$ *es distributiva frente a* $\wedge$	

FORMULACIÓN DE LP

(Franconi, 2002) dice que una formulación lógica es:

- ✓ ***Posible***, *si hay alguna interpretación I que la satisface.*
- ✓ ***Imposible,*** *cuando ninguna interpretación es cierta, es decir, que ninguno de los posibles mundos puede darse.*
- ✓ ***Valida o tautología,*** *si con cualquier* ***I*** *hace que la fórmula sea verdadera.*
- ✓ ***In-validable,*** *si hay al menos una interpretación que hace que la formula no sea verdadera.*

En este campo de trabajo una formulación será el conjunto de la base de conocimiento, a la que poco a poco se ira añadiendo sub-fórmula o frases nuevas para comprobar si se deducen de la KB, se contradicen, o solo son invalidas.

Las formulas algunas veces necesitan ser normalizadas, de modo que sean aplicables para varios algoritmos que necesiten la normalización; a continuación algunas formas normales:

Forma Normal Conjuntiva (FNC): Permite dejar todo como proposiciones atómicas (negadas o no) enlazadas entre sí por disyunciones (sub-fórmulas) y éstas enlazadas por conjunciones.

Si todas las sub-fórmulas contienen V (*Verdadero*) o proposiciones atómicas complementarias, es una tautología. (Alonso, Hidalgo, Martin, & Ruiz, 2003, p.4) **Forma Normal Disyuntiva (FND):** Deja todo como proposiciones atómicas (negadas o no) enlazadas entre sí por conjunciones y éstas a su vez enlazadas por disyunciones.

De esta manera, todas las sub-fórmulas contienen F (*Falso*) o proposiciones atómicas complementarias, es imposible. (Alonso & Cordón, 2003, p.8)

Forma Normal Negada (FNN): Sólo están negadas las proposiciones atómicas, de tal manera que no quede ningún paréntesis negado. Se utiliza para el procedimiento Tableux.

Forma Horn (HF): Deja todas las sub-fórmulas como máximo una proposición sin negar (no siempre se logra esta transformación). La parte que no se niega es la causa de la implicación y la negada es la consecuencia de dicha implicación.

A continuación un ejemplo:

$$(A \vee \neg C) \wedge (\neg C \wedge \neg B \wedge A) \qquad \text{[Fórmula. 2.8]}$$

Transformándose en im

$$(C \rightarrow A) y (A \rightarrow (C \vee B) \qquad \text{[Fórmula. 2.9]}$$

RAZONAMIENTO DE LP

El objetivo es permitir que un sistema inteligente compruebe conclusiones, basándose en la KB; considerada en la lógica proposicional como un conjunto de fórmulas lógicas, de forma que una interpretación **I** satisface a ésta y esto es posible sólo si satisface todas y cada una de sus fórmulas. (Romero, 2007, p.90)

I |= KB Si I|=A, para todo A $\in$ KB [Fórmula. 2.10]

Para saber si una fórmula Z se puede inferir a partir de la KB, se basa en obligar a que se cumpla Z (sea verdadera) en todas las interpretaciones que satisfacen la KB.

KB|=Z si I|=Z para todos los modelos I de la KB [Fórmula. 2.11]

Hay dos procedimientos para resolver las inferencias: R*azonamiento por enumeración* o por el *método tableaux* los cuales se explican a continuación.

- **Razonamiento por Enumeración:**

Consiste en poner en una tabla todas las combinaciones posibles (proposiciones atómicas) y ver las que son modelo de KB (todas con el resultado de verdadero en las fórmulas de KB) y del mismo modo que den como resultado *verdadero* en las fórmulas que se está infiriendo.

Éste permite realizar un estudio de todas las posibilidades, es decir; todos los métodos posibles; la ventaja es: realizar todos los tipos de comprobaciones al mismo tiempo. Una de las desventajas es que necesita un gran esfuerzo computacional, de tal manera que si se analiza ***x*** variables se realizar 2^x posibles combinaciones de verdadero y falso. (Romero, 2007, p.71)

Para una mejor explicación, se presenta el siguiente ejemplo:

$$KB = (A \vee C), (B \wedge \neg C) \qquad \text{[Fórmula. 2.12]}$$

$$\alpha = (A \vee B)$$ [Fórmula. 2.13]

KB |= α *(Comprobar si puede inferir* α *de KB)*

Tabla 11. Modelos Posibles 8(I1… I8)

A	*B*	*C*	*(AvC)*	$(B \wedge \neg C)$	*KB*	α	*I*
F	*F*	*F*	*F*	*F*	*F*	*F*	*1*
F	*F*	*V*	*V*	*F*	*F*	*F*	*2*
F	*V*	*F*	*F*	*V*	*F*	*V*	*3*
F	*V*	*V*	*V*	*F*	*F*	*V*	*4*
V	*F*	*F*	*V*	*F*	*F*	*V*	*5*
V	*F*	*V*	*V*	*F*	*F*	*V*	*6*
V	*V*	*F*	*V*	*V*	*V*	*V*	*7*
V	*V*	*V*	*V*	*F*	*F*	*V*	*8*

KB es verdadero en $\{I_7\}$

α Es cierto en $\{I_3 \ldots I_8\}$

M (KB) $\subseteq M(\alpha)$ (α cierto en TODOS los modelos posibles de KB)

KB|= α ***SI*** *(se puede inferir* α *de KB)*

- **Razonamiento del Método Tableaux**

Conocidas como tablas semánticas, es un método de demostración por refutación, él mismo que; se basa en la semántica más que en la sintaxis de las fórmulas; sin embargo se requiere de una clasificación de las fórmulas basadas en la sintaxis.

Es el método más óptimo computacionalmente hablando en comparación con el razonamiento por enumeración pero, sólo se puede resolver si es un conjunto de fórmulas o no. Por tal motivo se deberá realizar algunas modificaciones para llegar a otras conclusiones. (Miranda, Reyes; 2008, p.2)

Un tableaux es un árbol binario donde cada nodo está etiquetado con una fórmula. Éste consiste en reducir el sistema en su forma NNF de conjunciones y disyunciones (será el nodo raíz de nuestro árbol Tableaux). (Romero, 2007, p.94)

A continuación un ejemplo:

$$(A \vee C), (B \wedge \neg C), (A \vee B)$$ [Fórmula. 2.14]

Desarrollo del árbol:

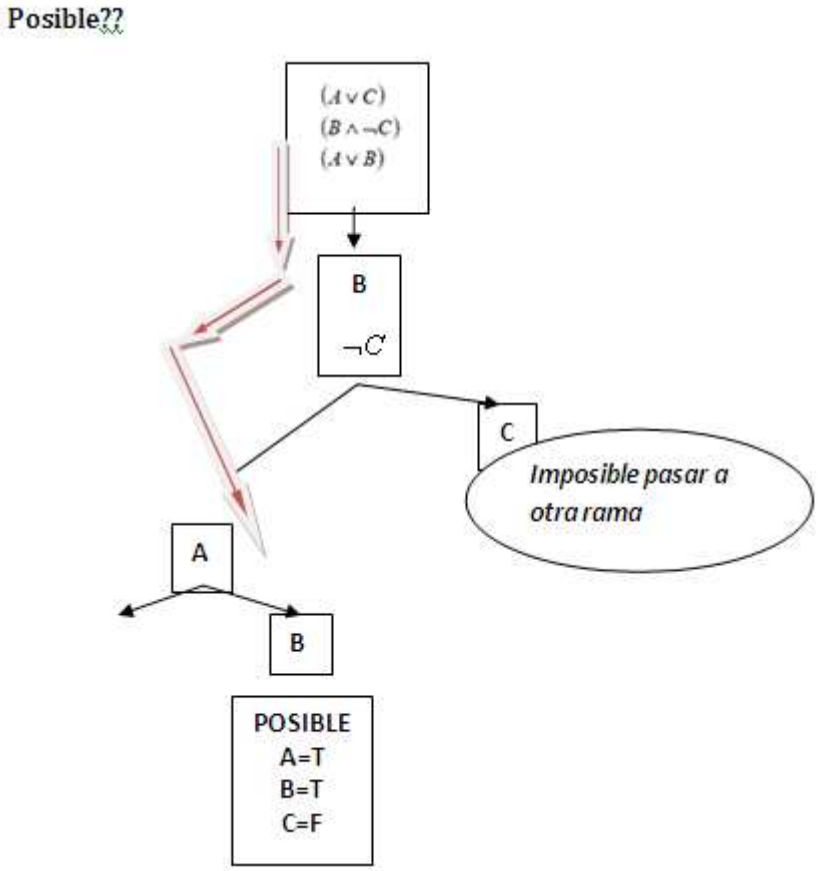

Figura 18. Árbol Binario Tableaux

- **i.** Por cada conjunción se sigue la rama
- **ii.** Por cada disyunción se divide la rama
- **iii.** Se sigue hasta que una rama es imposible (contradicción) o se han dividido todas las fórmulas a átomos(A, B, C).
- **iv.** Si todas las ramas son imposibles, el sistema es imposible.

El método *Tableaux* es más eficiente (si se parte de la forma NNF) que la enumeración, ya no es necesario recorrer todas las posibilidades.

2.3. Lógica de Primer Orden (LPO)

Es una implicación de la lógica proposicional; estudia las frases declarativas a un mayor tipo de detalle, respetando su estructura interna; se toman elementos básicos como las entidades individuales (*objetos*), con características distintivas (*propiedades*), que tienen varios tipos de relaciones, entre ellas (f*unciones*). (Labra & Fernandez, p.2)

Su importancia se debe a la estructura del mundo en objetos y relaciones que facilita el razonamiento, dando la libertad al usuario de acoplar su mundo a sus necesidades; además permite expresar sentencias sobre todos los objetos del universo.

Un ejemplo 1:

"La pelota es redonda"

REDONDA (pelota) [Fórmula. 2.15]

Donde la **pelota** es un **objeto**

Y **REDONDA** es una **propiedad del objeto**. (Símbolo predicativo)

Adicional a esto cuando se utiliza cuantificadores se pone variables en los predicados.

Ejemplo 2:

$$\exists x\left[REDONDA(X) \wedge BLANCA(X)\right] \quad \text{[Fórmula. 2.16]}$$

SINTAXIS EN LPO

Tabla 12. Sintaxis de la Lógica del Primer Orden

Símbolo	Denominación	Descripción
a,b	Constante	Un objeto del mundo
x,y	Variables	En el momento de hacer la aserción puede ser cualquier constante de nuestro mundo.
$F(t_1 \ldots t_n)$		T_i=a ó xó $F(t_1 \ldots t_n)$ (términos)
$F(a_1 \ldots a_n)$	Términos básicos	Términos que no incluyen variables
$\phi, \phi \equiv P(t_1 \ldots t_n)$	Fórmulas atómicas	Términos relacionados por un símbolo predicativo P
V	Verdadero	
F	Falso	
$\neg\phi$	Negación	
$\phi \wedge \phi$	Conjunción	
$\phi \vee \phi$	Disyunción	
$\phi \rightarrow \phi$	Implicación	
$\phi \leftrightarrow \phi$	Equivalencia	

$\exists x.[\phi]$	Cuantificador Existencial	Se refiere a al menos uno de los objetos que cumplen ϕ
$\forall x.[\phi]$	Cuantificador Universal	Se refiere a al menos todos los objetos que cumplen ϕ

SEMÁNTICA EN LPO

En esta lógica se llama a la interpretación ***I***

Tabla 13. Semántica LPO

Sintaxis	**Significado**
I, α |=P(t_1…..t_n)	$\langle t_1^{I,\alpha},...,t_n^{I,\alpha}\rangle \in P^I$
I, α |= $\neg\phi$	I, α |$\neq \phi, \phi$ no se satisface en I, α
I, α |= $\phi \wedge \phi$	I, α |= ϕ y además I, α |= ϕ
I, α |= $\phi \vee \phi$	I, α |= ϕ o (no exclusiva) I, α |= ϕ
I, α |= $\phi \rightarrow \phi$	I, α |= ϕ entonces I, α |= ϕ
I, α |= $\phi \leftrightarrow \phi$	Si, I, α |= ϕ, si y sólo si I, α |= ϕ ($\phi \equiv \phi$)en esa interpretación I, α
I, α |= $\exists x.[\phi]$	Para todo d $\in \Delta : I, \alpha[x/d]$ |= ϕ
I, α |= $\forall x.[\phi]$	Existe al menos un d $\in \Delta : I, \alpha[x/d]$ |= ϕ

A las relaciones entre operadores de la lógica proposicional hay que añadir la de los cuantificadores.

Tabla 14. Cuantificadores

Fórmula	**Equivalente**
$\forall x.[\forall y.[\phi]]$	$\forall y.[\forall x.[\phi]]$
$\exists x.[\exists y.[\phi]]$	$\exists y.[\exists x.[\phi]]$
$\forall x.[\phi]$	$\neg\exists x[\neg\phi]$
$\exists x.[\phi]$	$\neg\forall x[\neg\phi]$
$\left\{\begin{array}{l}[\otimes x.[\phi]]\bullet\varphi=[\otimes x.[\phi\bullet\varphi]],\\ \otimes\in\{\exists,\forall\},\\ \bullet\in\{\wedge,\vee\}\end{array}\right\} si\varphi\_no\_depende\_de\_x,$	
$\exists$ suele llevar dentro $\wedge$ ej $\exists x.[\phi \wedge \phi]$	
$\forall$ Suele llevar dentro $\rightarrow$ ej. $\forall x.[\phi \rightarrow \phi]$	
$\forall x.[\exists y.[\phi]] \neq \exists y.[\forall x.[\phi]]$	
$\exists x.[\forall y.[\phi]] \neq \forall y.[\exists x.[\phi]]$	

En LPO se concluye el concepto de herencia, por medio de la inclusión. Dado P y Q como símbolos predicativos del mismo orden, se dice que P incluye a Q en la Teoría KB:

KB|=P $\supseteq$ Q [Fórmula. 2.17]

2.4. Lógica de Descripción (LD)

Llamada lógica descriptiva (LD), es un conjunto de lenguajes para representar el conocimiento expresado de manera terminológica en un dominio de aplicación de una forma estructurada y comprendida formalmente.

El significado de lógica descriptiva se refiere a dos aspectos, primero a la descripción de conceptos usados para describir un dominio y, en segundo lugar a la semántica que establece una equivalencia entre las fórmulas de lógicas de descripción y expresiones en lógica de predicados de primer orden. (Tsarkov & Horrocks, 2003)

A diferencia de otros sistemas de representación (redes semánticas frames), estas lógicas están dotadas con una semántica formal basada en la lógica entre sus características importantes se presenta:

- ***Un formalismo descriptivo***: conceptos, roles, individuos y constructores.
- ***Un formalismo terminológico***: axiomas terminológicos que introducen descripciones complejas y propiedades de la terminología descriptiva.
- ***Un formalismo asertivo***: introduce propiedades de individuos. Son capaces de inferir nuevo conocimiento a partir de un conocimiento dado; tienen por tanto; algoritmos de razonamiento que son de decisión.

Los elementos centrales del alfabeto del lenguaje de la lógica de descripción son:

- ***Nombres de concepto***: asignan un nombre a un grupo de objetos.
- ***Nombres de rol***: asigna un nombre a una relación entre objetos.
- ***Nombres de individuos (u objetos):*** los individuos son instancias de los conceptos y también se pueden relacionar por medio de un rol.
- ***Constructores (constructor):*** relaciona nombres de conceptos y nombres de roles, y también crea conceptos complejos a partir de los atómicos.

Los sistemas DL se basan en una KB que contiene aserciones sobre las clases (*terminología, T-Box*) y un conjunto de aserciones sobre instancias, o dicho de otra forma, representaciones de una situación concreta (A-Box). Sobre esta KB se realiza inferencias.

RAZONAMIENTO DE LD

Existen dos tipos de razonamiento: TBox, ABox. El primero realiza un razonamiento sobre la estructura de la ontología. Mientras que el segundo es un método más completo de recuperación de instancias y consultas que permite capaz de inferir información no sólo

sobre la estructura de la ontología sino también sobre las instancias. Esto permite mejorar los resultados en las consultas. (Roldán & Aldana, 2007, p.2)

Haciendo una comparación con la base de datos, TBox es la descripción de la Base de Datos (esquemas) y ABox las tuplas de la Base de Datos.

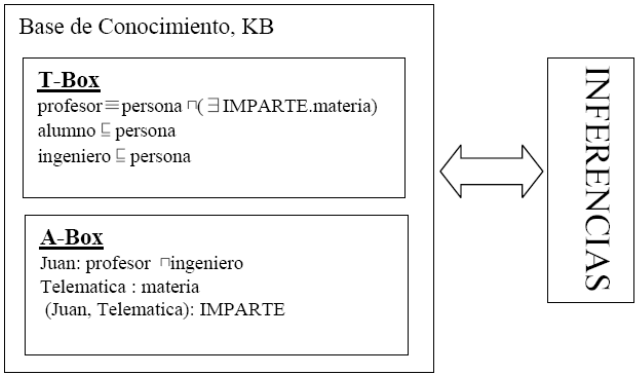

Figura 19. Arquitectura de un Sistema DL

T-Box contiene:

- ✓ Definición de conceptos(asignación de nombre)

  $A \equiv C$

  Siendo A *nombre del concepto* y C *su definición.*

  Profesor $\equiv$ persona $\cap \exists$ Imparte .materia
- ✓ Axiomas (restringe el modelo)

  $C_1 \subseteq C_2$ C_i son conceptos complejos

  Ejemplo: La clase alumno es subclase de la clase persona.

  Alumno $\subseteq$ persona

ABox contiene:

- ✓ Aserciones sobre conceptos

  a: C

  a *nombre de la instancia* que pertenece al concepto C.

  Ejemplo: María es maestra y además Doctora.

  María: maestra $\cap$ doctora

- ✓ Aserciones sobre roles

  (a, b): R ó R(a, b)

  a, b son objetos de nuestro sistema y R un rol (propiedad) del sistema.

  Ejemplo: María está relacionada por la propiedad ESTUDIA a la instancia Matemáticas.

  (María, Matemáticas): ESTUDIA

La T-Box contiene información relacionada únicamente con las clases, por lo que se la considera como parte del conocimiento general. En la A-Box se sitúa la parte de la KB

relacionada con las instancias, por lo que se considera a la A-Box como la parte concreta de la KB, la que trata de casos particulares.

SINTAXIS Y SEMÁNTICA DE LD

Existen varios lenguajes de lógica descriptiva, siendo un subconjunto de la sintaxis de LPO (aunque la notación se simplifica). Para objeto del estudio se toma el lenguaje denominado *ALC,* que es un lenguaje DL simple y restrictivo, tiene sólo predicados binarios que relacionan instancias abstractas (no números, string u otro tipo de dato definido ni la enumeración de sus elementos). (Alonso, Hidalgo, Martin, & Ruiz, 2006)

Además no permite asignar propiedades a los roles, pese a ser un lenguaje más sencillo, el resto utilizan la misma sintaxis, añadiendo solo las propiedades de los roles.

Tabla 15. Sintaxis y Semántica de ALC

Sintaxis	Semántica	Significado
C	$C^I \subseteq \Delta^I$	Concepto simple, primitivo o atómico. Es un subconjunto de Δ^I
R	$R^I \subseteq \Delta^I x \Delta^I$	Rol simple, primitivo o atómico
CцD	$C^I \bigcup D^I$	Conjunto de elementos definidos por la unión de los conceptos (unión de conjuntos)
$C \Pi D$	$C^I \bigcap D^I$	Conjunto de elementos definidos por la intersección de dos conceptos(intersección de conjuntos)
$\neg C$	Δ^I / C	Conjunto de elementos que abarcan todo nuestro espacio menos los individuos que agrupa C
$\exists R.C$	$\{x \mid \exists y.[R^I(x,y) \wedge C^I(y)]$	Conjunto de elementos que se relacionan al menos una vez por medio de R con otros elementos pertenecientes al concepto de C
$\forall R.C$	$\{x \mid \forall y.[R^I(x,y) \rightarrow C^I(y)]$	Conjunto de elementos que tienen todas sus relaciones con otros elementos pertenecientes al concepto C por medio de R, y no por otro medio.
T	Δ^I	Todo nuestro espacio (“top”)
$\perp$	ϕ	El conjunto vacío (“botón”)

Ahora se describe la semántica relacionada con LD, dada una interpretación I; definiendo de esta manera una Teoría del Modelo

$$I = (\Delta^I, .^I) \qquad \text{[Fórmula. 2.18]}$$

La teoría del modelo indica las reglas básicas de mi abstracción de la realidad, los únicos elementos son las instancias de los objetos y las instancias de parejas de objetos. Tanto los roles simples como los conceptos simples, son conjuntos de $[\Delta^I]$ y $[\Delta^I x \Delta^I]$ respectivamente. (Horrocks & Patel-Schneider, 2003)

2.5. Otro Tipo de Razonamiento

DATALOG (DATABASE LOGIC)

Es un lenguaje lógico que es la forma más simple de lógica desarrollada para el modelo relacional; está compuesto por un conjunto de cláusulas Horn. (Ruckhaus, 2009) Basado en la LPO para representar características estructurales de los datos.

El lenguaje de programación lógica es prolog, al igual que su sintaxis es muy semejante; el significado de los programas en datalog se define de manera declarativa, a diferencia de la semántica más procedimental de prolog.

Datalog simplifica la escritura de consultas simples y hace más sencilla la optimización de consultas. (Cohuo, s.f)

Modelo de datos de datalog

Datalog es una versión de prolog para base de datos, específicamente tienen dos diferencias:

- No admite símbolos de función en los argumentos.
- El significado de los programas datalog siguen el punto de vista de teoría de modelos.

El modelo de datos de datalog es similar al relacional, una relación se representa por un predicado. Sin embargo, sus argumentos siguen una notación proposicional, no explicita como en el modelo relacional (cada columna tiene un nombre).

Ejemplo:

Una instancia *r* de la relación R(A, B) en el modelo relacional definida

r

A	B
1	2
3	4

Se representa en Datalog por los hechos:
r(1,2)
r(3,4)
En datalog, el primer argumento de R se corresponde con el atributo A, y el segundo con B. El significado de la relación en ambos modelos de datos es el mismo, en el conjunto de tuplas {(1,2) (3,4)}. Es decir que hay una relación de tipo R entre 1, 2, 3 y 4.

2.6. Semejanzas y Diferencia entre lógica descriptiva y datalog

Datalog y lógica de descripción comparten estas características:

- ✓ Conjunción en la cabeza de las reglas.
- ✓ Cuantificación universal en la cabeza de las reglas.

- ✓ Cuantificación existencial en el cuerpo de las reglas.
- ✓ Predicados unarios y binarios
- ✓ Predicados/roles con inversas.
- ✓ Predicados transitivos, simétricos.

La lógica descriptiva tiene características que datalog no:

- ✓ Negación
- ✓ Disyunción
- ✓ Cuantificador universal en el cuerpo de una regla.
- ✓ Cuantificador existencial en la cabeza de una regla.
- ✓ Restricciones de cuantificación y cardinalidad.

También datalog tiene características que la lógica descriptiva no tiene:

- ✓ Predicados n-arios.
- ✓ Reglas que rompen el modelo de árbol.

2.7. Lenguajes Ontológicos

Una ontología se representa en base a un lenguaje ontológico que servirá para almacenar el conocimiento sobre algún dominio de interés. Las ontologías son utilizadas por personas y sistemas computacionales para compartir información de un dominio, entendiendo por dominio una porción determinada de un área de conocimiento. (Heflin, 2004)

Las ontologías tienen características que la diferencian de una base de datos:

- Almacena información de un sistema, donde las tablas con clases, las tuplas de las tablas son instancias y las relaciones entre instancias son relaciones simples o múltiples entre los identificadores de las tablas. (Philippe, 2002)
- Permite que los objetos tengan propiedades, que no estén en la definición de la clase a la que pertenecen.
- Admite múltiples definiciones de clases y propiedades. En una base de datos normalmente existe una tabla fija para el almacenamiento de información de un objeto.
- Un objeto no tiene por qué tener solo su información en un documento, a pesar de que existen bases de datos distribuidas, en estos lenguajes se permite mayor libertad.
- Al basarse en lenguajes lógicos se procede a deducir, que lo realizan sistemas no humanos. No se puede inferir automáticamente basándose en una base de datos.

Estas características permiten utilizar ontologías para mejorar aplicaciones web, ya sea para generar portales sobre temáticas determinadas para describir fuentes en internet, para describir la capa lógica de un sistema de tres capas, para diseñar cómo ha de ser la documentación, para su uso por agentes inteligentes.

- **Tipología de lenguajes ontológicos**

Hay variedad de lenguajes ontológicos, por lo que es necesario realizar una taxonomía de ellos. Una clasificación es: (Bechhofer S. , 2003)

- ✓ **Modelos de Representación**

Basados en lenguajes lógicos, sólo en la sintaxis de las ontologías (clases, instancias, propiedades que se relacionan entre sí). Dentro de este grupo están:

- **Redes semánticas.-** *entidades como un conjunto de nodos relacionados entre sí por fechas. Un ejemplo de esto es WorldNet*.
- **Mapas de conceptos.-** *denominado "topics maps"*, es una ampliación de redes semánticas que incluyen conceptos de ocurrencias, que son una referencia a fuentes externas al mapa de conceptos.
- **Uml.-** lenguaje que permite representar objetos y sus relaciones se puede entender como uno de estos lenguajes.
- **Rdf.-** *Infraestructura para la Descripción de Recursos,* brinda una base para procesar metadatos, proporciona interoperabilidad entre aplicaciones que intercambian información entendible por máquina en la web. (Lassila & Swick, 1999)

- ✓ **Lenguajes basados en lógica**

Se definen como lenguajes formales para representar la información, de manera que se infiera conclusiones. Tiene una sintaxis, que define el tipo de sentencias que pueden darse en el lenguaje, y una semántica, que define el significado de dichas sentencias. Los lenguajes que se utilizan para describir ontologías son los lenguajes *basados en lógica*.
Según la lógica tenemos:

- **Basados en Lógica Descriptiva.-** para este caso tenemos OIL, DAML + OIL, OWL.
- **Basados en Lógica de Primer Orden.-**como es el caso de KIF y sus sub-lenguajes. Se dice que la lógica de primer orden comprende a su vez la lógica descriptiva.
- **Basados en Reglas.-** se entiende como "unidades de conocimiento auto-contenidas que implican algún tipo de razonamiento" (Wagner, Tabet, & Bole, 2003)

- **Basados en lógicas de mayor orden.-** El mayor orden en este caso se refiere sintácticamente, es decir poder relacionar conjuntos de elementos de discusión (y no solo en pares)**.**
- **Lógicas no clásicas.-** No se basan en el concepto verdadero o falso, sino en grado de verdad (probabilidad) de una aserción.
- **Lógicas probabilísticas o difusas.-** Estos lenguajes suelen ser utilizados por sistemas de redes neuronales.

Es importante hacer la descripción del lenguaje ontológico más utilizado:

OWL: Es un lenguaje para publicar y compartir datos usando ontologías en la www. Tiene como objetivo facilitar un modelo de marcado construido sobre RDF y código XML. (Heflin, 2004)
Se representa en tres niveles de lenguaje, en función de su expresividad (Grigoris & Harmelen,s.f, p.4)

- **Owl Lite:** Es el más sencillo y sirve para aquellos usuarios que necesitan una jerarquía de clasificación y características simples de restricción.

  Ejemplo: Solo permite valores de cardinalidad de 0 o 1. Su lógica tiene unas restricciones numéricas muy limitadas.

- **Owl DL:** Útil para obtener la máxima expresividad sin perder la integridad de cómputo, los tipos de datos permitidos son los de XMLS.

  Ejemplo: Una clase puede ser sub-clase de otras clases, pero no un caso de una clase.

- **Owl Full:** Máximo nivel de expresión y libertad sintáctica de RDF. No tiene una garantía computacional.

  Ejemplo: Una clase es tratada a la vez como una colección de individuos y como un individuo por sí mismo.

SWRL: Un lenguaje desarrollado en owl ampliado con definiciones de reglas, las ampliaciones más relevantes son (Horrocks I. , 2004):

Está se basa en Owl Dl y Owl Lite y no es compatible con RDF ni RDFS,

- Definición de variables
- Definición de átomo
- Definición de conjunto de átomos

- Definición de reglas

Pero las herramientas que hay para la manipulación de este lenguaje son muy limitadas, por lo tanto su uso es un poco factible.

RuleML: Es un lenguaje basado en reglas estas comprenden a la lógica clásica; de forma que hay reglas presente en el lenguaje para definir más detalladamente las ontologías. Datalog sub-lenguaje de la lógica Horn es el núcleo de RuleML, es la intersección de SQL y Prolog. (Boley, Grosof, & Tabet, 2005)

Los tipos de reglas de RuleML:

- Regla de integridad.- sirve para definir por ejemplo una restricción de integridad.

- Regla de Derivación.- Es un conjunto de condiciones y una consecuencia si se dan las condiciones anteriores; representadas en una formula lógica.

- Reglas de Reacción.- Estas reglas tienen una condición inicial y una acción a realizar. Hay dos tipos Evento-Condición-Acción-Pos condición y Evento-Condición-Acción

- Reglas de Producción.- Son muy parecidas a las reglas de reacción; expresa que se dará cuando se dé una condición.

- Reglas de Transformación.- Son reglas que explican cómo pasar de una instancia de un tipo a su equivalente de otro tipo.

2.8. Aprendizaje del Capitulo

- Los tipos de lógica que existen son algunas como se ha demostrado en este capítulo, la diferencia radica en su forma de expresar el conocimiento unas son más expresivas que otras; la lógica descriptiva y la lógica de primer orden son una de las más utilizadas por la manera de descripción que consta de objetos y predicados de esos objetos. Pero también hay que destacar que todas las lógicas parte de la lógica proposicional que a pesar de ser muy limitada fue el punto de partida para que las demás lógicas puedan surgir con mejores características y cualidades de representación.

- Para la investigación se ha determinado que le uso de la Lógica Descriptiva es una buena opción, ya que es un subconjunto de la Lógica de Primero Orden que permite implementar razonadores que utilizan inferencias en tiempos razonables y por eso es el más utilizado en el campo de la web semántica.

- Como ya se ha dado a conocer existen muchos lenguajes que están a nuestro alcance para el desarrollo de ontologías pero hay algunos aspectos que se deben definir para la selección como: la definición del problema, el conocimiento que se desea representar, la expresividad que se necesita, y a que área será dirigida. Estos aspectos definirán el uso de un buen lenguaje que se acople a las necesidades del desarrollador.

CAPITULO III
ESTUDIO SOBRE LAS PRESTACIONES DE LOS RAZONADORES SEMÁNTICOS

Introducción

A continuación se hablará sobre los *Razonadores Semántico* que trabajan con la lógica descriptiva y el lenguaje owl*;* aquí se define que es un Razonador, y sus principales características, lo que servirá de base para la evaluación y comparación de los mismos.

Objetivos

- ✓ Crear conocimiento sobre los razonadores semánticos, de tal manera que esto represente la base del estudio.
- ✓ Realizar un estudio de las principales características de los razonadores más utilizados, de tal manera que se pueda seleccionar algunos de ellos para un estudio de comparación.

3. Razonador

- ➢ **Definición de Razonador Semántico**

(Carrol, Roo, 2004) define a un razonador como: "*Un comprobador de coherencia OWL tiene un documento como entrada y devuelve una palabra: de ser coherente, inconsistente, o desconocido*".

Un razonador es también conocido como motor de razonamiento o motor de reglas, una pieza de software capaz de deducir las consecuencias lógicas de un conjunto de axiomas. Realiza su funcionamiento como un motor de inferencia, proporcionando un conjunto de mecanismos para trabajar con reglas de inferencia que son definidos por medio de lenguajes de ontologías y con frecuencia en lenguajes de descripción. (Horrocks, Volz, & Decker, 2003)

En conclusión el razonador es un *motor de reglas que deduce conclusiones lógicas de un conjunto de axiomas, que nos permite comprobar la coherencia de una ontología.*

3.1. Tipos de Razonadores más comunes

Se estudia los razonadores más utilizados enfocando las características más relevantes como: su forma de razonamiento, arquitectura, lenguaje de programación que fue desarrollado, licencia, entre otras.

RAZONADOR KAON2

El KAON2 es una infraestructura para manejar a OWL-DL, SWRL, y ontologías lógicas. Fue producido por el esfuerzo unido de las siguientes instituciones (Motik, 2005):
La ingeniería de Proceso de Información del (IPE) en el Centro de Investigación para las tecnologías de la información (FZI).

- ✓ El instituto de Tecnología de la Información Aplicada y los Métodos Formales (AIFB) de Descripción en la Universidad de Karlsruhe.
- ✓ El grupo de la Dirección de Información del (IMG) en la Universidad de Manchester.

Es un razonador semántico, que permite realizar la manipulación y ejecución de inferencias con ontologías representadas en OWL DL, SWRL y F-logic; a diferencia de otros razonadores que trabajan con lógica descriptiva no implementa cálculo tableaux como base para su proceso de razonamiento; sino que utiliza una serie de algoritmos que reducen una base de conocimiento SHIQ(D) a un programa *datalog* disjunto (Hustadt, Motik, & Sattler, 2004) lo que aumenta de forma considerable el rendimiento del razonador a la hora de ejecutar los procesos de inferencia.

Características de Kaon2:

- ✓ Ofrece una librería de programación para la manipulación ontología en OWL DL y F-logic.
- ✓ Resuelve consultas conjuntivas que están expresadas en el lenguaje SPARQL. (Prud'hommeaux, 2008)
- ✓ Posee una interfaz DIG que permite el acceso a la funcionalidad de Kaon2 de otras herramientas, tales como Protégé. (Knublauch, Fergerson, Noy, & Musen, 2004)
- ✓ Posee un módulo para extraer datos de base de datos relacionales**.**
- ✓ Un acceso que provee servidor autónomo para las ontologías en una manera distribuida usando a RMI.
- ✓ Un motor de inferencia para las averiguaciones conjuntivas de contestación (la sintaxis utilizada y expresada SPARQL).

Formalismos Lógicos de Kaon2

El Api de kaon2 es capaz de manipular ontologías OWL-DL. Actualmente, éste puede leer sintaxis XML, presentación RDF y Sintaxis OWL. Para el razonador kaon2, soporta el subconjunto *Shiq (D)* de owl-dl. Esto incluye todas las características de Owl-dl con la excepción de las clases enumeradas.

Razonamiento con Kaon2

Los algoritmos nuevos que hacen más pequeño el conocimiento *SHIQ*, le permiten aplicar técnicas deductivas de la base de datos conocida como sets mágicos.

Los algoritmos dan contestación a las consultas en kaon2 en uno o más órdenes de magnitud más rápido que en los sistemas existentes.

Arquitectura de Kaon2

Es una arquitectura flexible y escalable, está organizada en tres estratos: los adaptadores para almacenamiento de la ontología; el software personalizado para los servicios ofrecidos por la suite de herramientas; y las aplicaciones del cliente que acceden al software personalizado de la ontología y ofrecen aplicaciones a los usuarios finales. Toda la suite de herramientas de kaon2 es implementada en java. (Gómez, Fernández y otros, 2004, p.332)

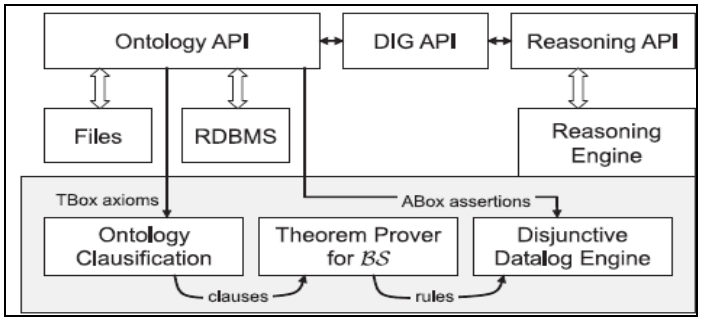

Figura 20. Arquitectura de KAON2 (Motik & Sattler, 2005)

En la Figura 20, se muestra la arquitectura usada por kaon2 y sus diferentes componentes:

Ontology API.- Facilita servicios de manipulación de una ontología, como la adición y recuperación de axiomas de la ontología, es completamente compatible por OWL y SWRL en el nivel sintáctico. Las ontologías pueden ser guardadas en archivos, utilizando sintaxis owl, rdf, xml. Alternativamente, las afirmaciones ABox se pueden almacenar en una base de datos relacional (RDBMS): asignando entidades a las tablas en una base de datos de la ontología, kaon2 consulta la base de datos sobre la marcha durante el razonamiento. (Bechhofer S. , 2003)

El razonamiento de API permite invocar diversos razonamientos, tareas, y recuperación de resultados. Kaon2 es utilizado como una biblioteca dinámica, o como servidor. En este último caso, el cliente puede acceder a las aplicaciones mediante el DL-KAON2 Grupo ejecutor (DIG) API. Por lo tanto, el sistema es utilizado con Protégé (*editor ontológico*).

- ✓ ***Ingeniería del Razonamiento.-*** Es el componente central de Kaon2 y este consta de tres sub-componentes.
- ✓ ***La Clasificación de la Ontología.-*** Es el responsable de la traducción de la base de conocimiento KB SHIQ en un conjunto de cláusulas de primer orden (KB).
- ✓ ***El teorema Prover BS.-*** Implementa supervisión en base a cálculos que se utilizan en algoritmo de reducción. (Bachmair,1995)
- ✓ ***Disjuntive Datalog Engine.-*** Se utiliza para responder a las preguntas en el programa obtenido por la reducción.

Modelo de conocimiento de Kaon2

El modelo de conocimiento trabaja con la suite de la herramienta Kaon (Lenguaje kaon) basado en una extensión de RDF (S). Con kaon2 se modela ontologías con los conceptos, las propiedades (para expresar relaciones y atributos), las instancias de conceptos y de propiedades.

Los conceptos son organizados en taxonomías de concepto (con la relación del subclassOf) y las propiedades son organizadas en jerarquías de la propiedad (con la relación PropertyOf). (Gomez, Fernandez, & Vicente, 2004, p.333)

Las extensiones para RDF (S) son relatadas para propiedades:

- ✓ Se define cardinalidades de la propiedad.
- ✓ Distingue entre relaciones y atributo según el rango, si es un concepto o un data-type (tipo de dato).
- ✓ Definir las relaciones simétricas o asimétricas.
- ✓ Y finalmente, definir relaciones inversas.

El modelo de conocimiento Kaon2 también coloca etiquetas, documentación, sinónimos, los componentes más complicados, como *axiomas formales*, no son expresados en koan2.

RAZONADOR PELLET

Es un razonador desarrollado por la Universidad de Maryland que permite realizar inferencias con ontologías OWL DL y es combinado con Jena y otras librerías de manipulación de owl. Basado en el algoritmo *tableaux* desarrollado para lógica descriptiva que incluso soporta el razonamiento sobre las clases enumeradas. (Lama & Sánchez, 2008)

Características de Pellet:

- Es capaz de analizar y reparar ontologías incorporando un conjunto de heurísticas para detectar y corregir automáticamente las ontologías owl para asegurarse que son owl-dl.
- Razonar con tipos de datos definidos en el estándar XML Shema
- Validar “especies” para garantizar que los individuos se clasifiquen de modo correcto en la taxonomía de conceptos de la ontología.
- Tiene un razonamiento con múltiples ontologías.

Arquitectura de Pellet

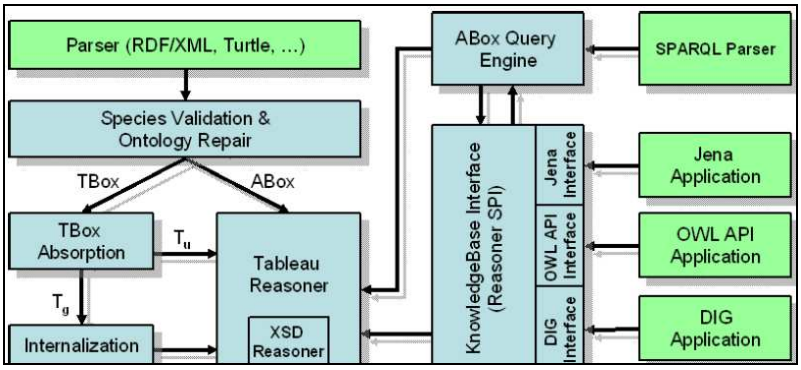

Figura 21. Arquitectura Pellet (Sirin, Parsia, Cuenca, Kalyanpur, & Katz, 2005)

La parte más importante del sistema es el razonador tableaux que comprueba la consistencia de una base de conocimiento. El razonador esta acoplado con Oracle datatype que comprueba la consistencia de conjunciones para (incorporar o derivar) XML-Shema simple tipo de dato (datatypes).

Las ontologías owl son cargadas en el razonador después de realizar la validación de especies y la reparación de ontologías; esto es para asegurar que todos los recursos tengan un apropiado tipo triple (un requisito para OWL-DL pero no OWL-Full) y el tipo de declaraciones pérdidas son añadidas según algunos heurísticos.

Análisis y Cargado

Pellet proporciona diferentes interfaces para la carga de ontologías, este no posee un *RDF/OWL* sino que está integrado a diferentes RDF/OWL kit de herramienta que proporciona el analizador. Las ontologías representadas en las estructuras de datos de tales herramientas son directamente cargadas para Pellet; también aplica la interfaz del razonador definida en esta herramienta para responder a las consultas.

Cada herramienta para la representación de ontología owl tiene diferente estructura, por esta razón; pellet incluye diferentes sub-módulos para cargar ontologías de diferentes representaciones.

Razonador Tableaux

Su función es, controlar la consistencia de la ontología; según el modelo owl teórico-semántico (Schneider, Hayes, & Horrocks, 2004), una ontología es coherente si hay una interpretación que satisfaga a todos los axiomas de la ontología. Esta interpretación es llamada modelo de la ontología.

Razonador Tipo de Dato

Es responsable de comprobar si la intersección de (la posible negación) de datos es coherente o no. Los tipos de datos owl se describe utilizando los esquemas XML que proporciona un conjunto de datos de simples, diversos tipos numéricos (enteros, flotantes), cadena y fecha/hora.

- ***Interfaz de Base del Conocimiento***

Todas las tareas de razonamiento se reducen a una prueba de coherencia de la KB con una apropiada transformación. Sin embargo, dichas transformaciones no siempre son triviales y hacen un control de coherencia arbitraría porque cada consulta requiere un mayor esfuerzo.

RAZONADOR RACER PRO

Tiene sus orígenes dentro de la LD; proporciona una base de criterios para normalizar los lenguajes de las ontologías en el contexto de la web semántica; este razonador gestiona ontologías basadas en owl. Sin embargo, también es visto como un repositorio web de información semántica con una optimización en su motor para la recuperación de información.

Es conocido como (Razonador para A-Boxes y expresiones de conceptos renombrados), fue uno de los primeros razonadores para A-boxes, el sistema también utiliza lógica modal. (RacerPro)

Características de Racer Pro

- Permite comprobar la coherencia de una ontología owl y un conjunto de descripciones de datos.
- Busca relaciones implícitas, sub-clases incluidas en la ontología.
- Encuentra sinónimos de los recursos (ya sean clases o los nombres de instancias).
- Apoya el uso de recursos computacionales: Las respuestas que necesitan menos recursos computacionales tienen prioridad, las otras aplicaciones se debe decidir si todas las respuestas valen o no tal esfuerzo.

Racer Pro es un sistema de representación del conocimiento, utiliza el método tableaux altamente optimizado para la LD. Con los avances que se realizan todos los días ahora es posible ofrece servicios de razonamiento para múltiples T-boxes y A-boxes. (RacerProNotes, 2007, p.1)

RacerPro como Web Semántica Razonamiento del Sistema de información y Repositorio

Un aspecto principal de los sistemas de explotación de los recursos es la capacidad de proceso de owl. Por ello es importante un sistema de razonamiento que este dentro del sistema de edición para una ontología. (GuideRacerPro, 2007, p.2)

RAZONADOR FACT ++

Conocido como un *Clasificación Acelerada de Terminología*, es un clasificador LD que sirve para satisfacer la lógica modal experimental. Este sistema incluye dos razonadores el SHIF y SHIQ (Horrocks I. , 2003).

FaCT++ es una versión nueva de FaCT OWL-DL, es la continuación del razonador libre para consultas T-Boxes. Además este razonador es implementado en C++ para crear una herramienta de desarrollo del software eficiente, y para el uso correcto de sus algoritmos. (Tsarkov & Horrocks, 2006)

Características de FaCT++

- Su lógica expresiva (*En particular el razonador SHIQ*); SHIQ es expresivo para ser usado como razonador de la lógica DL, porque razona con esquemas de base de datos.
- Tiene soporte para razonar con base de conocimiento arbitrarias.
- Su implementación optimizada tableaux (que ahora se ha convertido en un estándar para los sistemas DL).

Está diseñado como una plataforma para experimentar con nuevos algoritmos del tableaux. También incluyen una nueva arquitectura "Lista de tareas pendientes" que es adecuada más para los algoritmos Tableaux (como los usados para razonar con ontologías OWL), estos cuentan con un rango más amplio de optimizaciones heurísticas. (FaCT++, 2007)

Generalmente este razonador cumple algunas funciones: el chequeo de consistencia de una ontología, la relación que existe entre tipo clase-sub clase entre dos conceptos, clasificar toda la ontología para crear y corregir taxonomías, y comprobar la factibilidad de los conceptos de la ontología. (Lama & Sánchez, 2008, P.13)

Optimización de FaCT++

La simplificación y normalización léxica es una optimización estándar que re-escribe principalmente el diseño para detectar la inconsistencias, al simplificar los conceptos se

detectando incongruencias relativamente triviales. La idea básica de estos conceptos son transformarlos en una forma normal simplificada (SNF), donde este tiene algunos operadores como: *negación ($\neg$), conjunción (Π), restricción universal ($\forall$), o (restricción calificada ($\leq$).* (Tsarkov & Horrocks, 2006)

A continuación otros razonadores semánticos:

3.2. Tipos de Razonadores menos comunes

RAZONADOR CEL

Es el primer razonador para la descripción lógica *EL*+, una de las características más relevantes de CEL, es que implementar un algoritmo polinomio-tiempo. El apoyo a la descripción lógica *EL*+, le brinda los medios expresivos que se adapten a la formulación de ontologías médicas y biológicas. (Cel)

Sistema Cel

Tiene una interfaz muy interactiva, simple que provee a los usuarios de todas las funcionalidades esenciales, incluyendo un sistema de ayuda. El desarrollo de este razonador es un trabajo constante, se propone impulsar su poder expresivo lógico++, descomponer conceptos, dominios concretos, y nominales. Sé basa en los avances más recientes que han salido para la LD, tomando en cuenta las restricciones existenciales de conjunción. Ayuda como razonamiento práctico en las ontologías de gran extensión de la ciencia de la vida. (Baade, Lutz, & Suntisrivaraporn, 2002)

RAZONADOR MSPASS

Son un conjunto de varios teoremas con características automatizadas para numerosas lógicas, es una extensión de SPASS, es un cuentahílos para la LPO y la LD.

El razonador MSPASS es una extensión de SPASS que ayuda a manejar formulas de la lógica modal, lógica descriptiva y de cálculo relacional. Las fórmulas de primer orden se hacen de un vocabulario de símbolos de la proposición con dos tipos disjuntos, uno tipo booleano (concepto) y otro tipo relacional (rol). (Hustadt & Schmidt, 2000)

(Schmidt, 2007) especifico lo que integra el repertorio de construcciones lógicas:

- ✓ Los operadores estándar booleanos; el tipo booleano y las fórmulas de tipo de relaciones, tienen estos valores: Verdadero, Falso, no, y, o, etc.
- ✓ Los operadores relacionales adicionales: la composición, la suma relativa, el reverso, la relación de identidad, la relación de diversidad, la prueba, la restricción de dominio y alineación de restricción.

Los símbolos verdaderos y falsos tienen diversas interpretaciones, estas sirven también como booleano o fórmula de relación. El verdadero es usado como tipo Booleano, representa un conjunto universal, y es usado como relación universal. De manera similar, el falso es usado como el tipo de relación lo que representa al conjunto vacío o una relación vacía. (Schmidt, 2007)

RAZONADOR IRIS

Iris es un razonador open-sourse licenciado bajo GNU Lesser GPL y auspiciada por Sourceforge. Desarrollado en el lenguaje Java; se evalúa el registro de datos seguros o inseguros, tipo de esquema de datos xml, función de predicados. Su paquete de instalación contiene tres componentes: motor de razonamiento, analizador, y otros programas de utilidad, incluido dos aplicaciones que dan una interfaz de usuario para el motor IRIS. (IrisGuide, 2008, p.4)

Características de Iris

- Utiliza el método Datalog para su razonamiento.
- Conjunto completo y extensible de construcción en los predicados.
- Soporte para todos los tipos primitivos de datos de esquema XML

Proceso de Evaluación de Iris

Las consultas se realizan sobre una de base de conocimientos, está formada por hechos y normas. La combinación de los hechos, las normas y consultas se conoce como un programa de lógica y es la entrada a la tarea de razonamiento. Para dar la respuesta a una consulta, el razonador IRIS devuelve un conjunto de tuplas que se han deducido de la KB para responden a la consulta. (IrisGuide, 2008, p.5)

3.3. Aprendizaje del Capitulo:

Haciendo un resumen de las características más destacadas de cada razonador se presenta el siguiente cuadro.

Tabla 16. Características Principales de los Razonadores.

Características	RAZONADORES						
	Kaon2	**Pellet**	**Fact ++**	**RACER PRO**	**IRIS**	**MSPASS**	**CEL**
Expresividad de razonamiento	Razonamiento SHIQ	SROIQ(D)	SHF y SHIQ	--	--	--	--
Algoritmo de Razonamiento	Datalog	Tableux	Tableux	Tableaux	Datalog	--	polynomial - time
Lenguaje de Consulta	SPARQL	SPARQL		nRQL	--	--	
Permite extraer datos de BD Relacional	SI	SI	SI	SI	SI	--	--
Lenguaje	Java	Java	C++	Comercial	Java	C++	--
Soporte DIG	SI	SI	SI	SI	NO	NO	SI
Tipo de Licencia	Libre	Libre	Libre	No libre	Libre	Libre	--
Soporte de Reglas	SWRL-DL	SWRL-DL	NO	SWL	WSML	--	--
Disponibilidad del API	SI	SI	NO	SI	SI	--	NO
Multiplataforma	SI	SI	NO	NO	--	--	SI

CAPITULO IV
ANÁLISIS Y EVALUACIÓN DE LOS RAZONADORES

Introducción

El desarrollo de este capítulo es el más importante dentro de la Investigación ya que para su desarrollo se basa en el conocimiento de los capítulos anteriores todo esto ayudará a la Evaluación y Comparación de los razonadores semánticos.

Para lograr un desenvolvimiento óptimo en el proceso de evaluación se definen criterios importantes que caracterizan a un razonador y lo evalúen desde diferentes puntos de vista; cada uno de estos criterios debe cumplir con ciertas fases que se especificaron con la finalidad de lograr un proceso transparente.

Los resultados obtenidos serán un respaldo para las conclusiones que de ellos se deriven, para una mejor visualización y expresión de los datos; se aplicará fórmulas para determinar el rendimiento de cada una de las herramientas y así poder expresarlas en gráficas que reflejen los resultados.

Objetivos

- Integrar los razonadores seleccionados a la Herramienta de Protégé.
- Ejecutar pruebas por cada razonador seleccionado en base a las fases que se desarrollan para la evaluación y comparación.
- Evaluar y contrastar resultados de los razonadores en función a las fases realizadas.

4. Selección y Análisis de la Ontología

Ian Horrocks el principal impulsor de la investigación con razonadores semánticos junto a otros investigadores como T. Gardiner y D. Tsarkov mencionan la importancia de escoger ejemplos típicos, del mundo real con diferente expresividad; en las que el control manual no es factible como lo sería con ejemplos pequeños.

En base a esta recomendación se seleccionó la ontología OER-CC que fue desarrollada con el objetivo de que los usuarios y productores puedan describir recursos educativos licenciados bajo Creative Commons utilizando un vocabulario común. (Piedra, Chicaiza, Martinez, & Tovar, 2009)

Como una prueba adicional se realizó la evaluación con otro tipo de ontología que se encuentra en el Anexo 3, esto servirá para realizar un análisis y fundamentar las conclusiones.

En la Figura 22. Se muestra el Mapa Conceptual, él modelo visualiza claramente el diagrama que tendrá la ontología.

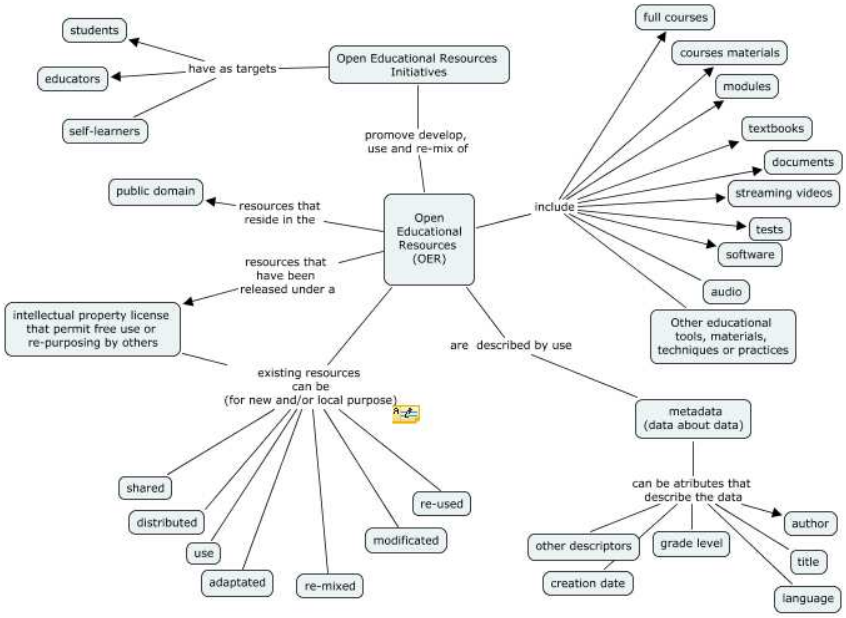

Figura 22. Mapa Conceptual de la Ontología OER-CC (Piedra, N. Chicaiza, J. López, J. Martínez, O. Tovar, M., 2010)

Características de OER-CC (Piedra, N. Chicaiza, J. López, J. Martínez, O. Tovar, M., 2010)

- Contiene información sobre los recursos educativos, en este caso utiliza el estándar de metadatos como es LOM (Learning Object Metada), sin embargo también se aplican algunos metadatos y vocabularios de Dublín Core.

- En cuanto a los metadatos para describir licencias Creative Commons, el estándar propuesto es ccREL, de esta forma se puede conocer que acciones se puede realizar sobre un recurso, si se puede desarrollar uno nuevo a partir de otros previos e incluso si es requerido citar al autor original del recurso.

- Para el proceso de desarrollo de la ontología se utilizó Cmaptools, Protégé y se utilizó las fases de construcción que propone la metodología METHONTOLOGY.

4.1. Requerimientos para Ejecutar la Evaluación

Para la presente investigación se ha seleccionado a los razonadores más utilizados (Pellet, Racer Pro, Kaon2, FaCT++), los mismos que comparten características comunes, que facilitan una mejor evaluación.

Las características de software y hardware que se emplearán para desarrollar las tres fases son:

Hardware:

- Equipo Portátil HP
- Core 2Duo.
- 2GHz.
- 1,99 GB de RAM.

Software:

- Sistema Windows XP
- Java 1.5
- Protégé 3.4

De igual manera se consideran algunas condiciones que permitan asegurar que las pruebas se realizaran de una manera transparente, se menciona dos de las principales:

- No re-iniciar el razonador.
- No detener el proceso del razonador durante las pruebas.

Estas, ayudan para que la ejecución de los razonadores se realice con normalidad sin afectar los tiempos de ejecución.

4.2. Criterios para medir el Rendimiento de un Razonador

Una vez realizada la investigación y seleccionado los razonadores para desarrollar el proyecto es conveniente hacer un análisis de medición del rendimiento de cada uno, razón por lo que se plantea los siguientes criterios que son de mayor interés: Eficiencia, Eficacia, Facilidad al instalar y Accesibilidad de la información de su funcionamiento

Tras el cumplimiento de cada criterio, se determinará el rendimiento al realizar las diferentes fases planteadas en la ontología, que faciliten escoger la mejor opción de los razonadores.

La medición de estos criterios se enfoca en dos tipos de usuarios con un escenario diferente, el principiante y el experto. Para el usuario principiante se necesitará que el razonador sea eficaz y que contenga todos los aspectos de los criterios planteados; en cambio sí es experto, necesitará saber sólo cual es el más eficiente* para probar su ontología.

- **Eficiencia (*),** en la ejecución del tiempo para cumplir cada una de las tareas principales que un razonador realiza. (l*a clasificación de taxonomías*, *chequear la consistencia de una ontología y cómputo de tipos a inferirse)*. Este se determina en la primera fase al realizar la medición de los tiempos al ejecutar cada tarea.

  A través de este primer criterio de medición, lo que se busca es demostrar la eficiencia que cumple el razonador, al procesarle, o pedirle información detallada

sobre los tiempos de ejecución que se necesita obtener. Para demostrar que los razonadores que se ha escogido para la investigación son los más apropiados.

- **Eficacia:** Se realiza en base a los resultados obtenidos en las tres fases planteadas, comparando los resultados. Así por ejemplo se compara su tiempo de respuesta, los procesos, la realización de las tareas y el alcance de la información.

  El usuario principiante tendrá información valiosa, contar con este criterio ayudará para evaluaciones futuras.

- **Facilidad al Instalar,** el demostrar la facilidad de instalación de un razonador es importante, considerando ciertos aspectos como: *la disponibilidad de los instaladores, una guía para la instalación, un sitio de soporte que brinde ayuda en caso de algún problema*. Este criterio se aplica en la instalación de cada razonador que se seleccionó para la evaluación.

  Se pretende demostrar que la utilización e instalación de razonadores, contando con la información adecuada, instaladores y la ayuda de expertos para resolver inquietudes y dudas, son procesos fáciles de ejecutar.

- **Accesibilidad a la Información de su funcionamiento:** El contar con la información de cada razonador permite tener una base sólida de su funcionamiento, siendo primordial conocer los aspectos principales que cada uno posee, conociendo sus ventajas y desventajas; que dan un respaldo a las deducciones que se realizan de ellos.

Mediante la investigación se ha establecido tres fases importantes para el cumplimiento de los criterios planteados, en la Figura 23. Diagrama del Desarrollo de Evaluación, se muestra el diagrama a cumplirse para la evaluación.

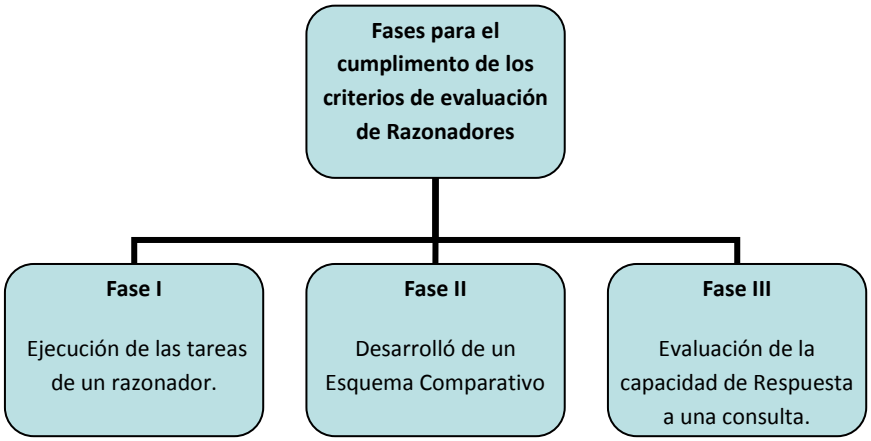

Figura 23. Diagrama del Desarrollo de Evaluación

4.3. Fases para la evaluación de cada razonador

Fase I: Ejecución de las tareas del razonador, se fundamenta en las tareas que cumple un razonador; se parte de la tarea de la *clasificación de taxonomías* que desarrollan la jerarquía de inferencia basándose en la jerarquía de aserción; las mismas que permiten obtener la construcción de la ontología. Como se explica en el Capítulo 1.

Luego se realiza, la prueba de *chequeo de consistencia* de la ontología, mediante una verificación o chequeo, para controlar la consistencia entre las propiedades de objetos y restricciones asignadas a las clases, con el fin de obtener hechos consistentes, verificando que cada subclase pertenezca a la clase correcta. Finalmente dentro de esta fase, se realizará el *cómputo de los tipos inferidos*; método que consulta al razonador los tipos a deducirse de todos los individuos en la ontología. (Floréz H. , 2008, p.11)

Para esta primera fase se realiza la instalación de cada razonador a la herramienta de Protégé para realizar pruebas y así conocer su funcionamiento de forma práctica, luego ya con la aplicación en la ontología seleccionada; se ejecuta las tres tareas principales haciendo una captura de pantallas sobre los resultados de cada tarea que realiza cada razonador. El registro de resultados se realizó a través de tablas que representen cada experimento realizado.

Fase II: Desarrolló de un esquema comparativo, se basa en la información disponible en la Web, acerca de cada razonador. Obteniendo los aspectos más importantes como: características principales, ventajas y desventajas; que ayuden a respaldar las conclusiones que de ellos se deriven.

En la investigación es necesario tener un dominio de información que permitan afirmar o refutar los resultados obtenidos en la práctica; por tal razón se creó un esquema que recopile los aspectos más relevantes y que reflejan la disponibilidad de información que hay sobre cada uno. En algunos casos la información es básica y otros en cambio ofrecen una gama de alternativas para despejar inquietudes.

Tercera III: Evaluación de la capacidad de respuesta a una consulta, sé tomó una ontología y consulta ejemplo para ver el comportamiento que tiene el razonador con respecto a las consultas realizadas a una ontología. Para la fase de consultas la ontología usada no prestaba los requerimientos necesarios para cumplir con la misma, entonces se procedió a obtener una ontología diferente a la ya utilizada, que contenga datos y de esta manera poder realizar la consulta y obtener sus resultados.

Una vez concluida esta última fase se espera despejar algunas dudas que surgen en torno a este tema, al saber cómo reacciona el razonador ante la consulta.

4.4. Desarrollo de las fases:

4.1.1 Fase I: Ejecución de las Tareas del razonador

La primera actividad que se realizó previo a las pruebas y desarrollo de ésta fase, es la instalación de cada razonador a Protégé; los aspectos a tomar en cuenta son la dirección y el número de puerto por el que trabajan, como ya se ha mencionado gracias al estándar DIG[3] es posible configurar éstos datos, y para una mejor percepción de todo el proceso de instalación revisar la parte de Anexos (Anexo 1), donde se encuentra detallado la instalación de cada razonador. La escala de equivalencia para los criterios de evaluación se presenta en la Tabla 17. Para una mejor interpretación de los resultados obtenidos

Tabla 17. Escala de Equivalencia para Criterios de Evaluación

Facilidad de Instalación	Porcentajes
Excelente	100% -80%
Bueno	50% - 70%
Malo	30%-40%
Deficiente	10% -20%

En la Tabla 18, se indican las características de cada razonador y sus respectivos porcentajes que reflejan la práctica realizada en el proceso de instalación de los razonadores.

Tabla 18. Facilidad de la Instalación de los razonadores

Razonador	Integrado en Protégé	Instalador	Puerto	Observaciones	Facilidad de Instalación /100%
Pellet	x		8080	Razonador integrado con Protégé.	100%
Racer Pro		x	8080	Versión de prueba y Licencia Comercial	80%
Kaon2		x	8080	Configuración de class-path, corrida mediante consola.	60%
FaCT++		x	3490	Número de puerto diferente a los demás razonadores.	90%

El siguiente paso es insertar la ontología a Protégé donde se realizaran tres actividades: la *comprobación de la coherencia,* de la ontología basada en la descripción de una clase, el razonador puede comprobar si es posible que se tenga instancias de la clase; *la clasificación de taxonomías* donde permite observar los resultados en cuanto a las inferencias generadas por el modelo que describe el dominio del problema y *el cómputo de tipos a inferir* que permite computar los tipos de objetos dentro de la ontología.

[3] DIG: estándar o protocolo que se introduce para proporcionar una interfaz común para los razonadores DL.

(Horrocks, Tsarkov, & Gardiner, 2006) dicen que: para evaluar el rendimiento no se puede repetir la misma prueba sobre el mismo razonador sin haberlo reseteado antes, si se hace de diferente manera el tiempo de cálculo precisa el resultado.

1. PRIMERA EJECUCIÓN CON LOS RAZONADORES

Cuando se habla de razonamiento de Protégé-OWL API, esto significa obtener una instancia de ProtegeReasoner (Comunicación directa con DIG que siempre este *sincronizado* con el modelo interno Protege OWL).

Esta instancia es utilizada para tener información acerca de la deducción del modelo, como inferir superclases, deducir clases equivalentes y deducir tipos de individuos (APIProtégé, 2009)

Este proceso que se realiza a través de ProtegeReasoner, permite consultar al razonador y obtener información e insertarla en el Modelo OWL. Esto significa que el modelo puede ser examinado "en línea" desde el razonador.

Un ejemplo de esto es la Clasificación de Taxonomías, donde se consulta al razonador de la coherencia, deduce superclases y clases equivalentes de todas las clases de la ontología y luego introduce la información en el modelo Protégé-OWL.

Lo mismo pasa con el Computo de clases a inferirse, método que consulta al razonador para los tipos a deducirse de todos los individuos en la ontología y actualizar el Modelo OWL.

Así, en la ejecución del primer experimento se observa el proceso que se realiza en cada actividad:

1. **Chequeo de la consistencia**
   a. **Sincronización del Razonador :**
      Proceso que realiza en este paso:

      - ✓ Limpieza de la base de conocimiento
      - ✓ Actualización del razonador
      - ✓ Sincronización

   b. **Chequeo de la consistencia de conceptos**
      - ✓ Tiempo de actualización

   c. **Tiempo Total**

Así mismo, para la clasificación de taxonomías existe un proceso previo que realiza el razonador:

2. **Clasificación de Taxonomías**

   a. **Sincronización del Razonador**
   - ✓ Limpieza de la base de conocimiento
   - ✓ Actualización del razonador
   - ✓ Sincronización

   b. **Chequeo de Consistencia de Concepto**

   - ✓ Tiempo de actualización Protege- OWL

   c. **Calcular la jerarquía a inferirse**

   - ✓ Tiempo de actualización Protege- OWL

   d. **Calcular las clases equivalentes**

   - ✓ Tiempo de actualización Protege- OWL

   e. **Tiempo Total**

Y por último en el cómputo de tipos a inferirse este es su proceso:

3. **Computo de los tipos a inferirse:**
   a. **Computo de tipos a inferirse**
   - ✓ Tiempo de actualización Protégé – OWL

   b. **Tiempo Total**

El orden de ejecución de las pruebas del razonador es: Pellet, Racer Pro, Kaon2, FaCT++. Una vez que se ha desarrollado el primer experimento con las tres pruebas, en la Tabla 199, se muestra los datos obtenidos.

Tabla 19. Primer Experimento con los razonadores

PRIMERA PRUEBA									
Razonadores	Chequeo de Consistencia			Clasificación de Taxonomías				Tipos a Inferirse	
	Sincronización del Razonador	Chequeo de Consistencia de Conceptos	Tiempo Total	Chequeo de Consistencia de Conceptos	Calcular la Jerarquía Inferida	Calcular clases equivalentes	Tiempo Total	Tipos a Inferirse	Tiempo Total
Pellet 1.5.2	0,75 seg	0,125	0,985	0,016	0,156	0,001	0,203	0,172	0,203
FaCT 1.3.0		0,032	0,062	0,017	0,033	0,003	0,062	0,001	error
Racer Pro 2.0									
Kaon2	0,968	0,204	1	0,017	0,033	0,018	0,078	0,001	error

Para el primer experimento se identifican algunos aspectos que requieren de un pequeño análisis y discusión, así en la tarea de *chequeo de la consistencia* dos razonadores Pellet y Kaon2 cumplen con la sub-tarea de sincronización que es la comunicación con el modelo owl.

FaCT++ y Kaon2 al realizar ésta tarea y la de clasificación de taxonomías indican la siguiente alerta: "*Not able to convert datatype cardinality restrictions to DIG (lenguaje used to communicate with the reasoner). Ignoring this restriction and attempting to continue*". Tal

como lo indica la alerta las restricciones de cardinalidad son omitidas por estos razonadores para dar paso a la realización de la tarea, esto se demuestra en la Figura 24.

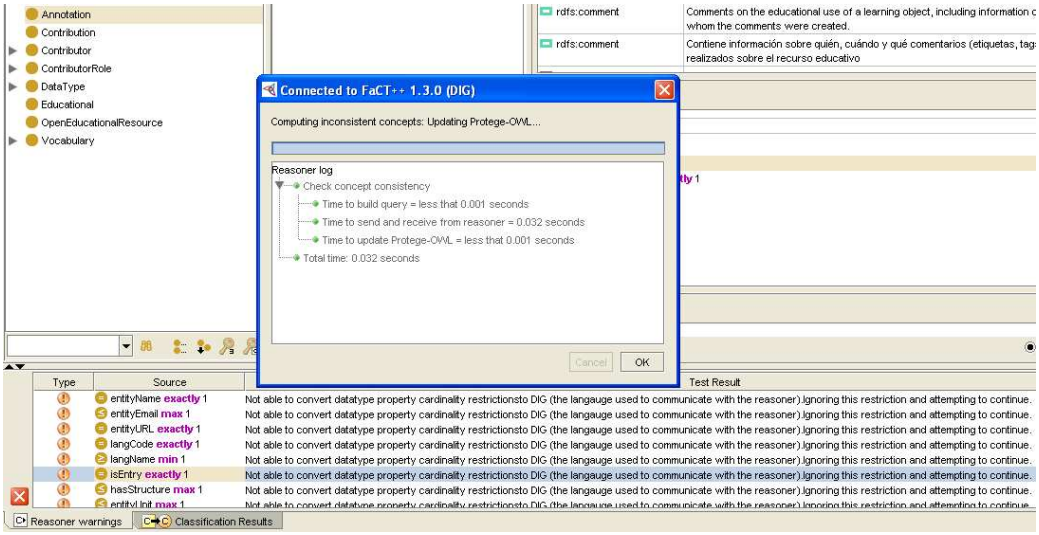

Figura 24. Resultado de Chequeo de la Consistencia con FaCT++ y Kaon2

Para la tarea de cómputo de tipos a inferirse FaCT++ y Kaon2 indican un error como se muestra en la Figura 25, en donde el razonador recibe un mensaje ASK en donde sí la sintaxis que contiene no es comprendida, el resultado será un ERROR.

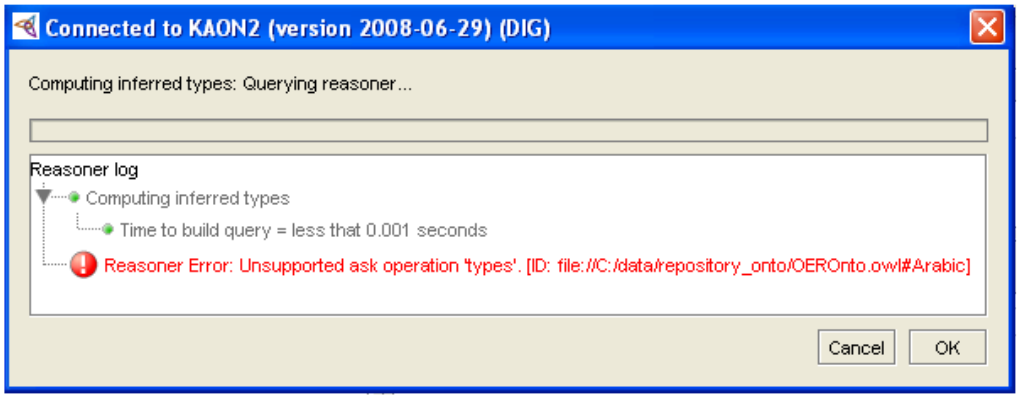

Figura 25. Cómputo de tipos a inferirse

Una solicitud de tipo ASK, en su estructura se compone de una serie de declaraciones, cada una de ellas debe tener un atributo *ID* que suministre una identificación única para la consulta permitiendo al razonador optimizar el proceso. Cada elemento de ASK debe tener un atributo que es una URI que identifica la KB en el razonador. (Bechhofer, 2003)

A consecuencia de esto la tarea no es realizada por ninguno de los dos razonadores.

Y por último, al aplicar Racer Pro su respuesta se indica en la Figura 26; en donde muestra que el contenido no es permitido. Ahora visualizando la herramienta de protégé para tratar

de identificar el error en la Figura 27 se indica la invocación de un método getIdentity[4] ( ) que permite obtener una identificación para la interfaz DIG que conecta al razonador; en este caso esta conexión está fallando y no permite que la ontología sea admitida.

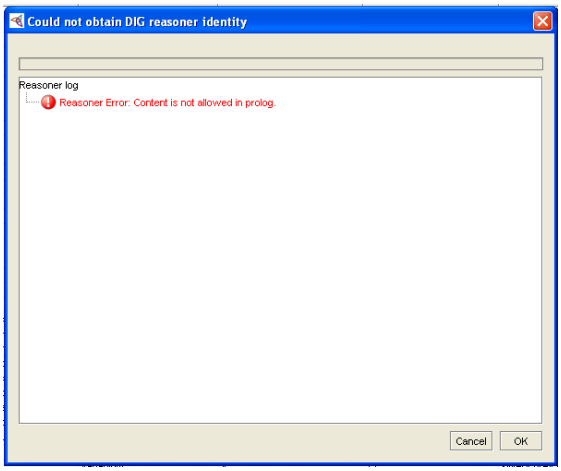

Figura 26. Respuesta del Razonador Racer Pro

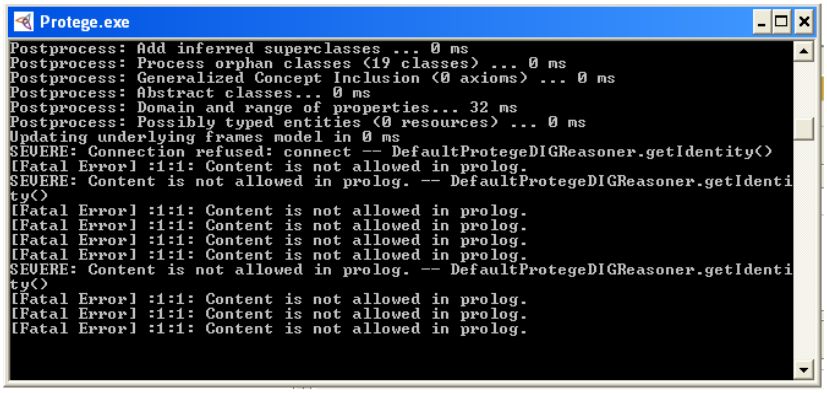

Figura 27. Identificación del Error en Protégé

Pero también se debe aclarar que sí se utiliza la herramienta propia de Racer Pro esta es capaz de realizar la tarea de clasificación de taxonomías como se muestra en la Figura 28, pero como el objetivo de nuestra investigación es la utilización del razonador con Protégé para poder medir su rendimiento, éste queda descartado para la evaluación con esta ontología. Por el problema que existe con la interfaz DIG y el razonador.

4 http://protege.stanford.edu/protege/3.4/docs/api/owl/edu/stanford/smi/protegex/owl/inference/dig/DefaultProtegeDIGReasoner.html

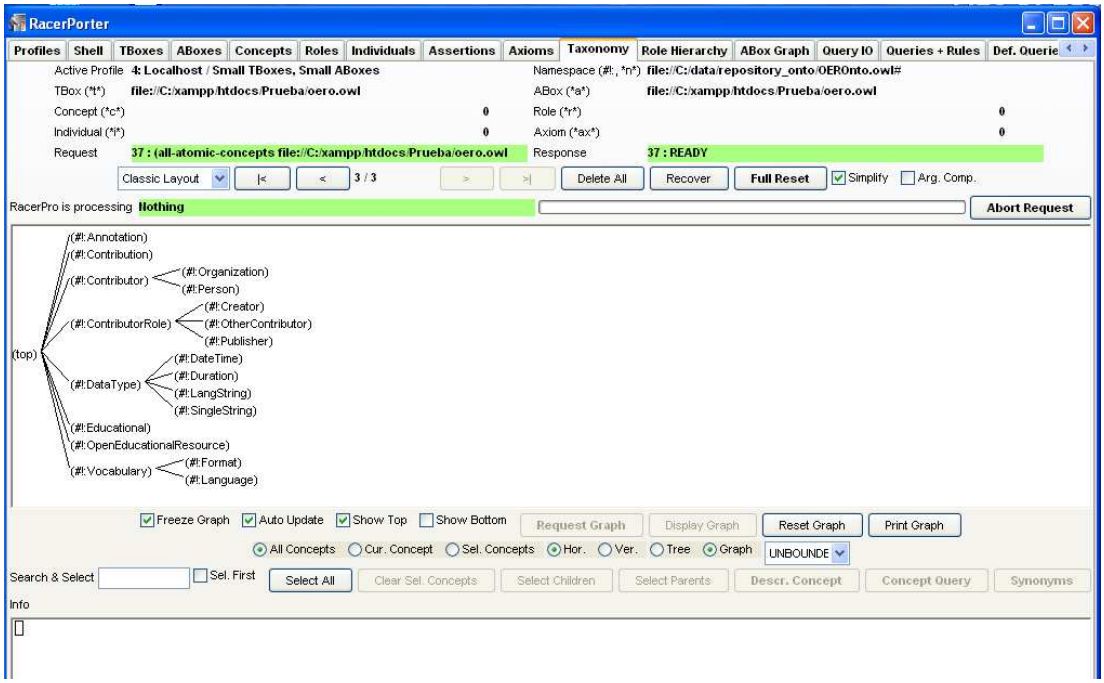

Figura 28. Clasificación de Taxonomías con Racer Pro

A consecuencia de esto el estudio se reduce a tres razonadores (Pellet, FaCT++ y Kaon2).

Cabe destacar que cada razonador actúa de una diferente manera, así obteniendo diferentes tiempos de respuesta a pesar de compartir algunas características cada uno es independiente.

2. SEGUNDA EJECUCIÓN CON LOS RAZONADORES

Para el segundo experimento, en las tres actividades se presentan los siguientes datos ilustrados en la Tabla 20.

Tabla 20. Segundo Experimento sobre la Ontología

SEGUNDA PRUEBA									
Razonadores	Chequeo de Consistencia			Clasificación de Taxonomías				Tipos a Inferirse	
	Sincronización del Razonador	Chequeo de Consistencia de Conceptos	Tiempo Total	Chequeo de Consistencia de Conceptos	Calcular la Jerarquía Inferida	Calcular clases equivalentes	Tiempo Total	Tipos a Inferirse	Tiempo Total
Pellet 1.5.2	0,74	0,109	0,484	0,016	0,032	0,001	0,078	0,109	0,125
FaCT 1.3.0		0,032	0,062	0,032	0,018	0,018	0,11	0,001	error
Racer Pro 2.0									
Kaon2		0,048	0,063	0,049	0,018	0,003	0,125	0,001	error

Para esta parte se observa que solo el razonador Pellet es quien realiza la actividad de sincronización del razonador en la tarea de chequeo de consistencia, de la misma forma hay un error en la actividad de computo de tipos a inferirse en los razonadores FaCT++ y Kaon2.

Por lo demás cada razonador brinda sus datos respectivos, y así contribuyen para la evaluación.

3. TERCERA EJECUCIÓN CON LOS RAZONADORES

En este último experimento no hay aspectos nuevos a recalcar, claramente sus tiempos son diferentes pero no hay un aspecto relevante que destacar como se muestra en la Tabla 21.

Tabla 21. Tercer Experimento sobre la Ontología

TERCERA PRUEBA									
Razonadores	Chequeo de Consistencia			Clasificación de Taxonomías				Tipos a Inferirse	
	Sincronización del Razonador	Chequeo de Consistencia de Conceptos	Tiempo Total	Chequeo de Consistencia de Conceptos	Calcular la Jerarquía Inferida	Calcular clases equivalentes	Tiempo Total	Tipos a Inferirse	Tiempo Total
Pellet 1.5.2	0,067	0,125	0,469	0,001	0,047	0,001	0,109	0,125	0,125
FaCT 1.3.0		0,017	0,079	0,032	0,003	0,018	0,109	0,001	error
Racer Pro 2.0									
Kaon2		0,048	0,063	0,017	0,033	0,018	0,093	0,001	error

Para un mejor detalle, las capturas de imágenes del proceso que se realizó y las respuestas que dio la herramienta a cada prueba se encuentran en la parte de Anexos (Anexo2).

4. FASE I: INTERPRETACIÓN DE RESULTADOS

Una vez que se ha obtenido los datos respectivos, se procede a extraer los tiempos totales de cada tarea para la evaluación en *tablas resumen* de cada una de las tareas. Con el objetivo de evaluar el rendimiento de los razonadores en sus tareas.

En la primera tarea de **Chequeo de la Consistencia** se presenta los datos en la Tabla 22. Resumen de la Tarea de Chequeo de Inconsistencias las tres experiencias bajo la misma tarea.

Tabla 22. Resumen de la Tarea de Chequeo de Inconsistencias

CHEQUEO DE LA CONSISTENCIA				
Razonadores	**Primer Experimento (segundos)**	**Segundo Experimento (segundos)**	**Tercer Experimento (segundos)**	**Promedio**
Pellet 1.5.2	0,985	0,484	0,469	0,65
FaCT 1.3.0	0,062	0,062	0,079	0,07
Kaon2	1	0,063	0,063	0,38

Proceso para obtención de la Eficiencia:

Tomamos como referencia un tiempo mayor de 0,7

T_Rendimiento = 0,7 - Promedio
Pellet → 0,7 – 0,65 = 0,054
FaCT++ →0,7 – 0,07 = 0,63
Kaon2 → 0,7 – 0,38 = 0,32

T_Rendimiento $\rightarrow$ 100%

T_Rendimiento = (T_Rendimiento *100%)/0,7

T_R_Pellet → = (0,05 *100%)/0,7 = 7,71%
T_R_FaCT++ →= (0,63 *100%)/0,7 = 90,33%
T_R_Kaon2 → = (0,32 *100%)/0,7 = 46,38%

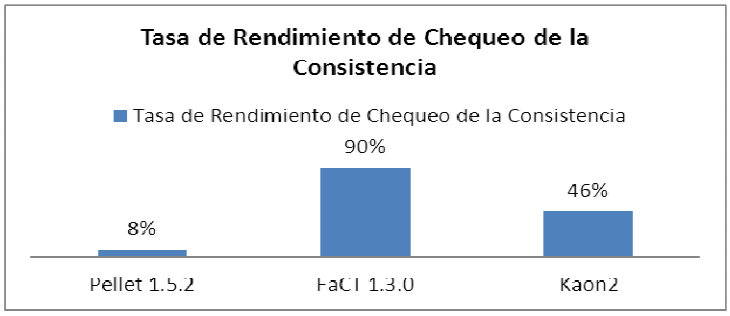

Figura 29. Tarea de Chequeo de Consistencia

De esta manera se ha podido identificar la mejor propuesta de los razonadores al realizar el chequeo de la consistencia basados en el tiempo de ejecución, obteniendo un 90% FaCT++ siendo el mejor al realizar esta tarea; seguido de Kaon2 con 46%

En la segunda tarea de la clasificación de Taxonomías en la Tabla 23, se encuentran los datos con los que se va a trabajar.

Tabla 23. Resumen de la Tarea de Clasificación de Taxonomías

CLASIFICACIÓN DE TAXONOMÍAS				
	Primer Experimento (segundos)	**Segundo Experimento segundos)**	**Tercer Experimento (segundos)**	**Promedio**
Pellet 1.5.2	0,203	0,078	0,109	0,13
FaCT 1.3.0	0,062	0,11	0,109	0,09
Kaon2	0,078	0,125	0,093	0,10

Proceso para obtención de la Eficiencia:

Tomamos como referencia un tiempo mayor de 0,2

T_Rendimiento = 0,2 - Promedio
Pellet → 0,2 – 0,13 = 0,07
FaCT++ → 0,2 – 0,09 = 0,11
Kaon2 →0,2 – 0,10 = 0,10

T_Rendimiento → 100%

T_Rendimiento = (T_Rendimiento *100%)/0,2
T_R_Pellet → = (0, 07 *100%)/0,2 = 35%
T_R_FaCT++→ = (0, 11 *100%)/0,2 = 53%
T_R_Kaon2 →= (0, 10 *100%)/0,2 = 51%

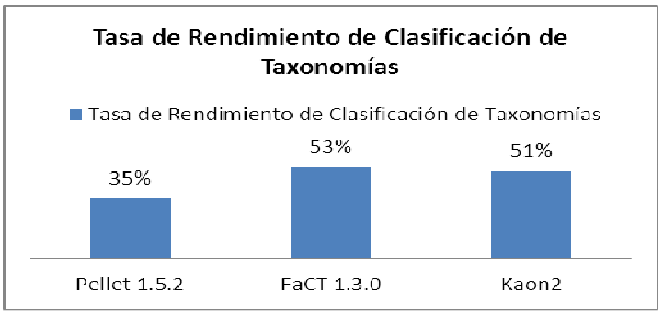

Figura 30. Tasa de Rendimiento de la Clasificación de Taxonomías

La **clasificación de las taxonomías** es una de las principales tareas que cumple un razonador permitiendo observar los resultados en cuanto a las inferencias generadas por el modelo que describe el dominio del problema.

En la Figura 30, se muestra como el razonador FaCT++ optimiza mejor el tiempo representando un 53% de eficiencia mejor que los demás razonadores, siguiéndole Kaon2 con el 51% de rendimiento.

En la última tarea de **Cómputo de tipos a inferirse** los valores se encuentran en la Tabla 24Tabla 24. Resumen de la Tarea de Cómputo de Tipos a Inferirse de aquí se parte para obtener el rendimiento de esta tarea.

Tabla 24. Resumen de la Tarea de Cómputo de Tipos a Inferirse

COMPUTO DE TIPOS A INFERIRSE				
	Primer Experimento (segundos)	**Segundo Experimento (segundos)**	**Tercer Experimento (segundos)**	**Promedio**
Pellet 1.5.2	0,203	0,125	0,125	0,15
FaCT 1.3.0	Error	Error	Error	Error
Kaon2	Error	Error	Error	Error

Proceso para obtención de la Eficiencia:

Tomamos como referencia un tiempo mayor de 0,2

T_Rendimiento = 0,2 - Promedio

0,2 – 0,151 = 0,049

T_Rendimiento $\rightarrow$ 100%

T_Rendimiento = (T_Rendimiento *100%)/0,2

= (0,049 *100%)/0,2 = 24,5%

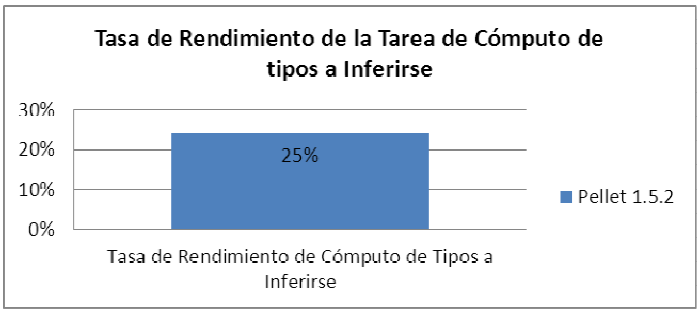

Figura 31. Cómputo de Tipos a Inferirse

En la Figura 31, se muestra los resultados obtenidos para la prueba de cómputo de tipos a inferirse (método que consulta al razonador los tipos a deducirse de todos los individuos en la ontología), para esta ontología en especial solo el razonador Pellet cumple con esta tarea en el caso de kaon2 y FaCT++ muestra un error de la operación a realizar.

5. FASE I: ANÁLISIS DE RESULTADOS

Para la obtención de los resultados de las tareas, se tomó un tiempo ***máximo de referencia*** (tiempo máximo que engloba todos los tiempos de las pruebas) para la medición del rendimiento. Para la tasa de rendimiento se utiliza la diferencia del ***tiempo de referencia*** (en segundos) con el tiempo de la actividad, se lo multiplicó por el 100% y se lo dividió para el valor tomado de referencia, estos resultados presentan la ***tasa de rendimiento*** que tiene cada razonador, se reflejó su eficiencia.

En base a estos resultados el "**Razonador FaCT++"**es el más eficiente por obtener los mayores porcentajes en el rendimiento en las dos tareas (chequeo de consistencia y clasificación de taxonomías), siendo estas tareas las más importantes de esta fase. Todo el desarrollo de esta fase ayuda a cumplir el criterio de Eficiencia que se da en cada razonador. A continuación en la Figura 32, se muestra un resumen de la eficiencia del mejor razonador en cada tarea.

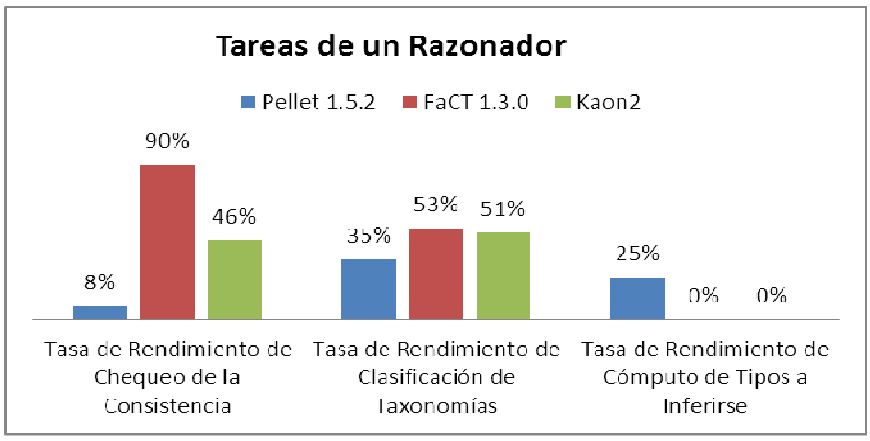

Figura 32. Demostración del Criterio de Eficiencia

4.1.2 Fase II: Desarrollo de un Cuadro Comparativo de las Características Principales

Como se puede observar en la Tabla 25, se muestra una tabla comparativa, los razonadores se efectúan con algunas características similares tal es el caso de que todos implementan el interfaz DIG, por lo que se puede utilizar dicho interfaz al implementar el razonador con Protégé.

De igual forma se aprecia que en tres de los razonadores se utiliza el mismo algoritmo de razonamiento el Tableaux (Mencionado en el Capítulo 2), a diferencia del KAON2 que utiliza el Datalog. Otra característica importante, es que tienen su propio razonador integrado, es la utilización de una API para poder atacar al razonador directamente a diferencia de FACT ++ 1.3. Se debe destacar que todos los razonadores seleccionados a diferencia de RACER PRO, son de tipo de licencia libre mientras que RACER en una licencia de propietario.

Una característica en la que los cuatro razonadores son compatibles, es que con Protégé y la utilización de una BD Relacional, me permite extraer los datos y poder utilizarlos

Tabla 25. Características de los Razonadores en Estudio

Características	Razonadores		
	Kaon2	**Pellet**	**Fact ++**
Expresividad de razonamiento	Razonamiento SHIQ	SROIQ(D)	SHF y SHIQ
Algoritmo de Razonamiento	Datalog	Tableaux	Tableaux
Lenguaje de Consulta	SPARQL	SPARQL	
Permite extraer datos de BD Relacional	SI	SI	SI
Lenguaje	Java	Java	C++
Soporte DIG	SI	SI	SI
Tipo de Licencia	Libre	Libre	Libre
Soporte de Reglas	SWRL-DL	SWRL-DL	NO
Multiplataforma	SI	SI	NO
Tipo de Consultas	A-Boxes extensas	A-Boxes y T-Boxes	T-Boxes

ANÁLISIS DE RESULTADOS

El análisis de rendimiento de cada razonador representa la parte experimental pero esta debe estar apoyada de la parte teórica para hacer una diferenciación entre los resultados; porque es verdad que la práctica es importante pero la base teórica también lo es.

Hay algunos factores importantes que se deben tomar en cuenta en un razonador, a continuación los siguientes:

- El algoritmo Tableaux utilizado para el razonamiento de algunos razonadores se ha convertido en un estándar.
- Datalog es una extensión de Prolog, usado por el razonador Kaon2 con nuevos algoritmos de razonamiento que permiten una optimización en sus tareas.
- La licencia con la que cuenta cada razonador es importante en una investigación y mucho más para el desarrollo de un proyecto aplicado a la vida real.
- El lenguaje de consultas que posea cada razonador es importante para poder extraer los datos de la ontología.
- Para los desarrolladores contar con herramientas que funcionen en varias plataformas es una ventaja, así son libres de escoger la de su mejor dominio para realizar un trabajo de calidad.
- La obtención del API del razonador, permite atacar al razonador directamente y así brindar un aporte a la investigación y desarrollo de mejoras para el razonador.

Todas las características son importantes a la hora de escoger el mejor razonador para una ontología; desde el punto de vista teórico.

En base al criterio accesibilidad a la información, vemos que FaCT++, a pesar de ser open-source, no cuenta con mucha información de ayuda, ni comunidad grande de soporte, además de ofrecer poca información al usuario en caso de error (aunque un poco más de la que ofrece Kaon2 en su interfaz DIG).

Sólo ***Pellet*** cuenta con el apoyo de una comunidad de desarrolladores que permite acelerar la madurez del producto, siendo este razonador el que cumple de mejor manera éste criterio.

4.1.3 Fase III: Capacidad de Respuesta a una Consulta

En esta tercera fase se realizó uno de los factores más importantes para promover el uso de ontologías, el desarrollo de razonadores que permitan realizar consultas en las ontologías.

Es primordial tener sistemas que garanticen la ejecución de las consultas y que razonen sobre la ontología de manera eficiente, de tal manera que es posible explotar el poder de la semántica. Las consultas a las ontologías no solo implican realizar una consulta interna de está, sino también realizar una consulta a un conjunto de ontologías para la búsqueda de un concepto determinado.

DESARROLLO DE LA CONSULTA

Se plantearon dos consultas como se muestra a continuación:

1. Consulta devuelve los nombres de los contribuyentes de recursos y su mail

```
PREFIX table: <file://C:/data/repository_onto/OEROnto.owl#>
SELECT DISTINCT ?contribution ?nombre ?mail
 WHERE { ?oer table: hasContribution ?scontribution.
          ?scontribution table:contributingEntity ?contribution.
          ?oer table:hasContribution ?grupoautor.
         ?grupoautor table:contributingEntity ?ggrupoautor.
         ?contribution table:entityName ?nombre.
         ?contribution table:entityEmail ?mail. }
```

2. Consulta para la recuperación de recursos:

```
PREFIX table: <file://C:/data/repository_onto/OEROnto.owl#>
SELECT DISTINCT ?recurso
WHERE {?autor  table:isContributingEntityOf ?rec.
          ?rec table:isContributionOf ?recurso. }
```

RESULTADOS DE LAS CONSULTAS

Lo que se desea lograr a través de la ejecución de la consulta planteada, es verificar la capacidad de respuesta y validar la eficiencia del razonador al realizar una consulta del tipo TBox.

Al realizar la prueba con los razonador Pellet, FaCT++, Kaon2 la respuesta es satisfactoria en los tres casos. Determinando así que cualquiera de estos razonadores responderán a las consultas que el usuario necesite.

Results

contribution	nombre	mail
UTPL	UTPL	info@utpl.edu.ec
ELIZABETH.CADME	Elizabeth Cadme	iecadme@utpl.edu.ec
PRISCILA.VALDIVIEZO	Priscila Valdiviezo	pmvaldiviezo@utpl.edu.ec
JANNETH.ALBA	Janneth Alba	jmalba@utpl.edu.ec
MANUEL.SALAZAR	Manuel Salazar	msalazar@utpl.edu.ec

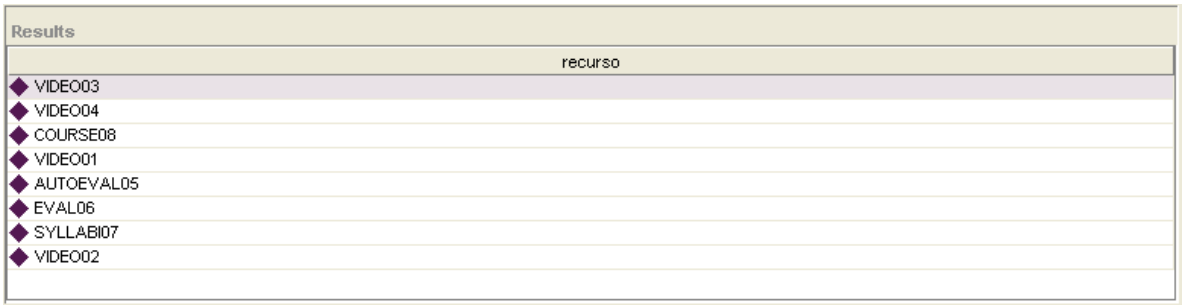

Figura 33. Respuestas a las Consultas Planteadas con los tres Razonadores

ANÁLISIS DE RESULTADOS

Una vez que se ha obtenidos los resultados que se muestran en la Figura 33, se determina la factibilidad de estos tres razonadores para hacer consultas con esta ontología.

Al obtener los resultados se analiza que no todas las ontologías podrán ser leídas por los razonadores y no específicamente porque estén mal diseñadas o estructuradas, sino que depende de las características que está posea para dejar desarrollar la consulta.

Para fundamentar está hipótesis se ha consultado en algunas Universidades como la Universidad de Maryland, Universidad de Valencia y la Universidad de Manchester; que conjuntamente con algunos de los impulsores como: Ian Horrocks junto a D. Tsarkov y T. Gardiner, han desarrollado pruebas e investigaciones sobre este tema de evaluación de los razonadores ya que es una herramienta fundamental y muy importante a la hora de valorar una ontología; tomando en cuenta que hoy en día la tendencia es la creación de ontologías para aplicarlas a la vida real y el mejoramiento de actividades del diario vivir.(Horrocks,Tsarkov, 2006)

Tabla 26. Resultados de la Capacidad de Respuesta a la consulta establecida

Razonador	SI	NO
Pellet	X	
Kaon2	X	
FaCT++	X	

En la Tabla 26, se muestra un resumen de fase de la consulta en una ontología de tal manera que se pueda identificar la acción de cada razonador.

4.2 Criterio de Eficacia de los Razonadores

Una vez que se han finalizado todas las fases, se desea realizar una recopilación de todos los resultados obtenidos y en base a estos evaluar el criterio de eficacia. Para ello se ha utilizado una ***matriz de satisfacción de criterios de evaluación***, éstos se muestran en la Tabla 27.

Tabla 27. Criterios de Evaluación para medir la eficacia de cada razonador

Criterios de Evaluación	Ponderación
Disponibilidad de la Información	20
Facilidad de Instalación	10
Chequeo de la Consistencia	10
Clasificación de la Taxonomía	20
Cómputo de Tipos a Inferirse	10
Respuesta a Consultas	30
	100%

Establecidos los criterios de evaluación se evalúa el cumplimiento en cada uno de los razonadores, en base a la experiencia y experimentos realizados, los resultados se muestran en la Tabla 28.

Tabla 28. Evaluación de Cumplimiento de los Criterios en cada Razonador

Métricas	Pellet	Kaon2	FaCT++
Disponibilidad de la Información	20	5	5
Facilidad de Instalación	10	5	10
Chequeo de la Consistencia	4	8	10
Clasificación de la Taxonomía	4	8	10
Cómputo de Tipos a Inferirse	10	0	0
Respuesta a Consultas	30	30	30
	78	**56**	**65**

Aplicando los valores planteados en la Tabla 27, se observa que el razonador de mejor rendimiento es "***Pellet",*** con un resultado final de rendimiento del 78% de eficacia mostrada en la Tabla 28.

Realizando un análisis comparativo entre Pellet y FaCT++ que son los razonadores que muestran un mejor rendimiento del 78% y 65% respectivamente, ambos poseen valores similares en varias de las actividades con la única diferencia en la tarea de Cómputo de tipos a inferirse que FaCT++ no lo realiza por que no tiene un buen soporte en las reglas de inferencia. No obstante ***Pellet*** cumple con todas las métricas y su rendimiento refleja su eficiencia.

4.3 Trabajos Relacionados:

Institución: Universidad Politécnica de Valencia
Área: Departamento de Comunicaciones
Temática: Especificación Owl de una Ontología para Teleeducación en la Web Semántica
Año: 2007
Autor: Roberto Romero Llop

Referencia: Romero, L. R. (2007). *Especificación owl de una ontología para teleeducación en la web semántica.* Recuperado el 21 de Marzo de 2009, de Universidad Politecnica de Valencia; Departamento de Comunicaciones.

Descripción:

Con el objetivo de aprovechar los recursos educativos en la web y debido a los altos costes de producción, nace la necesidad de potencializar la reutilización de estos contenidos. Para lograrlo se crea una ontología para teleeducación, basada en lógica descriptiva y lenguaje owl, se desarrolla e implementan conceptos relacionados con la interacción de los distintos agentes que intervienen en una experiencia de aprendizaje a través de la web.

En el proceso de desarrollo del tema se han creado apartados en los que se realiza un estudio de los razonadores para la evaluación de la ontología, ya que este proceso es de suma importancia para verificar que la Ontología sea coherente en su contenido a expresar.

Análisis:

Para el estudio de los razonadores como herramienta para evaluar ontologías, se analizan dos aspectos específicos: un análisis de las características de cada razonador y otra en base al rendimiento de cada razonador al procesar algunas actividades.

De estas pruebas el razonador ***Pellet*** tuvo el mejor rendimiento en el proceso de evaluación, ya que esta herramienta posee características flexibles para el usuario, información y un soporte para su uso.

Aporte inédito:

La afirmación de que los resultados obtenidos no significa que un razonador sea mejor que otro, sino que simplemente es el mejor para el uso propuesto con Protégé con la ayuda de la interfaz DIG y para las consultas Aboxes y TBoxes que se han planteado. Pellet al ofrecer valores más estables es el seleccionado como el mejor en esta investigación.

Aspectos que se utilizaron de esta Investigación:

La Figura 34, muestra la comparación en cuanto al rendimiento de cada razonador al realizar las dos tareas de chequeo de la consistencia y clasificación de taxonomías que son tomados en cuenta para la evaluación.

	valor (ms)	FaCT ++	Racer 1.9	Pellet 1.3
1) Consistencia TBox	media	**310**	**316**	**173**
	varianza	249	145	9,9
2) Consistencia ABox	media	**2765**	**1576**	**619**
	varianza	532	82	47
3) Clasificación TBox	media	**434**	**391**	**289**
	varianza	295	143	38
4) Clasificación ABox	media	**2488**	**1707**	**588**
	varianza	515	224	158
5) Tipos Inferidos ABox	media	**2450**	**2334**	**2134**
	varianza	501	504	49
6) Simulación de inferencia	media	**2908**	**6681**	**2277**
	varianza	527	1898	43

Figura 34. Comparativa del rendimiento de los razonadores

Para la presente investigación se tomó a la información como uno de los aspectos que determina el uso de estas herramientas y se ve respaldo este aspecto con la descripción de este trabajo desarrollado, para medir el rendimiento de cada razonador se tomaron algunos aspectos incluyendo las dos tareas evaluadas aquí; pero incrementando aspectos como la tarea de cómputo de tipos a inferirse, la capacidad de respuesta que tiene al razonador ante una consulta y se aportó con otras características de cada razonador.

Institución: Universidad de Manchester
Temática: Framework for an Automated Comparison of Description Logic Reasoners
Autor: Tom Gardiner, Dmitry Tsarkov, Ian Horrocks

Referencia: Gardiner, T., Tsarkov, D., & Horrocks, I. (26 de Agosto de 2006). *Framework For an Automated Comparison of Description Logic Reasoner.* Recuperado el 21 de Febrero de 2010

Descripción:

El diseño del lenguaje owl fue influenciado por la lógica de descripción y específicamente en la semántica formal. El enfoque de este trabajo es proporcionar interoperabilidad de los diferentes razonadores así como probar y comparar los razonadores owl con una biblioteca amplia de ontologías. En estudio se apoya en la comparación del rendimiento al realizar las

tareas, utilizar consultas SQL para analizar y presentar los resultados para todo lo que se considere oportuno.

Análisis:

Cada paso de la evaluación se rige en los tiempos para poder comparar el rendimiento de los razonadores.

Type	Status	FaCT++	KAON2	Pellet	RacerPro
All	Success	137	43	143	105
All	CouldNotProcess	24	119	20	60
All	ResourcesExceeded	4	3	2	0
Nominals	Success	4	0	2	0
Nominals	CouldNotProcess	0	5	3	5
Nominals	ResourcesExceeded	1	0	0	0
TransRoles	Success	9	5	9	6
TransRoles	CouldNotProcess	2	6	3	7
TransRoles	ResourcesExceeded	2	2	1	0
Datatypes	Success	91	0	98	62
Datatypes	CouldNotProcess	19	112	14	50
Datatypes	ResourcesExceeded	2	0	0	0
OWL-Lite	Success	43	41	42	43
OWL-Lite	CouldNotProcess	5	6	6	7
OWL-Lite	ResourcesExceeded	2	3	2	0

Figura 35. Tabla Comparativa de los Razonadores

La Figura 34. Muestra una visión general del desempeño de los cuatro razonadores con muchas ontologías de prueba que son capaces de clasificar dentro de los tiempos provistos, luego se centra en las ontologías con un determinado lenguaje. Además se muestra el desempeño en ontologías OWL-Lite. Cabe tener en cuenta que sólo las ontologías que son 100% compatibles son comparadas en este trabajo.

Aporte inédito:

A partir del trabajo realizado se desarrolló un sistema para probar razonadores con ontologías de la vida real, los beneficios son las pruebas autónomas, análisis flexible de los resultados, chequeo de la coherencia y el desarrollo de una biblioteca de prueba que es valioso recurso para la comunidad de ontologías. Se seguirán haciendo pruebas y analizando los resultados para encontrar las inconsistencias que han encontrado.

Aspectos que se utilizaron de esta Investigación:

Una de las hipótesis que se plantea en la investigación es el hecho de no poder asegurar que un razonador es universal y que podrá con el manejo de cualquier ontología, sino que dependerá mucho de los factores que integren la ontología para la utilización de un razonador solo para esa ontología. En este planteamiento este trabajo apoya esta hipótesis

y reafirma con el hecho de haber creado una biblioteca de ontologías que sean factibles para determinados razonadores.

4.4 Aprendizaje del Capítulo:

- ✓ Para el análisis de la ontología (**OER-CC**) se determinó el uso de cuatro razonadores por compartir algunas características y ser los más usados; pero en la práctica se presentó un problema con la interfaz DIG que no permite la comunicación de Protégé y el razonador por lo tanto la selección de razonadores a estudiar se limitó en tres: Pellet, Kaon2, FaCT++.

- ✓ Como un aporte adicional se evaluó otra ontología (***Definición de Conceptos de Investigación Académica)*****,** para la cual se determinó el uso de los cuatro razonadores más usados (Pellet, Kaon2, FaCT++ y Racer Pro) lo que no se dio con la primera ontología. Pero esto permitió un mejor análisis y apoyo en las conclusiones obtenidas.

- ✓ El tipo de licencia de un razonador pone un limitante al momento de usarlo para un proyecto determinado, para usos investigativos el que mejor que se adapta son los Open-Sourse por la libertad que tienen para su estudio y el aporte que se puede realizar para un mejoramiento de la herramienta.

- ✓ En el desarrollo de la primera fase se determinó que FaCT++ es el más eficiente al realizar las tareas de Chequeo de la Consistencia y Clasificación de Taxonomías. No así al realizar la tarea de cómputo de tipos a inferir donde sólo Pellet lo realizó.

- ✓ El razonador Pellet posee características muy flexibles para su uso, como ser multiplataforma, tipo de algoritmo de razonamiento tableaux uno de los más conocidos, licencia open-sourse y brinda un API para su manejo directo.

- ✓ El razonador Pellet soporta consultas de tipo A-box(infiere información de la estructura y sus instancias), en cambio el razonador FaCT++ soporta sólo razonamiento T-box solo sobre la estructura de la ontología. Por ultimo Kaon2 tiene los dos tipos de razonamiento que le da una mayor ventaja desde este punto de vista.

CONCLUSIONES

- En base al análisis desarrollado, se concluye que la web semántica, ha incursionado con grandes avances sobre la representación del conocimiento haciendo uso de las ontologías aplicadas a diferentes aspectos: empresariales, administrativos, médicos, etc. Para un buen desempeño de las ontologías dentro de estos aspectos, surge la necesidad de seleccionar un razonador eficiente que permita el chequeo y verificación del funcionamiento de la misma y así evitar errores cuando esta sea implementada a un problema de la vida real.

- Para la creación de una ontología se debe adoptar una buena metodología que guie el buen desempeño de la misma, en nuestro caso recomendamos Methontology como una de las mejores a recomendar, así mismo el uso de una herramienta para su gestión como lo es Protégé quien posee una amplia información junto con la interfaz DIG que permite la integración de algunos razonadores para el chequeo de la ontología.

- Se han determinado algunos criterios de evaluación como: eficiencia, eficacia, facilidad de instalación y accesibilidad a la información; los mismos, que permitirán evaluar los razonadores desde los distintos aspectos que lo benefician, siendo esto un aporte considerable para futuras investigaciones.

- En base a la experiencia dada, se ha determinado que los razonadores que facilitan la integración con Protégé son: Pellet, FaCT++, Racer Pro, Kaon2; siendo una de las razones para que estas herramientas sean seleccionados al momento de desarrollar el proceso de evaluación y comparación ante otros razonadores.

- Se ha verificado, que aplicando el criterio de eficiencia en el proceso de evaluación y comparación, el razonador más óptimo al obtener los mejores tiempos de ejecución, es FaCT++, debido a que obtuvo un tasa de rendimiento del 73% en el chequeo de la consistencia y un 53% en la clasificación de taxonomías en comparación con los otros razonadores evaluados.

- Se ha comprobado que en el criterio de eficacia el Razonador Pellet, es el que mejor rendimiento presentó al aplicar una matriz de satisfacción de criterios de evaluación, generando una eficacia del 61%, determinando así su factibilidad al momento de necesitar un razonador semántico; no obstante, se deja la apertura para realizar una investigación minuciosa con FaCT++ por los buenos resultados presentados ante las características que se han evaluado.

- Los razonadores son herramientas que verifican que el conocimiento que se desea representar mediante una ontología sea coherente y no tenga inconsistencias, basada en la investigación se determina que no se puede escoger el mejor razonador para todas las ontologías, sino que dependiendo del dominio a representar uno se acoplara a sus necesidades. Pero también es cierto que en varias pruebas y estudios relacionados al tema seleccionan a **Pellet** como el mejor porque no presenta muchos problemas y se acopla fácilmente a las necesidades del usuario.

RECOMENDACIONES

•

Es importante realizar una verificación de los requerimientos técnicos previa a la implementación de las herramientas tales como: Protégé y cada uno de los razonadores que se utilicen en el proceso de desarrollo de la investigación, de tal manera que permita ahorrar tiempo y optimizar el trabajo.

- Los aspectos básicos tomados en cuenta en la investigación como: ventajas, desventajas, instalación, información, plataforma, licencia son aspectos que se determinan como fundamentales e importantes en el desarrollo del presente trabajo.

- El tipo de consultas A-box implica un mayor esfuerzo de inferencia así como computacional, este aspecto debe explotarse más en los razonadores para optimizar su trabajo, ya que el principal objetivo de la creación de ontologías es la información que se pueda obtener de ella.

- Para el usuario principiante se recomienda utilizar un razonador eficaz que cumpla con los criterios de evaluación establecidos en la matriz de satisfacción, en este caso Pellet; en cambio para el usuario experto que tiene un dominio en el tema y su objetivo principal es probar la eficiencia en los tiempos de ejecución al realizar las tareas básicas del razonador en su ontología será de mejor uso FaCT++. Sin embargo el usuario es libre de seleccionar un razonador dependiendo de los criterios que necesita evaluar.

- Las consultas son un aspecto muy importante para incentivar el uso de las ontologías, un aspecto a considerar es el tipo de consulta a realizar, puede ser A-box o T-box, cada razonador tiene experiencia en una de ellas, pero actualmente la tendencia de los razonadores es la integración de ambos razonamientos pero aún están en investigaciones.

- La sintaxis con la que se representa el conocimiento en las ontologías juega un papel importante para una mejor descripción de las mismas, la selección de la lógica de razonamiento que cada razonador utiliza, delimitara el nivel de expresión del dominio de conocimiento.

- Una recomendación final e importante, es que la ontología que se desea evaluar se acople a un razonador según sus requerimientos, aquí se presenta una tabla, que identifica las características más importantes que se deben tomar en cuenta para la aplicación y uso de un razonador.

Kaon2	Razonador OWL Método eficaz para consultas A-boxes extensas. Basado en Lógica de Primer-Orden para la resolución de cálculos. Soporte de reglas SWRL-DL Utiliza el método de razonamiento Datalog
Pellet	Razona con tipo de datos definidos en el estándar XML-Shema Tipo de datos basado en el esquema XML Realiza consultas T-Box y A-box Insertar RDF/XML Formatos N-TRIPLE Uso del razonamiento Tableaux
Racer Pro	Ayuda al uso de recursos computacionales. Comprueba la coherencia en una ontología OWL Utiliza el razonamiento Tableaux Ofrece servicio de razonamiento para T-boxes y A-boxes.
FaCT++	No tiene soporte para datos que no sean String FacT++ no tiene soporte para el razonamiento con A-box de la ontología Utilización el razonamiento Tableaux

Tabla 1. Características Relevantes de los Razonadores

BIBLIOGRAFÍA

- Alonso, J., & Cordón, A. (2003). *Tema2: Equivalencias y Formas Normales.* Recuperado el 25 de Marzo de 2009, de Universidad de Sevilla, Departamento de Ciencias de la Computación e Inteligencia Artificial.

- Alonso, J., Hidalgo, M., Martin, F., & Ruiz, J. (2006). *Formalización de la lógica Descriptiva ALC en PVS.* Recuperado el 25 de Marzo de 2009, de Grupo de Lógica Computacional; Universidad de Sevilla.

- APIProtégé. (2009). *Protege-OWL API Razonamiento.* Recuperado el 23 de Septiembre de 2009, de http://protegewiki.stanford.edu/wiki/ProtegeReasonerAPI

- Areces, C., & Rijke, M. (2001). *From Description to Hybrid Logics, and Back.* Recuperado el 23 de Marzo de 2009, de Advances in Modal Logic, CSLI Publications.

- Arturo(2008)., F. F. (2008). *Construcción de Ontologías OWL .* Recuperado el 6 de Marzo de 2009, de Universidad del Bosque en Bogota. Investigacion y Desarrollo .

- Baade, f., Lutz, C., & Suntisrivaraporn, B. (2002). *CEL—A Polynomial-time Reasoner for Life Science Ontologies.* Recuperado el 7 de Julio de 2009

- Barceló, M., Guzmán, G., & Pérez, A. (2006). *La Web Semántica como apoyo a la Gestión del Conocimiento y al Modelado Organizacional.* Recuperado el 20 de Marzo de 2009, de Revista Ingeniería Informática.

- Barceló, P. (2008). *Introducción a la Lógica Proposicional.* Recuperado el 23 de Marzo de 2009

- Bechhofer, S. (2003). *The DIG Description Logic Interface: DIG/1.1.* Recuperado el 10 de Marzo de 2009

- Bechhofer, S. (2003). *The DIG Description Logic Interface: DIG/1.1.* Recuperado el 24 de Abril de 2009

- Boley, H., Grosof, B., & Tabet, S. (2005). *RuleML Tutorial , Disclaimer, The information presented in this document is preliminary.* Recuperado el 10 de Mayo de 2009, de http://ruleml.org/papers/tutorial-ruleml-20050513.html

- Chang, D., & Gramajo, F. (Junio de 2007). *Análisis e Implementación de una Ontología de Actividades Económicas y Productos para Pymes de Guatemala.* Recuperado el Enero de 2010, de http://biblioteca.usac.edu.gt/tesis/08/08_7964.pdf

Piedra, N., Chicaiza, J., López, J., Martínez, O., & Tovar, E. (s.f.). *An Approach for Description of Open Educational Resources based on Semantic Technologies.* Recuperado el 30 de Octubre de 2010

- Cel. (s.f.). *A polynomial-time Classifier for the description logic EL+.* Recuperado el 4 de Julio de 2009, de Página oficial de CEL: http://lat.inf.tu-dresden.de/systems/cel/

- Cohuo, A. M. (s.f.). *Base de Datos Relacionales.* Recuperado el 10 de Abril de 2009, de http://www.google.com.ec/url?sa=t&source=web&ct=res&cd=20&ved=0CDUQFjAJOAo&url=http%3A%2F%2Fwww.itescam.edu.mx%2Fprincipal%2Fsylabus%2Ffpdb%2Frecursos%2Fr23263.PPT&rct=j&q=Prolog%2BDatalog&ei=inRGS975DYWWtgeagO3zAQ&usg=AFQjCNE_48cng6Js28Eg3UZumPsMdQZ4cg

- Corcho, O., Fernández, M., & Goméz, A. (2004). *Construcción de ontologías legales con la metodología METHONTOLOGY y la herramienta WebODE, Facultad de Informática.* Recuperado el 24 de Junio de 2009

- Cuevas, A. (2006). *The IMS Service Platform - A Solution for Next-Generation Networks Operators to Be More than Bit Pipes.* Recuperado el 7 de Febrero de 2009, de IEEE Comm. Magazine.

- FaCT++. (2007). *Página oficial de FaCT.* Recuperado el 28 de Junio de 2009, de http://owl.man.ac.uk/factplusplus/

- Floréz, F. H. (2008). *Construcción de Ontologías OWL* . Recuperado el 18 de Febrero de 2009, de Universidad del Bosque en Bogota. Investigacion y Desarrollo .

- Floréz, H. (2008). *Construcción de Ontologías OWL. Disponible en:* . Recuperado el 22 de Julio de 2009, de http://www.udistrital.edu.co/comunidad/dependencias/revistavinculos/VINCULOS/revista/7edicion/22007702.pdf

- Franconi, E. (Marzo de 2002). *Description logics lecture notes.* Recuperado el 23 de Marzo de 2009, de Pres. slides, University of Manchester, .

- García, A., Samper, J., & Castillo, E. (2006). *Articulo Expresividad Limitada en el uso de Editores de Ontologías. .* Recuperado el 15 de Febrero de 2009, de Revista Digital Universitaria. Volumen7.

- Gardiner, T., Tsarkov, D., & Horrocks, I. (26 de Agosto de 2006). *Framework For an Automated Comparison of Description Logic Reasoner.* Recuperado el 21 de Febrero de 2010

- Gilliland, A. (2000). *Introduction to metadata: pathways to digital information.* Recuperado el 10 de Febrero de 2009, de Getty Information Institute.

- Goméz, M., & Pérez, J. (2006). *Una introducción a la web semántica.* Recuperado el 7 de Febrero de 2009

- Gomez, P. A., Fernandez, L. M., & Vicente, A. (2004). *Ontological Engineering with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. .* Recuperado el 27 de Mayo de 2009, de Facultad de Informatica, Universidad Politécnica de Madrid, Campus de Montegancedo.
- Grigoris, A., & Harmelen, F. V. (s.f.). *Web Ontology Languaje: OWL.* Recuperado el 2 de Mayo de 2009
- GuideRacerPro. (2007). *RacerPro User's Guide Version 1.9.2.* Recuperado el 12 de Junio de 2009, de Racer System GmbH& Co.KG. : www.racer-systems.com
- Heflin, J. (2004). *OWL Web Ontology Language Use Cases and Requirements.* Recuperado el 12 de Abril de 2009
- Horrocks, G. B., Volz, R., & Decker, S. (2003). *Description Logic Programs: Combining Logic Programs with Description Logic. .* Recuperado el 15 de Mayo de 2009
- Horrocks, I. (2004). *SWRL:A Semantic Web Rule Language Combining OWL and RuleML.* Recuperado el 4 de Mayo de 2009, de W3C Member Submission.
- Horrocks, I. (2003). *The FaCT System. .* Obtenido de http://www.cs.man.ac.uk/~horrocks/FaCT/
- Horrocks, I., & Patel-Schneider, P. (2003). *Three theses of representation in the semantic web. .* Recuperado el 5 de Abril de 2009, de In Proc. of the Twelfth International World Wide Web Conference.
- Horrocks, I., Tsarkov, D., & Gardiner, T. (2006). *Automated Benchmarking of Description Logic Reasone.* Recuperado el 4 de Agosto de 2009, de http://www.google.ca/search?hl=es&q=Automated+Benchmarking+of+Description+Logic+Reasoners&aq=f&aqi=&aql=&oq=&gs_rfai
- Hustadt, U., & Schmidt, R. A. (2000). *In Dyckhoff, R. (eds), Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2000). .* Recuperado el 6 de Julio de 2009, de Lecture Notes in Artificial Intelligence, Vol. 1847, Springer, 67-71. Abstract, BiBTeX, PostScript (Copyright © Sp.
- Hustadt, U., & Schmidt, R. A. (2000). *MSPASS: Modal reasoning by translation and first-order resolution.* Recuperado el 21 de Mayo de 2009
- Hustadt, U., Motik, B., & Sattler, U. (2004). *Reducing SHIQ- Descrption Logic to Disjunctive Datalog Programs. .* Recuperado el 24 de Mayo de 2009, de In Proceedings of the 9th International Conference on Knowledge Representation and Reasoning.

- IrisGuide. (2008). *IRIS-Integrated Rule Inference System-API and User Guide.* Recuperado el 26 de Julio de 2009, de http://www.iris-reasoner.org/pages/user_guide.pdf

- Jacobs, I. (2001). *Director de Comunicaciones del W3C.* Recuperado el 4 de Febrero de 2009

- Knublauch, A. (2005). *The Protégé OWL Plugin: An Open Development Environment for Semantic Web.* Recuperado el 5 de Marzo de 2009, de Applications.Stanford University. Extraído del World Wide Web. : www-scf.usc.edu/~csci586/ppt-2005/bhavin.ppt.

- Knublauch, H., Fergerson, R., Noy, N., & Musen, M. (2004). *The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications,.* Recuperado el 29 de Mayo de 2009, de In Proceedings of the 3rd International Semantic Web Conference, Hiroshima, Japon.

- Labra, J., & Fernandez, D. (s.f.). *Lógica de predicados.* Recuperado el 26 de Marzo de 2009, de Universidad de obviedo.

- Lama, M., & Sánchez, E. (2008). *Análsis de técnicas de aprendizaje adaptativo con ontolgías.* Recuperado el 30 de Mayo de 2009, de Proyecto Suma elearning multimodal y adaptativo. .

- Lamarca, L. J. (2009). *Ontologías.* Recuperado el 5 de Marzo de 2009, de http://www.hipertexto.info/documentos/ontologias.htm.

- Lassila, O., & Swick, R. (1999). *Resource Description Framework (RDF). .* Recuperado el 30 de Abril de 2009

- Lozano, A. (2001). *Ontologías en la Web Semántica; Area de Lenguajes y Sistemas Informático.* Obtenido de Universidad de Extremadura. España, I Jornadas de Ingeniería Web'01. .

- Millis, D. (2008). *Semantic Wave 2008 Report.* Recuperado el 6 de Febrero de 2009, de Managing Director, Project 10X.

- Motik, B. S. (2005). *KAON2 – A Scalable Reasoning Tool for the Semantic Web. .* Recuperado el 15 de Mayo de 2009, de In Proceedings of the 2nd European Semantic Web Conference (ESWC2005).

- Motik, B., & Sattler, U. (04 de Noviembre de 2005). *Practical DL Reasoning over Large Aboxes with KAON2.* Recuperado el 10 de Junio de 2009

- Muñoz, S. L. (2009). *Representación de Ontologías en la Web Semántica.* Recuperado el 11 de Febrero de 2009

- O'Reilly. (2005). *What Is Web 2.0. Design Patterns and Business Models for the Next Generation of Software.* Recuperado el 6 de Febrero de 2009
- OntoEdit. (2004). *Herramienta de creación de ontologías OntoEdit, Página Oficial.* Recuperado el 19 de Marzo de 2009
- Ortiz, A. (s.f.). *Web Semántica Hoy* . Recuperado el 5 de Febrero de 2009
- Ovejero , B., & Martín , S. (2001). *Avizora*. Recuperado el 20 de Julio de 2009, de ¿Qué son los tesauros y las ontologías?: http://www.avizora.com/publicaciones/pedagogia/textos/0047_que_es_tesauro_ontologia.htm
- Papataxiarhis, V. (2007). *Protege - OWL Tutorial, Course: Knowledge Technologies.* Recuperado el 18 de Marzo de 2009, de Department of Informatics and Telecommunications , University of Athens –Greece.
- Pérez, V. D. (2007). *Maestro de la Web.* Recuperado el 10 de Febrero de 2009, de http://www.maestrosdelweb.com
- Philippe, M. (2002). *Knowledge Representation, Sharing and Retrieval on the Web.* Recuperado el 22 de Abril de 2009, de Distributed System Technology Centre. .
- Prud'hommeaux, E. S. (2008). *SPARQL Query Language for RDF.* Recuperado el 26 de Mayo de 2009, de W3C Recommendation.: http://www.w3.org/TR/rdf-sparql-query
- RacerPro. (s.f.). *Página oficial de Racer Pro.* Recuperado el 4 de Junio de 2009, de http://www.racer-systems.com/index.phtml
- RacerProNotes. (2007). *Release Notes for RacerPro 1.9.2 beta.* Recuperado el 7 de Junio de 2009, de Racer Systems GmbH & Co. KG.
- Piedra, N., Chicaiza, J., Martinez, O., & Tovar, E. (2009). *Una Propuesta para la Descripción de Recursos Educativos Abiertos, basada en Tecnologías Semánticas.* Recuperado el 31 de Octubre de 2010, de http://ieeexplore.ieee.org/search/srchabstract.jsp?tp=&arnumber=5492453&searchWithin%3DAuthors%3A.QT.Chicaiza%2C+Janneth.QT.%26openedRefinements%3D*%26searchField%3DSearch+All
- Piedra, N. Chicaiza, J. López, J. Martínez, O. Tovar, M. (2010). *An approach for description of Open Educational Resources based on semantic technologies*. In Proceeding of the Education Engineering Conferences (EDUCON), IEEE, 2010, pp. 1111 -1119. Recuperado el 30 de Octubre de 2010ç

- Roldán, G. M., & Aldana, M. J. (2007). *DBOWL: Persistencia y Escalabilidad de Consultas y Razonamientos en la Web Semántica.* Recuperado el 3 de Abril de 2009, de Universidad de Málaga.
- Romero, L. R. (2007). *Especificación owl de una ontología para teleeducación en la web semántica.* Recuperado el 21 de Marzo de 2009, de Universidad Politecnica de Valencia; Departamento de Comunicaciones.
- Ruckhaus, E. (2009). *Clase 7.* Recuperado el 6 de Abril de 2009
- Schmidt, R. (2007). *MSPASS: Documentation, School of Computer Science.* Recuperado el 21 de Julio de 2009, de http://www.cs.man.ac.uk/~schmidt/mspass/documentation.html
- Schneider, P. P., Hayes, P., & Horrocks, I. (2004). *OWL web ontology language Abstract Syntax and Semantics.* Recuperado el 1 de Junio de 2009, de W3C Recommendation .
- Sirin, E., Parsia, B., Cuenca, B., Kalyanpur, A., & Katz, Y. (03 de 11 de 2005). *Pellet: A Practical OWL-DL Reasoner.* Recuperado el 03 de 12 de 2009
- Tsarkov, D., & Horrocks, I. (2003). *DL Reasoner vs.First-Order Prover.* Recuperado el 2 de Abril de 2009, de University of Manchester.
- Tsarkov, D., & Horrocks, I. (2006). *FaCT++ Desception Logic Reasoner: System Description. .* Recuperado el 23 de Junio de 2009, de School of Computer Science, The University of Manchester.
- Wagner, G., Tabet, S., & Bole, H. (2003). *MOF-RuleML: The Abstract Sytax of RuleML as a MOF Model.* Recuperado el 28 de Abril de 2009

ANEXOS

ANEXO1

Instalación de los Razonadores

Para realizar la debida evaluación de los diferentes razonadores el primer paso a tomar es la instalación de los razonadores, aquí una explicación detallada:

RAZONADOR PELLET 1.5.2

Con la utilización del Protégé 3.4.1 viene integrado el razonador Pellet 1.5.2 así que no es necesario realizar la instalación de este razonador.

RAZONADOR RACER PRO 2.0

Sitio de Descarga: http://www.racer-systems.com/products/download/preview20.phtml

Este Razonador Trabaja en el puerto ***8088*** así que debemos configurarlo en el Protégé como se muestra en la Figura 36.

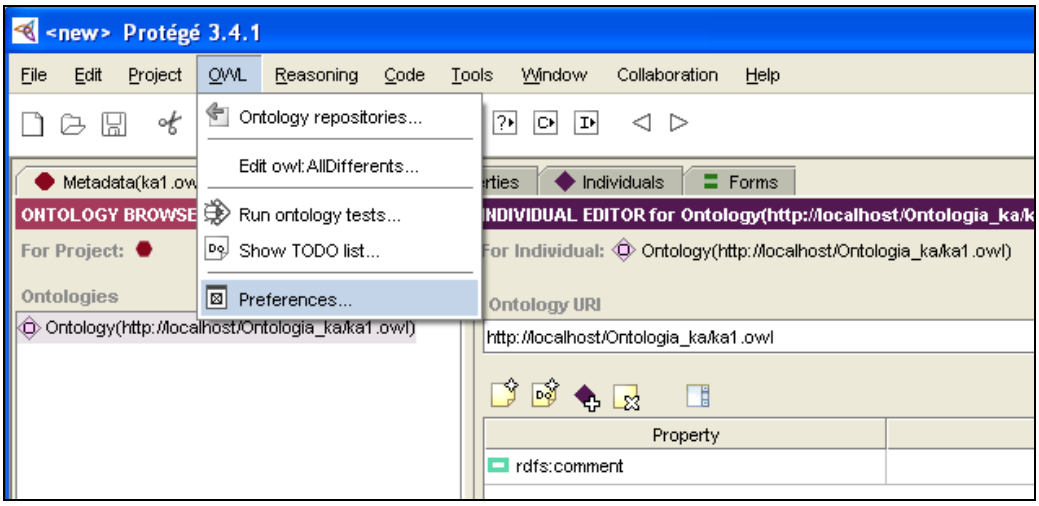

Figura 36. Opción de Preferencias

En la Figura 37.Se especifica el campo específico donde se debe realizar el cambio de puertos, ya que no todos los Razonadores trabajan en el mismo sino en diferentes.

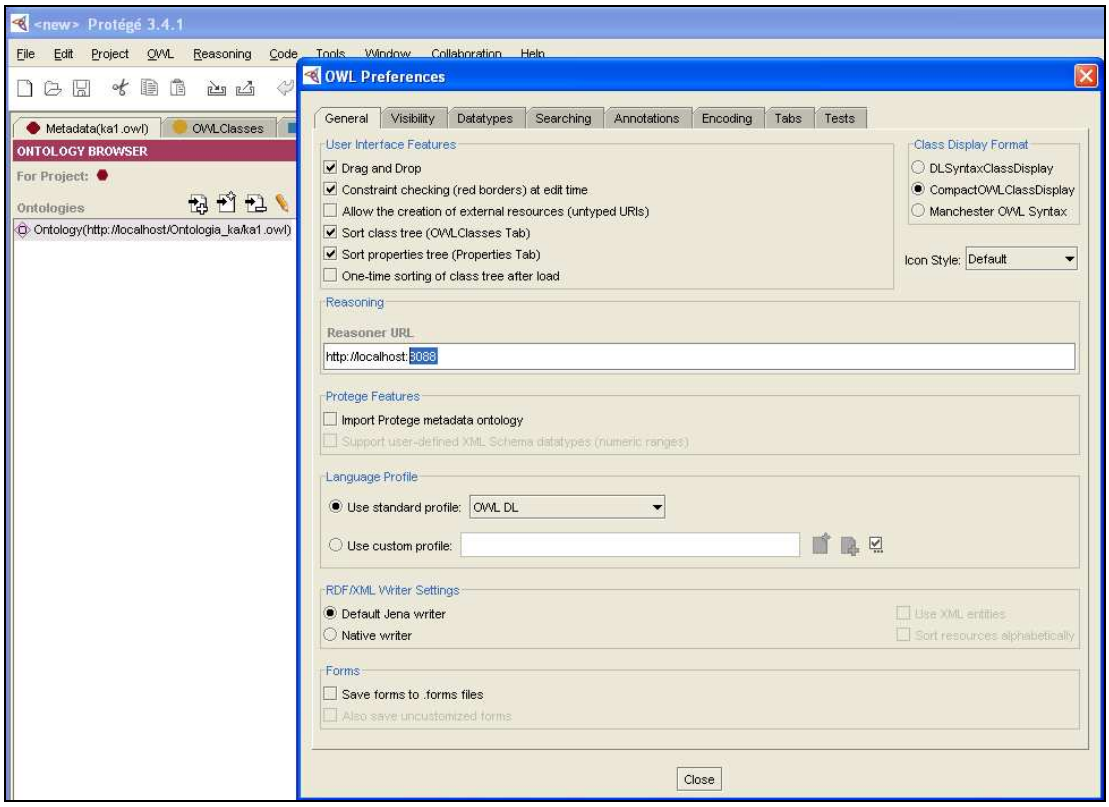

Figura 37. Modificando de Puerto

RAZONADOR KAON2

Este razonador trabaja con ***Java 1.5*** por lo tanto es importante como primer requerimiento tenerlo instalado conjuntamente con el ***jdk 1.5***.

Figura 38. Configuración de CLASSPATH

En las Variables de Entorno en la variable CLASSPATH debemos poder la dirección donde está instalado…. Java\jdk1.5\lib;. Hay que tomar en cuenta el punto al final de la dirección.

Para bajar el instalador de este Razonador: http://kaon2.semanticweb.org/#download

Una vez descargado él Razonador, se procede a colocarla dentro de la Carpeta de Protégé para ejecutarlo entramos en la Consola de Windows y manualmente entramos a la carpeta de Protégé y a la carpeta del Razonador Kaon2 para poder ejecutar el siguiente comando para la activación del Servicio del Razonador; tal como lo muestra en las siguientes imágenes.

Comando a Ejecutar:

java -cp kaon2.jar org.semanticweb.kaon2.server.ServerMain -registry -rmi -ontologies server_root -dig -digport 8088

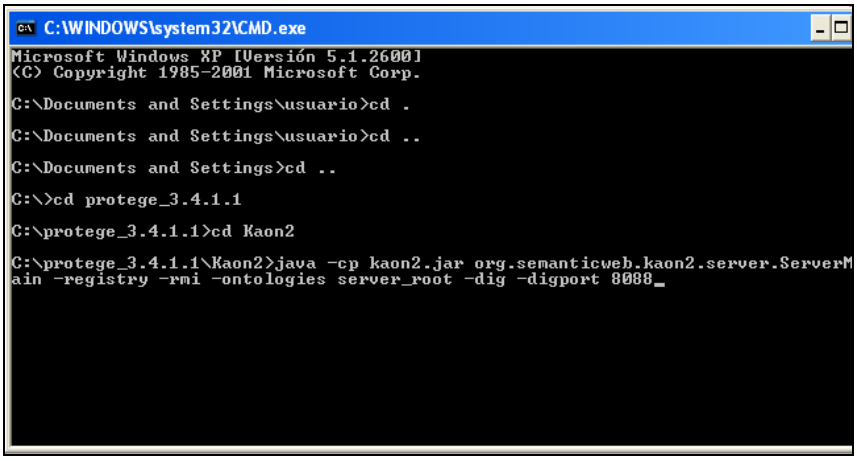

Figura 39. Consola de Windows

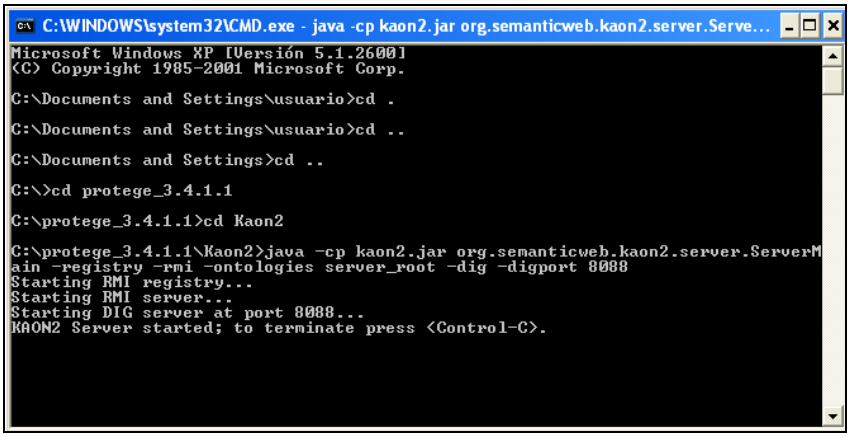

Figura 40. Levantamiento del Servicio

Ahora dentro de Protégé se debe hacer el cambio respectivo del puerto con el que funciona el Razonador Kaon2 que es el ***8088.***

Una vez cumplidos todos los pasos antes dichos el Razonador Kaon2 funcionara perfectamente.

RAZONADOR FACT++

Este razonador lo podemos descargar de la siguiente dirección: http://owl.man.ac.uk/factplusplus/

Este Razonador trabaja con el puerto ***3490.***

Prueba con los Razonadores

RAZONADOR PELLET

Es un razonador basado en el algoritmo *Tableaux* desarrollada para lógica descriptiva; como primera prueba se lo ha realizado con ayuda del Protégé que tiene integrado este razonador a continuación los resultados obtenidos analizando una ontología simple como lo es ***pizza.owl***

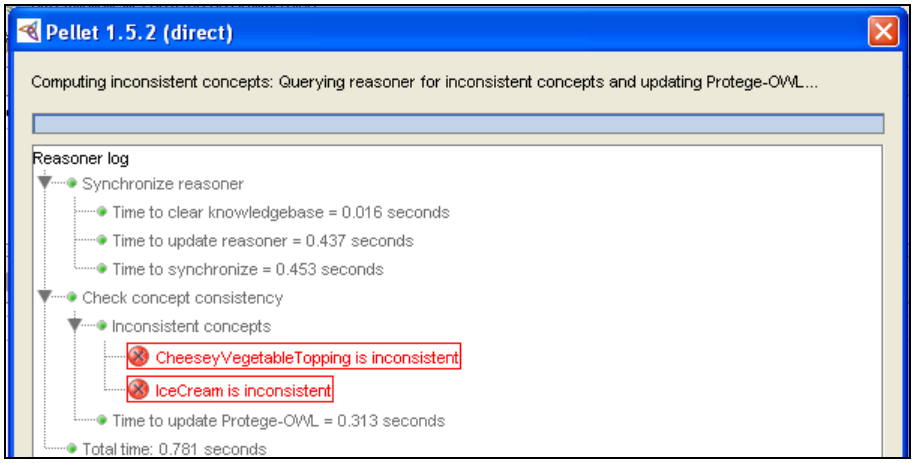

Figura 41. Chequeo de Inconsistencias

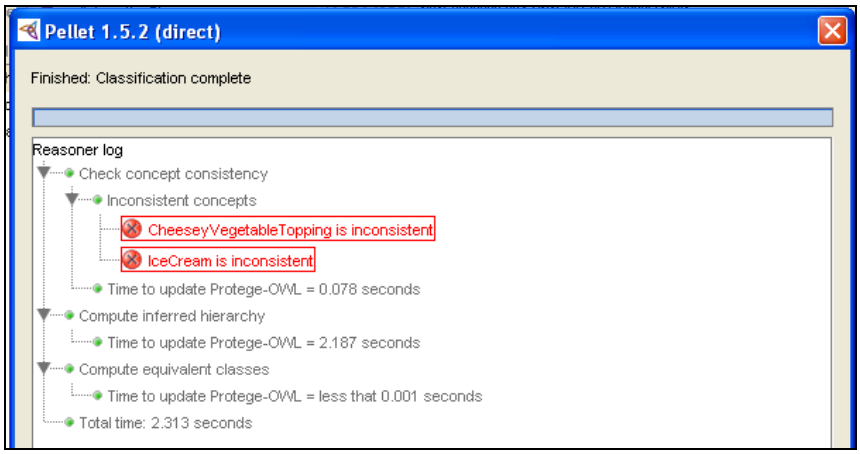

Figura 42. Clasificación de Taxonomías

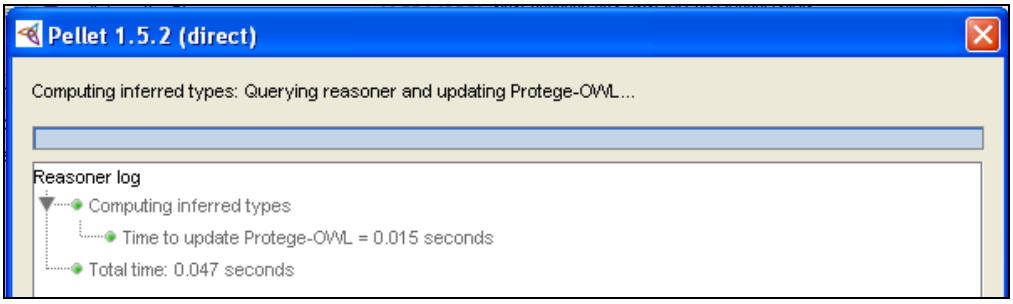

Figura 43. Tipos de Inferencias

RAZONADOR RACER PRO

Este razonador presta los siguientes servicios a las ontologias OWL y Descripciones RDF

- Comprobar la coherencia de una ontología OWL y un conjunto de descripciones de datos.
- Buscar relaciones implícitas de subclase inducida por la declaración en la ontología.
- Buscar sinónimos de los recursos (ya sean clases o los nombres de instancia).
- Dado la extensa información de documentos OWL (casos OWL y sus interrelaciones) debe ser consultado para las aplicaciones cliente, un OWL-QL consulta de procesamiento sistema está disponible como un proyecto de código abierto para Racer Pro.
- Cliente HTTP para la recuperación de los recursos importados de la web. Múltiples recursos pueden ser importados en una ontología.
- Responder a la consulta incremental para tareas de recuperación de información de información (recuperar los resultados *n* de las siguiente consulta).

Racer Pro como sistema de Lógica de Descripción

Es un sistema de representación del conocimiento que aplica un cálculo optimizado de tableaux para una lógica de descripción muy expresiva. Para las pruebas se lo realizo en el Protégé que arrojo los siguientes resultados:

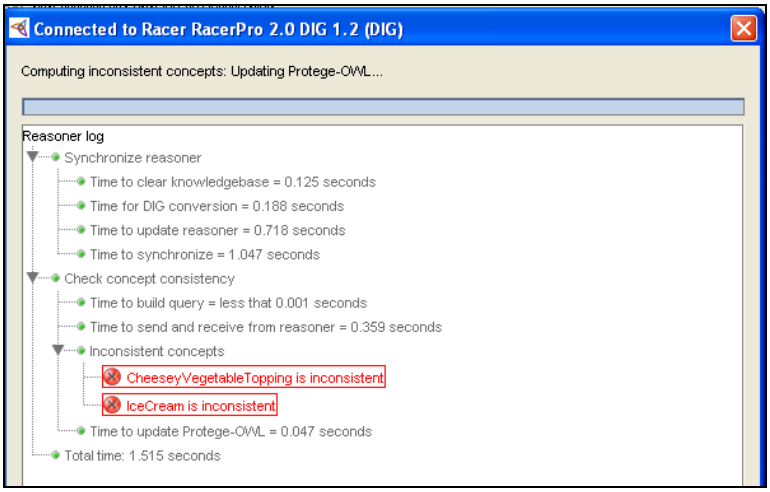

Figura 44. Chequeo de Inconsistencias

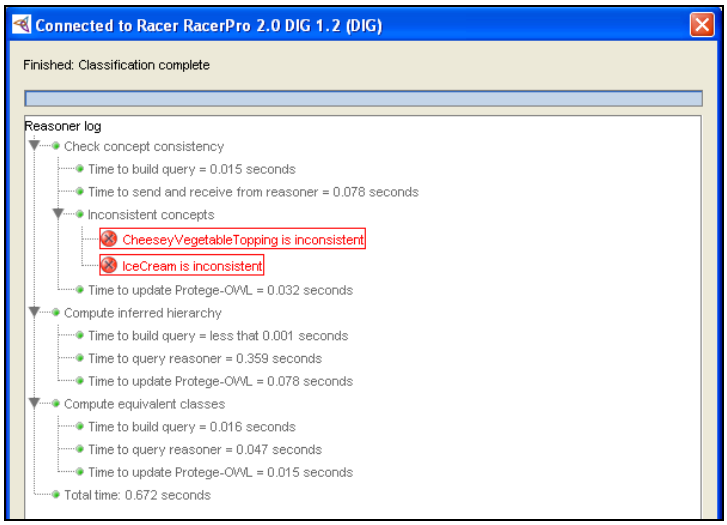

Figura 45. Clasificación de Taxonomías

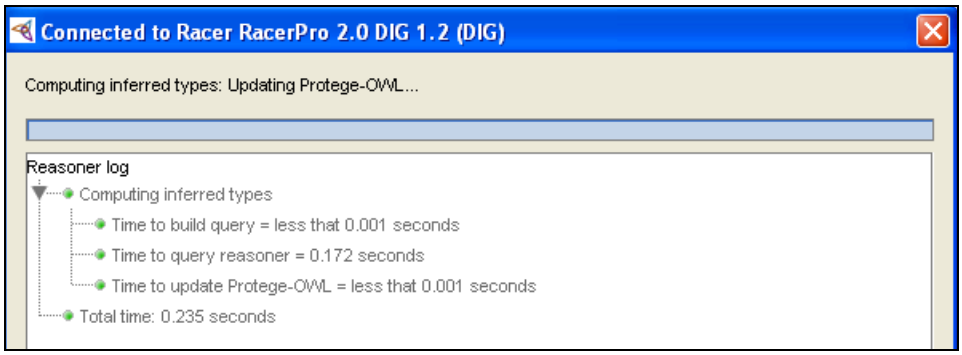

Figura 46. Tipos de Inferencias

RAZONADOR KAON2

Este razonador a diferencia de los otros no utiliza el algoritmo ***Tableaux***, sino que utiliza una serie de algoritmos que reducen una base de conocimiento SHIQ(D) a un programa *datalog* disjunto [HUSTADT], lo que aumenta de forma considerable el rendimiento del razonador a la hora de ejecutar los procesos de inferencia.

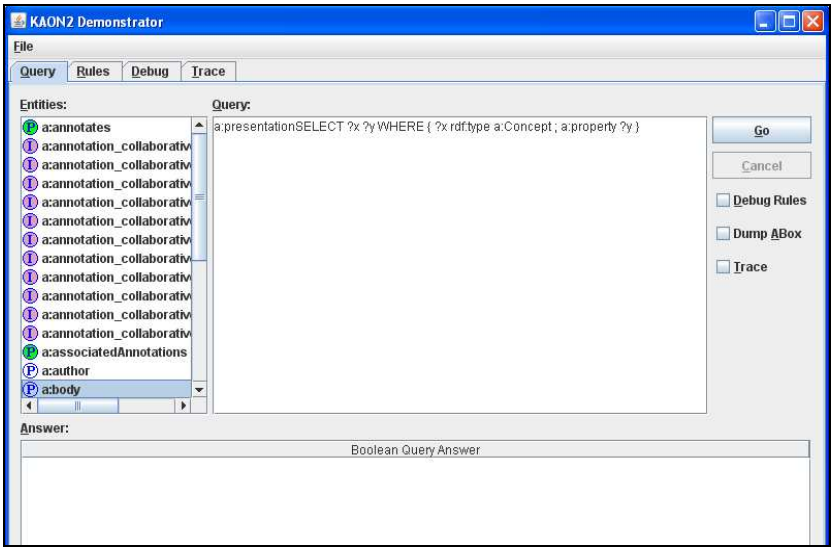

Figura 47. Razonador Kaon2

Proceso de Evaluación del Segundo y Tercer Experimento

SEGUNDO EXPERIMENTO

En esta segunda parte de pruebas la mayoría de los razonadores ya no realizan un análisis previo como lo realizan en el primer experimento sino directamente pasan a realizar el análisis de la consistencia de la ontología.

PELLET 1.5.2

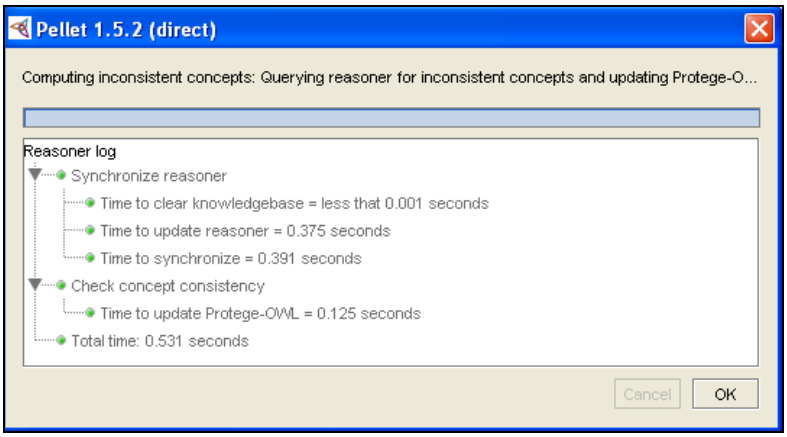

Figura 48. Chequeo de Consistencias

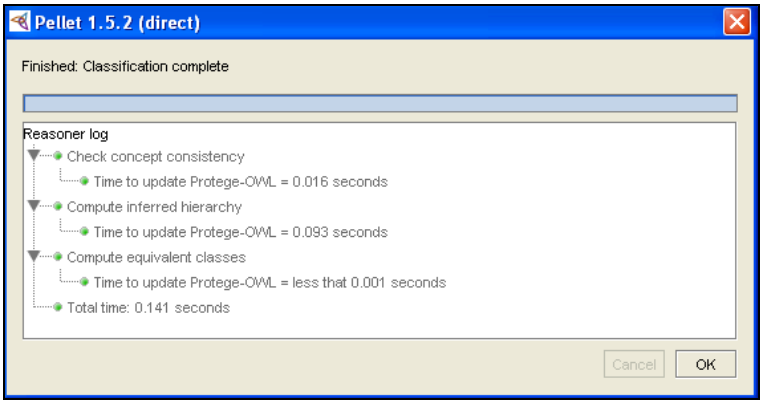

Figura 49. Clasificación de Taxonomías

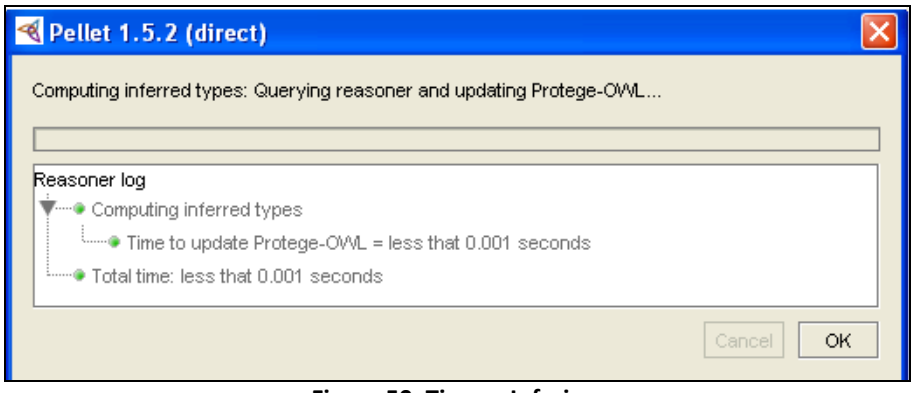

Figura 50. Tipos a Inferirse

RACER PRO 2.0:

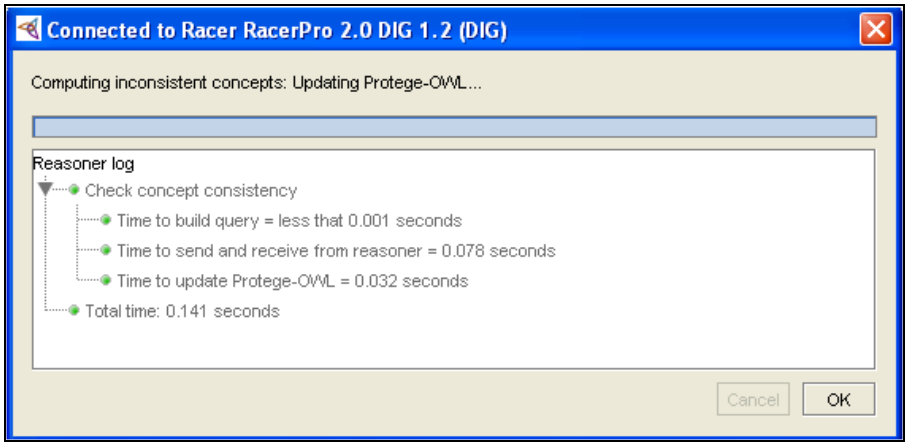

Figura 51. Chequeo de Consistencias

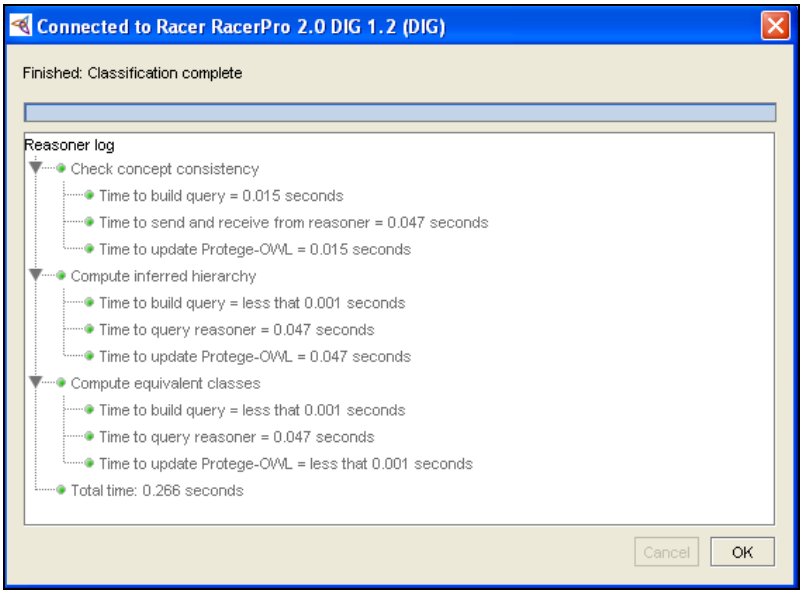

Figura 52. Clasificación de Taxonomías

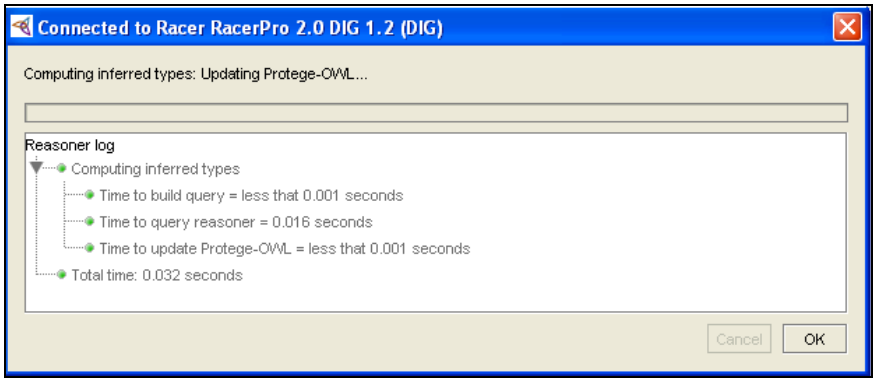

Figura 53. Tipos a Inferirse

KAON2 (VERSIÓN 2008/06/29)

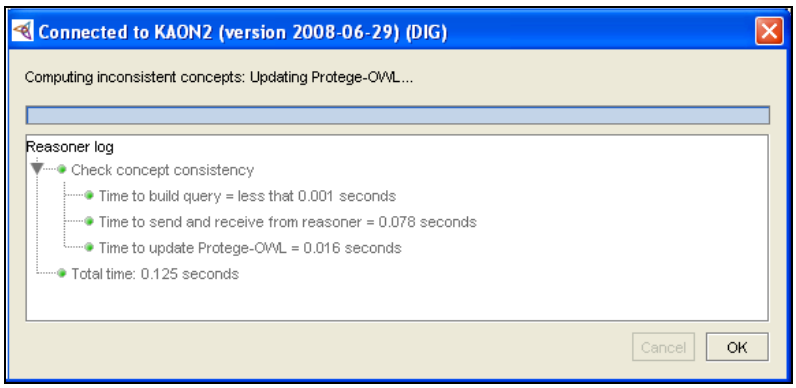

Figura 54. Chequeo de Consistencias

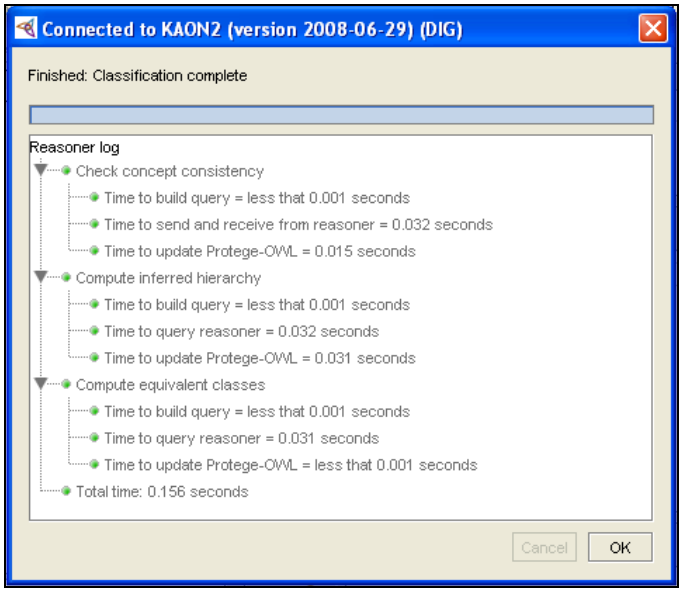

Figura 55. Clasificación de Taxonomías

FACT ++ 1.3.0

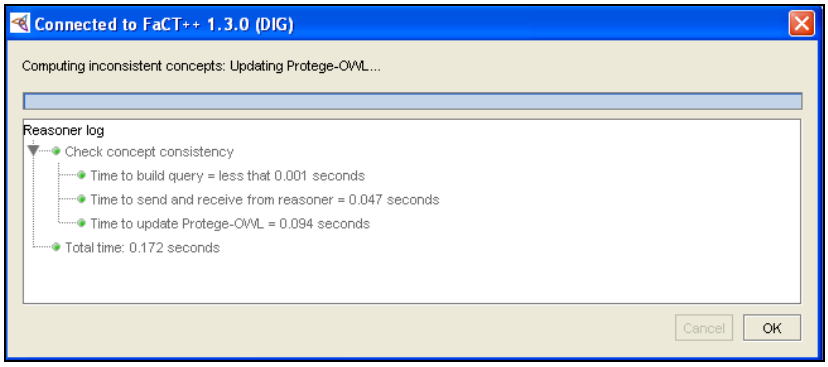

Figura 56. Chequeo de Consistencias

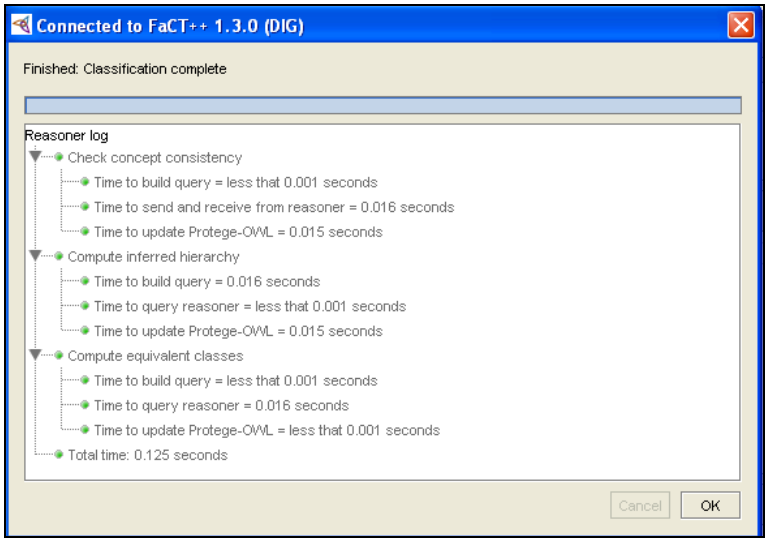

Figura 57. Clasificación de Taxonomías

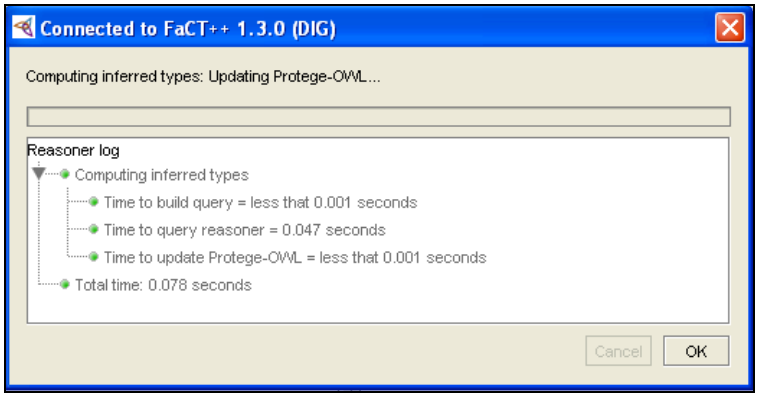

Figura 58. Tipos a Inferirse

TERCER EXPERIMENTO

Para esta tercera parte de las pruebas los razonadores tienen un promedio de tiempo mucho menor que al realizar las primeras experiencias.

PELLET 1.5.2:

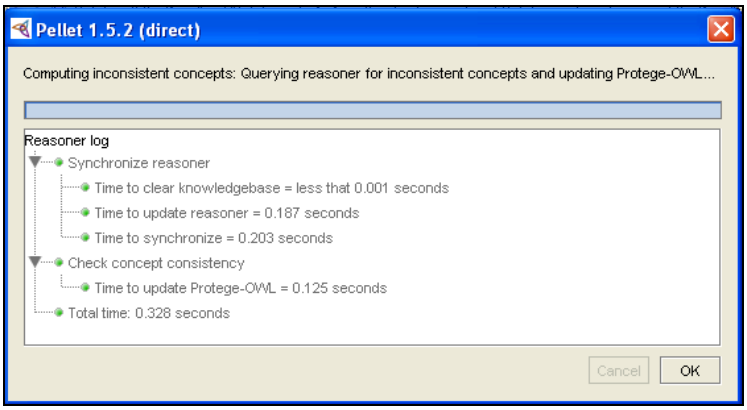

Figura 59. Chequeo de Consistencias

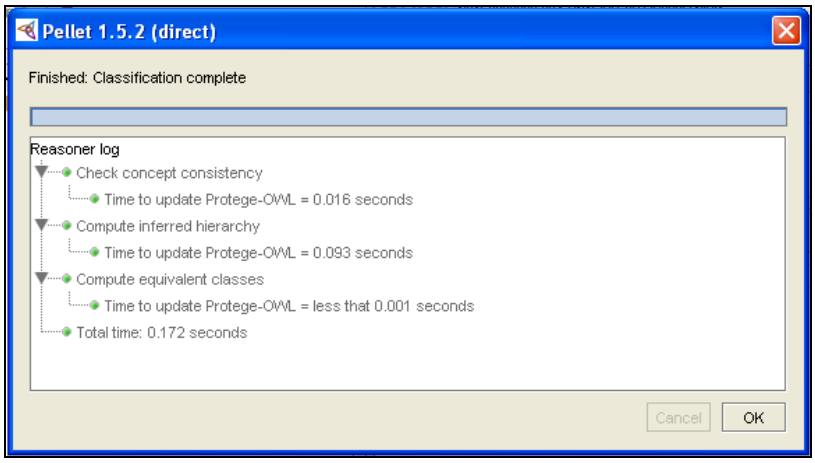

Figura 60. Clasificación de Taxonomías

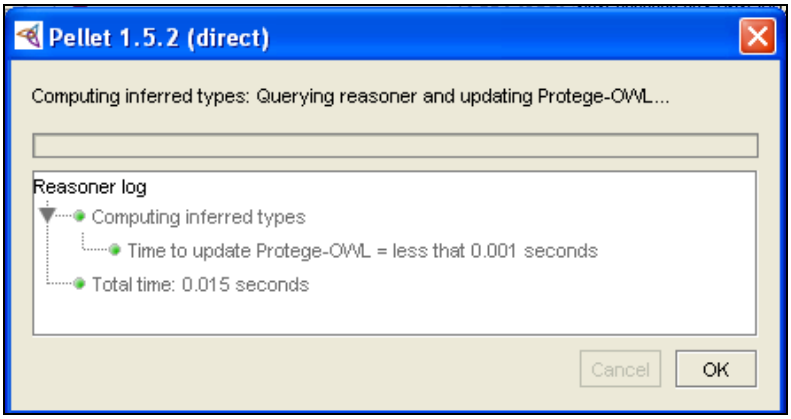

Figura 61. Tipos a Inferirse

RACER PRO 2.0:

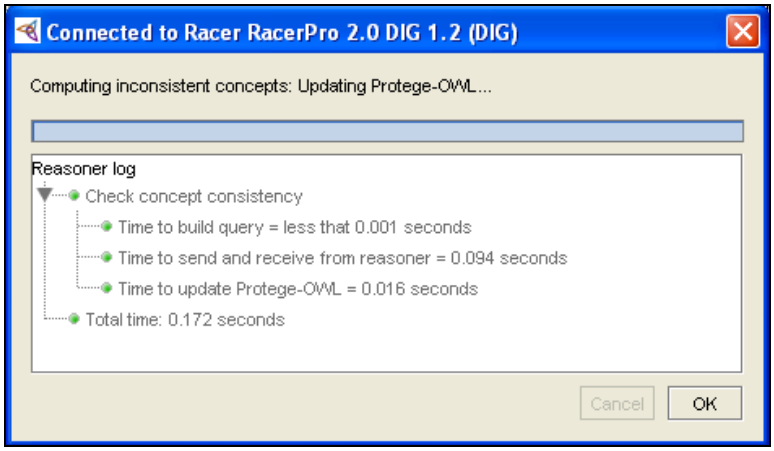

Figura 62. Chequeo de Consistencia

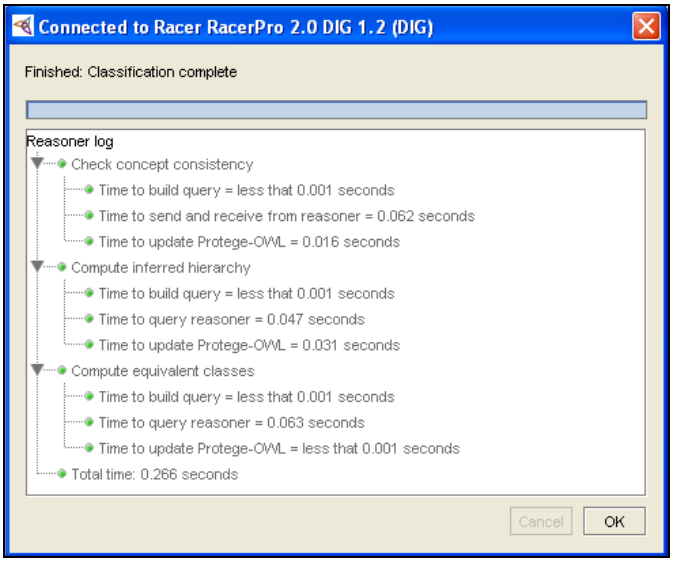

Figura 63. Clasificación de Taxonomías

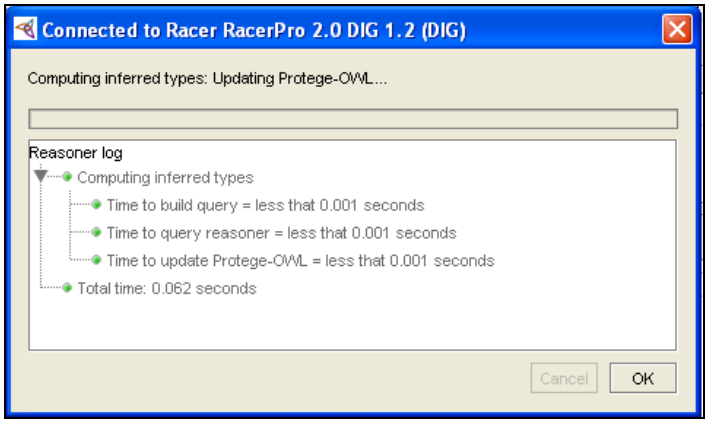

Figura 64. Tipos a Inferirse

KAON2 (VERSIÓN 2008/06/29)

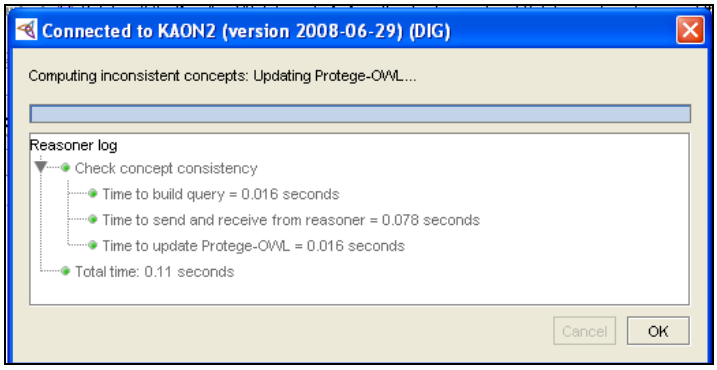

Figura 65. Chequeo de Consistencias

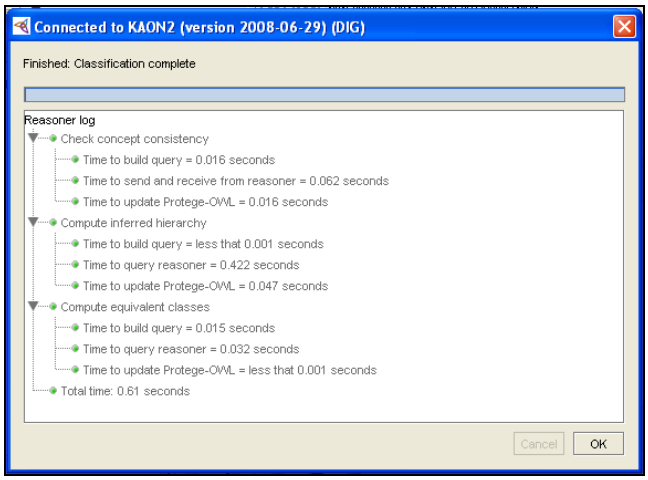

Figura 66. Clasificación de Taxonomías

FACT ++ 1.3.0

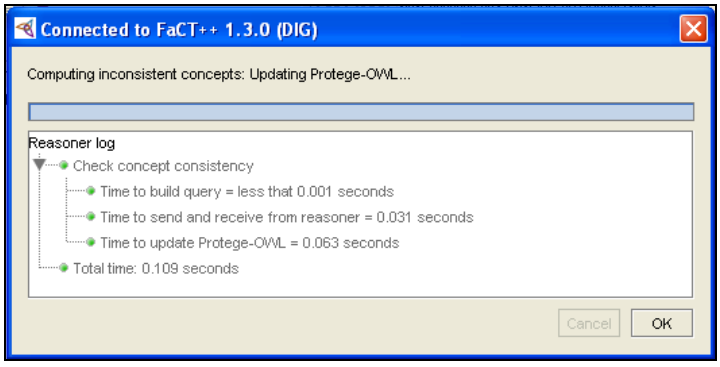

Figura 67. Chequeo de Consistencias

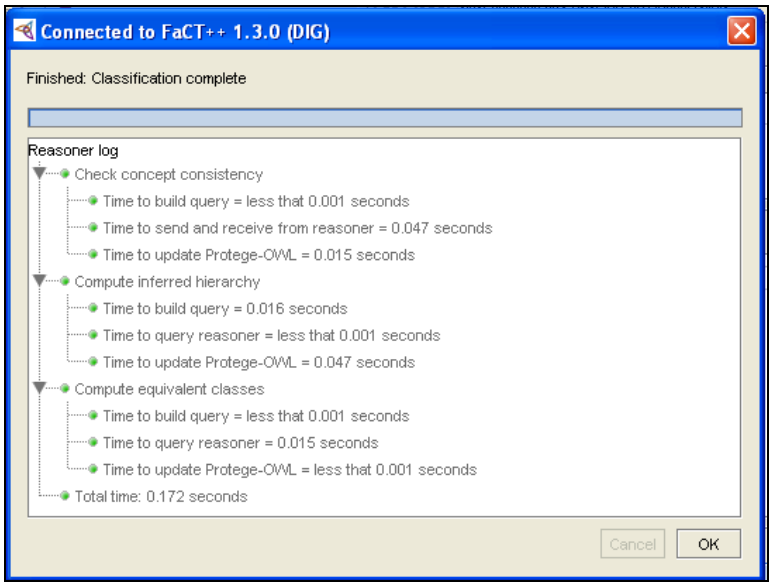

Figura 68. Clasificación de Taxonomías

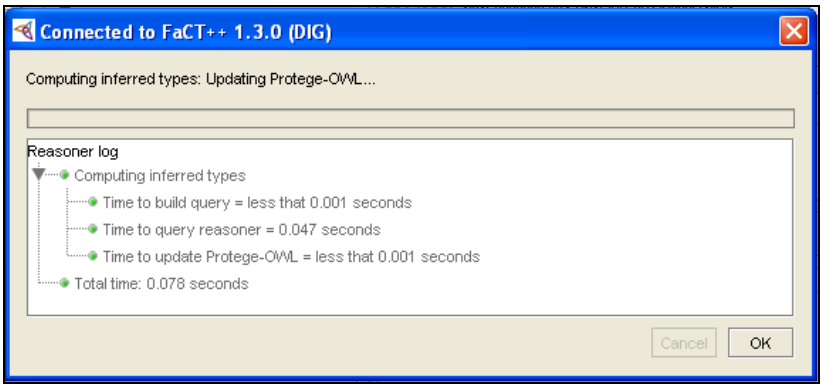

Figura 69. Tipos a Inferir

Debe realizarse estas actividades previas al desarrollo de las tres actividades del razonador:

- Instalación y configuración de los razonadores.
- Carga de la ontología.
- Prueba con cada razonador

La primera actividad que se realiza es el chequeo de la consistencia de la ontología

ANEXO 2

Primera Prueba con Razonadores

A continuación el primer razonador **Pellet 1.5.2:**

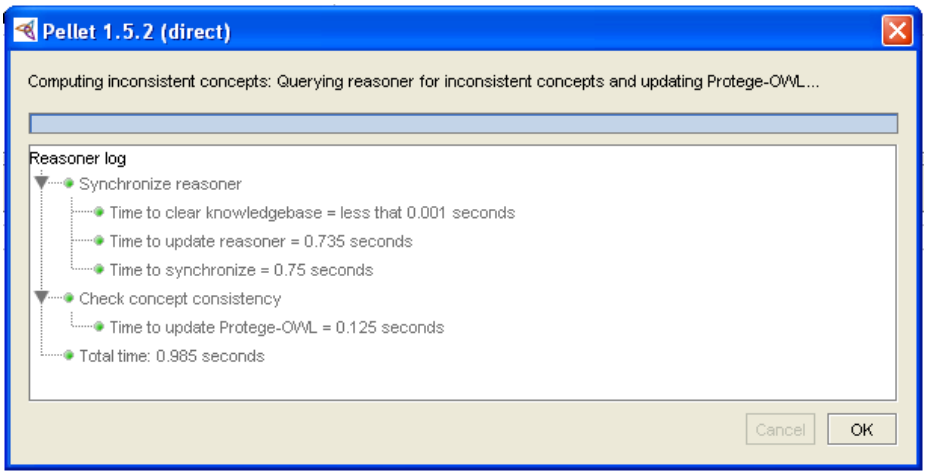

Figura 70. Comprobación de Coherencia

El primer punto de comparación es: Comprobar la Coherencia de la ontología, para esto el proceso realiza algunas actividades que le ayudan a determinar el tiempo total del chequeo.

I. Sincronización del Razonador **:**

Estas son los pasos que realiza en esta fase.

1. Tiempo de limpiar la base de conocimiento
2. Tiempo para actualizar el razonador
3. Tiempo para sincronización

II. Chequeo de la consistencia de conceptos

1. Tiempo de actualización

III. Tiempo Total

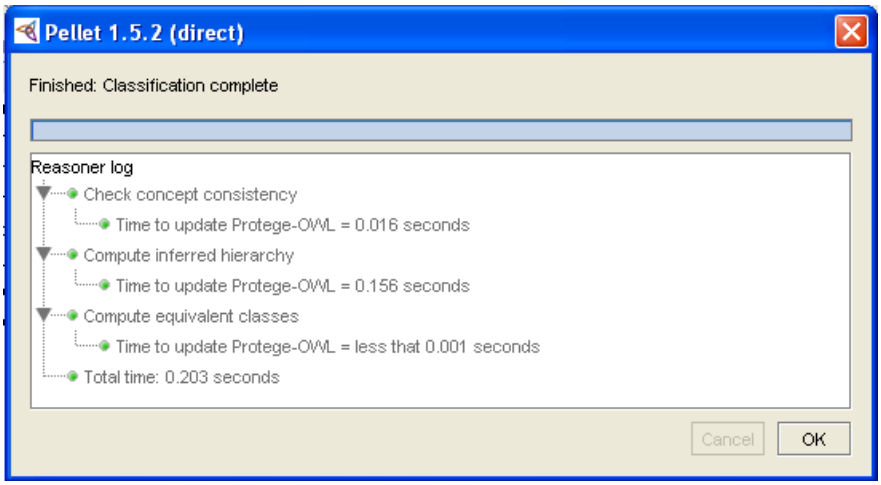

Figura 71. Clasificación de Taxonomías

Clasificación de Taxonomías

I. **Sincronización del Razonador**

   a) Tiempo para limpiar la Base del Conocimiento
   b) Tiempo de actualización del Razonador
   c) Tiempo para sincronizar

II. **Chequeo de Consistencia de Concepto**

   a) Tiempo de actualización Protege- OWL

III. **Calcular la jerarquía a inferirse**

   a) Tiempo de actualización Protege- OWL

IV. **Calcular las clases equivalentes**

   a) Tiempo de actualización Protege- OWL

V. Tiempo Total

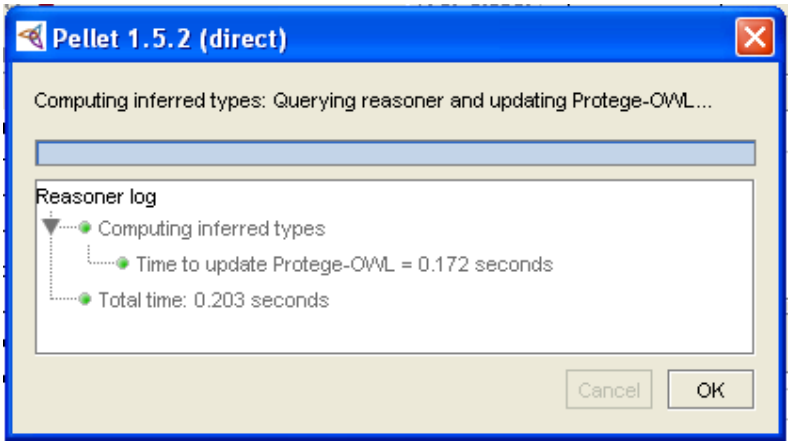

Figura 72. Computo Tipos a Inferirse

Computo de los tipos a inferirse:

I. **Computo de tipos a inferirse**

   a) Tiempo de actualización Protégé – OWL

II. **Tiempo Total**

KAON2 (VERSIÓN 2008/06/29)

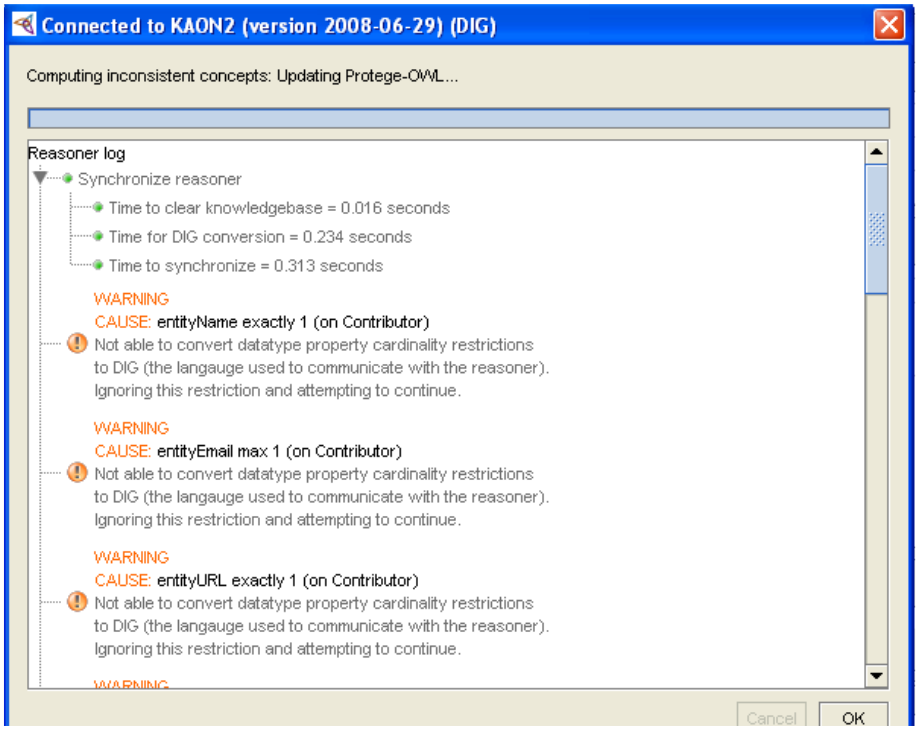

Figura 73. Chequeo de Consistencia

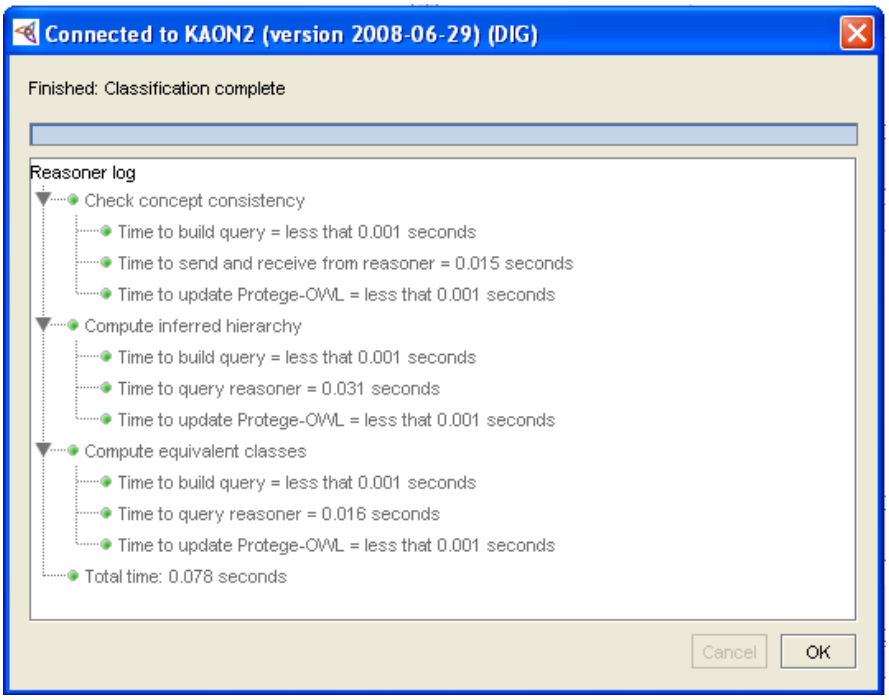

Figura 74. Clasificación de Taxonomías

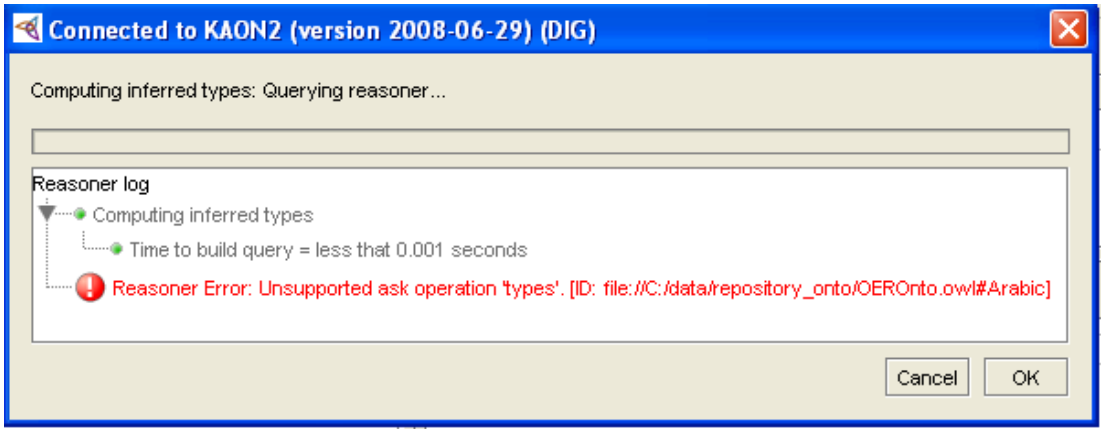

Figura 75. Tipos a Inferirse

FACT ++ 1.3.0

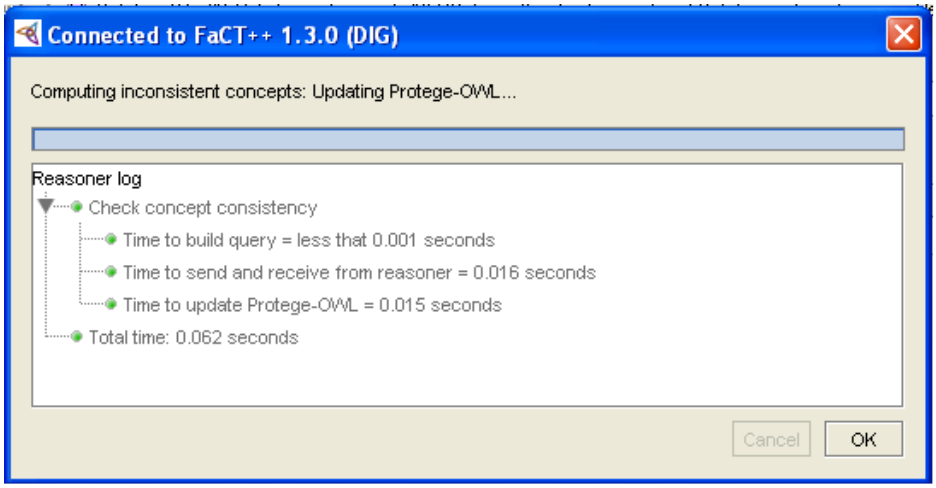

Figura 76. Chequeo de Consistencias

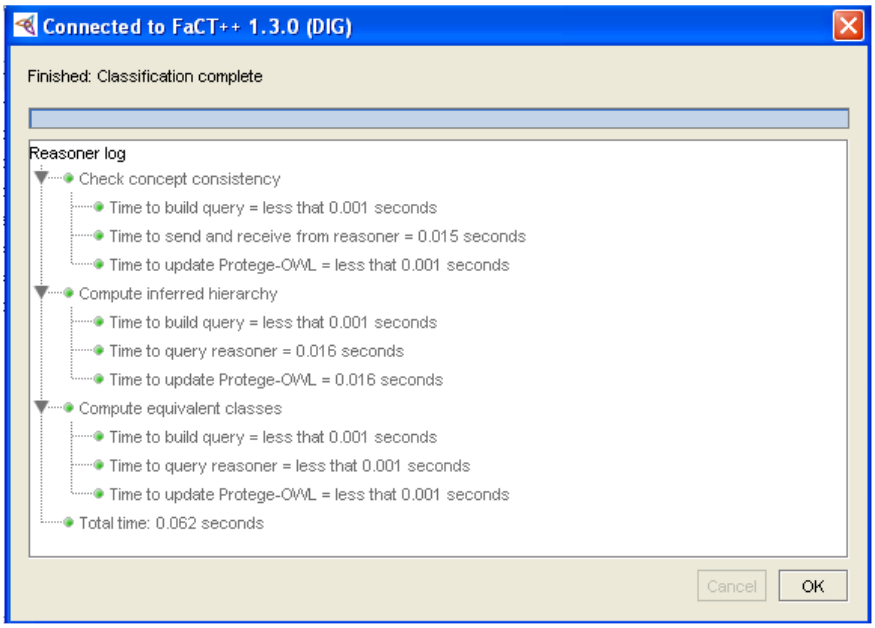

Figura 77. Clasificación de Taxonomías

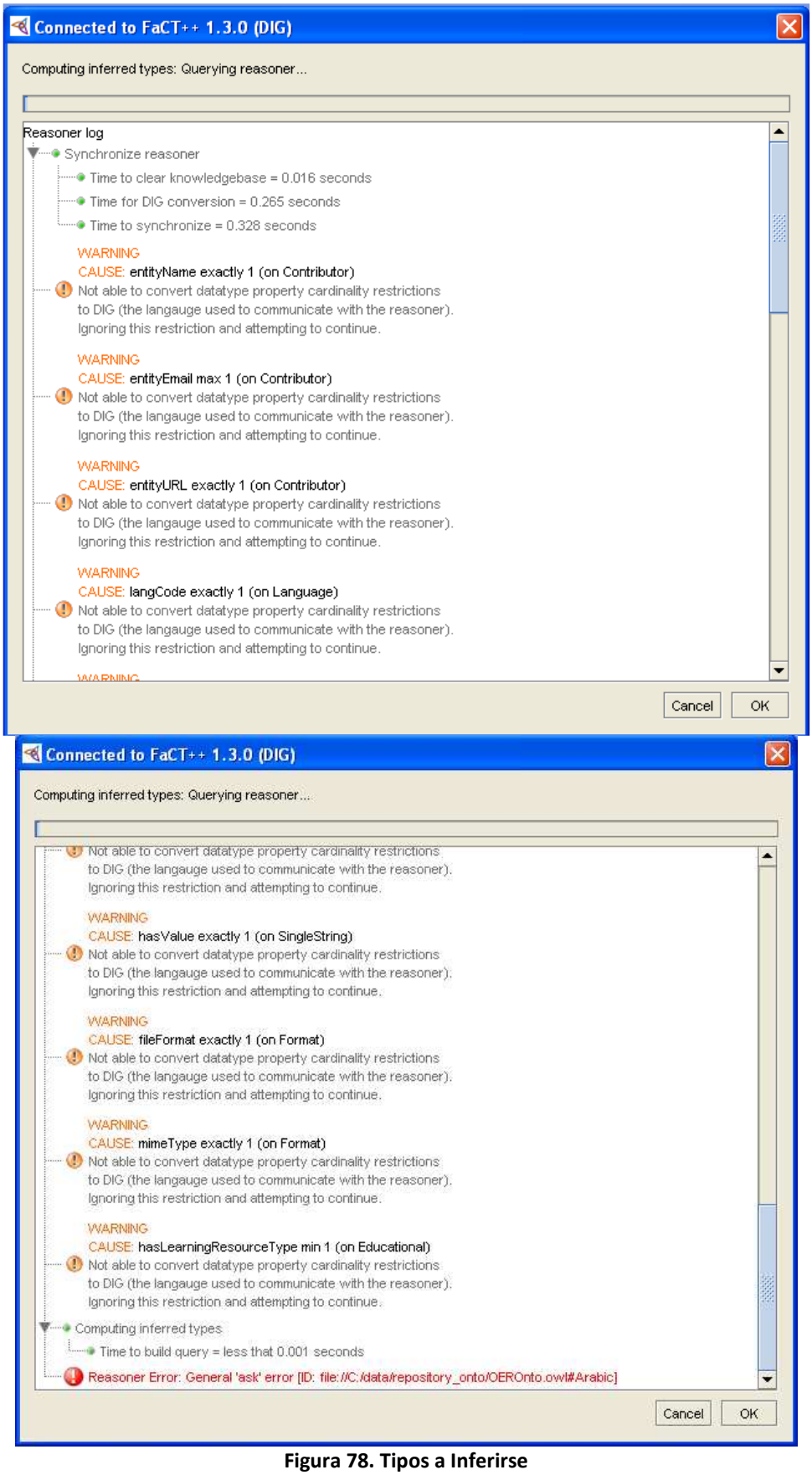

Figura 78. Tipos a Inferirse

SEGUNDO EXPERIMENTO

En esta segunda parte de pruebas la mayoría de los razonadores ya no realizan un análisis previo como lo realizan en el primer experimento sino directamente pasan a realizar el análisis de la consistencia de la ontología.

PELLET 1.5.2

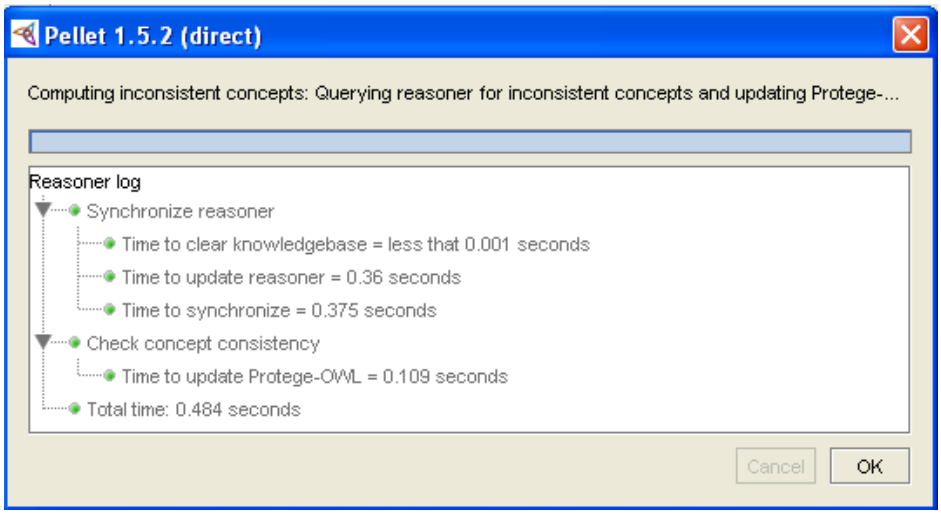

Figura 79. Chequeo de Consistencias

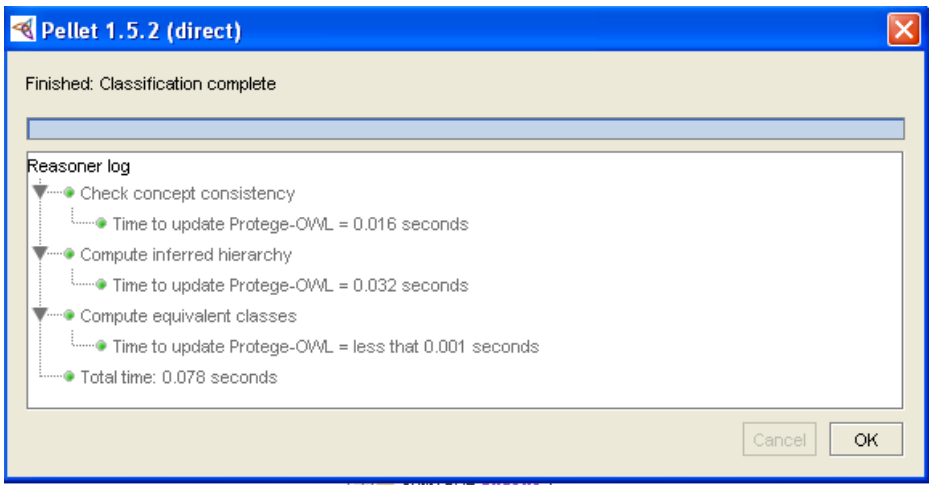

Figura 80. Clasificación de Taxonomías

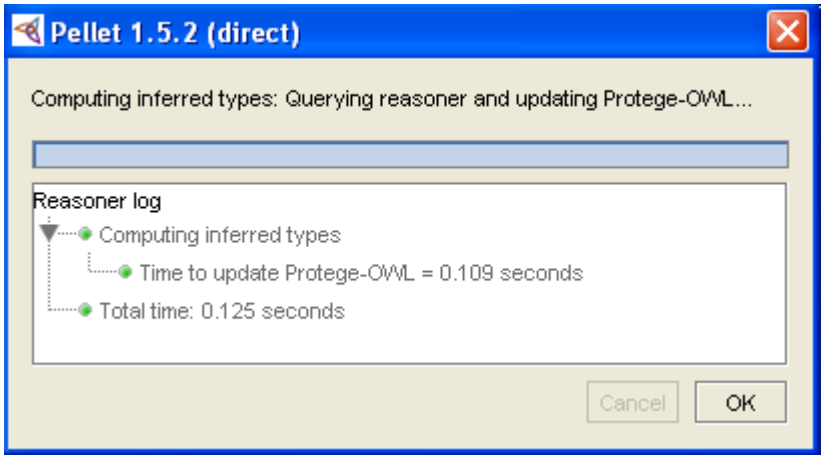

Figura 81. Tipos a Inferirse

RACER PRO 2.0:

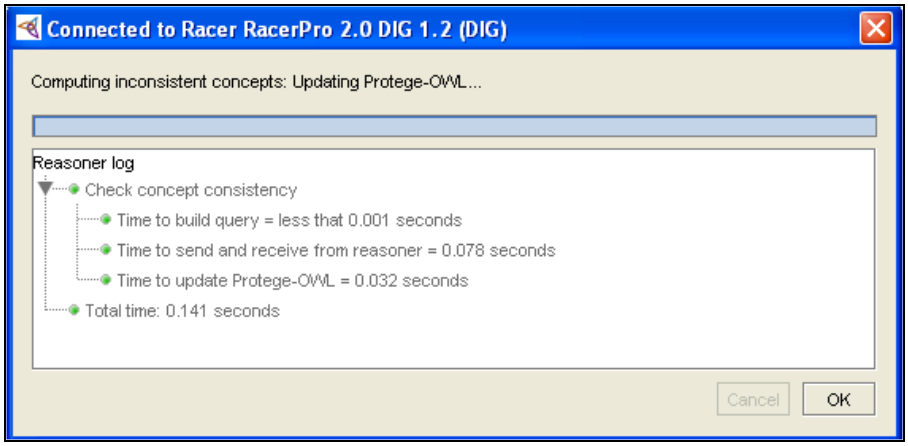

Figura 82. Chequeo de Consistencias

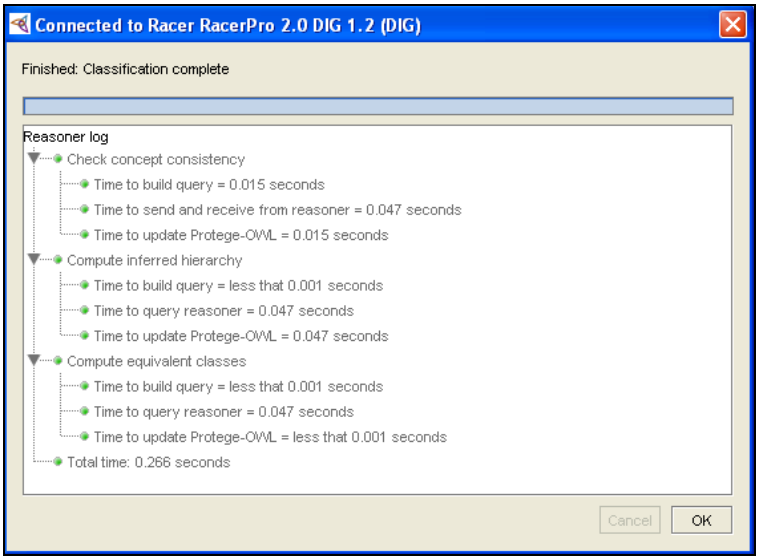

Figura 83. Clasificación de Taxonomías

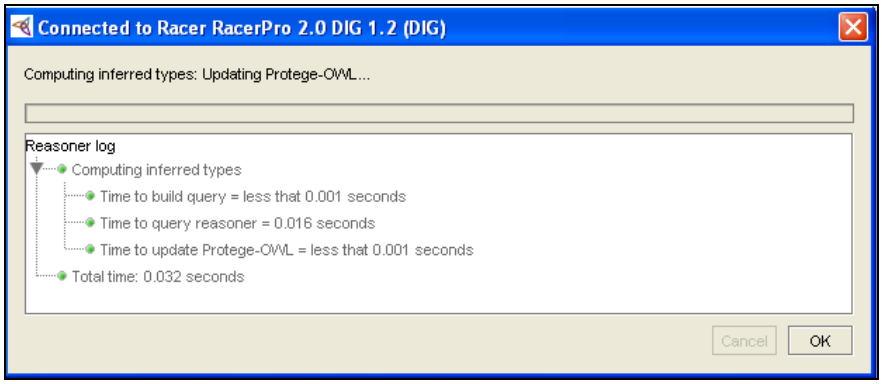

Figura 84. Tipos a Inferirse

KAON2 (VERSIÓN 2008/06/29)

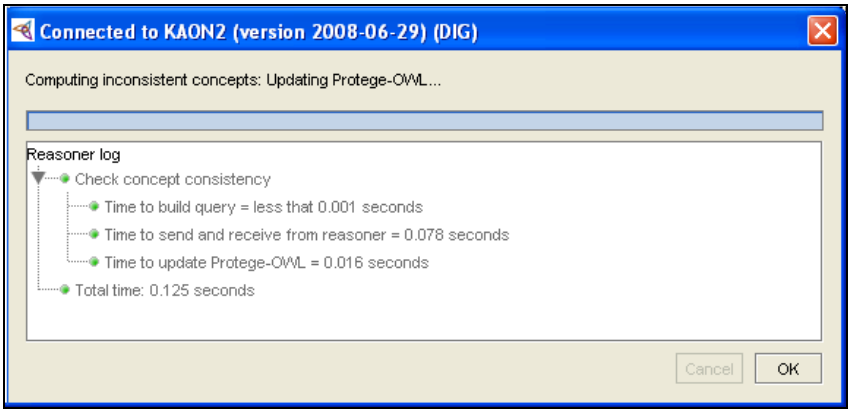

Figura 85. Chequeo de Consistencias

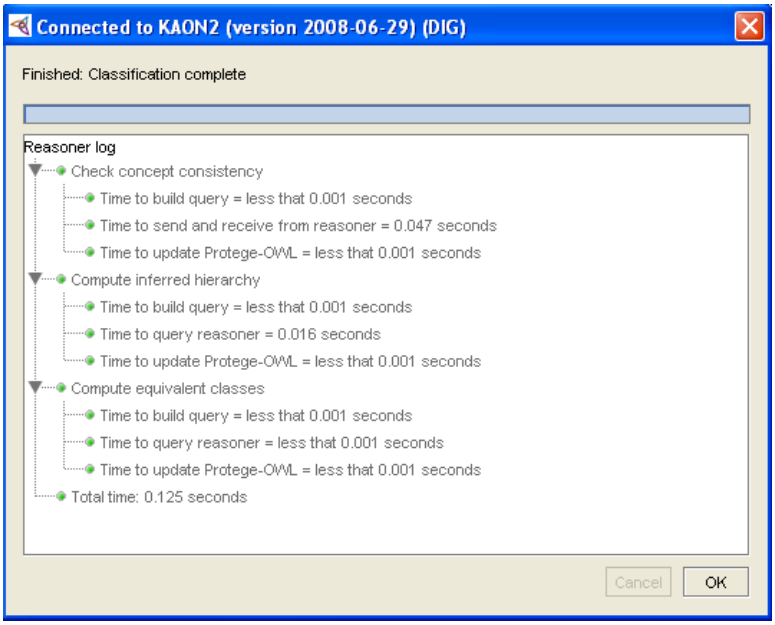

Figura 86. Clasificación de Taxonomías

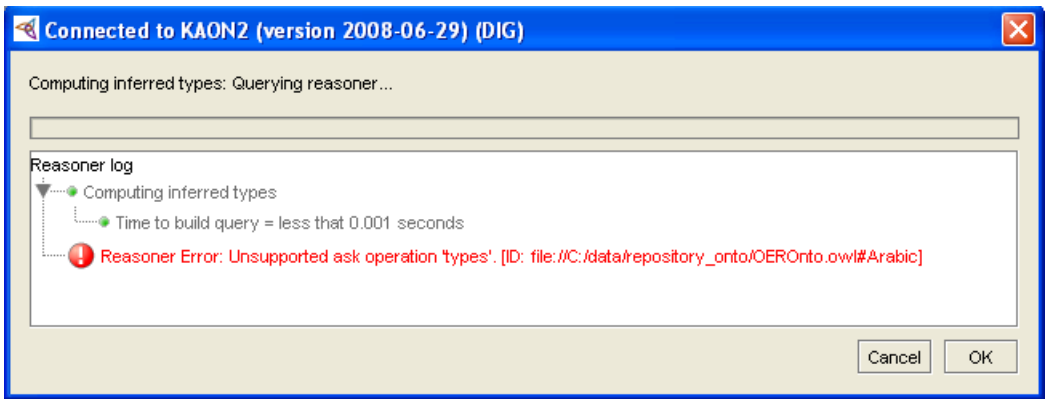

Figura 87. Calculo de Tipos a Inferirse

FACT ++ 1.3.0

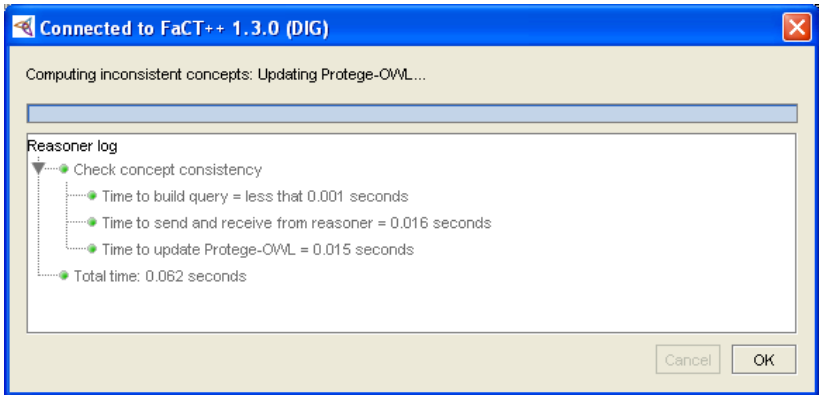

Figura 88. Chequeo de Consistencias

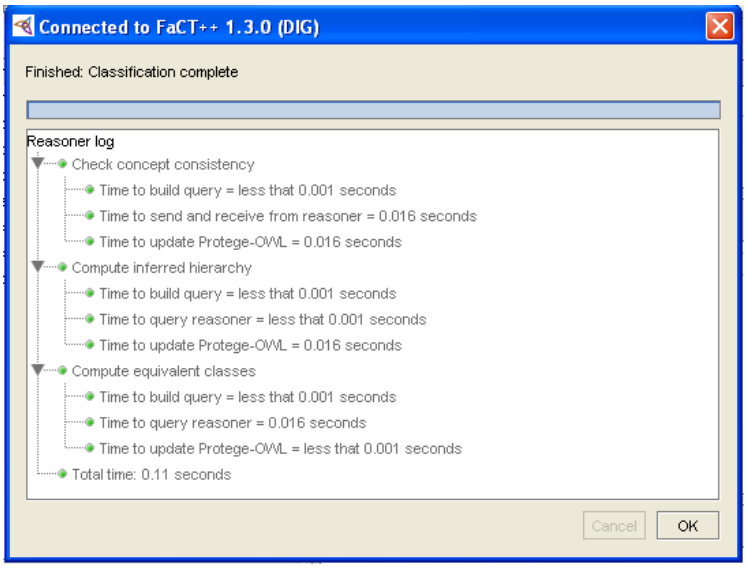

Figura 89. Clasificación de Taxonomías

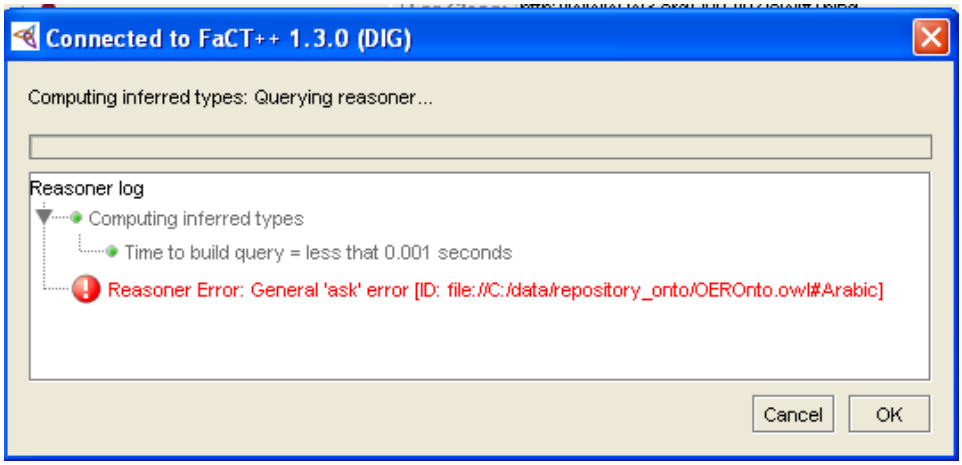

Figura 90. Tipos a Inferirse

TERCER EXPERIMENTO

Para esta tercera parte de las pruebas los razonadores tienen un promedio de tiempo mucho menor que al realizar las primeras experiencias.

PELLET 1.5.2:

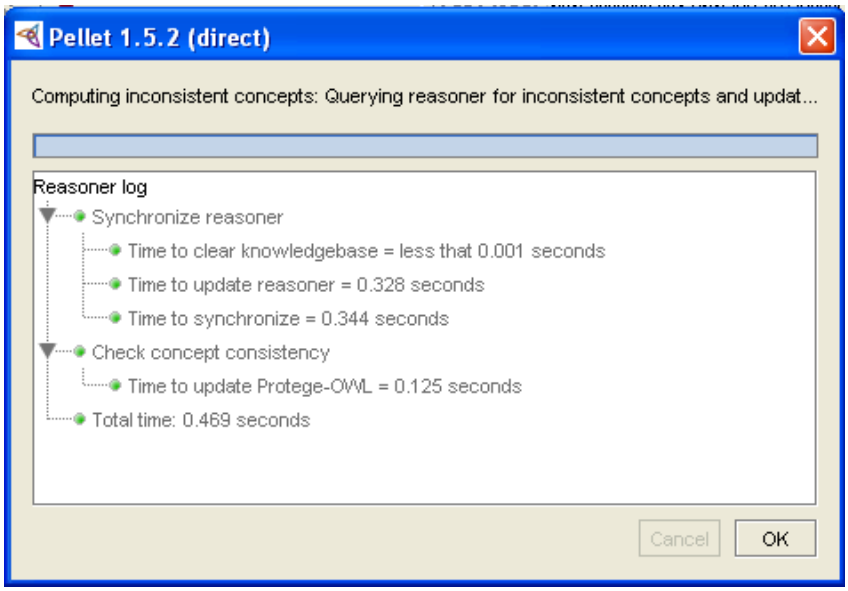

Figura 91. Chequeo de Consistencias

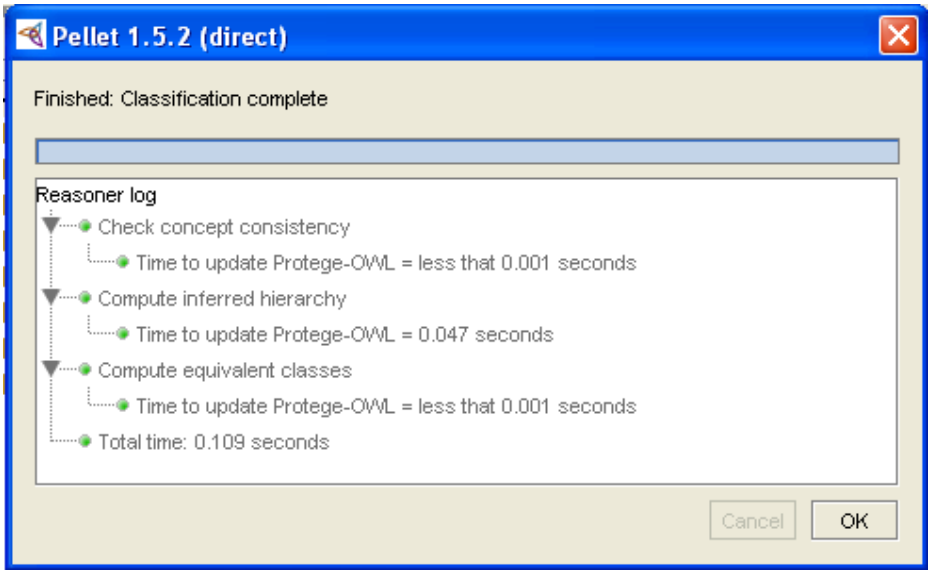

Figura 92. Clasificación de Taxonomías

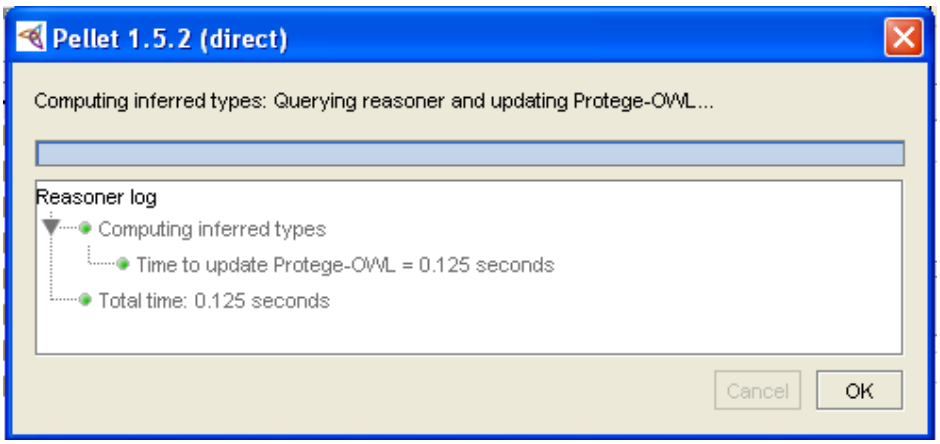

Figura 93. Tipos a Inferirse

RACER PRO 2.0:

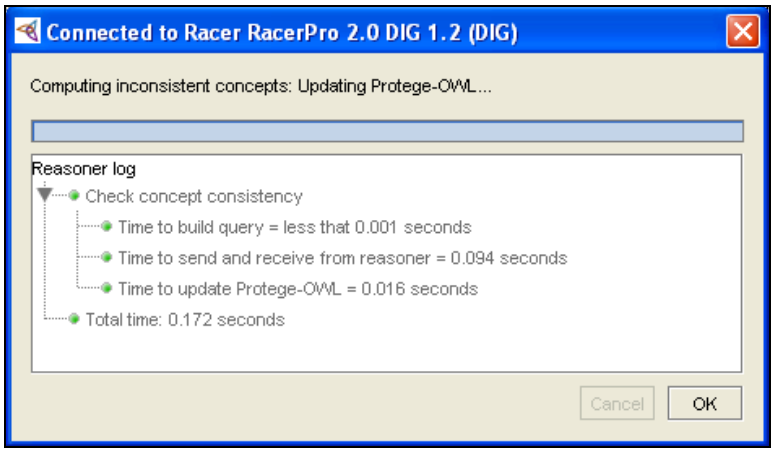

Figura 94. Chequeo de Consistencia

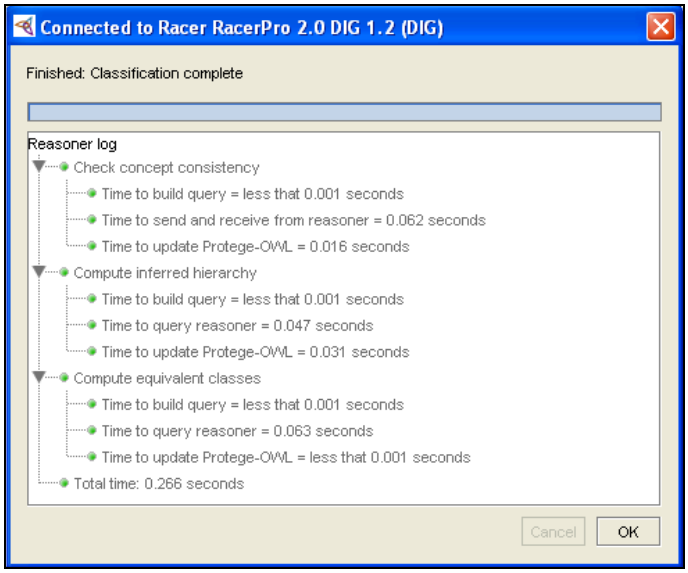

Figura 95. Clasificación de Taxonomías

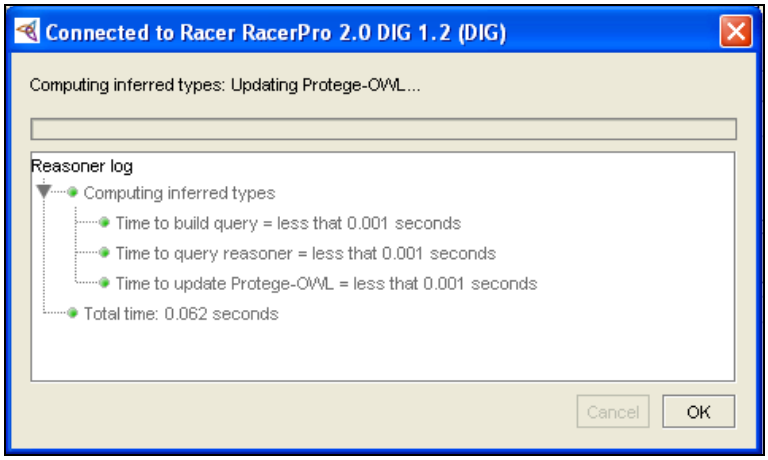

Figura 96. Tipos a Inferirse

KAON2 (VERSIÓN 2008/06/29)

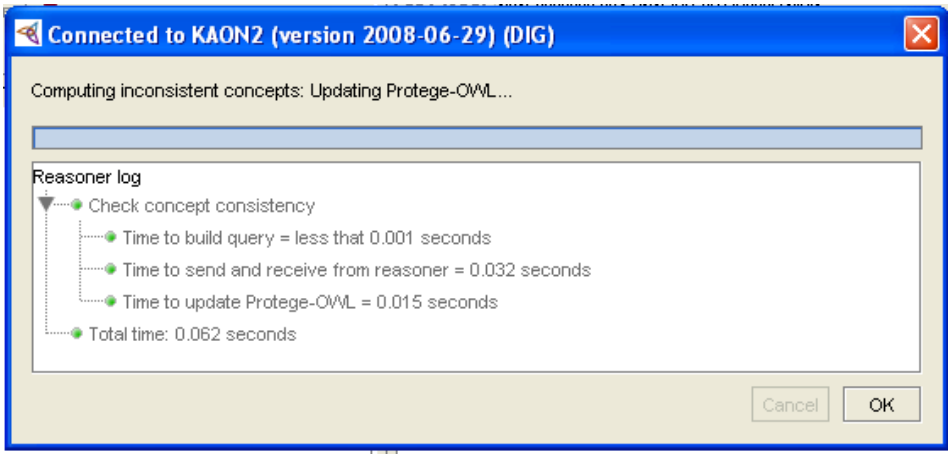

Figura 97. Chequeo de Consistencias

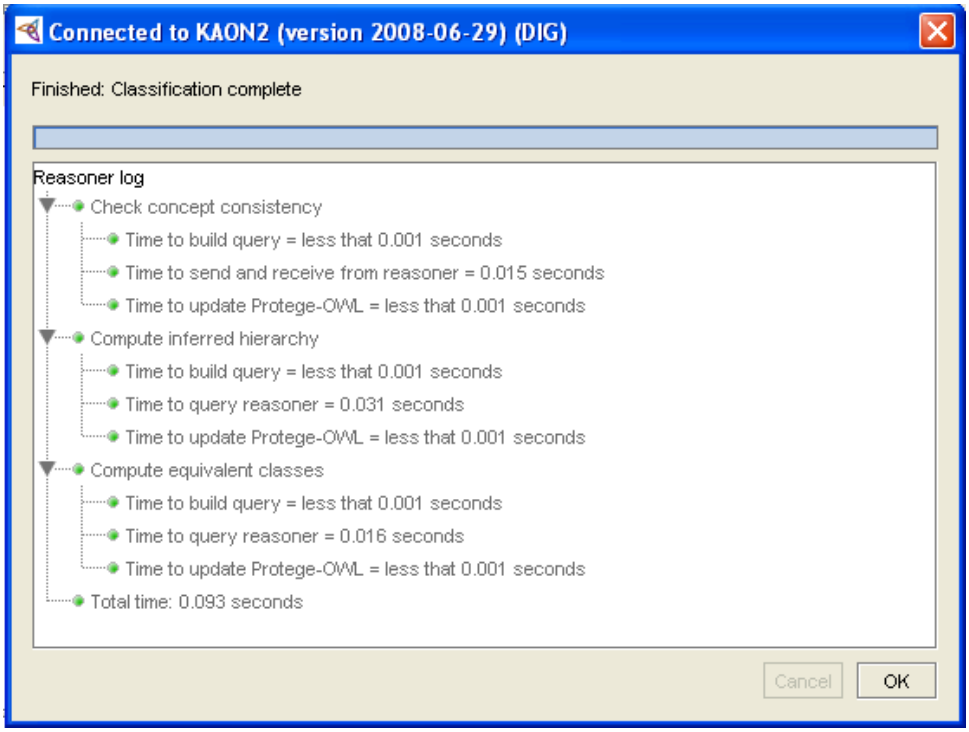

Figura 98. Clasificación de Taxonomías

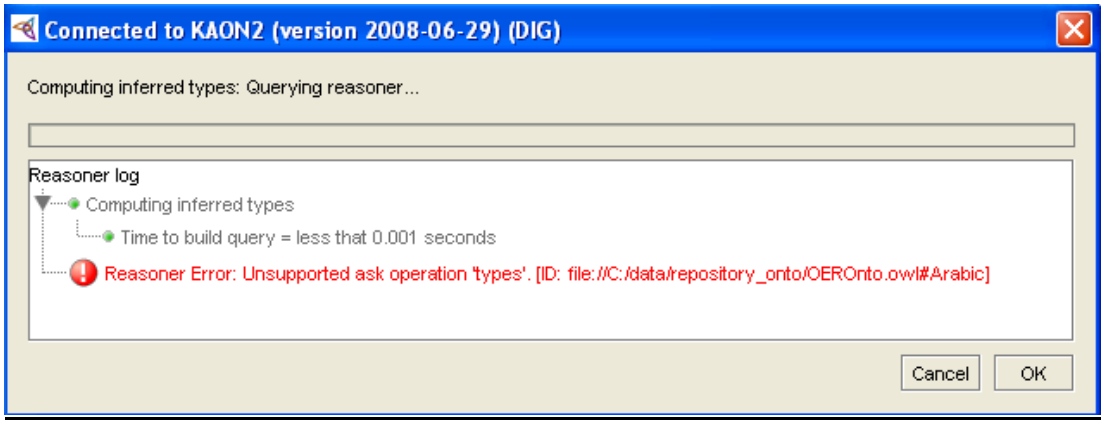

Figura 99. Calcular Tipos a Inferirse

FACT ++ 1.3.0

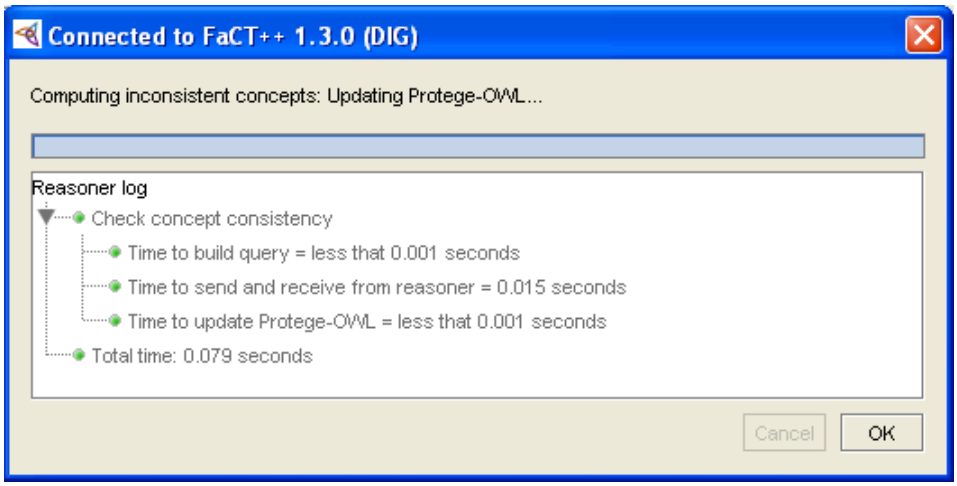

Figura 100. Chequeo de Consistencias

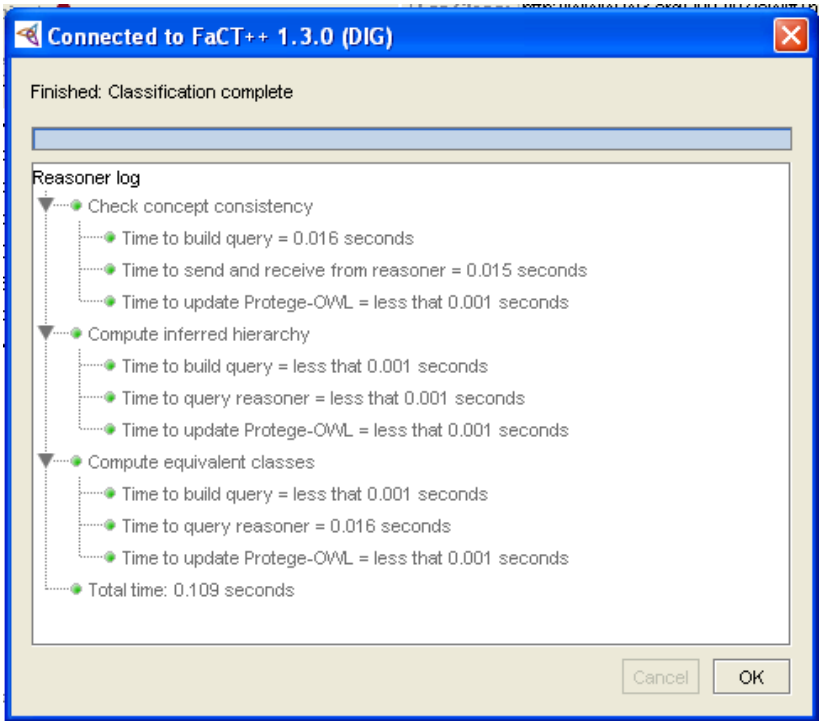

Figura 101. Clasificación de Taxonomías

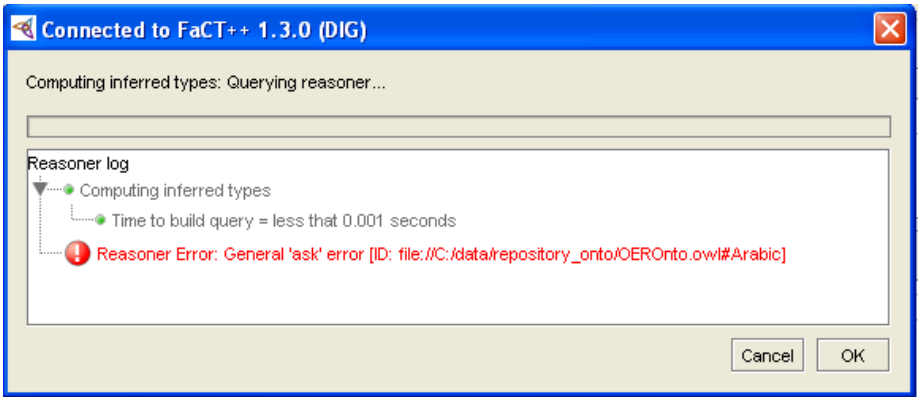

Figura 102. Calcular tipos a Inferir

ANEXO 3

Selección y Análisis de la Ontología

Ian Horrocks el principal impulsor de la investigación con razonadores semánticos junto a otros investigadores como T. Gardiner y D. Tsarkov mencionan la importancia de escoger ejemplos típicos, del mundo real con diferente expresividad; en las que el control manual no es factible como lo sería con ejemplos pequeños. En base a esta recomendación se seleccionó la siguiente ontología, ***Definición de Conceptos de Investigación Académica.***[5]

La ontología seleccionada es, con fines educativos posee una gran cantidad de datos (Clases y subclases) que utiliza el lenguaje OWL; es un estándar dentro de los lenguajes para ontologías y es admitido por la herramienta Protégé, por lo que se requiere de un mayor esfuerzo computacional, permitiendo de esta manera probar el tiempo de ejecución al realizar cada una de las actividades planteadas, que ejecuta el razonador.

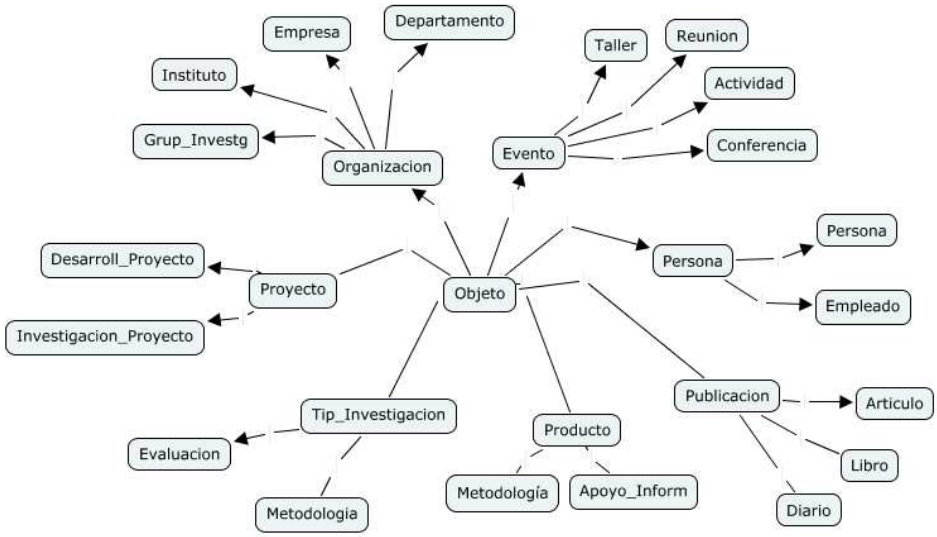

Figura 103. Clases de la Ontología Definición de Conceptos de Investigación Académica

Desarrollo de las fases:

Fase I: Ejecución de las Tareas del razonador

La primera actividad que se realizó previo a las pruebas y desarrollo de ésta fase, es la instalación de cada razonador a Protégé; los aspectos a tomar en cuenta son la dirección y el número de puerto por el que trabajan, como ya se ha mencionado gracias al estándar DIG[6] es posible configurando estos sencillos datos, y para una mejor percepción de todo el

[5] http://web.comlab.ox.ac.uk/people/Ian.Horrocks/

[6] DIG: estándar o protocolo que se introduce para proporcionar una interfaz común para los razonadores DL.

proceso de instalación revisar la parte de Anexos (Anexo 1), donde se encuentra detallado la instalación de cada razonador. En la tabla 29 se muestra los parámetros a tomar en cuenta en la instalación de cada razonador.

Tabla 29. Facilidad de la Instalación de los razonadores

Razonador	Integrado en Protégé	Instalador	Puerto	Observaciones	Facilidad de Instalación /100%
Pellet	x		8080	Razonador integrado con Protégé.	100%
Racer Pro		x	8080	Versión de prueba y Licencia Comercial	80%
Kaon2		x	8080	Configuración de class-path, corrida mediante consola.	60%
FaCT++		x	3490	Número de puerto diferente a los demás razonadores.	90%

El siguiente paso es insertar la ontología a Protégé donde se realizaran tres actividades: la *comprobación de la coherencia,* de la ontología basada en la descripción de una clase, el razonador puede comprobar si es posible que se tenga instancias de la clase; *la clasificación de taxonomías* donde permite observar los resultados en cuanto a las inferencias generadas por el modelo que describe el dominio del problema y *el cómputo de tipos a inferir* que permite computar los tipos de objetos dentro de la ontología.

La metodología que se aplicó se basó en (Horrocks, Tsarkov, & Gardiner, 2006) donde se concluyó que para evaluar el rendimiento no se puede repetir la misma prueba sobre el mismo razonador sin haberlo reseteado antes, si se hace de diferente manera el tiempo de cálculo precisa el resultado.

PRIMERA EJECUCIÓN CON LOS RAZONADORES

Cuando se habla de razonamiento de Protege-OWL API, esto significa obtener una instancia de ProtegeReasoner (Comunicación directa con DIG que siempre este *sincronizado* con el modelo interno Protege OWL).

Esta instancia es utilizada para tener información acerca de la deducción del modelo, como inferir superclases, deducir clases equivalentes y deducir tipos de individuos (APIProtégé, 2009)

Este proceso que se realiza a través de ProtegeReasoner, permite consultar al razonador y obtener información e insertarla en el Modelo OWL. Esto significa que el modelo puede ser examinado “en línea” desde el razonador.

Un ejemplo de esto es la Clasificación de Taxonomías, donde se consulta al razonador de la coherencia, deduce superclases y clases equivalentes de todas las clases de la ontología y luego introduce la información en el modelo Protégé-OWL.

Lo mismo pasa con el Computo de clases a inferirse, método que consulta al razonador para los tipos a deducirse de todos los individuos en la ontología y actualizar el Modelo OWL.

Así, en la ejecución del primer experimento se observa el proceso que se realiza en cada actividad:

4. **Chequeo de la consistencia**

   d. **Sincronización del Razonador :**
   Proceso que realiza en este paso:

   ✓ Limpieza de la base de conocimiento
   ✓ Actualización del razonador
   ✓ Sincronización

   e. **Chequeo de la consistencia de conceptos**
   ✓ Tiempo de actualización

   f. **Tiempo Total**

Así mismo, para la clasificación de taxonomías existe un proceso previo que realiza el razonador:

5. **Clasificación de Taxonomías**

   a. **Sincronización del Razonador**

   ✓ Limpieza de la base de conocimiento
   ✓ Actualización del razonador
   ✓ Sincronización

   b. **Chequeo de Consistencia de Concepto**

   ✓ Tiempo de actualización Protege- OWL

   c. **Calcular la jerarquía a inferirse**

   ✓ Tiempo de actualización Protege- OWL

   d. **Calcular las clases equivalentes**

   ✓ Tiempo de actualización Protege- OWL

   e. **Tiempo Total**

Y por último en el cómputo de tipos a inferirse este es su proceso:

6. **Computo de los tipos a inferirse:**

   a. **Computo de tipos a inferirse**

   ✓ Tiempo de actualización Protégé – OWL

   b. **Tiempo Total**

El orden de ejecución de las pruebas del razonador es: Pellet, Racer Pro, Kaon2, FaCT++. Una vez que se ha desarrollado el primer experimento con las tres pruebas se muestra los datos obtenidos:

Tabla 30. Primer Experimento con los razonadores

PRIMER EXPERIMENTO										
RAZONADORES	Chequeo de Consistencia			Clasificación de Taxonomías					Tipos a Inferirse	
	Sincronización del Razonador	*Chequeo de Consistencia de Conceptos*	*Tiempo Total*	*Sincronización del Razonador*	*Chequeo de Concepto de Consistencia*	*Calcular la Jerarquía a inferirse*	*Calcular clases equivalentes*	*Tiempo Total*	*Tipos a Inferirse*	*Tiempo Total*
Pellet 1.5.2	0.752	0.125	0.563	0.345	0.047	0.093	0.001	0.328	0.001	0.016
Racer Pro 2.0	1.984	0.219	1.281		0.079	0.157	0.064	0.343	0.003	0.047
Kaon2 (Versión 2008/06/29)	1.048	0.546	1.468		0.079	0.547	0.063	0.719		
FaCT++ 1.3.0		0.063	0.125		0.003	0.032	0.017	0.14	0.017	0.047

La actividad de sincronización que permite la comunicación con el modelo OWL; en la tarea de chequeo de la consistencia, en el primer experimento lo realizan tres razonadores Pellet1.5.2, Racer Pro2.0 y Kaon2. Para el razonador FaCT++ 1.3 no realiza esta tarea como se manifiesta en la Tabla 30.

En cuanto a la actividad de la clasificación de taxonomías sólo el razonador Pellet realiza una vez más ésta sincronización, los demás razonadores ya no la ejecutan. Otro aspecto que se destaca en esta primera prueba es que el razonador Kaon2 en su tiempo de respuesta a la tarea de Cómputo de Tipos a inferirse no brinda un tiempo exacto, esto puede deberse a dos aspectos que tienen que ver con el tiempo o con la capacidad del equipo.

Cabe destacar que cada razonador actúa de una diferente manera, así obteniendo diferentes tiempos de respuesta a pesar de compartir algunas características cada uno es independiente.

SEGUNDA EJECUCIÓN CON LOS RAZONADORES

Para el segundo experimento, en las tres actividades se presentan los siguientes datos ilustrados en la tabla 31.

Tabla 31. Segundo Experimento sobre la Ontología

SEGUNDO EXPERIMENTO										
RAZONADORES	Chequeo de Consistencia			Clasificación de Taxonomías					Tipos a Inferirse	
	Sincronización del Razonador	*Chequeo de Consistencia de Conceptos*	*Tiempo Total*	*Sincronización del Razonador*	*Chequeo de Concepto de Consistencia*	*Calcular la Jerarquía a inferirse*	*Calcular clases equivalentes*	*Tiempo Total*	*Tipos a Inferirse*	*Tiempo Total*
Pellet 1.5.2	0.767	0.125	0.531		0.016	0.093	0.001	0.141	0.001	0.001
Racer Pro 2.0		0.111	0.141		0.077	0.095	0.049	0.266	0.018	0.032
Kaon2 (Versión 2008/06/29)		0.095	0.125		0.048	0.064	0.033	0.156		
FaCT ++		0.142	0.172		0.032	0.032	0.018	0.125	0.049	0.078

Para esta parte se observa que solo el razonador Pellet es quien realiza la actividad de sincronización del razonador en la tarea de chequeo de consistencia, de la misma forma no hay respuesta de la actividad de computo de tipos a inferirse.

Por lo demás cada razonador brinda sus datos respectivos, y así contribuyen para la evaluación.

TERCERA EJECUCIÓN CON LOS RAZONADORES

En este último experimento no hay variaciones en aspectos nuevos a recalcar, claramente sus tiempos son diferentes pero no hay un aspecto relevante que destacar como se muestra en la tabla 32.

Tabla 32. Tercer Experimento sobre la Ontología

TERCER EXPERIMENTO										
RAZONADORES	Chequeo de Consistencia			Clasificación de Taxonomías					Tipos a Inferirse	
	Sincronización del Razonador	*Chequeo de Consistencia de Conceptos*	*Tiempo Total*	*Sincronización del Razonador*	*Chequeo de Concepto de Consistencia*	*Calcular la Jerarquía a inferirse*	*Calcular clases equivalentes*	*Tiempo Total*	*Tipos a Inferirse*	*Tiempo Total*
Pellet 1.5.2	0.391	0.125	0.328		0.016	0.093	0.001	0.172	0.001	0.015
Racer Pro 2.0		0.111	0.172		0.079	0.079	0.065	0.266	0.003	0.062
Kaon2 (Versión 2008/06/29)		0.11	0.11		0.094	0.47	0.048	0.61		
FaCT ++		0.095	0.109		0.063	0.064	0.017	0.172	0.049	0.078

Para un mejor detalle, las capturas de imágenes del proceso que se realizó y las respuestas que dio la herramienta a cada prueba se encuentran en la parte de Anexos (Anexo2).

FASE I: INTERPRETACIÓN DE RESULTADOS

Una vez que se ha obtenido los datos respectivos de los experimento, se procede a extraer los tiempos totales de cada tarea para la evaluación en tablas resumen de cada una de las tareas. Con el objetivo de evaluar el rendimiento de los razonadores en sus tareas.

En la primera tarea de **Chequeo de la Consistencia** se presenta los datos en la Tabla 22. Resumen de la Tarea de Chequeo de Inconsistencias, las tres experiencias bajo la misma tarea.

Tabla 33. Resumen de la Tarea de Chequeo de Consistencia

CHEQUEO DE CONSISTENCIA				
Razonadores	**1er Exp. (Segundos)**	**2do Exp. (Segundos)**	**3er Exp (Segundos)**	**Promedio**
Pellet 1.5.2	0,563	0,531	0,328	0,474
Racer Pro 2.0	1,281	0,141	0,172	0,531
Kaon2	1,468	0,125	0,11	0,568
FaCT++ 1.3	0,125	0,172	0,109	0,135

Proceso para obtención de la Eficiencia:

Tomamos como referencia un tiempo mayor de 0,6

Trendimiento = 0,6 - Promedio

Pellet → 0,6 – 0,568 = 0,032
Racer Pro → 0,6 – 0,531 = 0,069
Kaon2 → 0,6 – 0,474 = 0,126
FaCT++ → 0,6 – 0,135 = 0,465

T_Rendimiento → 100%

T_Rendimiento = (Trendimiento *100%)/0,6

T_R_Pellet → = (0,032 *100%)/0,6 = 5,3%
T_R_Racer Pro → = (0,069 *100%)/0,6 = 11%
T_R_Kaon2 → = (0,126 *100%)/0,6 = 21%
T_R_FaCT++→ = (0,465 *100%)/0,6 = 78%

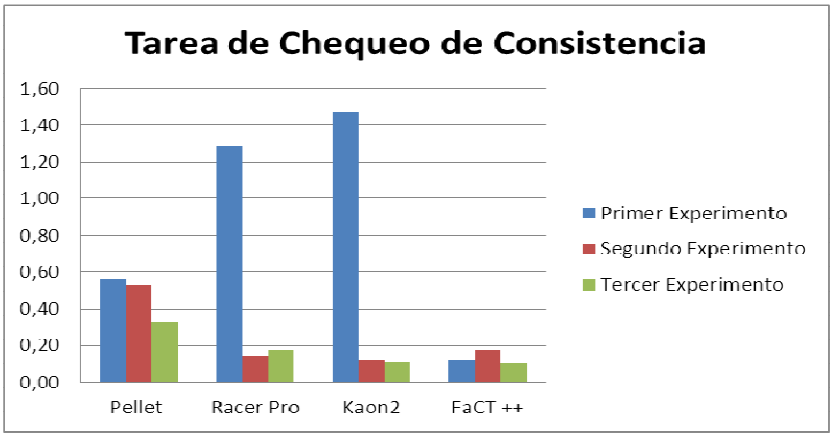

Figura 104. Tasa de Rendimiento del Chequeo de Consistencia

De esta manera se ha podido identificar la mejor propuesta de los razonadores al realizar el chequeo de la consistencia basados en el tiempo de ejecución, en este caso obteniendo un 78% de eficiencia FaCT++, se lo considera el mejor al realizar esta tarea. El segundo mejor es Pellet con el 21%.

En la segunda tarea de la clasificación de Taxonomías en la tabla 35Tabla 23, se encuentran los datos con los que se va a trabajar.

Tabla 34. Resumen de la Tarea de Clasificación de Taxonomías

CLASIFICACION DE TAXONOMIAS				
Razonadores	**1er Exp. (segundos)**	**2do Exp. (segundos)**	**3er Exp. (segundos)**	**Promedio**
Pellet 1.5.2	0,328	0,141	0,172	0,214
Racer Pro 2.0	0,343	0,266	0,266	0,292
Kaon2	0,719	0,156	0,61	0,495
FaCT++ 1.3	0,14	0,125	0,172	0,146

Proceso para obtención de la Eficiencia:

Tomamos como referencia un tiempo mayor de 0,5

Trendimiento = 0,5 - Promedio

Pellet → 0,5 – 0,214 = 0,286
Racer Pro → 0,5 – 0,292 = 0,208
Kaon2 → 0,5 – 0,495 = 0,005
FaCT++ → 0,5 – 0,146 = 0,35

T_Rendimiento → 100%

T_Rendimiento = (Trendimiento *100%)/0,6

T_R_Pellet → = (0,286 *100%)/0,6 = 57,20%
T_R_Racer Pro → = (0,208 *100%)/0,6 = 41,60%
T_R_Kaon2 → = (0,005 *100%)/0,6 = 1%
T_R_FaCT++ → = (0,354 *100%)/0,6 = 70,8%

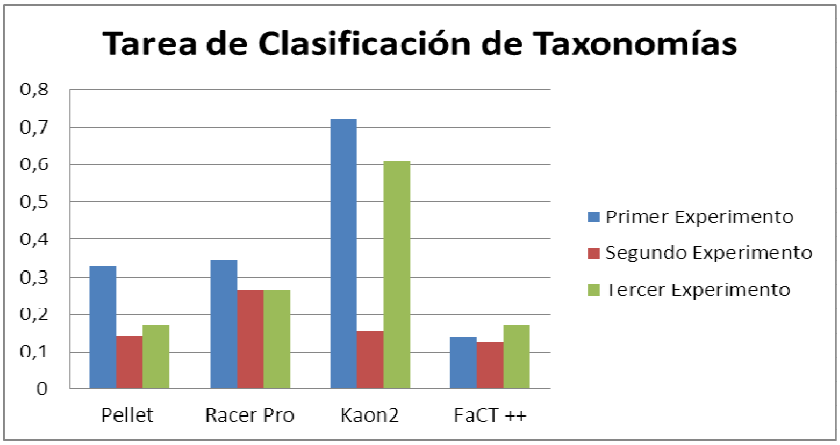

Figura 105. Tarea de la Clasificación de Taxonomías

La **clasificación de las taxonomías** es una de las principales tareas del razonador que como ya se ha mencionado permite observar los resultados en cuanto a las inferencias generadas por el modelo que describe el dominio del problema.

En la Figura 106, ilustra como el razonador FaCT++ optimiza su tiempo representando un 71% de eficiencia mejor que los demás razonadores, siguiéndole Pellet con el 57% de rendimiento.

En la última tarea de **Cómputo de tipos a inferirse** los valores se encuentran en la Tabla 36 de la tarea de cómputo de tipos a inferirse, se parte para obtener el rendimiento de esta tarea.

Tabla 35. Resumen de la Tarea de Cómputo de Tipos a Inferirse

COMPUTO DE TIPOS A INFERIRSE				
Razonadores	**1er Exp. (segundos)**	**2do Exp. (segundos)**	**3er Exp. (segundos)**	**Promedio**
Pellet 1.5.2	0,016	0,001	0,015	0,011
Racer Pro 2.0	0,047	0,032	0,062	0,052
Kaon2				
FaCT++ 1.3	0,047	0,078	0,078	0,068

Proceso para obtención de la Eficiencia:

Tomamos como referencia un tiempo mayor de 0,07

Trendimiento = 0,07 - Promedio
Pellet → 0,07 – 0,011 = 0,059
Pacer Pro → 0,07 – 0,052 = 0,018
FaCT++ → 0,07 – 0,068 = 0,002

T_Rendimiento → 100%

T_Rendimiento = (Trendimiento *100%)/0,07
T_R_Pellet →= (0,059 *100%)/0,07 = 84,280%

T_R_Racer Pro →= (0,018 *100%)/0,07 = 25,71%
T_R_FaCT++ →= (0,002 *100%)/0,07 = 2,85%

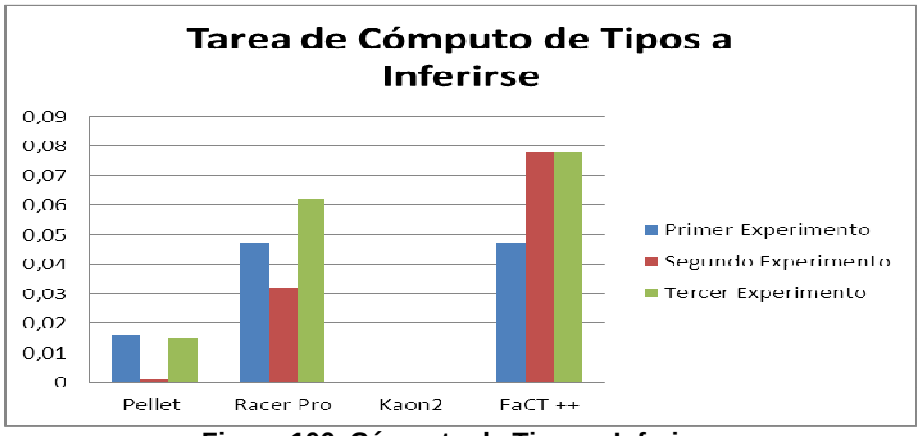

Figura 106. Cómputo de Tipos a Inferirse

En la figura 107, se muestra los resultados obtenidos para la prueba de cómputo de tipos a inferirse (método que consulta al razonador los tipos a deducirse de todos los individuos en la ontología), quedando como la mejor opción Pellet con el 84% y el segundo mejor es Racer Pro con 25%.

FASE I: ANÁLISIS DE RESULTADOS

Para la obtención de los resultados de las tareas, se tomó un tiempo máximo de referencia (tiempo máximo que engloba todos los tiempos de las pruebas) para la medición del rendimiento. Para la tasa de rendimiento se utiliza la diferencia del tiempo de referencia (menos de un segundo) con el tiempo de la actividad, se lo multiplicó por el 100% y se lo dividió para el valor tomado de referencia, con estos resultados se refleja la tasa de rendimiento que tiene cada razonador, se reflejó su eficiencia.

En base a estos resultados el "**Razonador FaCT++"**es el más eficiente por obtener los mayores porcentajes en el rendimiento en las dos tareas (chequeo de consistencia y clasificación de taxonomías), siendo estas tareas las más importantes de esta fase. Todo el desarrollo de esta fase ayuda a cumplir el criterio de Eficiencia que se da en cada razonador.

A continuación en la Figura 108, se muestra un resumen de la eficiencia del mejor razonador en cada tarea.

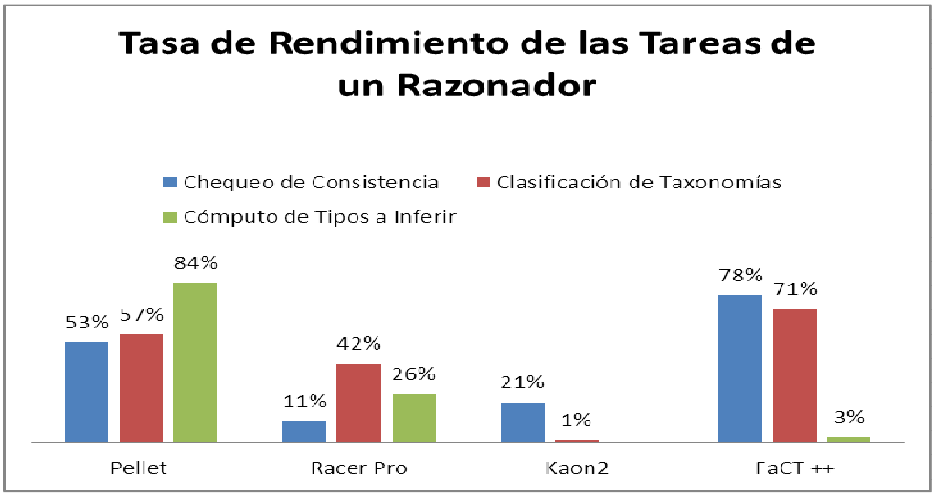

Figura 107. Tasa de Rendimiento de las Tareas de un Razonador

Fase II: Desarrollo de un Cuadro Comparativo de las Características Principales

Como se puede observar en la Tabla 37, se muestra una tabla comparativa, los razonadores se efectúan con algunas características similares tal es el caso de que todos implementan el interfaz DIG, por lo que se puede utilizar dicho interfaz al implementar el razonador con Protégé.

De igual forma se aprecia que en tres de los razonadores se utiliza el mismo algoritmo de razonamiento el Tableaux (Mencionado en el Capítulo 2), a diferencia del KAON2 que utiliza el Datalog. Otra característica importante, es que tienen su propio razonador integrado, es la utilización de una API para poder atacar al razonador directamente a diferencia de FACT ++ 1.3. Se debe destacar que todos los razonadores seleccionados a diferencia de RACER PRO, son de tipo de licencia libre mientras que RACER en una licencia de propietario.

Una característica en la que los cuatro razonadores son compatibles, es que con protégé y la utilización de una BD Relacional, me permite extraer los datos y poder utilizarlos

Tabla 36. Características de los Razonadores en Estudio

CARACTERISTICAS	RAZONADORES			
	KAON2	PELLET 1.5.2	FACT ++ 1.3	RACER PRO
Expresividad de razonamiento	Razonamiento SHIQ	SROIQ(D)	SHF y SHIQ	--
Algoritmo de Razonamiento	Datalog	Tableaux	Tableaux	Tableaux
Lenguaje de Consulta	SPARQL	SPARQL	--	nRQL
Permite extraer datos de BD Relacional	SI	SI	SI	SI
Lenguaje	Comercial	Java	C++	Comercial
Soporte DIG	SI	SI	SI	SI
Tipo de Licencia	Libre	Libre	Libre	No libre

Soporte de Reglas	SWRL-DL	SWRL-DL	NO	SWL
Disponibilidad del API	SI	SI	NO	SI
Multiplataforma	SI	SI	NO	NO

ANÁLISIS DE RESULTADOS

Una vez recopilada toda la información de las características, ventajas y desventajas de cada razonador se estructuro una tabla con los aspectos más relevantes que permitan realizan una valoración de los razonadores, en algunos aspectos:

- El algoritmo Tableaux, que utilizan algunos de los razonadores seleccionados, es considerado como un estándar para los razonadores semánticos.
- Datalog es una extensión de Prolog, usado por el razonador Kaon2 con nuevos algoritmos de razonamiento que permiten una optimización en sus tareas.
- La licencia con la que cuenta cada razonador es importante en una investigación y mucho más para el desarrollo de un proyecto aplicado a la vida real.
- El lenguaje de consultas que tiene cada razonador es importante para poder extraer los datos de la ontología.
- El que se pueda utilizar en diferentes plataformas es una ventaja en la utilización de un determinado razonador.
- La obtención del api del razonador, permite atacar al razonador directamente y así brindar un aporte a la investigación y desarrollo de mejoras para el razonador.

Todas las características son importantes a la hora de escoger el mejor razonador para una ontología; desde el punto de vista teórico.

En base al criterio de la disponibilidad de la información, vemos que FaCT++, a pesar de ser open-source, no cuenta con mucha información de ayuda, ni comunidad grande de soporte, además de ofrecer poca información al usuario en caso de error (aunque un poco más de la que ofrece Racer en su interfaz DIG).

Sólo Pellet cuenta con el apoyo de una comunidad de desarrolladores y beta testers que permite acelerar la madurez del producto, siendo este razonador el que cumple de mejor manera éste criterio.

Fase III: Capacidad de Respuesta a una Consulta

En esta tercera fase se realizó uno de los factores más importantes para promover el uso de ontologías, el desarrollo de razonadores que permitan realizar consultas en las ontologías.

Es primordial tener sistemas que garanticen la ejecución de las consultas y que razonen sobre la ontología de manera eficiente, de tal manera que es posible explotar el poder de la semántica. Las consultas a las ontologías no solo implican realizar una consulta interna de está, sino también realizar una consulta a un conjunto de ontologías para la búsqueda de un concepto determinado.

SELECCIÓN DE LA ONTOLOGÍA Y CONSULTA

La ontología ejemplo que se escogió es: ***La tabla periódica***[7]. La ontología seleccionada proporciona datos de referencia sobre los elementos químicos que tiene la tabla periódica para apoyar aplicaciones de la Web Semántica en la química y disciplinas afines.

La consulta a realizar es la: Petición de los elementos químicos que tenga el nombre, símbolo, número y color; expresada en lenguaje de consulta, queda de la siguiente manera:

```
PREFIX table: <http://www.daml.org/2003/01/periodictable/PeriodicTable#>
SELECT *
FROM <http://www.daml.org/2003/01/periodictable/PeriodicTable.owl>
WHERE
{
?elementtable:name ?name.
?elementtable:symbol ?symbol.
?elementtable:atomicNumber ?number.
 OPTIONAL {?elementtable:color ?color.}
}
ORDER BY ?name
```

DESARROLLO DE LA CONSULTA

Lo que se desea lograr a través de la ejecución de la consulta planteada, es verificar la capacidad de respuesta y validar la eficiencia del razonador al realizar una consulta del tipo TBox.

Al realizar la prueba con el razonador Pellet la respuesta es la misma que el razonador FaCT++ , es decir se obtiene una respuesta afirmativa sobre la consulta lo que permite validar la eficiencia del razonador como se indica en la Figura 109

[7] http://www.daml.org

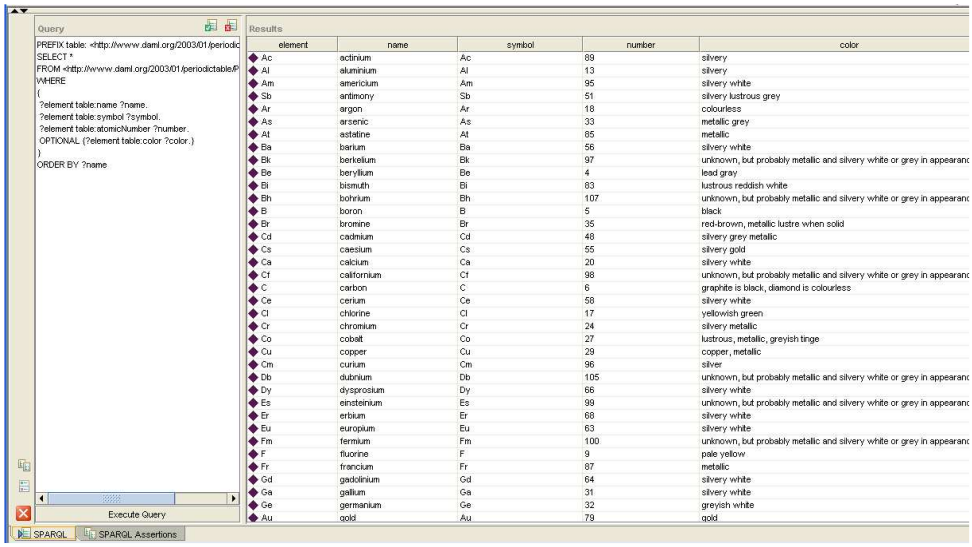

Figura 108. Respuesta de Consulta de Pellet y FaCT++

Al utilizar el razonador Racer Pro y Kaon2 (Figura 109) se puede visualizar la respuesta negativa a la consulta, a pesar de ser una consulta ejemplo y básica que no requiere de un procesamiento extenso.

En el sitio oficial de Kaon2[8] menciona un aspecto relevante y de importancia para esté tema de consultas, afirmando que esté razonador no puede manejar grandes cantidades de estados de cardinalidad, incluso han visto algunos problemas en cardinalidades máximas de dos y concluye diciendo que Kaon2 no es capaz de responder a cualquier consulta, y que sin embargo si hay problemas no sólo depende de la cardinalidad sino de otros axiomas.

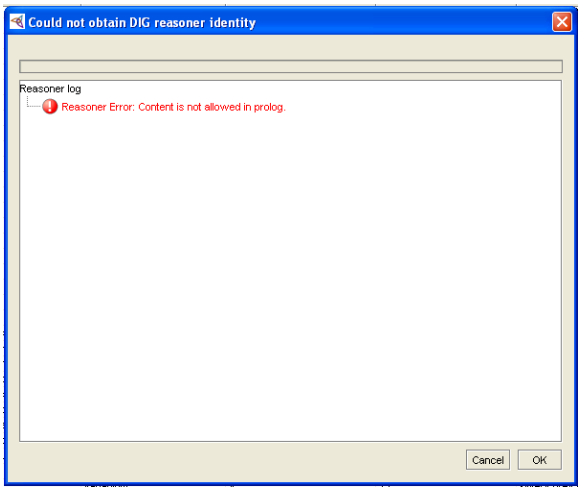

Figura 109. Resultado de la Consulta con Racer Pro y Kaon2

8 http://kaon2.semanticweb.org/#reasoning

ANÁLISIS DE RESULTADOS

Una vez que se ha obtenidos los resultados que se muestran en las Figuras 109 y 110, se distingue la capacidad de respuesta de cada razonador ante la ontología ejemplo.

Cuando se escogió la ontología para la consulta, un aspecto que se tomó en cuenta es, que está sea apta para realizar consultas, que tenga datos ingresados para poder ser gestionada; al obtener los resultados se analiza que no todas las ontologías podrán ser leídas por los razonadores y no específicamente porque estén mal diseñadas o estructuradas, sino que depende de las características que está posea para dejar desarrollar la consulta.

Para fundamentar está hipótesis se ha consultado en algunas Universidades como la Universidad de Maryland, Universidad de Valencia y la Universidad de Manchester; que conjuntamente con algunos de los impulsores como: Ian Horrocks junto a D. Tsarkov y T. Gardiner, han desarrollado pruebas e investigaciones sobre este tema de evaluación de los razonadores ya que es una herramienta fundamental y muy importante a la hora de valorar una ontología; tomando en cuenta que hoy en día la tendencia es la creación de ontologías para aplicarlas a la vida real y el mejoramiento de actividades del diario vivir.(Horrocks,Tsarkov, 2006)

Tabla 37. Resultados de la Capacidad de Respuesta a la consulta establecida

Razonador	**SI**	**NO**
Pellet	X	
Racer Pro		X
Kaon2		X
FaCT++	X	

En la tabla 38. Se muestra un resumen de fase de la consulta en una ontología de tal manera que se pueda identificar la acción de cada razonador.

4.5 Criterio de Eficacia de los Razonadores

Una vez que se han finalizado todas las fases planteadas, se desea realizar una recopilación de todos los resultados obtenidos y en base a estos evaluar el criterio de eficacia. Para ello se ha utilizado una matriz de satisfacción de criterios de evaluación, éstos se muestran en la tabla 39.

Tabla 38. Criterios de Evaluación para medir la eficacia de cada razonador

Criterios de Evaluación	Ponderación
Disponibilidad de la Información	20
Facilidad de Instalación	10
Chequeo de la Consistencia	10
Clasificación de la Taxonomía	20
Cómputo de Tipos a Inferirse	10
Respuesta a Consultas	30
	100%

Establecidos los criterios de evaluación se evalúa el cumplimiento en cada uno de los razonadores, en base a la experiencia y experimentos realizados, los resultados se muestran en la tabla 40.

Tabla 39. Evaluación de Cumplimiento de los Criterios en cada Razonador

Métricas	Pellet	Racer Pro	Kaon2	FaCT++
Disponibilidad de la Información	20	10	5	5
Facilidad de Instalación	10	5	5	10
Chequeo de la Consistencia	8	6	4	10
Clasificación de la Taxonomía	8	6	4	10
Cómputo de Tipos a Inferirse	10	8	4	6
Respuesta a Consultas	10	0	0	10
	66	**35**	**22**	**51**

Aplicando los valores planteados en la tabla 39, se observa que el razonador de mejor rendimiento es "***Pellet",*** con un resultado final de rendimiento del 66% de eficacia mostrada en la Tabla 28.

Realizando un análisis comparativo el FaCT++, quien tiene un rendimiento del 51%; dentro del primer parámetro se encuentra más información para consultar y ayuda es Pellet que del FaCT++, la instalación de ambos razonadores no es compleja por lo tanto en este aspecto ambos sirven y son de fácil instalación.

Si bien en el chequeo de la consistencia y la clasificación de las taxonomías el FaCT++ es más eficiente que el Pellet, en el cómputo de Tipos a inferirse se tiene un mejor rendimiento del Pellet, superior a todos los demás razonadores; con una respuesta en las consultas de igual valor que el FaCT++.

Una vez sumado cada uno de los valores se concluye que él que brinda mejor rendimiento y eficacia es el Pellet, con el 66% ante los demás razonadores.

GLOSARIO DE TÉRMINOS

AJAX: Es una forma de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente, y mantiene comunicación asíncrona con el servidor en segundo plano. De esta manera es posible realizar cambios sobre la misma página sin necesidad de recargarla. Disponible en: http://www.asp.net/

ASP: Es una tecnología de script que corre del lado del servidor y puede ser usado para crear aplicaciones web dinámicas e interactivas. Una página ASP es una página HTML que contiene script que corra alado del servidor que son procesados por un servidor web antes de ser utilizado por el navegador. Disponible en: http://www.asp.net/

CGI: (Interfaz Común Pasarela), es una tecnología de la Word wide web que permite a un cliente (explorador web) solicitar datos de un programa ejecutado en un servidor web. Además especifica un estándar para transferir datos entre el cliente y el programa. Disponible en: http://hoohoo.ncsa.uiuc.edu/cgi/

CSS: (Hojas de Estilo en Cascada), es un lenguaje que describe la presentación de los documentos estructurados en hojas de estilo para diferentes métodos de interpretación, es decir se describe como se mostrara un documento en pantalla, impresora o por voz. Disponible en: http://www.w3c.org/Style/CSS

DAML Y OIL: DAML se desarrolló como una extensión del lenguaje XML y del lenguaje RDF para establecer ontologías sobre la Web. Similar al lenguaje DAML es el lenguaje OIL se trata de una propuesta para una representación basada en la web y las leyes de indiferencia para ontologías. Disponible en:
http://www.daml.org/2001/03/daml+oil-walkthru.html

DATAMINING: Técnicas de "inteligencia artificial" que se apoyan en las matemáticas, estadística y neurología, para efectuar operaciones sobre información y de esta manera descubrir nueva información, tal como patrones y tendencias. Disponible en:
http://www.oracle.com/technology/products/bi/odm/index.html

FLASH: Es una tecnología para crear animaciones gráficas vectoriales independientes del navegador, la animación flash se ve igual en todos los navegadores, solo se necesita tener un plug-in para mostrar animaciones flash. Disponible en: http://www.macromedia.com/

FOAF: Construido utilizando tecnologías web semántica descentralizada, diseñada para permitir la integración de datos a través de una variedad de aplicaciones, sitios web, servicios, y sistemas de software. Disponible en: http://xmlns.com/foaf/0.1/

FRAMEWORK: es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado. Disponible en: http://es.wikipedia.org/wiki/Framework

HTML: (lenguaje de Marcado de Hipertexto), es un lenguaje de programación muy sencillo que se utiliza para crear los textos y las páginas web. Disponible en http://www.w3.org/MarkUp

HTTP: (Protocolo de Transferencia de hipertexto), es el método más común de intercambio en la Word Wide Web, mediante el cual se transfieren las páginas web al ordenador. El protocolo de transferencia es el sistema mediante el cual se transfiere información entre los

servidores y los clientes (ejemplo: navegadores). Disponible en: http://www.w3.org/Protocols/

JAVA: Leguaje Orientado a Objetos, permite crear programas que funcionan en cualquier equipo tipo ordenador y sistema operativo. Se usa Java para crear programas especiales llamados Applets, que se incorporan en páginas web para hacerlas interactivas. Disponible en: http://java.sun.com/

JAVA WEB START: El software permite descargar y ejecutar aplicaciones Java desde la Web. Permite activar aplicaciones con un simple clic; garantizar que se está ejecutando la última versión de la aplicación y eliminar complejos procedimientos de instalación o actualización. Disponible en: http://javasun.com

JAVASCRIPT: Es un lenguaje interpretado orientado a páginas web, con una sintaxis semejante a la del lenguaje java. Se utiliza en páginas web HTML, para realizar tareas y operaciones en el marco de la aplicación cliente. Disponible en: http://www.mozilla.org/js/

JSP: (Página de Servidor Java), es un tipo especial de páginas HTML en la cual se insertan script, se procesan en línea para finalmente desplegar un resultado final al usuario en forma de HTML. Disponible: http://java.sun.com/products/jsp/

KIF: (Formato de intercambio de conocimientos), formato de intercambio de conocimiento (KIF) es un lenguaje diseñado para su uso en el intercambio de conocimientos entre los sistemas informáticos dispares. Disponible en:
http://logic.stanford.edu/kif/dpans.html

OWL: Lenguaje de Ontologías Web, tiene mayor capacidad para expresar significado y semántica que XML, RDF, y RDF-S, va más allá de estos lenguajes en su capacidad para representar contenido interpretable por un ordenador en la Web. Disponible en: http://www.w3.org/2007/09/OWL-Overview-es.html#s1

PHP: (Preprocesador de Hiper-texto), es un lenguaje de script incrustado dentro del HTML. La mayor parte de su sintaxis es tomada de C, Java y Perl con algunas características específicas de sí mismo. La meta es permitir rápidamente a los desarrolladores la generación dinámica de páginas. Disponible en: http://php.apache.org/

RDF: Originalmente diseñada como un modelo de metadatos, pero se ha vuelto un método general para el modelado de información. Permite describir metadatos en sitios web, ofreciendo interoperabilidad entre las aplicaciones que intercambian información en lenguaje máquinas por la web. Disponible en: http://www.w3.org/RDF/

RDF SCHEMA: (Esquema RDF), lenguaje primitivo de ontologías que proporciona los elementos básicos para la descripción de vocabularios. Disponible en:
http://www.w3.org/TR/rdf-schema/#ch_introduction

RMI: (*Invocación Método Remoto de Java)*, es un mecanismo ofrecido por Java para invocar un método de manera remota, provee mecanismos simples para la comunicación de aplicaciones distribuidas basadas exclusivamente en Java. Disponible en: http://java.sun.com/javase/technologies/core/basic/rmi/index.jsp

SHOE: Extensiones de HTML simple Ontología, lenguaje de representación del conocimiento basado en HTML. Posee es un súper-conjunto del HTML que añade los

códigos necesarios para integrar datos arbitrarios en las páginas web semántica. Disponible en: http://www.cs.umd.edu/projects/plus/SHOE/

SQL (Structured Query Language): Lenguaje de consultas a bases de datos, implementado por primera vez por IBM, que se transformó en un estándar existente en todas las bases de datos, debido a su facilidad de uso (es similar al inglés), a la flexibilidad y a la potencia. Disponible en:

SWING: es una biblioteca gráfica para Java. Incluye widgets para interfaz gráfica de usuario tales como cajas de texto, botones, desplegables y tablas. Disponible: http://es.wikipedia.org/wiki/Swing_(Java)

SWRL (Lenguaje de Reglas de la Web Semántica): es una propuesta de normas de la web semántica, es la combinación de sub-lenguajes de OWL con los lenguajes de reglas de marcado (Unario/Binarios Datalog). Disponible en:
http://www.w3.org/Submission/SWRL/

XML: Lenguaje de programación desarrollada por W3C, es una versión de SGML diseñado específicamente para los documentos de la web; permite que los diseñadores creen sus propias etiquetas, dando la definición, transmisión, validación e interpretación de datos entre aplicaciones y entre organizaciones. Disponible en: http://www.w3.org/XML/

W3C: Siglas de Word Wide Web Consorcio, un consorcio fundado en 1994 para dirigir la web hacia su pleno potencial mediante el desarrollo del protocolos comunes que promueven su evolución y aseguren su interoperabilidad. Disponible en: http://www.w3.org/

WORRDL WIDE WEB: Conocida también como web, red ó www; básicamente es un medio de comunicación de texto, gráficos y otros objetos multimedia a través de internet. Es decir la web es un sistema de hipertexto que utiliza internet como su mecanismo de transporte. Disponible en: http://www.w3c.es/

WIKI: Significa rápido, en términos tecnológicos es un software para la creación de contenido de forma colaborativa. Un sistema de creación, intercambio y revisión de información en la web, de forma fácil y automática. Disponible en: http://www.wikipedia.org/

XUL: (Lenguaje basado en XML para interfaz de usuario) es la aplicación de XML a la descripción de la interfaz de usuario en el navegador Mozilla. XUL es un estándar, la principal ventaja es que aporta una definición de interfaces GUI simple y portable. Disponible en: http://www.xul.fr/